A Thesis

entitled

Estimation of Unmeasured Radon Concentrations in Ohio Using Quantile Regression Forest

by

Neel Kamal Bandreddy

Submitted to the Graduate Faculty as partial fulfillment of the requirements for the Masters of Science Degree in Electrical Engineering

Dr. Vijay Devabhaktuni, Committee Chair

Dr. Ashok Kumar, Committee Member

Dr. Mansoor Alam, Committee Member

Dr. Patricia R. Komuniecki, Dean
College of Graduate Studies

The University of Toledo
December 2014
An Abstract of

Estimation of Unmeasured Radon Concentrations in Ohio Using Quantile Regression Forest

by

Neel Kamal Bandreddy

Submitted to the Graduate Faculty as partial fulfillment of the requirements for the Masters of Science Degree in Electrical Engineering

The University of Toledo
December 2014

The most stable isotope of radon is Radon-222, which is a decay product of radium-226 and an indirect decay product of uranium-238, a natural radioactive element. According to the United States Environmental Protection Agency (USEPA), radon is the primary cause of lung cancer among non-smokers. The USEPA classifies Ohio as a zone 1 state because the average radon screening level is more than 4 picocuries per liter. To perform preventive measures, knowing radon concentration levels in all the zip codes of a geographic area is necessary. However, it is impractical to collect the information from all the zip codes due to its inapproachability. Several interpolation techniques have been implemented by researchers to predict the radon concentrations in places where radon data is not available. Hence, to improve the prediction accuracy of radon concentrations, a new technique called Quantile Regression Forests (QRF) is proposed in this thesis. The conventional techniques like Kriging, Local Polynomial Interpolation (LPI), Global Polynomial Interpolation (GPI), and Radial Basis Function (RBF) estimate output using complex mathematics. Artificial Neural Networks (ANN) have been introduced to overcome this problem. Although ANNs show better prediction accuracy in comparison to more conventional techniques, many issues arise, including local minimization and over fitting. To overcome the inadequacies of existing methods, statistical learning techniques such as Support Vector Regres-
sion (SVR) and Random Forest Regression (RFR) were implemented. In this thesis, Quantile Regression Forest (QRF) is introduced and compared with SVR, RFR, and other interpolation techniques using available operational performance measures. The study shows that QRF has least validation error compared with other interpolation techniques.
To my Parents (Sri B. Ram Babu and Sri B. Laxmi) and my loving brother (B. N. Abhiram) for their unconditional love, support and encouragement.
Acknowledgments

I am deeply indebted to my advisor Dr. Vijay Devabhaktuni whose stimulating motivation and valuable ideas helped me in the completion of my master’s degree study and research. I would like to thank Dr. Ashok Kumar for his patience, guidance and encouragement. I also thank Dr. Mansoor Alam for being part of my thesis committee.

The financial support from the EECS Department in the form of a graduate/research assistantship is gratefully acknowledged.

I would like to thank my dearest friend Ruthwik Reddy Junuthula for his continuous support and motivation to complete my research.

I also thank my friends Tejas Madireddi, Srujan Reddy Baddam, Rahul Yarlagadda, Sravan Kumar Akula, Hari Kishore Yarrabathina, Sudheer Kumar Kuppili, Satya Sai Dattu Vangala, Vijaya Sena Reddy Karnati, Ravi Chandran Reddy Alla, Varun Reddy Chinthlakunta, Santosh Adivi for their love and support at all times. You made my stay in Toledo a memorable one. I also thank Jayaram Gummadi, Srujana Adusumilli for their help and guidance.

I am also very thankful to my friends and colleagues in the Research lab (NE 2042) at the University of Toledo for their assistance.

Above all, I would like to thank my parents, and my brother who with their love, and encouragement have made all this possible.
Contents

Abstract iii

Acknowledgments vi

Contents vii

List of Tables x

List of Figures xii

List of Abbreviations xiii

1 Introduction 1
   1.1 Problem Statement ................................................. 1
   1.2 Research Approach .................................................. 3
   1.3 Organization of Thesis ............................................. 5

2 Literature Review 6
   2.1 Role of Interpolation .............................................. 6
   2.2 Conventional Interpolation Techniques ............................ 7
   2.3 Review of Neural Network Techniques ............................. 9
   2.4 Results and Discussions .......................................... 12

3 Artificial Neural Networks 14
   3.1 Artificial Neural Network Approach .............................. 14
3.1.1 Methodology ........................................ 14
3.1.2 Results and Discussions .......................... 17
3.2 Knowledge-Based Neural Networks .................. 18
  3.2.1 Methodology .................................... 19
  3.2.2 Results and discussion ......................... 21
3.3 Correction-Based ANN Model Approach .............. 25
  3.3.1 Methodology .................................... 25
  3.3.2 Results and Discussions ......................... 27

4 Support Vector Regression and Random Forest Regression 34
  4.1 Support Vector Regression .......................... 34
    4.1.1 Review of Support Vector Machines ........... 34
    4.1.2 Methodology .................................. 35
    4.1.3 Results and Discussions ..................... 37
  4.2 Random Forest Regression .......................... 38
    4.2.1 Review of Random Forest Regression .......... 38
    4.2.2 Methodology .................................. 39
    4.2.3 Results and Discussions ..................... 41

5 Quantile Regression Forest ............................. 42
  5.1 Overview of Quantile Regression Forest ............ 42
  5.2 Methodology ...................................... 42
    5.2.1 Results and Discussions ..................... 44
  5.3 Performance Measures .............................. 44
    5.3.1 Mean Absolute Error .......................... 45
    5.3.2 Factor of Two ................................. 45
    5.3.3 Root Mean Square Error ....................... 46
    5.3.4 Fractional Bias ................................ 46
5.3.5 Normalized Mean Square Error ........................................ 46
5.3.6 Fractional variance (FV) .................................................. 46
5.3.7 Revised IOA ($IOA_r$) .................................................... 47
5.3.8 Geometric mean variance (VG) ......................................... 47
5.3.9 Accuracy for paired peak ($A_p$) ....................................... 48
5.3.10 Geometric mean bias (MG) ............................................. 48
5.3.11 Coefficient of correlation ($r$) ......................................... 48
5.3.12 Spearman correlation coefficient ($\rho$) ................................. 48

5.4 Results and Discussion ...................................................... 49

6 Conclusions and Future Work .............................................. 55
6.1 Conclusion ........................................................................ 55
6.2 Suggestions for Future Work .............................................. 56

References ............................................................................. 57

A Source Code ........................................................................ 65
A.1 Source code for Correction-Based ANN modeling approach ........ 65
A.2 Source code for nu-SVR using LIBSVM package ..................... 71
A.3 R Software commands for executing ‘randomForest’ package ...... 72
A.4 R Software commands for executing ‘quantregForest’ package ..... 73
List of Tables

2.1 Conventional Interpolation Techniques commonly used in Environmental Science ............................................. 7
2.2 Performance Measures of Conventional Interpolation Techniques ......... 12
3.1 Training and validation errors of ANN approach using Backpropagation algorithm .................................................. 17
3.2 Training and validation errors of ANN approach using Quasi-Newton algorithm .................................................. 18
3.3 Training and validation errors of PKI approach using Backpropagation algorithm .................................................. 22
3.4 Training and validation errors of PKI approach using Quasi-Newton algorithm .................................................. 22
3.5 Training and validation errors of difference models in SDM approach using Quasi-Newton algorithm .......................... 23
3.6 Training and validation errors of SM Model in SMNN approach using Quasi-Newton algorithm .......................... 24
3.7 Training and validation errors of coarse model in SMNN approach using Quasi-Newton algorithm .......................... 25
3.8 Training and validation errors of candidate correction model 1 using Quasi-Newton algorithm .......................... 28
3.9 Training and validation errors of candidate correction model 2 using Quasi-Newton algorithm .......................... 29
4.1 Validation error of Support Vector Regression with varying gamma value 38
4.2 Performance of RFR Based on Validation Error ($E_{avg}$) by Varying the Number of Trees  .................. 41

5.1 Performance of QRF Based on Validation Error ($E_{avg}$) by Varying the Number of Trees  .................. 44
5.2 Comparison of all interpolation techniques based on the training error, validation error and some operational performance measures  ........... 52
5.3 Comparison of all interpolation techniques based on some operational performance measures  ........... 53
5.4 Comparison of all interpolation techniques based on some operational performance measures  ........... 54
List of Figures

1-1 Counties having school rooms with radon concentration greater than 4 pCi/l. 2
1-2 Geometric Mean of the Indoor Radon Concentrations in Ohio Zip Code Areas. 4

2-1 Decay Chain of Uranium 238. 10
2-2 General Representation of PKI Method 10
2-3 General Representation of SDM Method 11
2-4 General Representation of SMNN Method 12

3-1 The MLP3 architecture 15
3-2 The MLP3 Architecture for PKI approach 19
3-3 The MLP3 Architecture for SDM approach 20
3-4 The MLP3 Architecture for SMNN approach 30
3-5 The MLP3 network architecture for the candidate correction model with latitude as output 31
3-6 The MLP3 network architecture for the candidate correction model with longitude as output 32
3-7 Flow Chart for Correction-Based Neural Networks using sensitivity based root finding algorithm 33

4-1 Work Flow of Random Forests 40
**List of Abbreviations**

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANN</td>
<td>Artificial Neural Networks</td>
</tr>
<tr>
<td>CAD</td>
<td>Computer Aided Design</td>
</tr>
<tr>
<td>CBNN</td>
<td>Correction-Based Artificial Neural Networks</td>
</tr>
<tr>
<td>CV</td>
<td>Cross-Validation</td>
</tr>
<tr>
<td>$F_{a_2}$</td>
<td>Factor of Two</td>
</tr>
<tr>
<td>FB</td>
<td>Fractional Bias</td>
</tr>
<tr>
<td>IARC</td>
<td>International Agency for Research on Cancer</td>
</tr>
<tr>
<td>KBNN</td>
<td>Knowledge Based Neural Networks</td>
</tr>
<tr>
<td>LPI</td>
<td>Local Polynomial Interpolation</td>
</tr>
<tr>
<td>MAE</td>
<td>Mean Average Error</td>
</tr>
<tr>
<td>NMSE</td>
<td>Normalized Mean Square Error</td>
</tr>
<tr>
<td>ODH</td>
<td>Ohio Department of Health</td>
</tr>
<tr>
<td>pCi/L</td>
<td>Pico-Curie per liter of air</td>
</tr>
<tr>
<td>PKI</td>
<td>Prior Knowledge Input</td>
</tr>
<tr>
<td>QRF</td>
<td>Quantile Regression Forest</td>
</tr>
<tr>
<td>RBF</td>
<td>Radial Basis Function</td>
</tr>
<tr>
<td>RFR</td>
<td>Random Forest Regression</td>
</tr>
<tr>
<td>RMSE</td>
<td>Root Mean Square Error</td>
</tr>
<tr>
<td>SDM</td>
<td>Source Difference Method</td>
</tr>
<tr>
<td>SM</td>
<td>Space Mapping</td>
</tr>
<tr>
<td>SMNN</td>
<td>Space Mapped Neural Networks</td>
</tr>
<tr>
<td>SVR</td>
<td>Support Vector Regression</td>
</tr>
<tr>
<td>USEPA</td>
<td>United States Environmental Protection Agency</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Problem Statement

Radon (Rn) is a tasteless, colorless, odorless, radioactive noble gas with an atomic number 86 that naturally occurs from the decay of uranium. Radon gas can accumulate in buildings through cracks and pores in floor slabs, basement walls, and openings around sump pumps and pipes that penetrate floors and walls. The International Agency for Research on Cancer (IARC) classified Radon-222 as carcinogenic to humans. The United States Environmental Protection Agency (USEPA) estimates that roughly 15,000 to 22,000 lung cancer deaths in the United States every year are related to radon [1]. It is the leading cause of death among nonsmokers, and the second overall leading cause of lung cancer. The units of radon concentration are in picocuries per liter of air (pCi/L), which are measured using radon-monitoring devices. The USEPA and the U.S. Geological survey categorized all 3,141 counties of the United States into three zones based on the radon potential: Zone 1 with Highest Potential (greater than 4 pCi/L), Zone 2 with Moderate Potential (from 2 to 4 pCi/L), and Zone 3 with Low Potential (less than 2 pCi/L). All of Ohio’s counties are listed as either Zone 1 or Zone 2 (Highest and Moderate potential, respectively). The USEPA recommends that homes or schools exceeding the radon concentration
of 4pCi/L should be remediated and advises people to take preventive measures in order to reduce the radon concentration [2]. Figure 1-1 shows the counties having school rooms with radon concentration greater than 4 pCi/L.

Figure 1-1: Counties having school rooms with radon concentration greater than 4 pCi/L. (http://www.eng.utoledo.edu/aprg/radon/information/schoolsfig.html)

The USEPA has shown through considerable research that it is possible to bring down the radon levels in homes or schools [3]. Since 1989, the Ohio Department of Health (ODH), in conjunction with radon testing laboratories, local health departments, and university researchers has been collecting radon information from homes and schools across Ohio in order to identify the buildings with high radon
concentration levels and perform a cost effective radon mitigation plan [4–7]. Various database management systems manage the data collected by the ODH [8–13]. The Ohio Radon Information System (ORIS) includes most of the Ohio indoor radon data, which were collected by The University of Toledo in association with Ohio Air Quality Development Authority and Ohio Department of Health (ODH) [11]. In Ohio, radon concentration data are available for around 1257 out of 1500 zip codes, but it is not possible to survey all the zip codes, for reasons such as, inapproachability. For the remaining zip codes where data cannot be collected, radon concentration can be estimated by using various interpolation techniques. Figure 1-2 shows the indoor radon concentrations in Ohio [14].

1.2 Research Approach

Initial research to fill the missing data was focused on conventional interpolation techniques such as Kriging, Local Polynomial Interpolation (LPI), Global Polynomial Interpolation (GPI) and Radial Basis Function (RBF) to estimate the radon concentration [15–18]. These techniques involve large data and complex mathematics, hence Artificial Neural Networks (ANN) such as 3-layer Multi-Layer Perceptron (MLP3) were introduced [19]. Due to poor extrapolation capability, limited generalization, and hard interpretation of data of MLP3 ANNs, researchers had to move on to more efficient ANNs such as Knowledge Based Neural Networks (KBNN) and Correction-Based Neural Networks (CBNN). There are different KBNN models such as Prior Knowledge Input (PKI), Source Difference Method (SDM), and Spatial Mapped Neural Networks (SMNN) that improve the generalization ability and extrapolation capability [20]. In this thesis, uranium concentrations provided by the Ohio Department of Natural Resources were used as existing knowledge for PKI-based model training. To overcome the drawbacks of structural complexity, an improved
KBNN called Correction-Based Neural Networks was introduced [22]. To overcome the disadvantages of neural networks, two regression techniques were also introduced, namely Random Forest Regression (RFR) and Support Vector Regression (SVR) for estimating and modeling the indoor radon concentrations. This thesis presents a new regression technique called Quantile Regression Forest (QRF), an advancement to Random Forest Regression (RFR). Comparisons between conventional interpolation techniques, neural network approaches, SVR and RFR techniques, and the newly
introduced QRF techniques with different performance measures will be discussed in this thesis.

1.3 Organization of Thesis

The remaining thesis is structured as follows. Chapter 2 discusses the role of interpolation and the conventional interpolation techniques and review of artificial neural networks. Chapter 3 discusses ANN approach, KBNN, and CBNN. Chapter 4 gives the detailed discussion of SVR and RFR. Chapter 5 discusses the use of newly introduced regression technique QRF and some operational performance measures to predict the accuracy of radon concentration. Finally, chapter 6 concludes the thesis and suggests the future work.
Chapter 2

Literature Review

2.1 Role of Interpolation

Interpolation is a mathematical function that estimates the values where no actual value can be measured. Spatially continuous data of environmental variables are required to estimate missing values. Thus, the attribute values at unmeasured areas need to be predicted to produce spatially continuous data. In such cases, spatial interpolation methods have been implemented where attribute values are not available to create the spatially continuous data [23]. Spatial interpolation assumes that the attribute data is spatially dependent and are continuous over space where estimation at any unmeasured area is within the data boundary.

The data used in this thesis were provided by ODH, collected from testing laboratories for estimation of radon and mapping radon concentration across Ohio. These predicted results are further used for quantitative research for reducing the radon related cancer deaths. Therefore, accuracy in the interpolation method is a crucial factor for unmeasured radon concentration estimation as it is the basis for further research studies.

Spatial interpolation techniques may be either deterministic or stochastic. Deterministic methods have no estimates of possible errors, whereas stochastic methods
Table 2.1: Conventional Interpolation Techniques commonly used in Environmental Science

<table>
<thead>
<tr>
<th>Interpolation Technique</th>
<th>Working Principle</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kriging</td>
<td>Linear unbiased prediction including spatial arrangement of data</td>
<td>Fits the polynomial function without any edge-effects</td>
<td>Involves complex and sophisticated programming</td>
</tr>
<tr>
<td>GPI</td>
<td>Captures coarse-scale patterns in the data and fits the polynomial function</td>
<td>Easy to Compute and can predict data with global variations</td>
<td>As the complexity increases the estimation error also increases exponentially</td>
</tr>
<tr>
<td>LPI</td>
<td>Similar to GPI but fits the polynomial function to a local subset of data</td>
<td>Data with short range variations can be interpolated effectively</td>
<td>Unable to handle global variations</td>
</tr>
<tr>
<td>RBF</td>
<td>Fits the polynomial function but the curve is irregular</td>
<td>Requires less data</td>
<td>Fails for the data having short range variations and does not fit for extrapolation</td>
</tr>
</tbody>
</table>

provide probabilistic estimates [24]. The working principle, advantages and disadvantages of various interpolation techniques are discussed in Table 2.1 [16].

### 2.2 Conventional Interpolation Techniques

Conventional interpolation techniques such as Kriging, GPI, LPI, and RBF have been employed in this thesis to predict the unmeasured radon concentrations in Ohio using split sample validation [15]. These interpolation methods are as follows.

Kriging is based on spatial auto correlation of data and is the best linear unbiased spatial predictor. Kriging not only predicts the attribute values (i.e., radon concentration) but also estimates the accuracy in prediction. Kriging can compute unique to stochastic methods. The ability to fit the polynomial function without any
edge effects, is another advantage. However, it has its fair share of drawbacks, such as non-stationary real world data sets and also involves complex programming. Recently, Kriging has been used in remote sensing image analysis and geostatistics [25].

GPI uses least-squares regression to fit the smooth surface over the input sample points. GPI is useful in determining the global data but fails at minor details. GPI is easy to compute and can predict data with global variations. GPI is based on the order of the polynomial function. It works well for lower order polynomial function, but as the order goes higher it results in arbitrary results. Also, as the complexity increases the estimation error also increases exponentially. Earlier, GPI was used in the agriculture field (Komuscu et al. 1998) and also in the field of climatology (Nalder and Wein 1998; Ninyerola et al. 2000), etc.

LPI is useful in determining the local variations but fails at global trends. Unlike GPI, the LPI fits many polynomial functions, within the individual neighborhoods. LPI is based on neighborhood distance. LPI produce surfaces by varying the shape, number of points to be used, and sector configuration that constitute more local variations. Previously, LPI have been used in hydro-logic modeling of daily precipitations [26], environmental sciences for estimating the climate [27]. GPI captures the long-range variations, however LPI can only capture the short-range variations.

RBF is an interpolation technique which approximates multivariable functions by linear combination of different radial basis functions. The five different radial basis functions used by RBF are spline with tension, thin-plate spline, multi-quadric function, inverse multi-quadric function, and completely regularized spline. Conceptually, RBF is like exactly fitting a rubber film through the given sample values, minimizing the total curvature of surface. Less data sample values are sufficient for RBF to predict the attribute values independent of the direction. The drawback of RBF interpolation is that RBF can only predict the values which are below minimum or above maximum, which is not feasible at all points. RBF has been previously em-
ployed to control the supercritical main stream temperature of a power plant [28] and in location of ores (Journel and Huijbregts 1978; Richmond 2003)

### 2.3 Review of Neural Network Techniques

Due to generalization capability and faster learning ability of ANNs, they are being used in computer aided modeling applications. Conventional interpolation techniques fail to derive the input-output relationship for the environmental data as the data pattern is complex, however ANNs proved to be efficient. The effectiveness of ANNs depends on the number of hidden neurons and training samples. ANNs are being used for regression and data classification and in many fields irrespective of its deficiency in extrapolation capability.

Recently, ANNs are being used to diagnose several cancers [37]. ANNs are being applied in various categories like data processing, robotics, function approximation etc. Due to their learning ability and data classification, ANNs are also used to identify hand written zip-code by the U.S. Postal Service [29]. ANNs play a vital role in estimating the measurements of air quality [30, 42] and water quality [43, 44] using available data for years [46–48]. ANNs are known for their generalization capability. Unlike conventional methods, ANNs can model the data with less complexity in programming.

The advanced ANNs called KBNN were introduced because they improve both the generalization and the extrapolation capabilities of a model [41]. As the name indicates, the neural networks model the data by considering additional knowledge other than inputs and outputs. There are different KBNNs such as the Prior Knowledge Input (PKI) method, the Source Difference Method (SDM), and the Space-Mapped Neural Network (SMNN) [49].

Uranium concentration is considered as the additional input to the neural network
as radon is the decay product of uranium. Figure 2-1 represents the Decay Chain of Uranium 238. Buildings or Schools that were built above uranium-bearing rocks may have higher radon concentration levels as mentioned in Harrell et al [6].

The Prior Knowledge Input (PKI) method is a KBNN, proposed in [35], and was limited to split-sample validation in the process of estimating radon concentration. This method uses the uranium concentration as the additional knowledge input to the existing MLP3 neural network. A general representation of PKI method is shown in Figure 2-2.

![General Representation of PKI Method](image)

The Source Difference Method (SDM) was proposed in [36]. This method involves two MLP3 neural networks; namely, coarse model and a difference model. The addi-
tional knowledge in this method is the estimation error. The SDM method proposed in [36] is a combination of two different MLP3 models. The coarse model is trained and validated using the original inputs and actual radon concentration (attribute value) as output. After selecting the best coarse model, estimated error is calculated using the predicted radon values. The difference model is trained and validated using original inputs and estimated error as the desired output. Final output is obtained by summing the predictions from both the models. A general representation of SDM method is shown in Figure 2-3.

![General Representation of SDM Method](image)

The Space-Mapped Neural Network (SMNN) was proposed in [37]. The SMNN maps the actual radon concentration into a coarse model input-space by Space-Mapping (SM) technique. Actual radon concentration is named as a fine model. The SM technique is used to map the fine model input-space to a coarse model input-space. Finally, by simulating the coarse model, the radon concentration is obtained. A general representation of SMNN method is shown in Figure 2-4.

Recently, another kind of ANN was introduced in [22] other than KBNN, which is Correction based ANN.

The correction based ANN model approach proposed in [22] lessens the structural complexity of a neural model and improves both the generalization and extrapolation capabilities [22]. A sensitivity-based root finding algorithm has been employed to
2.4 Results and Discussions

ArcGIS v10 software [15] has been used to implement the conventional interpolation techniques: Kriging, LPI, GPI and RBF for the estimation of radon concentrations. The performance measures of these conventional interpolation techniques are compared in Table 2.2.

Table 2.2: Performance Measures of Conventional Interpolation Techniques

<table>
<thead>
<tr>
<th>Conventional Interpolation Technique</th>
<th>Training Error</th>
<th>$E_{avg}$</th>
<th>MAE</th>
<th>$F_{a2}$</th>
<th>RMSE</th>
<th>FB</th>
<th>NMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kriging</td>
<td>2.13</td>
<td>3.89</td>
<td>1.516</td>
<td>0.789</td>
<td>2.616</td>
<td>6.69e-04</td>
<td>0.586</td>
</tr>
<tr>
<td>LPI</td>
<td>4.05</td>
<td>3.77</td>
<td>1.466</td>
<td>0.785</td>
<td>2.361</td>
<td>7.66e-04</td>
<td>0.480</td>
</tr>
<tr>
<td>GPI</td>
<td>5.15</td>
<td>4.70</td>
<td>1.828</td>
<td>0.682</td>
<td>2.627</td>
<td>6.81e-04</td>
<td>0.580</td>
</tr>
<tr>
<td>RBF</td>
<td>0.39</td>
<td>3.79</td>
<td>1.477</td>
<td>0.793</td>
<td>2.576</td>
<td>6.42e-04</td>
<td>0.559</td>
</tr>
</tbody>
</table>

From Table 2.2, it can be observed that LPI and RBF has performed better compared to other interpolation techniques to estimate $E_{avg}$, and calculate MAE and
The ideal value is zero for $E_{avg}$, MAE, RMSE, FB and NMSE, and one for $F_{a2}$. The magnitude error (RMSE and NMSE) is low for LPI as shown in Table 2.2. RBF is closer to the ideal value of $F_{a2}$ compared to other conventional interpolation techniques. Though Kriging and RBF have least FB, it can be concluded that they are not suitable for short-range data interpolation as these techniques are comparatively high in FB. The drawback of conventional interpolation techniques is their complex mathematical functions. In order to overcome the drawbacks, the ANN approach was introduced and discussed in Chapter 3.
Chapter 3

Artificial Neural Networks

3.1 Artificial Neural Network Approach

Artificial Neural Networks (ANNs), due to their generalization capability and faster learning ability, are being used in computer aided modeling applications. ANNs are feed forward multilayer networks. MLP3, a 3 layer perceptron is the most commonly used ANN. ANNs are widely used in applications such as signal processing and pattern recognition [38,39]. In this thesis, ANN is implemented using a split sample validation technique [19]. Unlike the conventional techniques, ANNs are not based on complex mathematical modeling [40].

A MLP3 network consists of an input layer, a hidden layer and an output layer. A relay function, a sigmoid activation function, and a linear function are used for the input layer, the hidden layer, and the output layer respectively. ANNs are implemented using a Computer Aided Design (CAD) tool called Neuromodeler.

3.1.1 Methodology

In Figure 3-1, let \( x \) be the vector that contains model inputs \( X_1 \) and \( X_2 \). In radon modeling, latitude and longitude are considered as inputs. Let \( y \) be the vector that contain model output. The relationship between vectors \( x \) and \( y \) can be defined as
\[ y = f(x) \] (3.1)

In Eq. 3.1, \( f \) represents the functional relationship between \( x \) and \( y \). In radon modeling, \( f \) is a neural network that is derived from the training process using a set of sample pairs given by

\[ [(x_p, d_p), p = 1, ..., N] \] (3.2)

In Eq. 3.2, \( d_p \) represents the desired output corresponding to \( p^{th} \) training input vector \( x_p \), \( N \) represents number of data samples available for training and \( p \) represents sample index.
In reality, model parameters referred as weights $w$ are included in neural network which are first initialized and then iteratively adjusted to minimize the error, given by

$$E(w) = \frac{1}{2} \sum_{p=1}^{N} \sum_{p=1}^{N} (y_{pq}(x_p, w) - d_{pq})^2$$  \hspace{1cm} (3.3)

In Eq. 3.3, $y_{pq}(x_p, w)$ is the $q^{th}$ output of the neural network with input $x_p$, $N$ is the number of data samples for training. In radon modeling, as ANN has only one output, $m$ is considered as 1 (i.e. $m=1$) and the model parameter $w$ is constantly updated, given by

$$w_{\text{next}} = w_{\text{now}} + \eta g$$ \hspace{1cm} (3.4)

In Eq. 3.4, $\eta$ represents positive step-size and $g$ represents update direction. Conceptually, $w_{\text{next}}$ is the adjusted weight from the current weight $w_{\text{now}}$ along the update direction $g$.

The output at every $j^{th}$ hidden neuron is given by

$$z_j = \frac{1}{1 + e^{-(\sum_{i=1}^{m} u_{ij}x_i + u_{0j})}}$$ \hspace{1cm} (3.5)

In Eq. 3.5, $u_{ij}$ is the weight between the $i^{th}$ input neuron and the $j^{th}$ hidden neuron, $x_i$ is latitude or longitude in present case, and $u_{0j}$ is the bias parameter for $j^{th}$ hidden neuron.

Finally, the model output (radon concentration) is calculated as

$$y = \sum_{j=1}^{h} (z_j v_j) + v_0$$ \hspace{1cm} (3.6)

In Eq. 3.6, $z_j$ is the sigmoid activation function, $v_j$ is the weight between the $j^{th}$
hidden neuron and the output neuron, \( h \) is the total number of hidden neurons and \( v_0 \) is the bias parameter.

The objective is to minimize the validation error of Artificial Neural Network (ANN) i.e., \( E_{\text{avg}} \) is given as

\[
E_{\text{avg}} = \frac{1}{N_v} \sum_{i=1}^{N_v} \left( \frac{f_{\text{ann}}(x_i, w) - y_i}{y_{\text{max}} - y_{\text{min}}} \right)
\]

(3.7)

In Eq. 3.7, \( N_v \) is the number of training samples, \( y_i \) is the actual output and \( f_{\text{ann}}(x_i, w) \) is the predicted output to the input vector \( x_i \).

### 3.1.2 Results and Discussions

MLP3 is the most common used ANN which is trained and validated by varying number of hidden neurons. There are many supervised learning algorithms to model the data. In this thesis, Backpropagation and Quasi-Newton are used for modeling. Based on the validation error \( E_{\text{avg}} \), the best model can be identified. The Ideal value of \( E_{\text{avg}} \) is zero. Table 3.1 and Table 3.2 represent the training and validation errors of ANN approach using Backpropagation and Quasi-Newton algorithms, respectively.

#### Table 3.1: Training and validation errors of ANN approach using Backpropagation algorithm

<table>
<thead>
<tr>
<th>No. of Hidden Neurons</th>
<th>Training Error</th>
<th>Validation Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( E_{\text{avg}}(%) )</td>
<td>( E_{\text{worst}}(%) )</td>
</tr>
<tr>
<td>10</td>
<td>4.63</td>
<td>4.87</td>
</tr>
<tr>
<td>20</td>
<td>4.44</td>
<td>4.19</td>
</tr>
<tr>
<td>30</td>
<td>4.79</td>
<td>4.38</td>
</tr>
<tr>
<td>40</td>
<td>4.36</td>
<td>4.91</td>
</tr>
<tr>
<td>50</td>
<td>4.47</td>
<td>4.29</td>
</tr>
<tr>
<td>60</td>
<td>4.42</td>
<td>4.83</td>
</tr>
<tr>
<td>70</td>
<td>4.65</td>
<td>4.26</td>
</tr>
<tr>
<td>80</td>
<td>4.29</td>
<td>4.80</td>
</tr>
<tr>
<td>90</td>
<td>4.46</td>
<td>4.03</td>
</tr>
<tr>
<td>100</td>
<td>4.57</td>
<td>4.18</td>
</tr>
</tbody>
</table>
Comparatively, from Table 3.1 and Table 3.2, it can be observed that training by Quasi-Newton algorithm has better results compared to Backpropagation. From Table 3.1, it can be observed that neural network using the Backpropagation algorithm has least validation error \( E_{avg} \) of 4.03 for 90 hidden neurons. From Table 3.2, it can be observed that neural network using Quasi-Newton algorithm has the least validation error \( E_{avg} \) of 3.98 for 40 hidden neurons. In Table 5.2, Table 5.3 and Table 5.4, Artificial Neural Network (ANN) is compared with conventional interpolation techniques, KBNN, CBNN, SVR, RFR, and QRF using some performance measures.

### 3.2 Knowledge-Based Neural Networks

The advanced ANNs called KBNN were introduced because they improve both the generalization and the extrapolation capabilities of a model [41]. As the name indicates, the neural networks model the data by considering additional knowledge other than inputs and outputs. In simple terms, a knowledge-based neural network is ‘addition of a prior knowledge to the existing neural networks’ i.e., adding prior knowl-
edge like uranium concentration [20] as an additional input to the conventional ANN. There are different KBNNs such as the Prior Knowledge Input (PKI) method, the Source Difference Method (SDM), and the Space-Mapped Neural Network (SMNN).

### 3.2.1 Methodology

The Prior Knowledge Input (PKI) method is a KBNN, proposed in [35], was limited to split-sample validation in the process of estimating radon concentration. This method uses the uranium concentration as the additional knowledge input to the existing neural network, which is MLP3. The PKI method is trained and validated using latitude, longitude (original inputs) and uranium concentration (knowledge input) as inputs and radon as output, thus improving the prediction accuracy. Availability of uranium concentration for each sample in predicting radon concentration is its limitation. Figure 3-2 represents the MLP3 Architecture for PKI approach.

![Figure 3-2: The MLP3 Architecture for PKI approach](image)

19
The Source Difference Method (SDM) was proposed in [36]. This method involves two MLP3 neural networks namely coarse model and a difference model. The additional knowledge in this method is the estimation error. The coarse model is trained and validated using latitude, longitude as inputs and radon as the desired output. By changing the number of hidden neurons, the best coarse model can be evaluated. Figure 3-3 represents the MLP3 Architecture for SDM approach.
Once the best coarse model is evaluated, an estimation error is calculated by subtracting the predicted radon concentration obtained from coarse model from the actual radon concentration. The difference model is trained and validated using latitude, longitude as inputs and estimation error as the desired output. Once the best difference model is evaluated the outputs of both the models (coarse model, difference model) are combined to get the predicted values of radon concentrations.

The Space-Mapped Neural Network (SMNN) was proposed in [37]. The SMNN is mapping the actual radon concentration into a coarse model input-space by Space-Mapping (SM) technique. Actual radon concentration is named as a fine model. The process starts with training the neural network by interchanging output and inputs i.e., radon as input, and latitude and longitude as outputs to obtain the predicted values latitude’ and longitude’. The SM technique is used to map the fine model input-space $x_f$ to a coarse model input-space $x_c$. The SM model is trained using latitude, longitude and uranium concentration as inputs while latitude’ and longitude’ as outputs to obtain the predicted latitude’ and longitude’, say latitude” and longitude”. Finally, the coarse model is used to train the neural network using latitude” and longitude” as inputs and radon concentration as the output to obtain the predicted radon concentration. Figure 3-4 represents the MLP3 Architecture for SMNN approach.

### 3.2.2 Results and discussion

In PKI approach, MLP3 network is trained and validated by varying number of hidden neurons. In this thesis, Backpropagation and Quasi-Newton are used for modeling PKI approach. Based on the validation error $E_{avg}$, the best model can be identified. The Ideal value of $E_{avg}$ is zero. Tables 3.3 and 3.4 represent the training and validation errors of PKI approach using Backpropagation and Quasi-Newton algorithms, respectively.
Table 3.3: Training and validation errors of PKI approach using Backpropagation algorithm

<table>
<thead>
<tr>
<th>No. of Hidden Neurons</th>
<th>Training Error</th>
<th>Validation Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$E_{avg}$ (%)</td>
</tr>
<tr>
<td>10</td>
<td>4.62</td>
<td>4.22</td>
</tr>
<tr>
<td>20</td>
<td>4.91</td>
<td>4.03</td>
</tr>
<tr>
<td>30</td>
<td>4.36</td>
<td>4.14</td>
</tr>
<tr>
<td>40</td>
<td>4.64</td>
<td>4.21</td>
</tr>
<tr>
<td>50</td>
<td>4.15</td>
<td>4.26</td>
</tr>
<tr>
<td>60</td>
<td>4.54</td>
<td>4.33</td>
</tr>
<tr>
<td>70</td>
<td>4.16</td>
<td>4.51</td>
</tr>
<tr>
<td>80</td>
<td>4.87</td>
<td>4.32</td>
</tr>
<tr>
<td>90</td>
<td>4.25</td>
<td>4.10</td>
</tr>
<tr>
<td>100</td>
<td>4.33</td>
<td>4.11</td>
</tr>
</tbody>
</table>

From Table 3.3, it can be observed that PKI approach using the Backpropagation algorithm has least validation error $E_{avg}$ of 4.03 for 20 hidden neurons.

Table 3.4: Training and validation errors of PKI approach using Quasi-Newton algorithm

<table>
<thead>
<tr>
<th>No. of Hidden Neurons</th>
<th>Training Error</th>
<th>Validation Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$E_{avg}$ (%)</td>
</tr>
<tr>
<td>10</td>
<td>4.38</td>
<td>4.01</td>
</tr>
<tr>
<td>20</td>
<td>4.02</td>
<td>3.92</td>
</tr>
<tr>
<td>30</td>
<td>4.20</td>
<td>4.13</td>
</tr>
<tr>
<td>40</td>
<td>4.36</td>
<td>3.97</td>
</tr>
<tr>
<td>50</td>
<td>3.98</td>
<td>3.84</td>
</tr>
<tr>
<td>60</td>
<td>4.20</td>
<td>4.05</td>
</tr>
<tr>
<td>70</td>
<td>4.43</td>
<td>4.12</td>
</tr>
<tr>
<td>80</td>
<td>4.14</td>
<td>3.96</td>
</tr>
<tr>
<td>90</td>
<td>4.34</td>
<td>3.89</td>
</tr>
<tr>
<td>100</td>
<td>4.39</td>
<td>4.03</td>
</tr>
</tbody>
</table>

From Table 3.4, it can be observed that PKI approach using the Quasi-Newton algorithm has least validation error $E_{avg}$ of 3.84 for 50 hidden neurons.

In SDM modeling, the coarse model is trained and validated using quasi-newton
algorithm by changing the number of hidden neurons in the hidden layer. In this thesis, the coarse model is considered as the ANN modeling approach, for which the training and validation errors are shown in Table 3.2 using Quasi-Newton algorithm. The best coarse model is observed for 40 hidden neurons.

The difference model is trained and validated by changing the number of hidden neurons which is shown in Table 3.5, and the best difference model has the least $E_{\text{avg}}$ of 3.86 at 20 hidden neurons. The sum of the predicted values of the best coarse model and the best difference model gives the SDM model. Hence, by summing the two models, the resulting validation error $E_{\text{avg}}$ for SDM is 4.09.

Table 3.5: Training and validation errors of difference models in SDM approach using Quasi-Newton algorithm

<table>
<thead>
<tr>
<th>No. of Hidden Neurons</th>
<th>Training Error</th>
<th>Validation Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$E_{\text{avg}}$(%)</td>
</tr>
<tr>
<td>10</td>
<td>4.43</td>
<td>4.38</td>
</tr>
<tr>
<td>20</td>
<td>4.72</td>
<td>3.86</td>
</tr>
<tr>
<td>30</td>
<td>4.55</td>
<td>3.92</td>
</tr>
<tr>
<td>40</td>
<td>4.95</td>
<td>4.12</td>
</tr>
<tr>
<td>50</td>
<td>4.25</td>
<td>4.09</td>
</tr>
<tr>
<td>60</td>
<td>4.78</td>
<td>3.99</td>
</tr>
<tr>
<td>70</td>
<td>4.32</td>
<td>4.66</td>
</tr>
<tr>
<td>80</td>
<td>4.60</td>
<td>4.65</td>
</tr>
<tr>
<td>90</td>
<td>4.33</td>
<td>4.26</td>
</tr>
<tr>
<td>100</td>
<td>4.40</td>
<td>4.55</td>
</tr>
</tbody>
</table>

From Table 3.5, it can be observed that difference models using Quasi-Newton algorithm has least $E_{\text{avg}}$(%) of 3.86 for 20 hidden neurons.

In modeling the SMNN approach, two models namely the SM model and coarse model are trained and validated by varying number of hidden neurons. Table 3.6 represent the training and validation errors of SM model using Quasi-Newton algorithm. Once, the SM model is trained and validated, a best model is selected. The best coarse model is observed for 60 hidden neurons based on the validation error.
$E_{avg}$ (%). The coarse model is then trained by using space mapped latitude’ and longitude’ as inputs by varying number of hidden neurons, while radon concentration as the output.

Table 3.6: Training and validation errors of SM Model in SMNN approach using Quasi-Newton algorithm

<table>
<thead>
<tr>
<th>No. of Hidden Neurons</th>
<th>Training Error</th>
<th>Validation Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$E_{avg}$ (%)</td>
</tr>
<tr>
<td>10</td>
<td>7.68</td>
<td>7.53</td>
</tr>
<tr>
<td>20</td>
<td>7.34</td>
<td>7.39</td>
</tr>
<tr>
<td>30</td>
<td>7.51</td>
<td>7.65</td>
</tr>
<tr>
<td>40</td>
<td>8.03</td>
<td>7.23</td>
</tr>
<tr>
<td>50</td>
<td>7.96</td>
<td>7.06</td>
</tr>
<tr>
<td>60</td>
<td>8.22</td>
<td>7.95</td>
</tr>
<tr>
<td>70</td>
<td>7.84</td>
<td>7.49</td>
</tr>
<tr>
<td>80</td>
<td>7.60</td>
<td>7.36</td>
</tr>
<tr>
<td>90</td>
<td>7.37</td>
<td>7.52</td>
</tr>
<tr>
<td>100</td>
<td>7.88</td>
<td>7.30</td>
</tr>
</tbody>
</table>

From Table 3.6, it is observed that, SM model using Quasi-Newton algorithm has least $E_{avg}$ of 7.06 for 50 hidden neurons.

From Table 3.7, it is observed that coarse model using Quasi-Newton algorithm has least $E_{avg}$ of 4.01 for 30 hidden neurons. In Table 5.2, Table 5.3 and Table 5.4, Knowledge-Based Neural Network (KBNN) is compared with conventional interpolation techniques, ANN, CBNN, SVR, RFR, and QRF using some performance measures.
Table 3.7: Training and validation errors of coarse model in SMNN approach using Quasi-Newton algorithm

<table>
<thead>
<tr>
<th>No. of Hidden Neurons</th>
<th>Training Error</th>
<th>Validation Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$E_{\text{avg}}$(%)</td>
</tr>
<tr>
<td>10</td>
<td>4.46</td>
<td>4.20</td>
</tr>
<tr>
<td>20</td>
<td>4.61</td>
<td>4.37</td>
</tr>
<tr>
<td>30</td>
<td>4.76</td>
<td>4.01</td>
</tr>
<tr>
<td>40</td>
<td>4.37</td>
<td>4.09</td>
</tr>
<tr>
<td>50</td>
<td>4.27</td>
<td>4.61</td>
</tr>
<tr>
<td>60</td>
<td>4.62</td>
<td>4.55</td>
</tr>
<tr>
<td>70</td>
<td>4.51</td>
<td>4.29</td>
</tr>
<tr>
<td>80</td>
<td>4.03</td>
<td>4.45</td>
</tr>
<tr>
<td>90</td>
<td>4.59</td>
<td>4.64</td>
</tr>
<tr>
<td>100</td>
<td>4.74</td>
<td>4.97</td>
</tr>
</tbody>
</table>

3.3 Correction-Based ANN Model Approach

Implementation of KBNNs are structurally complex. In order to avoid the structural complexity, the Correction-Based model using the sensitivity technique is implemented.

3.3.1 Methodology

The correction based ANN model approach was proposed in [22] lessens the structural complexity of a neural model and improves both the generalization and extrapolation capabilities [22]. The conventional neural model $f_{\text{ann}}$, candidate correction model with latitude as the output $f_{\text{ann,1}}$, and candidate correction model with longitude as the output $f_{\text{ann,2}}$ are included in this approach in order to improve the prediction accuracy of $f_{\text{ann}}$ and achieve the condition $E_{\text{obj}} < E_{\text{user}}$. User defined error is considered as $E_{\text{user}}$ and the validation error $E_{\text{obj}}$ of each sample in $f_{\text{ann,j}}$ is given as.
\[ E_{\text{obj}} = \frac{\text{original} - \text{predicted}}{\text{original}} \times 100 \] 

The set of candidate correction models are denoted by \( f_{\text{ann},j} \) which is unknown initially. Let the desired neural model be

\[ y = f_{\text{ann}}(x_1, x_2, w) \] 

(3.9)

And the corresponding set of candidate correction models be

\[ x_{1,c} = f_{\text{ann},1}(y, x_2, w_1) \] 

(3.10)

\[ x_{2,c} = f_{\text{ann},2}(x_1, y, w_2) \] 

(3.11)

where \( x_{1,c} \) represents the candidate correction model with latitude as the desired output, \( x_{2,c} \) represents the candidate correction model with longitude as the desired output, \( y \) is the radon concentration, \( x_1 \) and \( x_2 \) are latitude and longitude respectively, \( w_1 \) and \( w_2 \) are the weight vectors of the first candidate correction model and second candidate correction model respectively. Figure 3-5 shows the MLP3 network architecture for the candidate correction model with latitude as output and Figure 3-6 shows the MLP3 network architecture for the candidate correction model with longitude as output. Figure 3-7 shows the Flow Chart for Correction-Based Neural Networks using sensitivity based root finding algorithm.

The first step after modeling the two candidate correction models is to select the best candidate correction model based on the error criterion \( E_{\text{avg}} \) [22]. The root finding algorithm is executed iteratively considering the output from \( f_{\text{ann}} \) as the initial solutions, such that the condition \( E_{\text{obj}} < E_{\text{user}} \) is attained. A sensitivity-based root finding algorithm uses the partial derivatives (Chain rule) to find the step-size and
update the direction. The partial derivatives of the candidate model correction model \( f_{ann,j} \) with respect to output \( y \) is given by

\[
\frac{\partial x_{j,c}}{\partial y} = \sum_{k=1}^{r} \frac{\partial x_{j,c}}{\partial z_k} \frac{\partial z_k}{\partial y_k} \frac{\partial y_k}{\partial y} \tag{3.12}
\]

In Eq. 3.12, \( r \) is the number of hidden neurons, \( y_k \) is weighted sum of all inputs to \( f_{ann,j} \) and \( z_k \) is the sigmoid activation function. The sigmoid activation function is employed in ANN training because of its better computational performance compared to the other functions. The sigmoid activation function \( z_k \) is given as

\[
z_k = \frac{1}{1 + e^{-y_k}} \tag{3.13}
\]

Finally, after implementing the root finding algorithm iteratively, the updated output \( y \) which is more accurate compared with ANN and KBNN approaches is obtained.

### 3.3.2 Results and Discussions

The correction-based ANN approach has been implemented using MATLAB software [59]. The two candidate correction models, one with the latitude as the output and the other with longitude as the output are trained and validated. Tables 3.8 and 3.9 represent the training and validation errors of candidate correction model 1 (latitude as the desired output) using Quasi-Newton algorithm and the training and validation errors of candidate correction model 2 (longitude as the desired output) using Quasi-Newton algorithm respectively. Therefore, fixed number of hidden neurons are taken for both the models, say 20 hidden neurons, to keep the same structure for both the models. Once, the least validation error candidate correction model is identified, it is paired with ANN model, and the root finding algorithm has been implemented such that \( E_{obj} < E_{user} \) is achieved. In the algorithm, \( E_{obj} \) is calculated
for every sample. After executing the root finding algorithm, the resulting $E_{avg}$ for correction-based ANN model is 4.33.

Table 3.8: Training and validation errors of candidate correction model 1 using Quasi-Newton algorithm

<table>
<thead>
<tr>
<th>No. of Hidden Neurons</th>
<th>Training Error</th>
<th>Validation Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$E_{avg}$ (%)</td>
<td>$E_{worst}$ (%)</td>
</tr>
<tr>
<td>10</td>
<td>15.92</td>
<td>15.33 80.34</td>
</tr>
<tr>
<td>20</td>
<td>15.01</td>
<td>15.12 83.64</td>
</tr>
<tr>
<td>30</td>
<td>14.65</td>
<td>15.86 91.06</td>
</tr>
<tr>
<td>40</td>
<td>16.03</td>
<td>15.68 87.36</td>
</tr>
<tr>
<td>50</td>
<td>15.36</td>
<td>16.32 90.67</td>
</tr>
<tr>
<td>60</td>
<td>16.59</td>
<td>16.67 86.69</td>
</tr>
<tr>
<td>70</td>
<td>15.49</td>
<td>16.43 85.09</td>
</tr>
<tr>
<td>80</td>
<td>15.60</td>
<td>15.94 92.37</td>
</tr>
<tr>
<td>90</td>
<td>16.29</td>
<td>16.03 96.78</td>
</tr>
<tr>
<td>100</td>
<td>14.98</td>
<td>16.64 89.33</td>
</tr>
</tbody>
</table>

From Table 3.8, it is observed that candidate correction model 1 using Quasi-Newton algorithm has least $E_{avg}$ of 15.12 for 20 hidden neurons.

From Table 3.9, it is observed that candidate correction model 2 using Quasi-Newton algorithm has least validation $E_{avg}$ of 22.38 for 40 hidden neurons.

In Table 5.2, Table 5.3 and Table 5.4, Correction-Based Neural Network (CBNN) is compared with conventional interpolation techniques, ANN, KBNN, SVR, RFR, and QRF using some performance measures.
Table 3.9: Training and validation errors of candidate correction model 2 using Quasi-Newton algorithm

<table>
<thead>
<tr>
<th>No. of Hidden Neurons</th>
<th>Training Error</th>
<th>Validation Errors</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$E_{\text{avg}}$ (%)</td>
<td>$E_{\text{worst}}$ (%)</td>
</tr>
<tr>
<td>10</td>
<td>20.69</td>
<td>20.22</td>
<td>86.03</td>
</tr>
<tr>
<td>20</td>
<td>21.32</td>
<td>23.55</td>
<td>89.25</td>
</tr>
<tr>
<td>30</td>
<td>21.41</td>
<td>21.07</td>
<td>96.56</td>
</tr>
<tr>
<td>40</td>
<td>21.54</td>
<td>20.36</td>
<td>93.15</td>
</tr>
<tr>
<td>50</td>
<td>22.33</td>
<td>22.06</td>
<td>95.20</td>
</tr>
<tr>
<td>60</td>
<td>22.65</td>
<td>22.67</td>
<td>94.03</td>
</tr>
<tr>
<td>70</td>
<td>20.56</td>
<td>21.63</td>
<td>91.18</td>
</tr>
<tr>
<td>80</td>
<td>22.05</td>
<td>20.42</td>
<td>93.33</td>
</tr>
<tr>
<td>90</td>
<td>21.67</td>
<td>23.52</td>
<td>88.06</td>
</tr>
<tr>
<td>100</td>
<td>22.07</td>
<td>22.14</td>
<td>90.55</td>
</tr>
</tbody>
</table>
Figure 3-4: The MLP3 Architecture for SMNN approach
Figure 3-5: The MLP3 network architecture for the candidate correction model with latitude as output
Figure 3-6: The MLP3 network architecture for the candidate correction model with longitude as output
Figure 3-7: Flow Chart for Correction-Based Neural Networks using sensitivity based root finding algorithm
Chapter 4

Support Vector Regression and Random Forest Regression

4.1 Support Vector Regression

4.1.1 Review of Support Vector Machines

The Support Vector Machines (SVMs) are learning machines that are used in a wide variety of applications for classification [42, 44] and regression [43]. SVMs use the structural risk minimization inductive principle to obtain good generalization on a limited number of learning patterns. SVM techniques have the ability to avoid local minima and sparse representation of the solution. To approximate the nonlinear regression function, SVM regression techniques are implemented [52]. The most commonly used versions of SVM regression are ‘\( \epsilon \)-SVR’ and ‘nu-SVR’. \( \epsilon \) and nu are two different parameters in SVR to apply a penalty to optimize points that are not accurately anticipated. In the parameter nu \((0, 1]\), support vectors lie under lower bound and the badly predicted errors lie under upper bound. Hence, nu-SVR has a more meaningful interpretation than \( \epsilon \)-SVR [49], and for this reason the authors only compared nu-SVR. SVR’s has been applied to the geolocation problem to smooth location estimates in a mobile tracking scenario [45]. Recently, SVR’s have been applied
in modeling and estimation of MEMS sensor which showed that nu-SVR performed better than ANN [43].

4.1.2 Methodology

Since, nu-SVR has a more meaningful interpretation than \( \epsilon \)-SVR; nu-SVR is employed in this thesis for the approximation of nonlinear regression function given in Eq. 4.1

\[ f(x) = w^T \Phi(x) + b \]  

(4.1)

In Eq. 4.1, \( w^T \) is the weight vector to the corresponding nonlinear mapping function \( \Phi(x) \) which maps the input space to a higher dimensional space, and \( b \) is a bias. In simple terms, the function \( f(x) \) is the sum of the dot product of \( w^T, \Phi(x) \), and bias. The parameters \( w \) and \( b \) need to be estimated in order to approximate the nonlinear regression function, such that the function \( f(x) \) and the desired output \( y \) should be as close as possible and should be flat to control the problem of over-fitting. To achieve the above objectives, minimization of the problem of nu-SVR given in Eq.4.1 is required.

\[
\frac{1}{2} \| w \|^2 + C \left\{ \gamma \epsilon + \frac{1}{n} \sum_{i=1}^{n} n(\xi + \xi^*) \right\} 
\]

(4.2)

Limitations:

\[
y_i - (w^T \Phi(x)) - b \leq \epsilon + \xi_i^* ,
\]

\[
(w^T \Phi(x)) + b - y_i \leq \epsilon + \xi_i ,
\]

(4.3)

\[ \xi_i^*, \xi_i \leq 0 \]
In Eq.4.2, $\|w\|^2$ represents parameter norm of $f(x)$ for measuring the flatness, $\epsilon$ represents the insensitive loss function for estimating the absolute error between actual and predicted value, $C$ is a normalized parameter that determines the balance between tolerance of error above $\epsilon$ and parameter norm, $\xi_i$ and $\xi_i^*$ are slack variables introduced by Vapnik and Cortes [44]. $\gamma(0 \leq 1)$ has a lower bound and upper bound on the fraction of support vectors and margin errors in the training set respectively. The dual problem of nu-SVR can be solved if $\gamma$ equals to both the fractions, by constructing a Lagrange function ($L$), given as,

$$L(\alpha, \alpha^*, \beta, \eta, \eta^*) = \frac{1}{2} \|w\|^2 + C \left\{ \gamma \cdot \epsilon + \frac{1}{n} \sum_{i=1}^{n} (\xi_i + \xi_i^*) \right\} - \frac{1}{n} \sum_{i=1}^{n} (\eta_i \xi_i + \eta_i^* \xi_i^*)$$

$$-\frac{1}{n} \sum_{i=1}^{n} (\epsilon + \xi_i + y_i - w^t \Phi(x) - b)$$

$$-\frac{1}{n} \sum_{i=1}^{n} (\epsilon + \xi_i - y_i + w^t \Phi(x) + b) - \beta \cdot \epsilon$$

where $\alpha, \alpha^*, \eta, \eta^*, \beta$ are the Lagrange multipliers. Thus, optimizing the dual problem of nu-SVR using partial derivatives of $L$ with respect to variables such as $w, \epsilon, b, \xi_i$ yields to

$$\text{maximum} - \frac{1}{2} \sum_{i,j=1}^{n} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)K(x_i, x_j) + \sum_{i=1}^{n} y_i(\alpha_i - \alpha_i^*)$$

(4.5)

Limitations:

$$\sum_{i,j=1}^{n} (\alpha_i - \alpha_i^*) = 0,$$

$$\sum_{i,j=1}^{n} (\alpha_i + \alpha_i^*) \leq C_\gamma,$$  (4.6)
where \( k(x_i, x_j) \) is a kernel function given as \( k(x_i, x_j) = \Phi(x_i)t.\Phi(x_j) \). Indeed, optimization of the dual problem of nu-SVR yields to Lagrange multipliers and weight parameter \( w \) as

\[
w = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*)\Phi(x_i)
\] (4.7)

Substituting \( w \) in Eq. 4.1 gives the approximation of the prediction function \( f(x) \) as:

\[
f(x) = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*)K(x_i, x_j) + b
\] (4.8)

In Eq. 4.8, \( \alpha \) and \( \alpha^* \) are Lagrange multipliers, \( K(x_i, x_j) \) is a kernel function, and \( b \) is a bias parameter. The selection of kernel function plays a vital role in SVM in increasing the prediction accuracy of the model. The frequently used kernel functions in SVMs are Polynomial Function (PF), Radial Basis Function (RBF), Sigmoid Function (SF) and Linear Function (LF). In general, RBF kernel function is the reasonable first choice as it has better prediction accuracy and a relatively less complex model for implementation [48]. Bias parameter is identified using Kuhn, Tucker and Karush conditions as given in [46, 47]. Using nu-SVR and \( \epsilon \)-SVR approaches, the desired output can be predicted by identifying the Lagrange multipliers \( \alpha, \alpha^*, b \) and selecting an appropriate kernel function, the approximated function in Eq. 4.8.

### 4.1.3 Results and Discussions

nu-SVR technique has been implemented using LibSVM software [50]. Kernel RBF function has been chosen as it produces accurate results and also easy to implement. By changing the value of gamma, kernel function can control the shape of the separating hyper plane. In general, the gamma value is directly proportional to the
number of support vectors. Hence, prediction accuracy increases with the increase in the gamma value. Table 4.1 represents the validation error of Support Vector Regression with varying gamma value.

Table 4.1: Validation error of Support Vector Regression with varying gamma value

<table>
<thead>
<tr>
<th>Gamma Value</th>
<th>(E_{\text{avg}})</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>4.424</td>
</tr>
<tr>
<td>0.1</td>
<td>4.235</td>
</tr>
<tr>
<td>1</td>
<td>3.725</td>
</tr>
<tr>
<td>10</td>
<td>3.842</td>
</tr>
</tbody>
</table>

From Table 4.1, it can be observed that as the gamma value increases the prediction accuracy increases and the validation error decreases in SVR. The best SVR model using kernel function is for gamma value as 1 and is compared with other interpolation techniques as shown in Table 5.2, Table 5.3 and Table 5.4.

4.2 Random Forest Regression

4.2.1 Review of Random Forest Regression

Random Forest Regression (RFR), proposed by Breiman in 2001, is an improved non-parametric regression approach known for its robustness and flexibility in modeling the input-output functional relationship appropriately [51]. In RFR, individual regression trees are constructed using bootstrap samples (samples obtained from random re-sampling of data) from training data. An ensemble of these individual trees is called a Random Forest. When samples are selected at random, some of the training data may not be used, and some may be repeated in the sample. These unused samples constitute the out-of-bag samples, which are used to estimate the learning error. Recently, RFR is being applied in various research fields like Global Positioning System (GPS) and Inertial Navigation System (INS) [51], and speech recognition.
4.2.2 Methodology

The Mean Square Error (learning error), which determines how efficient the Random Forest prediction would be after the validation is given as

$$MSE \approx MSE^{OOB} = n^{-1} \sum_{i=1}^{n} [\hat{y}(x_i) - y_i]^2$$  \hspace{1cm} (4.9)

In Eq. 4.9, $\hat{y}(x_i)$ is the predicted output corresponding to a given input sample whereas $y_i$, is the actual output and $n$ represents the total number of out of bag samples.

Due to RFR’s built-in cross-validation capability assisted by out-of-bag samples, this model provides prediction error estimates during the process of training and improves the generalization capability of the Random Forests.

Given an input-output dataset $\{(X_1, Y_1), \ldots, (X_n, Y_n)\}$, where $X_i, i = 1, \ldots, n$ is an input vector containing latitude and longitude and $Y_i$ is the output vector containing radon concentration. The procedure for training the given dataset is as follows:

1. A bootstrap sample is collected from the available dataset.
2. Grow a tree using the bootstrap sample to maximum size (i.e., until the point where no further splits are possible) without pruning.
3. Repeat Step 2 until a pre-specified number of trees are grown.

Let us suppose the above procedure results in a set of $M$ trees $\{R_1(X), R_2(X), \ldots, R_M(X)\}$, where $X = \{x_1, x_2, \ldots, x_p\}$ is a $p$-dimension input vector that forms a forest. In this case $X$ is a 2-dimensional vector containing latitude and longitude. The ensemble produces $M$ outputs corresponding to each tree $\hat{y}_1 = R_1(X), \ldots, \hat{y}_m = R_m(X)$, where $(\hat{y}_m, m = 1, \ldots, M)$, is the $m^{th}$ tree output [52].
Random Forest trees predict the output by estimating the conditional mean of all the data points that fall into a node. The conditional mean can be approximated by minimization of a squared error loss function given in Eq. 4.10.

\[
E(Y|X = x) = \arg \min E\{(Y - \mu)^2|X = x}\]  

(4.10)

where \(X\) is a real value predictor, \(Y\) is a response variable, and \(\mu\) is the estimate of the response variable. \(E(Y|X = x)\) is approximated over the observations of \(Y\) [53].

Figure 4-1: Work Flow of Random Forests

Figure 4-1 shows the work flow of Random Forests where the input training samples \(X_1, X_2, \ldots, X_n\) are used to grow the user-defined number of trees. The final
output \( Y_1, Y_2, \ldots, Y_n \) can be estimated by passing the testing samples along the
trees and average of all the user defined number of trees.

RFRs are powerful for high-dimensional regression and classification. The RFR
model performs better than the conventional interpolation techniques and ANN ap-
proach, as it aims to minimize the error function by evaluating the conditional mean
of the distribution. The conditional mean illuminates just one aspect of the con-
ditional distribution of the response variable and neglects all the other aspects of
possible interest [53].

### 4.2.3 Results and Discussions

The model in random forest regression is trained and validated by varying the
number of trees using an integrated development environment (IDE) software named
R [54]. In this work, RFR is implemented using ‘randomForest’ package. Table
4.2 represents the validation error criterion \( E_{avg} \) by varying number of trees. The
equation for error criterion \( E_{avg} \) is the same as given in Eq. 3.7.

<table>
<thead>
<tr>
<th>Number of Trees</th>
<th>( E_{avg} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>3.95</td>
</tr>
<tr>
<td>1000</td>
<td>3.94</td>
</tr>
<tr>
<td>1500</td>
<td>3.95</td>
</tr>
<tr>
<td>2000</td>
<td>3.94</td>
</tr>
</tbody>
</table>

From Table 4.2, it can be observed that, even with varying number of trees, the
performance of RFR is not improved. RFR has the least validation error \( E_{avg} \) of 3.94
which is comparatively higher than nu-SVR approach.

In Table 5.2, Table 5.3 and Table 5.4, Random Forest Regression (RFR) is com-
pared with conventional interpolation techniques, ANN, KBNN, CBNN, SVR, and
QRF using some performance measures.
Chapter 5

Quantile Regression Forest

5.1 Overview of Quantile Regression Forest

Quantile Regression Forest (QRF) is a generalization of Random Forest, proposed by Meinshausen (2006) [56]. Quantile Regression Forests give a non-parametric and precise way of approximating conditional quantiles for high-dimensional predictor variables, unlike RFRs, which approximate the conditional mean. Random forests have only the average of the observations that drop into each node in each tree and neglect all other data. But, quantile regression forests keep the value of all observations in each node in each tree, not only their mean, and also estimate the conditional distribution [53, 55]. There may be no predictive relationship between the measured predictive factors (X) and the mean of the response variable (Y) distribution. But there may be strong, useful predictive relationship with the other parts of the response variable distribution [57]. QRF holds all the predictive relationships with the response variable distribution by accurately approximating its conditional quantiles.

5.2 Methodology

Random Forest trees predict the output by estimating the conditional mean. The conditional mean can be approximated by minimization of a squared error loss func-
tion. Let the loss function be $L_{\alpha}$ given in Eq. 5.1.

$$L_{\alpha}(Y, q) = \begin{cases} \alpha |Y - q| & Y > q \\ (1 - \alpha)|Y - q| & Y \leq q \end{cases}$$  \hspace{1cm} (5.1)$$

where $Y$ and $q$ are actual and predicted response variables respectively and $0 < \alpha < 1$. Conditional quantiles minimizes the Expected Loss $E(L_{\alpha})$ as given in Eq. 5.2.

$$Q_{\alpha}(x) = \arg \min_{q} \ E\{L_{\alpha}(Y, q) | X = x\}$$  \hspace{1cm} (5.2)$$

where, $Q_{\alpha}$ is the $\alpha$- quantile.

The conditional mean of $Y$, given $X = x$, is approximated in Eq. 4.10, and the conditional distribution function of $Y$ given $X = x$, is given by

$$F(y | X = x) = P(Y \leq y | X = x) = E(1_{\{Y \leq y\}} | X = x)$$  \hspace{1cm} (5.3)$$

$E(1_{\{Y \leq y\}} | X = x)$ is approximated by the weighted mean over the observations of $1_{\{Y \leq y\}}$. The final estimate of the distribution function $\hat{F}(y | X = x)$ is given by

$$\hat{F}(y | X = x) = \sum_{i=1}^{n} w_i(x)1_{\{Y_i \leq y\}}$$  \hspace{1cm} (5.4)$$

where $w_i(x)$, is the average of weight vector.

Algorithm 1: QRF working procedure in order to estimate the distribution function $\hat{F}(y | X = x)$

Step 1: Collect the required data and identify the inputs (latitude and longitude) and outputs (Radon concentration).

Step 2: Grow the user defined number of trees such that each node of each tree has a note of all observations.

Step 3: For given $X = x$, in all the trees, compute weight $w_i(x)$ for every observation by taking the average of weight vector.
Step 4: Using weights in step 3, compute the estimate of the conditional distribution function as in Eq. 5.4 for all \( y \in R \).

5.2.1 Results and Discussions

The model in quantile regression forest is trained and validated by varying the number of trees using R software [54]. In this work, QRF is implemented using ‘quantregForest’ package. Table 5.1 represents the validation error criterion \( E_{\text{avg}} \) by varying number of trees.

Table 5.1: Performance of QRF Based on Validation Error \( (E_{\text{avg}}) \) by Varying the Number of Trees

<table>
<thead>
<tr>
<th>Number of Trees</th>
<th>( E_{\text{avg}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>1.94</td>
</tr>
<tr>
<td>1000</td>
<td>1.95</td>
</tr>
<tr>
<td>1500</td>
<td>1.94</td>
</tr>
<tr>
<td>2000</td>
<td>1.94</td>
</tr>
</tbody>
</table>

From Table 5.1, it can be observed that, even with varying number of trees, the performance of QRF is not varied. QRF has the least validation error \( E_{\text{avg}} \) of 1.94 which is the better approach comparing with all the above mentioned techniques.

In Table 5.2, Table 5.3 and Table 5.4, Random Forest Regression (RFR) is compared with conventional interpolation techniques, ANN, KBNN, CBNN, SVR, and QRF using some performance measures.

5.3 Performance Measures

The performance measures are used to estimate the difference between the predicted values \( p(k) \) and the actual values \( a(k) \) of an attribute. Recently, Kadiyala and Kumar [58] have given some recommendations for the better performance of
air-quality models. Some of the statistical performance measures are: Correlation (CORR), revised Index of Agreement (IOA_r), factor of two of the observations (FA_2), Fractional Bias (FB), Fractional Variance (FV), Model Comparison Measure (MC_M_2), geometric Mean Bias (MG), Mean Absolute Error (MAE), Normalized Mean Square Error (NMSE), Root Mean Square Error (RMSE), geometric mean variance (VG), Correlation Coefficient (ρ), and Accuracy for paired peak (A_p). Besides these performance measures, E_{avg} is the commonly used validation error to measure the performance of an interpolation technique.

5.3.1 Mean Absolute Error

The MAE is used to measure the average magnitude of errors in a set of predictions and is given as

\[
MAE = \frac{1}{N_v} \sum_{k=1}^{N_v} |a(k) - p(k)|
\]  

(5.5)

In Eq. 5.5, \( p(k) \) is the predicted value from the model for every \( k^{th} \) sample in validation set, \( a(k) \) is the actual or observed value of an attribute and \( N_v \) is the total number of samples. The ideal value of MAE is zero.

5.3.2 Factor of Two

The \( Fa_2 \) defines the percentage of predicted value over the actual value and is given as

\[
Fa_2 = \frac{a(k)}{p(k)}
\]  

(5.6)

The ratio of predicted value to actual value lies between 0.5 and 2.0. The ideal value of \( Fa_2 \) is one
5.3.3 Root Mean Square Error

The RMSE is used to calculate the average magnitude of error. The ideal value of RMSE is zero. RMSE is given as

\[
RMSE = \sqrt{\frac{\sum_{k=1}^{N_v} (a(k) - p(k))^2}{N_v}}
\]

(5.7)

5.3.4 Fractional Bias

The fractional bias or normalized bias is used to calculate the fraction of mean concentrations. The fractional bias lies between -2 and +2. The ideal value of FB is zero. FB is given as

\[
FB = \frac{\sum_{k=1}^{N_v} (a(k) - p(k))}{\frac{1}{2} \sum_{k=1}^{N_v} (a(k) + p(k))}
\]

(5.8)

5.3.5 Normalized Mean Square Error

The NMSE is used to assess the scatter in the complete data set and is given as

\[
NMSE = \frac{\sum_{k=1}^{N_v} (a(k) - p(k))^2}{\frac{1}{N_v} \sum_{k=1}^{N_v} (a(k) \ast p(k))}
\]

(5.9)

In general, NMSE values are inversely related to the performance of a model. The ideal value of NMSE is zero.

5.3.6 Fractional variance (FV)

The fractional variance (FV) is a normalization of the mean bias of the variances of the actual and predicted values. FV can be calculated using Eq. 5.10. The ideal
value of $FV$ is zero.

$$FV = 2 \left[ \frac{\sigma^2_a - \sigma^2_p}{\sigma^2_a + \sigma^2_p} \right]$$ (5.10)

### 5.3.7 Revised IOA ($IOA_r$)

A revised version of IOA ($IOA_r$) is a modification to the modified index of agreement (IOA). $IOA_r$ indicates the sum of the magnitudes of the differences between the model-predicted and observed deviations about the observed mean relative to the sum of the magnitudes of the perfect model and observed deviations about the observed mean. $IOA_r$ ranges from -1 to +1, and can be computed using Eq. 5.11 [Willmott et al. (2011)].

$$IOA_r = \begin{cases} 
  1 - \frac{\sum |p(k) - a(k)|}{c \sum |a(k) - M_0|} \\
  & \text{when } \sum |p(k) - a(k)| \leq c \sum |a(k) - M_0| \\
  \frac{c \sum |a(k) - M_0|}{\sum |p(k) - a(k)|} - 1 \\
  & \text{when } \sum |p(k) - a(k)| > c \sum |a(k) - M_0| \\
  \text{with } c = 2
\end{cases} \quad (5.11)$$

### 5.3.8 Geometric mean variance (VG)

Geometric mean variance (VG) is the exponential function of mean of square of the difference between logarithmic function of predicted values and the logarithmic function of actual values and can be computed using Eq. 5.12 and ideal value of VG is one [Chang and Hanna (2004)].

$$VG = exp[(\ln a(k) - \ln p(k))^2]$$ (5.12)
5.3.9 Accuracy for paired peak ($A_p$)

Accuracy measures for paired ($A_p$) peaks serve as complimentary tests of model performance and can be mathematically computed using Eq. 5.13 [McNally and Tesche (1995)].

$$A_p = \frac{p(\hat{k}) - a(\hat{k})}{a(\hat{k})} \times 100\% \quad (5.13)$$

5.3.10 Geometric mean bias (MG)

Geometric mean bias (MG) is the exponential function of the difference between means of logarithmic function of predicted values and the logarithmic function of actual value and is given in Eq. 5.14. The ideal value of MG is one [Chang and Hanna (2004)].

$$MG = exp\left[\left(\ln a(k) - \ln p(k)\right)\right] \quad (5.14)$$

5.3.11 Coefficient of correlation (r)

Coefficient of correlation (r) can be described as the relative change with respect to the means of two quantities, but cannot distinguish the magnitude of covariance. The ideal value of r is 1. r can be mathematically computed using Eq. 5.15.

$$r = \frac{(a(k) - \bar{a}(k)) \times (p(k) - \bar{p}(k))}{\sigma_p(k) \times \sigma_a(k)} \quad (5.15)$$

5.3.12 Spearman correlation coefficient ($\rho$)

Spearman correlation coefficient ($\rho$) is defined as the correlation coefficient (r) between the ranked variables. Spearman correlation coefficient can be mathematically computed using Eq. 5.16.
\[ \rho = \frac{\sum_i (a(k) - \bar{a}(k)) \ast (p(k) - \bar{p}(k))}{\sqrt{\sum_i (a(k) - \bar{a}(k))^2 \ast \sum_i (p(k) - \bar{p}(k))^2}} \]  

(5.16)

5.4 Results and Discussion

This thesis evaluates the interpolation techniques used to estimate radon levels in Ohio zip codes, namely conventional ANN approach, KBNN approaches, correction model-based ANN approach, Support Vector Regression, Random Forest Regression, and Quantile Regression forest based on the statistical performance measures compiled by Kadiyala and Kumar [58]. Some of the statistical performance measures are: Correlation (CORR), revised Index of Agreement (IOAr), factor of two of the observations (FA2), Fractional Bias (FB), Fractional Variance (FV), Model Comparison Measure (MCM2), geometric Mean Bias (MG), Mean Absolute Error (MAE), Normalized Mean Square Error (NMSE), Root Mean Square Error (RMSE), geometric mean variance (VG), Correlation Coefficient (\(\rho\)), and Accuracy for paired peak (\(A_p\)).

The models in this thesis are trained and validated using the same data sets for both training and validating in order to compare all the techniques without any bias. The authors considered latitude and longitude as inputs and radon concentration as the output.

In the Conventional Neural Network approach, the models are trained and validated with the quasi-newton algorithm by varying the number of hidden neurons. The best ANN model has the least amount of validation error \(E_{avg}\) of 3.98 for 40 hidden neurons.

In SDM modeling, the coarse model is trained and validated using quasi-newton algorithm by changing the number of hidden neurons in the hidden layer. The best coarse model is observed for 80 neurons. The difference model is trained and validated by changing the number of hidden neurons, and the best difference model has the least \(E_{avg}\) at 30 hidden neurons. The sum of the predicted values of the best coarse model
and the best difference model gives the SDM model. Hence, by summing the two models, the resulting validation error $E_{avg}$ for SDM is 4.09.

In SMNN modeling, the Space Map model and the coarse model are trained and validated with varying numbers of hidden neurons. The best SM model is observed for 40 hidden neurons. Hence, by obtaining the best SM model, the coarse model is trained and validated. The SMNN has the least $E_{avg}$ of 3.89 for 40 hidden neurons.

The correction-based ANN approach has been implemented using the root finding algorithm has been implemented such that $E_{obj} < E_{user}$ is achieved. In the algorithm, $E_{obj}$ is calculated for every sample. After executing the root finding algorithm, the resulting $E_{avg}$ for correction-based ANN model is 4.33.

SVR technique has been implemented by taking gamma value as 1 and by using kernel RBF function. The validation error of SVR has the least $E_{avg}$ of 3.72.

The RFR is implemented using the ‘randomForest’ package whereas the QRF is implemented using ‘quantRegForest’. In this article, the authors constructed 500 trees for both the Random Forest Regression and Quantile Regression Forest. The validation error for RFR has the least $E_{avg}$ of 3.94. The validation error for QRF has the least $E_{avg}$ of 1.94.

Based on the recommendations made by Kadiyala and Kumar [58], the interpolation technique is considered to be ideal if it satisfies the performance measuring criteria. MATLAB software [59] is used to compute each model’s performance using the following: MAE, $F_a$, RMSE, FB, NMSE, CORR, FV, IOA, $\rho$, MG, VG, and Ap. Each of the performance measures have a range of acceptable values. The ideal value for MAE, RMSE, FB, and NMSE is zero, whereas for CORR, FV, IOA, MG, and VG, the ideal value is 1. The mathematical expressions of these performance measures are as in [19]. An ideal interpolation technique has all of its performance measures within the acceptable range and closer to ideal values. However, it is unusual to find a technique that meets all the performance measurement criteria. From
Table 5.2, Table 5.3, and Table 5.4, it can be observed that QRF is much closer to the ideal values of Validation Error, MAE, $Fa_2$, FB, FV, IOA, MG, and VG, compared to the other interpolation techniques. PKI approach is closer to the ideal values of RMSE, NMSE, CORR, and when compared with other interpolation techniques. SVR meets the ideal value of $A_p$, while the Radial Basis Function and Kriging has the least training error.

Considering the relativity of predicted values to the actual values of each performance measure, one can infer that the QRF proposed in the thesis is the best technique for modeling and estimating the radon concentrations.
Table 5.2: Comparison of all interpolation techniques based on the training error, validation error and some operational performance measures

<table>
<thead>
<tr>
<th>Techniques for interpolating radon concentration</th>
<th>Training Error (IV=0)</th>
<th>Validation Error (IV=0)</th>
<th>MAE (IV=0)</th>
<th>$F_{a2}$ (IV=1)</th>
<th>RSME (IV=0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kriging</td>
<td>2.13</td>
<td>3.89</td>
<td>1.516</td>
<td>0.789</td>
<td>2.616</td>
</tr>
<tr>
<td>Local Polynomial Interpolation (LPI)</td>
<td>4.05</td>
<td>3.77</td>
<td>1.466</td>
<td>0.785</td>
<td>2.361</td>
</tr>
<tr>
<td>Global Polynomial Interpolation (GPI)</td>
<td>5.15</td>
<td>4.70</td>
<td>1.828</td>
<td>0.682</td>
<td>2.627</td>
</tr>
<tr>
<td>Radial Basis Function (RBF)</td>
<td>0.39</td>
<td>3.79</td>
<td>1.477</td>
<td>0.793</td>
<td>2.576</td>
</tr>
<tr>
<td>Artificial Neural Networks (ANN)</td>
<td>4.54</td>
<td>3.98</td>
<td>1.533</td>
<td>0.742</td>
<td>2.414</td>
</tr>
<tr>
<td>Prior Knowledge Input (PKI)</td>
<td>3.98</td>
<td>3.84</td>
<td>0.783</td>
<td>0.912</td>
<td>1.168</td>
</tr>
<tr>
<td>Source Difference Method (SDM)</td>
<td>4.35</td>
<td>4.09</td>
<td>1.593</td>
<td>0.746</td>
<td>2.466</td>
</tr>
<tr>
<td>Space-Mapped Neural Networks (SMNN)</td>
<td>3.65</td>
<td>3.89</td>
<td>1.515</td>
<td>0.769</td>
<td>2.338</td>
</tr>
<tr>
<td>Correction-Based ANN model (CANN)</td>
<td>4.26</td>
<td>4.33</td>
<td>1.687</td>
<td>0.734</td>
<td>2.496</td>
</tr>
<tr>
<td>Support Vector regression (SVR)</td>
<td>4.19</td>
<td>3.72</td>
<td>1.447</td>
<td>0.769</td>
<td>2.479</td>
</tr>
<tr>
<td>Random Forest Regression (RFR)</td>
<td>2.37</td>
<td>3.94</td>
<td>1.532</td>
<td>0.785</td>
<td>2.497</td>
</tr>
<tr>
<td>Quantile Regression Forest (QRF)</td>
<td>1.35</td>
<td>1.94</td>
<td>0.758</td>
<td>0.960</td>
<td>1.577</td>
</tr>
</tbody>
</table>
Table 5.3: Comparison of all interpolation techniques based on some operational performance measures

<table>
<thead>
<tr>
<th>Techniques for interpolating radon concentration</th>
<th>FB (IV=0)</th>
<th>NMSE (IV=0)</th>
<th>CORR (IV=1)</th>
<th>FV (IV=0.5)</th>
<th>IOA (IV=1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kriging</td>
<td>6.69e-04</td>
<td>0.586</td>
<td>0.576</td>
<td>0.784</td>
<td>0.586</td>
</tr>
<tr>
<td>Local Polynomial Interpolation (LPI)</td>
<td>7.66e-04</td>
<td>0.480</td>
<td>0.657</td>
<td>1.371</td>
<td>0.6</td>
</tr>
<tr>
<td>Global Polynomial Interpolation (GPI)</td>
<td>6.81e-04</td>
<td>0.580</td>
<td>0.592</td>
<td>1.951</td>
<td>0.501</td>
</tr>
<tr>
<td>Radial Basis Function (RBF)</td>
<td>6.42e-04</td>
<td>0.559</td>
<td>0.558</td>
<td>0.836</td>
<td>0.5972</td>
</tr>
<tr>
<td>Artificial Neural Networks (ANN)</td>
<td>7.24e-04</td>
<td>0.495</td>
<td>0.626</td>
<td>1.162</td>
<td>0.582</td>
</tr>
<tr>
<td>Prior Knowledge Input (PKI)</td>
<td>0.001</td>
<td>0.118</td>
<td>0.919</td>
<td>0.215</td>
<td>0.786</td>
</tr>
<tr>
<td>Source Difference Method (SDM)</td>
<td>7.06e-04</td>
<td>0.509</td>
<td>0.615</td>
<td>1.149</td>
<td>0.565</td>
</tr>
<tr>
<td>Space-Mapped Neural Networks (SMNN)</td>
<td>7.41e-04</td>
<td>0.464</td>
<td>0.641</td>
<td>1.284</td>
<td>0.586</td>
</tr>
<tr>
<td>Correction-Based ANN model (CANN)</td>
<td>6.59e-04</td>
<td>0.488</td>
<td>0.594</td>
<td>1.199</td>
<td>0.540</td>
</tr>
<tr>
<td>Support Vector regression (SVR)</td>
<td>8.14e-04</td>
<td>0.631</td>
<td>0.640</td>
<td>1.631</td>
<td>0.605</td>
</tr>
<tr>
<td>Random Forest Regression (RFR)</td>
<td>6.67e-04</td>
<td>0.514</td>
<td>0.585</td>
<td>0.937</td>
<td>0.582</td>
</tr>
<tr>
<td>Quantile Regression Forest (QRF)</td>
<td>9.97e-04</td>
<td>0.234</td>
<td>0.818</td>
<td>0.617</td>
<td>0.793</td>
</tr>
</tbody>
</table>
Table 5.4: Comparison of all interpolation techniques based on some operational performance measures

<table>
<thead>
<tr>
<th>Techniques for interpolating radon concentration</th>
<th>$\rho$ (IV=1)</th>
<th>MG (IV=1)</th>
<th>VG (IV=1)</th>
<th>$A_p$ (-15 &lt; IV &lt; 15)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kriging</td>
<td>0.682</td>
<td>0.864</td>
<td>1.572</td>
<td>1.260</td>
</tr>
<tr>
<td>Local Polynomial Interpolation (LPI)</td>
<td>0.675</td>
<td>0.820</td>
<td>1.554</td>
<td>0.704</td>
</tr>
<tr>
<td>Global Polynomial Interpolation (GPI)</td>
<td>0.595</td>
<td>0.752</td>
<td>1.846</td>
<td>3.283</td>
</tr>
<tr>
<td>Radial Basis Function (RBF)</td>
<td>0.560</td>
<td>0.837</td>
<td>1.533</td>
<td>2.972</td>
</tr>
<tr>
<td>Artificial Neural Networks (ANN)</td>
<td>0.628</td>
<td>0.839</td>
<td>1.661</td>
<td>2.058</td>
</tr>
<tr>
<td>Prior Knowledge Input (PKI)</td>
<td>0.922</td>
<td>0.959</td>
<td>0.999</td>
<td>0.055</td>
</tr>
<tr>
<td>Source Difference Method (SDM)</td>
<td>0.617</td>
<td>0.813</td>
<td>1.043</td>
<td>3.566</td>
</tr>
<tr>
<td>Space-Mapped Neural Networks (SMNN)</td>
<td>0.643</td>
<td>0.813</td>
<td>1.043</td>
<td>2.102</td>
</tr>
<tr>
<td>Correction-Based ANN model (CANN)</td>
<td>0.596</td>
<td>0.750</td>
<td>1.085</td>
<td>10.68</td>
</tr>
<tr>
<td>Support Vector regression (SVR)</td>
<td>0.642</td>
<td>0.964</td>
<td>1.001</td>
<td>-15.54</td>
</tr>
<tr>
<td>Random Forest Regression (RFR)</td>
<td>0.588</td>
<td>0.813</td>
<td>1.043</td>
<td>5.170</td>
</tr>
<tr>
<td>Quantile Regression Forest (QRF)</td>
<td>0.818</td>
<td>1.010</td>
<td>1.000</td>
<td>-7.970</td>
</tr>
</tbody>
</table>
Chapter 6

Conclusions and Future Work

6.1 Conclusion

This thesis presents the interpolation techniques used to estimate radon levels in Ohio zip codes, namely conventional ANN approach, KBNN approaches, correction model-based ANN approach, Support Vector Regression, Random Forest Regression, and Quantile Regression forest and are compared based on some statistical performance measures. From Table 5.2, Table 5.3 and Table 5.4, it can be observed that QRF is much closer to the ideal values of Validation Error, MAE, $F_{a2}$, FB, FV, IOA, MG, and VG, compared to the other interpolation techniques. PKI approach is closer to the ideal values of RMSE, NMSE, CORR, and $\rho$ when compared with other interpolation techniques. SVR meets the ideal value of $A_p$, while the Radial Basis Function and Kriging has the least training error. The results in this thesis conclude that QRF is a better regression technique when compared to other interpolation techniques in estimating the radon concentration. The performance measures of QRF are acceptable from extreme-end and mid-range modeling perspectives and are better than the other interpolation techniques. The concept of approximating the conditional quantiles allows QRF to perform better than the existing techniques, even for limited training data.
6.2 Suggestions for Future Work

Although the results presented in this thesis proved the effectiveness of the QRF technique, further research can be done for the selection of best quantile. In RFR, the absence of secondary variables effects the performance of the model, hence finding the secondary variables could improve the performance of the model. Besides, there are different interpolation techniques such as deep neural networks, Grey Wolf Optimization and Particle Swarm Optimization with parallel processing could solve the problem of local minimization in the Correction-Based ANN.
References


Appendix A

Source Code

A.1 Source code for Correction-Based ANN modeling approach

clc;
clear all;
close all;
n = 1;
p = 0;
q=3;
train=input('enter the number of training samples');
test=input('enter the number of testing samples');
total=test+train;
h=input('enter the number of hidden neurons');
for k=1:test
    file1 = fopen('data.dat');
    input = fscanf(file1, '%g',train);
    input1 = input(n); % Latitude
    input2 = input(n+1); % Longitude
outputp = input(n+2); % Radon
inputv = [input1 input2];
fclose(file1);
%Initial radon predictions
x=fopen('plot.dat');
y=fscanf(x, '%g',total);
yout=y(q);
file2 = fopen('trad.txt','a');
fprintf(file2, '%f
',yout);
fclose(file2);
input1c = yout; % Radon
input2c = input2; % Longitude
inputc = [input1c input2c];
outputpc = model(inputc);
file3 = fopen('mod.txt','a');
fprintf(file3, '%f
',outputpc);
fclose(file3);
% Difference between original and predicted latitude values
Diff = outputpc-input1;
file4 = fopen('error.txt','a');
fprintf(file4, '%f
',Diff);
fclose(file4);

%input scaling
x(1)=-1.0+(2.0)*(input(1)-(0.1)) / ((39.0) - (0.1));
x(2)=-1.0+(2.0)*(input(2)-(-84.79165)) / ((-80.53175) - (-84.79165));
% calculating hidden neurons
z(1) = 1.0 / ( 1.0 + exp(-1.0 * (1.65275+x(1))*(-9.16317) + x(2)*(1.16345)));
z(2) = 1.0 / ( 1.0 + exp(-1.0 * (-4.61926+x(1)*(6.28396) + x(2)*(12.1244))));
z(3) = 1.0 / ( 1.0 + exp(-1.0 * (-2.91127+x(1)*(-8.17225) + x(2)*(5.96476))));
z(4) = 1.0 / ( 1.0 + exp(-1.0 * (3.29503+x(1)*(4.0502) + x(2)*(-0.386808))));
z(5) = 1.0 / ( 1.0 + exp(-1.0 * (15.683+x(1)*(13.3762) + x(2)*(2.37671))));
z(6) = 1.0 / ( 1.0 + exp(-1.0 * (0.955023+x(1)*(2.37426) + x(2)*(5.53347))));
z(7) = 1.0 / ( 1.0 + exp(-1.0 * (3.93092+x(1)*(6.8071) + x(2)*(7.82879))));
z(8) = 1.0 / ( 1.0 + exp(-1.0 * (-9.71716+x(1)*(-5.9505) + x(2)*(-13.3094))));
z(9) = 1.0 / ( 1.0 + exp(-1.0 * (5.67257+x(1)*(1.74286) + x(2)*(-3.30721))));
z(10) = 1.0 / ( 1.0 + exp(-1.0 * (2.66828+x(1)*(3.73156) + x(2)*(9.34202))));
z(11) = 1.0 / ( 1.0 + exp(-1.0 * (-2.08968+x(1)*(-6.20615) + x(2)*(5.15899))));
z(12) = 1.0 / ( 1.0 + exp(-1.0 * (4.50482+x(1)*(5.51839) + x(2)*(7.17969))));
z(13) = 1.0 / ( 1.0 + exp(-1.0 * (0.0636359+x(1)*(2.27516) + x(2)*(7.54973))));
\[
z(14) = \frac{1.0}{1.0 + \exp(-1.0 \times (-5.3709 + x(1) \times (-2.9655) + x(2) \times (-9.32033)))};
\]
\[
z(15) = \frac{1.0}{1.0 + \exp(-1.0 \times (22.2943 + x(1) \times (21.176) + x(2) \times (3.09422)))};
\]
\[
z(16) = \frac{1.0}{1.0 + \exp(-1.0 \times (6.16266 + x(1) \times (-1.63508) + x(2) \times (0.907263)))};
\]
\[
z(17) = \frac{1.0}{1.0 + \exp(-1.0 \times (-3.83843 + x(1) \times (5.94363) + x(2) \times (-3.23456)))};
\]
\[
z(18) = \frac{1.0}{1.0 + \exp(-1.0 \times (1.04308 + x(1) \times (0.402742) + x(2) \times (-1.15482)))};
\]
\[
z(19) = \frac{1.0}{1.0 + \exp(-1.0 \times (-3.38877 + x(1) \times (4.74998) + x(2) \times (-0.225763)))};
\]
\[
z(20) = \frac{1.0}{1.0 + \exp(-1.0 \times (-4.18563 + x(1) \times (-6.45004) + x(2) \times (-6.3824)))};
\]

\% calculating output neurons
\[
y(1) = 0.76738 + z(1) \times (-2.0645) + z(2) \times (-0.596627) + z(3) \times (-7.45756) + z(4) \times (1.71317) + z(5) \times (-10.923) + z(6) \times (4.47803) + z(7) \times (6.21772) + z(8) \times (-2.21119) + z(9) \times (-5.19056) + z(10) \times (-4.5298) + z(11) \times (10.2907) + z(12) \times (8.12513) + z(13) \times (-2.40936) + z(14) \times (5.28407) + z(15) \times (4.24325) + z(16) \times (-3.2887) + z(17) \times (-3.39669) + z(18) \times (2.26929) + z(19) \times (1.84739) + z(20) \times (10.2539);
\]

\% output scaling
\[
output(1) = 38.448563 + (y(1) - (0.0)) \times ((41.934916 - (38.448563))/((1.0) - (0.0)));
\]
bw = [2.31087 -1.39897 8.59608 -1.57637 -8.03895 -3.49334 2.7054
-2.72137 2.03351 1.06151 3.00522 0.85773 7.69355 1.9282 2.45342
-7.74941 -0.6754 -2.58227 -0.376907 8.10244];

dw = [-2.0645 -0.596627 -7.45756 1.71317 -10.923 4.47803 6.21772
-3.2887 -3.39669 2.26929 1.84739 10.2539];

ew = 0.76738;
sensitivity_in = 0.0;

% Calculating sensitivity based on chain rule
for i = 1:h
sensitivity_in = sensitivity_in + dw(i)*z(i)*(1-z(i))*bw(i);
end
sensitivity_in2 = sensitivity_in * (((39.0) - (0.1))/2.0);
sensitivity_final = sensitivity_in2 * ((41.934916) - (38.438617));
S = sensitivity_final;
alpha = 0.1;
Euser = 0.1; iteration = 0; yout1 = yout;

% Defining error
Eobj = ((input1-outputpc)/input1)*100;
while(abs(Eobj)>Euser)
delta_out = abs(-Diff/S);
% Updating the Radon value using sensitivity value
if(yout<outputp)
yout = yout1 + (alpha * delta_out);
elseif(yout>outputp)
yout = yout1 - (alpha * delta_out);
end
yout1 = yout;
if(yout>outputp)
break;
end
input = [yout input2];
outputpc2 = model(input);
Eobj = ((input1-outputpc2)/input1)*100;
R = outputpc2 -input1;
iteration = iteration + 1;
if (iteration > 1000)
p= p+1;
break;
end
end
end
yout;
file5 = fopen('pred.txt','a');
fprintf(file5, '%f
',yout);close(file5);
n = n+3;
q=q+4;
end
A.2 Source code for nu-SVR using LIBSVM package

clear all;
close all;
clc
SPECTF = csvread('training.csv');
labels1 = SPECTF(:, 5);
features = SPECTF(:, 2:3);
features_sparse = sparse(features);
libsvmwrite('rad_train', labels1, features_sparse);
[label_vector4, instance_matrix4] = libsvmread('rad_train');
model_rad = svmtrain(label_vector4, instance_matrix4, 
    '-h 0 -s 4 -t 2 -g 10');
SPECTF = csvread('testing.csv');
labels = SPECTF(:, 5);
features = SPECTF(:, 2:3);
features_sparse = sparse(features);
libsvmwrite('rad_test', labels, features_sparse);
[label_vector, instance_matrix] = libsvmread('rad_test');
output = svmpredict(label_vector, instance_matrix, model_rad);
x=[output labels];
csvwrite('pred.csv', x);
A.3  \( R \) Software commands for executing ‘random-Forest’ package

```
training_data<-read.table("training.csv", header=TRUE, sep="",)
testing_data<-read.table("testing.csv", header=TRUE, sep="",)
train_inp<-subset(training_data, select=c(y2,y3))
train_out<-subset(training_data, select=c(y5))
test_inp<-subset(testing_data, select=c(y2,y3))
test_out<-subset(testing_data, select=c(y5))
rf<-randomForest(train_inp, train_out[,1], test_inp, test_out[,1],
keep.forest=TRUE, ntree=500)
pred<-predict(rf, test_inp)
y<-data.frame(pred, test_out)
write.table(y, file="pred.csv", sep="", row.names=pred)
```
A.4  \textit{R} Software commands for executing ‘quantreg-Forest’ package

\begin{verbatim}
train_data<-read.table("training.csv",header= TRUE,sep=",")
test_data<-read.table("testing.csv",header= TRUE,sep=",")
train_inp<-subset(train_data, select = c(-y3))
train_out<-subset(train_data, select = c(y3))
test_inp<-subset(test_data, select = c(-y3))
test_out<-subset(test_data, select = c(y3))
qrt1<-quantregForest(train_inp, train_out[,1],ntree=500)
pred1<-predict(qrt1,test_inp,quantiles = c(0.1,0.4,0.5,0.55,0.6))
pred2<-predict(qrt1,train_inp,quantiles = c(0.1,0.4,0.5,0.55,0.6))
y1<-data.frame(pred1,test_out[,1])
y2<-data.frame(pred2,train_out[,1])
write.table(y1, file = "pred_test_n_qrt3.csv", row.names = FALSE)
write.table(y2, file = "pred_train_n_qrt3.csv", row.names = FALSE)
\end{verbatim}