ARTIFICIAL NEURAL NETWORK MODEL FOR THE PREDICTION OF ELASTIC MODULUS OF CONCRETE

BY

CYRIL ODINAKA NWOSU (B.ENG)
Reg. No 20064549308

THESIS SUBMITTED TO THE POSTGRADUATE SCHOOL,
FEDERAL UNIVERSITY OF TECHNOLOGY, OWERRI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE OF MASTER OF
ENGINEERING (M. ENG) IN STRUCTURAL ENGINEERING

NOVEMBER, 2014.

CERTIFICATION

I certify that this work "Artificial Neural Network Application for the Prediction of Elastic Modulus of Concrete" was carried out by Cyril Odinaka Nwosu (Reg. No 20064549308) in partial fulfillment for the award of the degree of Master of Engineering (M. Eng) in Civil Engineering (Structures) of the Federal University of Technology, Owerri.

Dr. D.O. Onwuka
Principal Supervisor

Engr. Dr. (Mrs.) C. E. Okere
Co-Supervisor

Engr. Dr. (Mrs.) B. U. Dike
Head of Department

Engr. Prof. E. E. Anyawu
Dean, SEET

Engr. Prof. K. B. Oyoh
Dean, School of Postgraduate Studies

Engr. Prof. A.U. Elinwa
External Examiner
DEDICATION

I dedicated this work to my children, namely; Engr. Davis Munachimso Agughalam, Chiamaka Agughalam, Dr. Ifunanya Agughalam, Divine Agughalam and Meemee Agughalam. The Lord that started a good work in us will surely see us through to the end.
ACKNOWLEDGEMENTS

I would like to gratefully and sincerely thank my supervisor Engr. Dr. D. O. Onwuka for his guidance, understanding, and patience in correcting my writing during this research work. For everything you have done for me, Dr. (Mrs.) C. E. Okere, I thank you.

I wish to record my sincere thanks to the Head of Department, Engr. Dr. (Mrs.) B. U. Dike, for the concern she showed towards the completion of this work.

I would also like to thank Engr. Dr. J. C. Ezeh, Engr. Dr B.C. Okoro, Engr. Dr. J. C. Osuaqwu, Engr. Dr L.O. Ettu, Engr. Dr. H.U. Nwoke, Engr. (Mrs.) Arimanwa, Engr. C.F. Njoku, Engr. S. I. Agbo, Engr. Lewechi Anyaogu, Engr. Levi Nwakwasi, Engr. R. A. Egbulonu, Dr. Ujam, Engr. Ujiogu, Engr. Raphael Ogweoko, Mrs. Nkeiru Asodike, Mr. Joshua Egwuato, Mr. Nwachukwu.I.Nwachukwu, Mr. Chigozie Anyawu, Mr. Samuel Bamisheleye and all my colleagues, friends and family members who assisted and encourage me during the process of this work. God will reward your labour of love towards me.

Finally, to my lovely family, Munachimso, Chiamaka, Ifunanya, Divine and Meemee, thanks for always being there for me. Thank you all for upholding me even when I wanted to let go. Thank you for your understanding even when I had to give to you all, little or no attention. I love you all, so much.


**ABSTRACT**

This research presents Artificial Neural Network Model for the prediction of the Modulus of Elasticity of Concrete. Egbulonu (2011) equation derived from Scheffe’s (4, 2) simplex equation for predicting the Modulus of Elasticity (MOE) was used to generate 800 values. These data represent different values of Modulus of Elasticity (MOE) out of which, 571 values were selected randomly by the artificial neural network. From the selected values, 400 were used for training the network, 89 for testing the network and the remaining 82 for validating the network. The network was used to predict the Modulus of Elasticity (MOE) of the concrete. Its result was compared to the experimental results and was found to be 1.49% which is very close. The network was tested for good fit. It was found that the coefficient of correlation, R values for training, testing and validating data were 0.95237, 0.93731 and 0.91905 respectively, which showed that the data used for training, testing and validating the network have good fit since their R values is greater than 0.9. The network was also tested for adequacy at 0.05 significance level using statistical student’s test and was found to be adequate. Thus, the model can be used to predict the Modulus of Elasticity of concrete for any given mix ratio or vice versa.

**Keywords:** Concrete, Elasticity, Artificial Neural Network, Training, Testing, Validating, Mix Ratios, Optimization.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>TITLE</td>
<td>i</td>
</tr>
<tr>
<td>CERTIFICATION</td>
<td>ii</td>
</tr>
<tr>
<td>DEDICATION</td>
<td>iii</td>
</tr>
<tr>
<td>ACKNOWLEDGEMENTS</td>
<td>iv</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>v</td>
</tr>
<tr>
<td>TABLE OF CONTENTS</td>
<td>vi</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>xi</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>xii</td>
</tr>
</tbody>
</table>

## CHAPTER ONE: INTRODUCTION

1.1 Background of Study       1
1.2 Statement of Problem       3
1.3 Objectives of the Study    5
1.4 Significance of the Study  5
1.5 Scope of Study             6

## CHAPTER TWO: LITERATURE REVIEW

2.1 Strength of Concrete       8
2.1.1 Effect of Coarse Aggregate on Strength of Concrete       13
2.1.2 Prediction of Strength of Concrete         14
2.2 Elastic Modulus of Concrete    18
2.2.1 Definition and determination of Elastic Modulus of Concrete 18
2.2.2 Significance of Elastic Modulus of Concrete 19
2.2.3 Effect of Coarse Aggregate on Elastic Modulus of Concrete 20
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2.4</td>
<td>Previous Models for Predicting Elastic Modulus of Concrete</td>
<td>24</td>
</tr>
<tr>
<td>2.2.5</td>
<td>Scheffe’s Modeling Method</td>
<td>26</td>
</tr>
<tr>
<td>2.2.5.1</td>
<td>The Simplex Lattice</td>
<td>26</td>
</tr>
<tr>
<td>2.2.5.2</td>
<td>Scheffe’s Simplex Lattice Design</td>
<td>28</td>
</tr>
<tr>
<td>2.2.5.3</td>
<td>The Simplex Canonical Polynomial</td>
<td>29</td>
</tr>
<tr>
<td>2.2.6</td>
<td>Genetic Algorithm</td>
<td>30</td>
</tr>
<tr>
<td>2.3</td>
<td>Artificial Neural Networks</td>
<td>31</td>
</tr>
<tr>
<td>2.4</td>
<td>Current Development and Future Directions in Utilization of ANNs</td>
<td>34</td>
</tr>
<tr>
<td>2.4.1</td>
<td>Model Robustness</td>
<td>35</td>
</tr>
<tr>
<td>2.4.2</td>
<td>Model Transparency and Knowledge Extraction</td>
<td>37</td>
</tr>
<tr>
<td>2.4.3</td>
<td>Model Extrapolation</td>
<td>42</td>
</tr>
<tr>
<td>2.4.4</td>
<td>Model Uncertainty</td>
<td>42</td>
</tr>
<tr>
<td>2.5</td>
<td>Ability of Multilayer Neural Networks to Approximate Arbitrary Functions</td>
<td>44</td>
</tr>
<tr>
<td>2.5.1</td>
<td>One–Hidden–Layer Network</td>
<td>47</td>
</tr>
<tr>
<td>2.5.2</td>
<td>Two–Hidden–Layer Network</td>
<td>62</td>
</tr>
<tr>
<td>2.5.3</td>
<td>Optimum Network Architecture</td>
<td>65</td>
</tr>
<tr>
<td>2.6</td>
<td>Radial Basis Function Network (RBFN)</td>
<td>67</td>
</tr>
<tr>
<td>2.6.1</td>
<td>Ability of RBFN to Approximate Arbitrary Function</td>
<td>69</td>
</tr>
<tr>
<td>2.6.2</td>
<td>Optimum Network Architecture</td>
<td>70</td>
</tr>
<tr>
<td>2.7</td>
<td>Performance Comparison between MLN and RBFN</td>
<td>71</td>
</tr>
<tr>
<td>2.8</td>
<td>Probability Framework of Artificial Neural Networks</td>
<td>73</td>
</tr>
<tr>
<td>2.8.1</td>
<td>Probability Model of Feedforward Networks</td>
<td>73</td>
</tr>
<tr>
<td>2.8.1.1</td>
<td>Maximum Likelihood Estimation Model</td>
<td>76</td>
</tr>
<tr>
<td>2.8.1.2</td>
<td>Choice of Transfer Function: A Probabilistic View</td>
<td>79</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>2.8.2</td>
<td>Probabilistic Model of Radial Bases Function Networks</td>
<td>84</td>
</tr>
<tr>
<td>2.9</td>
<td>Artificial Neural Network Method of Optimization</td>
<td>89</td>
</tr>
<tr>
<td>2.10</td>
<td>Artificial Neural Networks for Function Approximation</td>
<td>95</td>
</tr>
<tr>
<td>2.11</td>
<td>The Multilayer Feed Forward Network with Back Propagation Learning Algorithm (MLP with BP)</td>
<td>96</td>
</tr>
<tr>
<td></td>
<td><strong>CHAPTER THREE: MATERIALS AND METHODS</strong></td>
<td>104</td>
</tr>
<tr>
<td>3.1</td>
<td>Materials</td>
<td>104</td>
</tr>
<tr>
<td>3.1.1</td>
<td>Cement</td>
<td>104</td>
</tr>
<tr>
<td>3.1.2</td>
<td>Aggregates</td>
<td>105</td>
</tr>
<tr>
<td>3.1.3</td>
<td>Fine Aggregate</td>
<td>105</td>
</tr>
<tr>
<td>3.1.4</td>
<td>Coarse Aggregate</td>
<td>106</td>
</tr>
<tr>
<td>3.1.5</td>
<td>Water</td>
<td>106</td>
</tr>
<tr>
<td>3.2</td>
<td>Constituents of Concrete Mix</td>
<td>106</td>
</tr>
<tr>
<td>3.2.1</td>
<td>Weighing of Aggregates</td>
<td>108</td>
</tr>
<tr>
<td>3.2.2</td>
<td>Workability</td>
<td>108</td>
</tr>
<tr>
<td>3.2.2.1</td>
<td>Mixability</td>
<td>109</td>
</tr>
<tr>
<td>3.2.2.2</td>
<td>Transportability</td>
<td>109</td>
</tr>
<tr>
<td>3.2.2.3</td>
<td>Mouldability</td>
<td>109</td>
</tr>
<tr>
<td>3.2.2.4</td>
<td>Compactibility</td>
<td>110</td>
</tr>
<tr>
<td>3.2.2.5</td>
<td>Factors Affecting Workability</td>
<td>110</td>
</tr>
<tr>
<td>3.3</td>
<td>Tests Carried Out</td>
<td>111</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Bulk Density</td>
<td>111</td>
</tr>
<tr>
<td>3.3.2</td>
<td>Slump Test</td>
<td>112</td>
</tr>
<tr>
<td>3.3.3</td>
<td>Silt Content Test on Fine Aggregate</td>
<td>116</td>
</tr>
<tr>
<td>3.3.4</td>
<td>Specific Gravity Determination and Coarse Aggregate</td>
<td>116</td>
</tr>
<tr>
<td>3.3.5</td>
<td>Compressive Strength Test</td>
<td>117</td>
</tr>
<tr>
<td>3.3.6</td>
<td>Sieve Test</td>
<td>118</td>
</tr>
<tr>
<td>3.4</td>
<td>Concrete Mix Design</td>
<td>119</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>4.1.4</td>
<td>Sieve Analysis of Crushed Rock Aggregate</td>
<td>134</td>
</tr>
<tr>
<td>4.1.5</td>
<td>Results of Modulus of Elasticity Test</td>
<td>134</td>
</tr>
<tr>
<td>4.2</td>
<td>Analysis of Results</td>
<td>137</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Artificial Neural Network-based Modeling of Modulus of Elasticity of Concrete</td>
<td>137</td>
</tr>
<tr>
<td>4.2.1.1</td>
<td>Learning Rates and Momentum Constant</td>
<td>138</td>
</tr>
<tr>
<td>4.2.1.2</td>
<td>Architecture of the Artificial Neural Network</td>
<td>138</td>
</tr>
<tr>
<td>4.2.1.3</td>
<td>Selection of the Training Data</td>
<td>138</td>
</tr>
<tr>
<td>4.2.1.4</td>
<td>Validation of the Network Performance</td>
<td>139</td>
</tr>
<tr>
<td>4.2.1.5</td>
<td>Comparison of Predicted and Experimental Values</td>
<td>142</td>
</tr>
<tr>
<td>4.3</td>
<td>Test of Adequacy of the Artificial Neural Network Model</td>
<td>143</td>
</tr>
<tr>
<td>4.3.1</td>
<td>Statistical Hypothesis</td>
<td>143</td>
</tr>
</tbody>
</table>

**CHAPTER FIVE: CONCLUSIONS, RECOMMENDATIONS AND CONTRIBUTION TO KNOWLEDGE**

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>Conclusions</td>
<td>146</td>
</tr>
<tr>
<td>5.2</td>
<td>Recommendations</td>
<td>147</td>
</tr>
<tr>
<td>5.3</td>
<td>Contribution to Knowledge</td>
<td>148</td>
</tr>
</tbody>
</table>

Appendices

References 158
LIST OF TABLES

Table 2.1  Recommended Coarse Aggregate Coefficients  22
Table 2.2  Compressive Strengths and Moduli of very high Strength Concrete Mixtures Containing Different Aggregate Types  23
Table 2.3  Comparison of Genetic Algorithm with Classic Algorithm  31
Table 2.4  The Training Time of a Feed Forward Network  64
Table 2.5  Types of Transfer Function of Different Artificial Neural Network Application  83
Table 3.1  Mixture Proportion for Concrete Beams (150x150x600mm)  125
Table 3.2  Artificial Neural Network (ANN) Information  131
Table 4.1  Physical Properties of Dangote Portland Cement  132
Table 4.2  Chemical Properties of Dangote Portland Cement  133
Table 4.3  Grain Size Distribution of Sand  134
Table 4.4  Results of Sieve Analysis of Crushed Rock Aggregate  134
Table 4.5  Experimental Results Compared and Scheffe’s Results of Modulus of Elasticity, E  136
Table 4.6  Experimental Results compared with Artificial Neural Network Prediction  142
Table 4.7  Check of Adequacy of the Network  144
| Fig. 2.1 | Relationship between Strength and Water/Cement Ratio | 15 |
| Fig. 2.2 | Strength Development of Various Concrete Mixes | 18 |
| Fig. 2.3 | Stress-strain Relation | 19 |
| Fig. 2.4 | Stress-strain Relation for Cement Paste, Aggregate and Concrete | 21 |
| Fig. 2.5 | Stress-strain Curve at 28 days | 24 |
| Fig. 2.6 | Typical Structure and Operation of ANNs | 32 |
| Fig. 2.7 | The Main Steps in ANN Development | 34 |
| Fig. 2.8 | Graphical Classification of Modeling Techniques | 39 |
| Fig. 2.9 | Function Approximation by Artificial Neural Networks | 45 |
| Fig. 2.10 | More Function Approximation by Artificial Neural Networks | 46 |
| Fig. 2.11 | Target-Sigmoid Functions | 48 |
| Fig. 2.12 | Numerical Function Approximation using One-Hidden-Layer Feedforward Networks with Different Size | 52 |
| Fig. 2.13 | Numerical Function Approximation using One-Hidden-Layer Feedforward Networks with 70 Processing Elements in the Hidden Layer | 53 |
| Fig. 2.14 | Numerical Function Approximation using One-Hidden-Layer Feedforward Networks with 30 Processing Elements in the Hidden Layer | 54 |
| Fig. 2.15 | Numerical Function Approximation using a One-Hidden-Layer Network | 56 |
| Fig. 2.16 | Effect of the Number of Training Samples to the Accuracy of Artificial Neural Network | 57 |
| Fig. 2.17 | Data for Classification Problem | 58 |
Fig. 2.18  Classification of Result using One-Hidden-Layer Network 61

Fig. 2.19  The Effect of Number of Processing Elements to the Classification Accuracy 62

Fig. 2.20  Cross Validation Model 66

Fig. 2.21  Typical RBFN 67

Fig. 2.22  Processing Element of a Artificial Neural Network 74

Fig. 2.23  A Simulated System 77

Fig. 2.24  A Simulated System 79

Fig. 2.25  A Simple Artificial Neuron 92

Fig. 2.26  A very Simple Artificial Neural Network 93

Fig. 2.27  Feedforward Network 97

Fig. 3.1  Slump Test Apparatus 114

Fig. 3.2  The Making of the Test Specimens and the Gauge Length I 118

Fig. 4.1  Regression Plot Network Outputs and Network Targets 139

Fig. 4.2  Bar Chart showing the Plot of Artificial Neural Network Values against Experimental Values 140

Fig. 4.3  (a) Plot of Network against Epoch
(b) Plot of Validation of Network against Epoch 141
CHAPTER ONE
INTRODUCTION

1.1 Background of Study
Concrete is one of the cement–based composite materials. It is one of the popular artificial materials because it is energy–conserving, cost–efficient, easy to cast into any shape, and highly resistant to marine environments. One of the major properties of concrete is its elastic modulus. The elastic modulus of concrete is a key factor in structural and material engineering. Designers need the elastic modulus for estimating immediate and time–dependent deformation, determining modulus of rupture and evaluating the stiffness of building and any structural members. Thus, in reinforced and prestressed concrete design, the elastic moduli of structural materials are very important parameters based on stiffness. It is also important in reinforced and prestressed concrete for creep and shrinkage evaluation, as well as crack control especially at an early age (Amir et. al, 2010).

The modulus of elasticity is defined in the region of stress–strain characteristics in which Hooke’s law, is obeyed as the ratio of stress over strain. In mechanics, Hooke’s law of elasticity is an estimation that states that the amount of strain is linearly related to stress. This can be determined from the slope of a
stress–strain curve created during the tensile tests on a specimen of concrete.

Despite its importance, elastic modulus is not usually measured in the site for compliance purposes. It is often estimated from the compressive strength based on empirical relationships proposed by various codes of practices. This is mainly to avoid performing laborious and time-consuming direct measurements from load-deformation curve (Kezhen et.al, 2010).

Analytical methods, including statistical ones used to predict elastic modulus of concrete in the past may be very complex and mathematical regression models are usually imperfect description of complex physical phenomena. The accuracy of the estimated results depends on the size of available data.

One way to have near perfect description of elastic modulus is by the use of soft computing technique. Such techniques may include artificial neural network (ANN) method, fuzzy logic (FL), linear genetic algorithm method, adaptive neuro–fuzzy interference etc.

In this research, the elastic modulus of concrete will be predicted using artificial neural network (ANN) method, because it can capture highly non-linear and complex relations among the input/output variables in a system with prior knowledge about
the nature of these interactions. Unlike traditional parametric models, this model is also able to construct a supposedly complex relationship between the input and output variable with an excellent level of accuracy compared with conventional methods (Shahin et.al, 2009).

The main advantage of this method is that one does not have to assume an explicit model form, which is a prerequisite in the parametric approaches. Indeed, in ANN models, a relationship of a possibly complicated nature between input and output variables, is generated by the data points.

In comparison to parametric methods, ANNs can deal with relatively imprecise or incomplete data and approximate results, and are less vulnerable to outliers. They are highly parallel, that is their numerous independent operations can be executed simultaneously (Haykin, 1994).

Therefore, in this research, ANN will be completely utilized to find the mathematical relationships between the elastic modulus and compressive strength of concrete.

1.2 Statement of Problem

At present, the modulus of elasticity property of concrete used in structural design, is based on literature, design codes or based on limited research of locally available material. The possible
errors in the estimation of this parameter, can cause a serious
problem in the design of reinforced and pre-stressed structures.
Thus, there is a great need for a comprehensive testing and
evaluation of locally available concrete mixes to determine the
mechanical and physical properties of lightweight as well as
normal weight concrete so that correct values of this property
can be used in structural designs.
In the time past, many researchers and engineers have tried to
predict the elastic modulus of concrete using theoretical and
empirical approaches based on compressive strength of concrete.
Most of these empirical approaches were developed by using
statistical regression models. These models are usually imperfect
description of complex physical phenomena since the data
collected during the prediction are always in complex form.
In this research, artificial neural network model is developed for
predicting elastic modulus of concrete using artificial neural
network (ANN) method. This method is preferable to regression
models because it is able to construct a supposedly complex
relationship between the input and variable with an excellent
level of accuracy. Also, it can deal with relatively imprecise or
incomplete data and approximate results and are less vulnerable
to outliers.
1.3 Objectives of Study

The aim of this work is to develop artificial neural network model for the prediction of the elastic modulus of concrete. The following are the specific objectives of the work:

(a) To determine experimentally the elastic modulus of concrete mixes.

(b) To develop an artificial network model for predicting the elastic modulus of concrete.

(c) To predict the elastic modulus of concrete mix using Artificial Neural Network.

(d) To compare the result obtained using Artificial Neural Network method and that obtained using regression model.

1.4 Significance of Study

Different formulae are proposed by building codes for computing the modulus of elasticity of concrete structures. These formulae were developed mostly from regression models and other empirical approaches, that cannot perfectly predict elastic modulus of concrete from complex data.

None of them however, is able to correctly predict the elastic modulus of high strength concrete, especially when it involves use of large number of samples. Hence, in this study, a soft–computing method which can predict elastic modulus of normal
strength concrete (NSC) more accurately than regression model of analysis, is employed. This method is capable of handling complex data; using them to predict the approximate value of elastic modulus of any concrete strength or type. The method is artificial neural network (ANN) method.

When this research is completed, the elastic modulus of any type of concrete can easily be predicted with ease by engineers and designers using the developed artificial neural network model. This will solve problem of faulty design resulting from wrong estimation of elastic modulus using regression models, which are not capable of handling complex data collected during experiment. And, the use of correct elastic modulus in structural designs will in-turn ensure the safety of structures and its occupants.

1.5 Scope of Study

This work is limited to prediction of elastic modulus of concrete using artificial neural network method. The work was covered in five chapters.

The work was introduced in chapter one. This chapter discussed the background of study, statement of problem, objective of study, significance of study and scope of study. In chapter two, the literature of the past works related to the subject matter was
reviewed. Chapter three dealt with in-depth description of artificial neural network method, materials used and various tests carried during the research.

The results collected during the experiment was analyzed and discussed in chapter four. The research was concluded in chapter five with recommendations.
2.1 Strength of Concrete

Strength of a material is one of the important properties of materials. Strength is the ability of a material to resist an induced amount of stress that would make it fail. Generally, it is expressed in pounds per square inch (psi) or megapascals (MPa). It is one of the most sought-after properties of concrete, especially by designers and quality control engineers. The value of strength gives an indication of overall quality and uniformity of a material. In addition, it provides a load-carrying capacity of a material used for design strength purposes. Besides, other technically important non-strength properties, such as modulus of elasticity and flexural strength which are measured by more complicated tests, can be related to strength. Even though the strength tests are relatively easy to perform, strength is a very volatile property. Consistent fabrication and testing procedures of concrete still give significant variations in the results of concrete strength.

These variations in strength of results concrete can be attributed to the following:

(a) **Mixing water**

As a rule, if water is potable, it can be used as mixing water. Water should not have excess undesirable organic
or inorganic substances which can potentially have an adverse effect, not only in the strength, but also in the setting time, surface efflorescence (deposits of white salts on the surface of concrete), and resistance to degradation. Water that is not drinkable might be used as mixing water as long as it in compliance with the acceptance criteria listed in ASTM-C94 (1994) "Ready-Mixed Concrete". ASTM C-94 (1994) states that the mixing water needs to be clear and apparently clean, free of substances that discolor it, makes it taste or smell in unusual way. Other sources, such as sea water increases the risk of corrosion in reinforced and pre-stressed concrete, but it does not have any effects on the strength of plain concrete, as long as the pH is between 6.0 and 8.0 and even 9.0. It needs to be mentioned, that this also applies to water present on the surface and pores of aggregates used for concrete production.

(b) **Age of concrete**

Based on typical strength-time curves, it will be noticed that the older the concrete is, the stronger it will be because of a longer hydration process.
(c) **Characteristics and proportions of the materials**

Gradation, shape, and surface texture of the aggregate, are of great importance. The desired coarse aggregate characteristics include strength, high resistance to abrasion, and high modulus of elasticity. These desired characteristics are discussed in more detail in the next section.

(d) **Mixing time**

Longer mixing times results in loss of moisture, which in turn results into drier concrete, which is usually hard to compact. Consequently, this leads to high volume of voids, and thus weak concrete results.

(e) **Physical condition and properties of the specimen**

To ensure uniform load transfer to test specimens, ASTM-C39 (2009) states that the ends of the specimen to be tested must be within 1/8 inches for every 12 inches in the perpendicular direction from the specimen axis, and that the ends must be plane to within 0.002 inches after being ground in the grinder. According to Lamond and Pielert (1996), specimens that do not meet the ASTM requirements yield relatively lower strength test results. Specimens must have consistent physical conditions at the time of testing.
as well. For example, specimens that are allowed to dry up give relatively higher strength results.

(f) **Degree of consolidation**

Neville (1996) states that the volume of pores in concrete influences the strength of concrete according to the following power function type:

\[ f_c = f_{c,0}(1 - p)^n \]

where:

- \( p \) = the volume of voids;
- \( f_c \) = the strength of concrete with porosity \( p \);
- \( f_{c,0} \) = strength at zero porosity; and
- \( n \) = coefficient obtained by regression analysis.

If the previous formula was plotted on a log scale, a general linear relationship can be established which implies that as porosity increases, the compressive strength decreases. Specimens that have low degrees of compaction, have as a result, high volume of voids and, therefore, yield lower strength results.

(g) **How and how well the testing procedure is performed**

Rate of loading can have an effect on the strength of concrete. Specimen loaded at a very slow rate, fail at loads that are about 25% lower than the standard test strength,
and specimens loaded at a very high rate fail at loads about 15% higher than the standard test strength. If the rate of loading is adjusted during testing, it can have a similar, but milder effect. If the diameter of the specimen is not located in the center, the use of a faulty measuring device, can lead to errors in strength calculations. The calibration of the loading apparatus is of great importance, as well as a smooth and continuous application of the load by the apparatus. Surface plainness of both the spherically seated and the solid bearing blocks must satisfy requirements of surface plainness. The spherically seated bearing blocks, must have the proper dimensions to run the test on a 6" diameter specimen, and the specimen needs to be under the center of the bearing face to avoid reduction in strength results.

Because of the potential variability of the strength test results, the appropriate test must be performed whenever a specific non-strength property is of interest. Characteristics and proportions of coarse aggregate and how it affects the strength of concrete, are discussed below.
2.1.1 Effect of Coarse Aggregate on Strength of Concrete

Usually it is the water to cementitious materials (w/c) ratio that governs concrete strength, but strength is influenced by properties of coarse aggregates as well. Coarse aggregate makes up about 75% of concrete by volume, so its characteristics cannot be overlooked. Coarse aggregate is defined as aggregate that does not pass the No 4 ASTM sieve, which is 4.75 mm (\(\frac{3}{16}\) in.) in size, and it contributes to both economic and technical advantages of concrete considerably. Undesired properties of coarse aggregate may affect the strength, durability, and performance of concrete negatively; therefore tests on coarse aggregate are needed to evaluate its suitability in concrete.

According to Kaplan, "no relationship between the strength of the coarse aggregate and the strength of concrete, was established. This finding should not be taken to mean that aggregates of low strength will not affect the concrete strength". This seems to be the case of non high–strength concretes (Wu. et.al.). They conducted a study on the effects of coarse aggregate type on mechanical properties of concrete, using four different types of aggregate: crushed granite, limestone, marble, and crushed quartzite. They found that concretes containing granite were stronger than concretes containing limestone, and had higher elastic moduli for different water/cement (w/c) ratios such as (0.26, 0.44, and 0.55). As the w/c ratio increased, the
effect of the type of aggregate used became insignificant. So, the type of aggregate makes an important role in concretes with low w/c ratios because the bond between mortar and coarse aggregate in concretes, has similar strength, allowing full strength potential.

2.1.2 Prediction of Strength of Concrete

As a structural design becomes more complex, flexible, and bigger, a prediction of the strength of concrete, can have a beneficial impact on the construction project budget by reducing the amount of formwork needed knowing that the cost of formwork can be as much as 60% of the total cost of the concrete in place (ACI 347R).

Since compressive strength is the most common measure by which the adequacy of concrete is judged, different equations to predict the compressive strength of concrete at different ages, have been developed over the years. These equations are simplified expressions representing the relationship between average laboratory data obtained under steady environmental and loading conditions. It must also be mentioned that these equations are for concretes containing no admixtures.

In engineering practice, water-cementitious material ratio is one of the principal factors governing the strength of concrete.
Abrams (1918) developed a relationship between water-cementitious material ratio and strength of fully compacted concrete containing about 1% of air voids. Abram found that the actual compressive strength is equal to:

\[ f_c = \frac{K_1}{w/c} \frac{1}{K_2} \]

where:

- \( w/c \) represents the water/cement ratio of the mix
- \( K_1 \) and \( K_2 \) are empirical constants obtained through regression analysis.

The following figure (i.e Fig 2.1) shows the relationship between a range of typical strength and water-cement ratio based on over 100 different concrete mixtures.

**Fig. 2.1:** Relationship between compressive strength and water/cement ratio of concrete
It can be observed from the fig. 2.1 that strength is inversely proportional to the water/cement ratio. The following equation is similar to that of Abram and relates the actual compressive strength of concrete to the absolute volumetric proportions of cement, water, and air. This equation was formulated by Féret (1896) and has the form:

\[ f_c = K \left( \frac{c}{c+w+a} \right)^2 \]  \hspace{1cm} 2.3

where:

c, w, and a represent the values of the volumetric proportion of cement, water, and air, respectively and

K is a constant that is obtained through regression analysis.

The following empirical equation has been provided by ACI 209R for the prediction of compressive strength,

\[ (f_c)_t = \frac{t}{\alpha + \beta t} \left( f_c \right)_{28} \]  \hspace{1cm} 2.4

where:

\[ \alpha \] and \( \beta \) are constants dependent on type of cement used and type of curing, \( (f_c)_{28} \) is a 28th -day strength

\[ t = \text{the age of the concrete in days.} \]
For this study type I cement was used, specimens were moist cured, and constants $\alpha$ and $\beta$ become 4.0 and 0.85 respectively.

It is also of importance to note the relationship between the 28-day compressive strength and the compressive strength at other ages, if one wants to predict strength development. For normal concrete, the strength at 7 days is about 75% of the 28 day strength; the strength at 56 and 91 days are about 10% to 15% greater than what relative to 28 day strength (Kosmatka et.al, 2000). This trend is shown in Figure 2.2. If the strength development is known, formwork can be removed, economy can be achieved by reusing them in other areas of the project, productivity of the construction process can reach maximum efficiency, and thus time of completion can be improved drastically. While these equations can help predict the strength of concrete at different ages, it must be noted that without a doubt, the best prediction is made based on historical data rather than by batching trial mixes. Whenever data is not available, it is highly recommended spending time on trial batching to ensure that the desired results can be achieved on the field at the time of construction.
2.2 Elastic Modulus of Concrete

2.2.1 Definition and Determination of Elastic Modulus of Concrete

According to ASTM E6-89, the modulus of elasticity, denoted as $E$, is defined as the ratio between normal stress to strain below the proportional limit of a material, and it is a measure of instantaneous elastic deformation. Since no test exist to evaluate the direct elastic modulus of concrete, the proportional limit, is evaluated by means of ASTM C 39 first, and this value is used to establish the limit used for the curves in the repeated applications of load. The elastic modulus is then calculated by determining the slope of the straight line portion of the stress-

**Fig 2.2:** Strength development for various concrete mixes
strain diagram. Fig 2.3 illustrates the stress-strain plot of a concrete as it goes through one loading cycle.

**Fig 2.3: Stress-strain relation for concrete**

### 2.2.2 Significance of Elastic Modulus of Concrete

When concrete is subjected to loading, it exhibits a linear stress-strain relationship in the elastic range. The ratio, which is the slope of this linear portion of the relationship, is known as the modulus of elasticity. The elastic limit is “the greatest stress which a material is capable of sustaining without any deviation from proportionality of stress to strain”.

As defined in the previous section, the modulus of elasticity is the ratio between stress and the reversible strain. When a load is applied to concrete, it will deform depending on the magnitude of the load and its rate of application. The value of strain is of
immense importance because it represents the rigidity of the structural design and the stress at which the concrete will experience permanent deformation if exceeded. Most structures are subject to cyclic loading and it is, therefore, important to know the elastic portion for design purposes, especially the amount of steel required for reinforcement.

According to Klieger and Lamond (2006), modulus of elasticity can be measured in tension, compression, or shear. For this research project, the modulus elasticity was predicted by using Artificial Neural Network method, using compressive strength of the concrete to train the network.

### 2.2.3 Effect of Coarse Aggregate on Elastic Modulus of Concrete

According to the Canadian Portland Cement Association (CPCA), the elastic modulus of concretes is influenced by the properties of different aggregate types with different elastic moduli. Typical stress-strain curves for aggregate, cement paste and concrete as contained in a report to the Florida Department of Transportation by Tia, Liu, and Brown (2001), are shown in fig. 2.4.
From Fig 2.4, it can be seen that aggregate has a substantially larger elastic modulus as compared to concrete and cement paste, and so aggregate has a significant effect on the elastic modulus of concrete. Parrot (1979) developed an equation that can estimate the elastic modulus of concrete; this is the only model that relates modulus of elasticity to strength and type of aggregate.

\[ E_c = K_0 + 0.2f_c \]  

where:

- \( E_c \) = the modulus of elasticity
- \( K_0 \) = a factor depending on the type of aggregate

\[ \text{Eqn} \: 2.5 \]
\( f_c' = \) the compressive strength of the concrete ranging from 20MPa to 70 MPa.

Iravani (2000) reported that concrete containing granite aggregate yield lower elastic modulus relative to concrete containing limestone aggregate. He concluded that the amount of coarse aggregate in concrete has a "major effect on the static modulus of elasticity" of concrete.

Table 2.1: Recommended coarse aggregate coefficients

<table>
<thead>
<tr>
<th>Kind of coarse aggregate</th>
<th>Recommended K₀ factor</th>
<th>Coefficient of Variation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sandstone gravel</td>
<td>0.71</td>
<td>0.066</td>
</tr>
<tr>
<td>Siliceous gravel</td>
<td>0.76</td>
<td>--</td>
</tr>
<tr>
<td>Limestone</td>
<td>0.92</td>
<td>0.093</td>
</tr>
<tr>
<td>Dolomite</td>
<td>0.92</td>
<td>0.087</td>
</tr>
<tr>
<td>Quartzite</td>
<td>0.97</td>
<td>0.055</td>
</tr>
<tr>
<td>Granite</td>
<td>0.82</td>
<td>0.072</td>
</tr>
<tr>
<td>Trap rock</td>
<td>0.97</td>
<td>0.018</td>
</tr>
<tr>
<td>Sandstone</td>
<td>0.61</td>
<td>0.143</td>
</tr>
</tbody>
</table>

Source: Iravani (2000)

Yilmaz et.al, (2006) stated that aggregate influences the elastic modulus of concrete significantly when the water/cement (w/c) ratio of the mix is less than 0.40.

The following table given by Aïtcin and Mehta (1990) shows that the elastic modulus of concretes made with limestone aggregate is higher at 28 and 56 days than the elastic modulus of concretes made with granite aggregate.
**Table 2.2:** Compressive strengths and elastic moduli of very high strength concrete mixtures containing different aggregate types

<table>
<thead>
<tr>
<th>Age, days</th>
<th><strong>Compressive Strengths of Concrete</strong> $f'_c$, MPa (psi)</th>
<th>Diabase</th>
<th>Limestone</th>
<th>Gravel</th>
<th>Granite</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>41.1 (5960)</td>
<td>42.5 (6160)</td>
<td>40.6 (5880)</td>
<td>37.2 (5390)</td>
</tr>
<tr>
<td>28</td>
<td></td>
<td>100.7 (14,600)</td>
<td>97.3 (14,200)</td>
<td>92.1 (13,350)</td>
<td>84.8 (12,300)</td>
</tr>
<tr>
<td>56</td>
<td></td>
<td>104.8 (15,200)</td>
<td>101.3 (14,700)</td>
<td>95.9 (13,900)</td>
<td>88.6 (12,850)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Age, days</th>
<th><strong>Elastic Modulus of concrete</strong> $E'$, GPa (10^6 psi)</th>
<th>Diabase</th>
<th>Limestone</th>
<th>Gravel</th>
<th>Granite</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td></td>
<td>36.6 (5.3)</td>
<td>37.9 (5.5)</td>
<td>33.8 (4.9)</td>
<td>31.7 (4.6)</td>
</tr>
<tr>
<td>56</td>
<td></td>
<td>37.9 (5.5)</td>
<td>40.7 (5.9)</td>
<td>35.9 (5.2)</td>
<td>33.8 (4.9)</td>
</tr>
</tbody>
</table>

*Source:* (Aitcin and Mehta, 1990)

They concluded that even as granite is "as hard and strong" as the other types of aggregates investigated in the study, concretes with granite yielded lower elastic modulus at all test ages. From the Table 2.2, it can be seen that concretes containing granite have a compressive strength of about 87% to that of limestone concretes at 28 days, and about 84% elastic modulus of limestone concrete. They attributed the results to the failure surface of the concretes which showed a fracture path through the granite aggregate particles, which was not the case in the limestone concrete. It must be noted that all concretes had identical water/cement ratios. The fig 2.5 shows the stress-
strain curves for concretes containing different types of aggregate. It can be seen from the figure that concretes containing limestone have a steeper stress-strain slope as compared to concretes containing granite which translates into higher elastic moduli for limestone concrete.

![Stress-strain curves at 28 days](image)

**Fig 2.5:** Stress – strain curves at 28 days

### 2.2.4 Previous Models For Predicting Elastic Modulus of Concrete

It is safe to assume that if concrete is very strong, its elastic modulus would be very high because it would resist deformations at high stress applications. This also means that the initial slopes of the stress-strain diagrams are steeper in
stronger concretes. Some of the models used in past to predict the elastic modulus of concrete are as follow:

(a) **Model recommended by ACI (209):**

The American Concrete Institute (ACI), section 8.5.1 recommended the Eqn (2.6) for calculating elastic modulus of concrete

\[ E_c = 33(w^{1.5})\sqrt{f_c} \]  

where:

- \( w \) = unit weight of the concrete in lb/ft\(^3\), \( f_c' \), the design compressive strength in psi and is valid for unit weights ranging from 90 to 155 lb/ft\(^3\)

(b) **Model recommended by ACI for normal-weight concrete:**

The American Concrete Institute (ACI), section 8.5.1 recommended the following modulus of elasticity, \( E_c \) for normal weight concrete with a density of 145 lb/ft\(^3\)

\[ E_c = 57,000\sqrt{f_c} \]  

(c) **Model recommended by ACI for high-strength concretes**

\[ E_c = 40,000\sqrt{f_c} + 1.0 \times 10^6 \]
(d) **Model by CEB-FIP Code (1990):**

The CEP-FIP code specifies the following equation for the calculation of modulus elasticity, $E_{ci}(t)$:

$$E_{ci}(t) = \left( \text{Exp} \left( s \left( 1 - \left( \frac{28}{t/t_1} \right)^{0.5} \right) \right) \right)^{0.5} E_{ci} 2.9$$

where:

$s = \text{coefficient depending on the type of cement}$

$t = \text{age of concrete (days)}$

$t_1 = 1 \text{ day}$

$E_{ci} = \text{elastic modulus of concrete at 28 days.}$

It is important to realize that the results of tests with different types of aggregates will give a widely scattered data for all the models. Regression analysis needs to be performed to account for different types of aggregates. In this research, elastic modulus of concrete was predicted using artificial neural network method.

### 2.2.5 Scheffe’s Modelling Method

#### 2.2.5.1 The Simplex Lattice

A simplex lattice can be described as a structural representation of lines joining the atoms of a mixture. This lattice can be used as a mathematical space in model experiment involving mixtures by considering the atoms as the constituent components of the
mixture, the constituent element are water, cement, coarse aggregate and fine aggregate. A mixture experiment involves mixing various proportions of two or more components to make different compositions of an end product (Aggarwal, 2002). These mixture components are subject to the constraints:

\[
\begin{align*}
0 & \leq X_i \leq 1 \quad 2.10 \\
\sum X_i & = 1 \quad 2.11 
\end{align*}
\]

where:

\[q = \text{number of components}\]

\[X_i = \text{the proportion of the components of the ith components in the mixture.}\]

This shows that if we assume the mixture to be a unit quantity, then the sum of all the proportions of the component, must be unity. Therefore, the space factor reduces to a regular \((q - 1)\) dimensional simplex.

A simplex is simply the projection of an n-dimensional space into an n-1 dimensional co-ordinate system (Goeiz, 2011). This is possible because the sum of proportions of a mixture, is constrained to unity. Therefore, a combination of 3 components can be projected into a 2-dimensional triangular field. The simplex of a mixture of 4 components is a 3-dimensional solid.
equilateral tetrahedron. The degree of the simplex lattice is defined by the degree of the polynomial that may be used to fit the response surface over the simplex. A theory for experiment with mixture of the q-components whose purpose is the empirical prediction of the response to any mixture of the components, when the response depends only on the proportion of the component and not the total amount was developed. Scheffe’s (1958) introduced the (q, m) simplex lattice.

2.2.5.2 Scheffe’s Simplex Lattice Design

Scheffe’s method of optimization is applicable to mixtures in which the desired response depends on the proportion of components present in the mixture, rather the quality of the mixture (Onwuka et.al, 2013). This method of optimization uses the simplex lattice factor space described in section 2.2.5.1 to determine the number of mix ratios for any concrete mixture components.

The (q,m) simplex lattice design introduced by Scheffe’s has the following properties:

(i.) The factor space has uniformly spaced distribution of points
(ii.) The proportions used for each factor have (m+1) equally spaced values from (0 to 1). That is to say:
\[ X_{ij} = 0, \frac{1}{m}, \frac{2}{m}, \frac{3}{m}, \ldots, \ldots, 1 \]  \hspace{1cm} 2.12

components used.

Scheffe’s showed that the number of points in \((q, m)\) lattice is given by:

\[ \frac{m+q+1}{m} = \frac{q(q+1)\ldots(q+m+1)}{m} \]  \hspace{1cm} 2.13

2.2.5.3 The Simplex Canonical Polynomial

The \((q, m)\) simplex lattice designs are characterized by the symmetric arrangements of points within the experimental region and a well chosen polynomial equation to represent the response surface over the entire simplex region. The polynomial has exactly as many parameters as there are numbers of points in the associated simplex lattice design. The response represent the property studied and is normally assumed to be a multi-varied function.

Scheffe’s (1958) introduced canonical polynomials to be used with his simplex lattice designs. These polynomials are obtained by modifying the usual polynomial model in \(X_i\) by using the restriction \(\sum X_i = 1\). He assumed that a polynomial function of degree ‘n’ in ‘q’ variables (i.e. \(X_1, X_2, \ldots, X_q\)) will be called a \((q, n)\) polynomial and that it will be in the form:

\[ Y = b_0 + \sum b_iX_i + \sum b_{ij}X_iX_j + \sum b_{ijk}X_iX_jX_k + \ldots + e \]  \hspace{1cm} 2.14
where:

\[ 1 \leq i \leq q, \quad 1 \leq i \leq j \leq q \quad \text{and} \quad 1 \leq i \leq j \leq k \leq q; \]

\[ \text{respectively.} \]

\[ b = \text{constant coefficient} \]

The number of terms in a \((q,n)\) polynomial is given by:

\[ \text{No of terms} = \frac{n+q}{n} \quad 2.15 \]

The condition \( \sum X_i = 1 \) enables the \( \text{nth} \) component to be eliminated; hence it reduces the number of coefficients to the following

\[ \text{No of coefficient} = \frac{n+q-1}{n} \quad 2.16 \]

This implies that the values of a \([q,n]\) polynomial can be assigned arbitrarily on a \((q,n)\) lattice and its values on a simplex are then uniquely determined.

**2.2.6 Genetic Algorithm**

The genetic algorithm created in 1975 by John Holland is a method for solving both constrained and unconstrained optimization problems that are based on natural selection, the process that drives biological evolution. (Castellani et al, 2009).

The genetic algorithm repeatedly modifies a population of individual’s selections. At each stage, the genetic algorithm selects individuals at random from the current population to be parents and uses them to produce the children for the next
generation. Over successive generations, the population 'revolves' toward an optimal solution. (Matlab, 2007).

**Table 2.3:** Comparison of Genetic Algorithm with Classic Algorithm

<table>
<thead>
<tr>
<th>S/N</th>
<th><strong>Classic Algorithm</strong></th>
<th><strong>Genetic Algorithm</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Generates a single point at each iteration. The sequence of point approaches an optimal solution.</td>
<td>Generates a population of points at each iteration. The best point in the population approaches an optimal solution.</td>
</tr>
<tr>
<td>2.</td>
<td>Select the next point in the sequence by a deterministic computation.</td>
<td>Select the next population by computation which uses random number generators.</td>
</tr>
</tbody>
</table>

**Source:** (Shahin et.al, 2005)

They are often hybridized with other search algorithms like the back propagation algorithm, to speed up the learning process.

The genetic algorithm is able to avoid being trapped into secondary peaks of performance as the case is, in back-propagation algorithm, and therefore provide effective and robust solution to the problem of automated artificial neural network design and training. (Castellani et al, 2009).

**2.3 Artificial Neural Networks**

Many authors have described the structure and operation of ANNs (Fausett, 1996). ANNs are data driven artificial intelligence approach that attempts to mimic, in a very simplistic way, the cognition capability of the human brain. ANNs learn by examples
of data inputs and outputs presented to them, so that the subtle functional relationships among the data are captured, even if the underlying relationships are unknown or the physical meaning, is difficult to explain. This is in contrast to most traditional empirical and statistical methods, which need prior knowledge about the nature of the relationships among the data. This is one of the main benefits of ANNs when compared with most empirical and statistical methods.

Typically, the architecture of ANNs consists of a series of processing elements (PEs) or nodes, that are usually arranged in the following layers: an input layer, an output layer and one or more hidden layers, as shown in Fig 2.6.

![Fig 2.6: Typical structure and operation of ANNs](image)

The input from each PE in the previous layer $x_i$ is multiplied by an adjustable connection weight $W_{ji}$. At each PE, the weighted input signals are summed and a threshold value $\theta_j$ is added. This combined input $I_j$ is then passed through a nonlinear
transfer function $f(.)$ to produce the output of the PE $y_j$. The output of one PE provides the input to the PEs in the next layer. This process is summarized in Eqns. (2.17) and (2.18) and illustrated in Fig 2.6.

Summation,  
\[ I_j = \sum W_{ji}x_i + \theta_j \]  
\[ Y_j = f(I_j) \]

The propagation of information in an ANN starts at the input layer, where the input data are presented. The network adjusts its weights on the presentation of a training data set and uses a learning rule to find a set of weights that will produce the input/output mapping that has the smallest possible error. This process is called “learning” or “training”. Once the training phase of the model has been successfully accomplished, the performance of the trained model needs to be validated using an independent testing set. The main steps involved in the development of an ANN, as suggested by Maier and Dandy (2000), are illustrated in fig 2.7. Several of these steps are discussed in some depth in the following section.
2.4 Current Development and Future Directions in Utilization of ANNS

One issue that needs to be addressed in order to improve the performance of ANN models is the utilization of a systematic approach in their development. Such an approach needs to address major factors, including the determination of adequate model inputs, data division and preprocessing, choice of suitable network architecture, careful selection of some internal parameters that control the optimization method, stopping criteria, and model validation. For example, in relation to the second step of *choice of data sets and method for data division*, Shahin et.al (2004) provided guidance using a geotechnical engineering example. He recommended the use of three, statistically consistent but independent data sets, one for each of
training, testing, and validation. In this context, Shahin et al. (2004) have introduced three approaches so that data division can be carried out in a systematic manner, including trial-and-error, self-organizing maps, and fuzzy clustering. A detailed treatment of each of the steps in the model development process, is contained in the work by Shahin et.al (2008).

Other key issues in relation to ANN modeling that have received recent attention and require further research in the future include developing approaches that:

(i.) Ensure the development of robust models.
(ii.) Increase models transparency and enable knowledge to be extracted from trained ANNs.
(iii.) Improve extrapolation ability.
(iv.) Deal with uncertainty.

Each of these is discussed below.

2.4.1 Model Robustness

Model robustness is the predictive ability of ANN models to generalize over a range of data similar to that used for model training. Kingston et.al (2005) stated that if “ANNs are to become more widely accepted and reach their full potential..., they should not only provide a good fit to the calibration and validation data,
but the prediction should also be plausible in terms of the relationship modeled and robust under a wide range of conditions,” and that “while ANNs validated against error alone may produce accurate predictions for situations similar to those contained in the training data, they may not be robust under different conditions unless the relationship by which the data were generated has been adequately estimated.” This is in agreement with the investigation into the robustness of ANNs carried out by Shahin et.al (2005) on the settlement of shallow foundations on granular soils. They found out that good performance of ANN models on the data used for model calibration and validation, does not guarantee that the models will perform well in a robust fashion over a range of data similar to those used in the model calibration phase. For this reason, they proposed a method to test the robustness of the predictive ability of ANN models by carrying out a sensitivity analysis to investigate the response of ANN model outputs to changes in its inputs. The robustness of the model can then be determined by examining how well model predictions are in agreement with the known underlying physical processes of the problem at hand over a range of inputs.
In addition, Shahin et.al (2005) advised that the connection weights should be examined as part of the interpretation of ANN model behavior, using, for example, the method suggested by Garson (1991). On the other hand, Kingston et al. (2005) adopted the connection weight approach proposed by Olden et.al (2004) hydrological modeling in their own work, Olden et al. (2004) found out that, this approach provided the best overall methodology for quantifying ANN input importance, in comparison to other commonly used methods, though with a few limitations. Support vector machines (SVMs) are an alternative data driven modeling approach that is claimed to provide better generalization capabilities and higher accuracy than ANNs and are therefore worth further consideration particularly for its improved model robustness Goh (2006) and Goh (2007) gave details of this technique. Recent applications of SVMs in civil engineering include the prediction of compressive strength of concrete (Pal, 2006), analysis of slope stability (Goh and Goh, 2007), and modeling of friction capacity of driven piles (Padmini et.al, 2008).

2.4.2 Model Transparency and Knowledge Extraction

Model transparency and knowledge extraction are the feasibility of interpreting ANN models in a way that provides insights into
how model inputs affect outputs. Fig 2.8 shows the classification of modeling techniques based on colors (Giustolisi, 2007). In the technique, the higher the physical knowledge used during model development, the better the physical interpretation of the phenomenon that the model provides to the user. The color coding of mathematical modeling, can be classified into: white-, black-, and grey-box models, each of which can be explained as follows (Giustolisi, 2004). White-box models are systems that are based on first principles (e.g., physical laws), in which model variables and parameters are known and have physical meaning, and the underlying physical relationships of the system can be explained. Black-box models are data-driven or regressive systems in which the functional form of relationships between model variables, is unknown and needs to be estimated. Black-box models rely on data to map the relationships between model inputs and corresponding outputs, rather than to find a feasible structure of the model input-output relationships. Grey-box models are conceptual systems in which the mathematical structure of the model can be derived, allowing further information of the system behavior to be resolved.
ANNs belong to the class of black-box models due to their lack of transparency and the fact that they do not consider, nor explain the underlying physical processes explicitly. This is because the knowledge extracted by ANNs is stored in a set of weights that are difficult to interpret properly, and due to the large complexity of the network structure. This issue has been addressed by many researchers with respect to hydrological engineering. For example, Jain et al. (2004) wanted to know whether or not the physical processes in a watershed, were inherent in a trained ANN rainfall-runoff model. This was carried out by assessing the strengths of the relationships between the distributed components of the ANN model, in terms of the responses from the hidden nodes, and the deterministic components of the network.
hydrological process, computed from a conceptual rainfall runoff model, along with the observed input variables, using correlation coefficients and scatter plots. They concluded that the trained ANN, in fact, captured different components of the physical process and a careful examination of the distributed information contained in the trained ANN, can be informative about the nature of the physical processes captured by various components of the ANN model. Sudheer (2005) performed perturbation analysis to assess the influence of each individual input variable on the output variable and found it to be an effective means of identifying the underlying physical process inherent in the trained ANN. (Olden et al, 2004), Sudheer and Jain (2004), and Kingston et al. (2006) also addressed this issue of model transparency and knowledge extraction.

In a particular civil engineering analysis, Shahin et.al. (2002) and Shahin and Jaksa (2005), expressed the results of the trained ANNs in the form of relatively straightforward equations. This was possible due to the relatively small number of input and output variables, and hidden nodes. Neurofuzzy applications are another means of knowledge extraction that facilitate model transparency. Neurofuzzy networks use the fuzzy logic system to store knowledge acquired from a set of input variables \( x_1, \)
and the corresponding output variable \( y \) in a set of linguistic fuzzy rules that can be easily interpreted. Examples of such applications in civil engineering include the works of Ni et.al (1996), Gokceoglu et.al (2004), Provenzano et.al. (2004), and Padmini et.al. (2008).

A recent technique that belongs to the class of grey box models, and therefore does not suffer from problem of model transparency and knowledge extraction, is genetic programming (GP). Several researchers have recently used the GP technique as an alternative to ANNs. GP is a computing method that attempts to mimic the biological evolution of living organisms. GP makes use of the principles of genetic algorithms (GAs) for parameter optimization, in which a population of expressions (or computer programs) for a function \( F \), coded in tree structures of variable size, is generated and executed. The generated expressions are then modified by means of artificial evolution in order to perform a global search needed to arrive at the best fit mathematical expression for a function, \( F \) that solves a certain problem. However, the number and combination of terms, as well as the values of GP modeling parameters, are all evolved automatically during model calibration. However, hybrid approaches, in which genetic algorithms are used to evolve optimal ANN structures
and connection weight values, can be used in modeling of properties of materials. It should be noted that while white-box models provide maximum transparency, their construction may be difficult to obtain for many civil engineering problems, because the underlying mechanism is not entirely understood.

2.4.3 Model Extrapolation

Model extrapolation is the ability of ANN models to predict well outside the range of the data used for model calibration. However, it is generally accepted that ANNs perform best when they do not extrapolate beyond the range of the data used for calibration (Flood, et.al, 1994, Tokar, et.al, 1999). Extreme value prediction is of particular concern in several areas of civil engineering, such as hydrological engineering, when floods are forecast, as well as in geotechnical engineering when, for example, liquefaction potential and the stability of slopes are assessed. Sudheer et.al (2003) highlighted this issue and proposed a methodology based on the Wilson–Hilferty transformation, for enabling ANN models to predict extreme values with respect to peak river flows.

2.4.4 Model Uncertainty

This is necessary for the assessment of the quality of model predictions. In an effort to address this, a few researchers have applied Bayesian techniques to ANN training (e.g., (Buntine,
1991; Mackay, 1992 and Kingston, 2005) in the context of hydrological engineering and Goh et.al. (2005) with respect to geotechnical engineering. Goh et.al. (2005) observed that the integration of the Bayesian framework into the back-propagation algorithm enhanced artificial neural network prediction capabilities and provided assessment of the confidence associated with network predictions. Research to date has demonstrated the value of Bayesian artificial neural networks, although further work is needed in the area of structural and material engineering. Shahin et.al. (2005) also incorporated uncertainly in the ANN process by developing a series of design charts expressing the reliability of settlement predictions for shallow foundations on cohesion less soils.

In the field of structural engineering, it is possible to encounter some types of problems that are very complex and not well understood. In this regard, ANNs provide several advantages over more conventional computing techniques. For most traditional mathematical models, the lack of physical understanding is usually supplemented by either simplifying the problem or incorporating several assumptions into the models. Mathematical models also rely on assumption structure of a model in advance which may be suboptimal. Consequently, many mathematical models fail to simulate the complex behavior
of most structural engineering problems. In contrast, ANNs are a data driven approach, in which the model can be trained on input-output data pairs, to determine the structure and parameters of the model. In this case, there is no need to either simplify the problem or incorporate any assumptions. Moreover ANNs can always be updated to obtain better results by presenting new training examples as new data become available. These factors combine to make ANNs a powerful modeling tool in structural/material engineering.

2.5 Ability of Multilayer Neural Networks to Approximate Arbitrary Functions
Cybenko (1988) has shown that two hidden layers are adequate for approximating an arbitrary function, provided there are sufficient numbers of units per layer. It was also established later that a network with only one hidden layer can represent any continuous function (Cybenko, 1989 and Hornik et.al, 1989). However, the relation between the accuracy of the approximation and the number of layers, or the number of units per layer, needs to be established. Fig. 2.9 and 2.10 illustrate the approximation (shown by dot line) of a set of function of the type $y=f(x)$, which are shown by solid line, with various multilayer architecture.

An in–depth analysis of function approximation ability of multilayer network (MLN) and back propagation (BP) was discussed by Poggio (1991).
Function Approximation by Artificial Neural Network

Fig 2.9:  

(a) \( Y = 0 \), \( X < 0 \)  
\( Y = c \), \( X \geq 0 \)

(b) \( Y = e^{-a} \sin(bX) \)
\( Y = a e^{-q(X-K)} + b e^{-r(X-m)} + c e^{-s(X-n)} \)

(b) \( Y = aX \sin(bX) \)

**Fig 2.10:** More Function Approximation by Artificial Neural Networks
2.5.1 **One – Hidden – Layer Network**

Since, it has been established that a hidden layer is sufficient to represent any continuous function; some characteristics of MLN are investigated by conducting numerical experiments on one–hidden–layer network. These include:

(a) The relationship between the number of processing elements and

(i.) Accuracy
(ii.) Training time
(iii.) Local minima
(iv.) Underfitting
(v.) The amount of training samples needed

(b) The proper network architecture for approximating a given function

(c) The effects of using different set of training samples

The network is taken as fully connected, with single element in the input and output layers, and multiple elements, in the hidden layer. The transfer function for the hidden layer element is the target–sigmoid function shown in fig. 3.3. A linear function is used for the output layer elements. The input and output data for the network, are scaled down to the range from -0.8 to 0.8, and the initial values of all weighting parameters are random
numbers ranging from -1 to 1. This choice of weights avoids saturating the transfer function in the hidden layer at the beginning of the training process.

$$f_{\beta}(h) = \frac{1}{1 + \exp(-2\beta h)}$$

Fig 2.11: Target-Sigmoid functions
Fig 2.12(a-e) contains a sequence of approximations, which are generated by networks having different numbers of interior elements, the function

$$y = 100e^{-x/10} (\sin x)$$  \hspace{1cm} 2.19

The training and testing data sets contain the values of $f$ for $x$ ranging from -10 to 10 with an interval of 0.1. The results show that, for a given number of training cycles, increasing the number of units improves the accuracy.

Fig 2.12(a) Network with 30 processing elements in the hidden layer
Fig 2.12(b) Network with 40 processing elements in the hidden layer
Fig 2.12(c) Network with 50 processing elements in the hidden layer

Fig 2.12(d) Network with 60 processing elements in the hidden layer
Underfitting is a problem associated with small networks. Fig. (2.12a-e) shows the effect of under–fitting when a network does not have sufficient parameters to fit a function. Increasing the size of the network lessens this problem. Small networks also tend to converge on local minimum of the error function, which is the situation when there is no improvement of accuracy with increasing number of training cycles, even though the network has enough parameters to fit the function. Increasing the number of units decreases the likelihood of converging to local minima, and also improve the accuracy.
Fig 2.12(a-e) and 2.13 demonstrate that the appropriate configuration of a network depends on the nature of the function to be approximated. Fig 2.12(a-e) shows that a network with 70 units in the hidden layer cannot accurately represent the function,

\[ y = 100x + x^2 + x^3 + 500 \sin(x), -10 \leq x \leq 10 \]  \hspace{1cm} 2.20

with 0.1 interval. On the other hand, fig 2.13 shows that only 10 units required to represent the function.

\[ y = 100e^{-20x^2} + 70e^{-30(x-2)^2} + 80e^{-100(x+3)^2} \]  \hspace{1cm} 2.21

for the same range and interval.

**Fig 2.13:** Numerical function approximation using a one-hidden-layer network with 70 processing elements in the hidden layer
**Fig 2.14:** Numerical function approximation using a one-hidden-layer network with 30 processing elements in the hidden layer

Next important issue is the significant of training data properly prepared training data can improve the convergence and help to avoid local minima. To demonstrate this effect, three different training data sets are used for training three network, each having 25 units in the hidden layer, to represent the function.

\[ y = 100 \sin(x), -10 \leq x \leq 10 \]  

The data interval for each training data set varies from 0.1 to 0.3, while the testing data set interval size is 0.33. As shown in fig 2.15(a-c), the network that is trained with the data having an
interval size of 0.1, which corresponds to 200 training data pairs, gives the best accuracy for this choice of test data.

Fig. 2.15a  Interval = 0.1

Fig. 2.15b  Interval = 0.2
Fig. 2.15c Interval = 0.3

**Fig. 2.15(a-c):** Numerical function approximation using a one-hidden-layer network

Fig 2.16 shows the effect of the amount of training data on the approximation accuracy, and indicates that the performance of networks increases with the amount of training data. However, there is no way of establishing the “a – priori” optimum size of the interval, or optimum size of training data set, for an arbitrary function. The proper size has to be validation method (Wahba, 1980 and Liu, 1995), which is described later in this chapter.
For pattern classification, a Boolean type representation is used to denote the categorical output. This is implemented in feed forward network by using a sigmoid transfer function in the output layer so that the output of each output unit ranges from 0 to 1. The number of output units is set equal to the number of categories that network is supposed to differentiate between, and each output unit corresponds to a specific category. The classification of a particular category is considered to occur when the output of the corresponding output unit is sufficiently close to unity, and the other outputs are sufficiently close to zero. Figure 2.17 shows four (4) classes of Gaussian distributed data, with
different standard deviations, which are used to evaluate the classification performance of various network. The symbols x, 0, + and * represent the different data classes.

**Fig. 2.17: Data for a Classification Problem**

These classes are nonlinearly separated and partially overlap. Each network has 2 input units and 4 output units. Fig 2.17 shows a classification buy the networks of different size. The performance is measured by “the percentage of correct classification” plotted in the fig. 2.18(a-d). The symbol 0 indicates that the output of the particular input did not reach the threshold level, which is set at 0.8 at every output node, and
therefore the classification cannot be mode. Generally, the threshold is allowed to be lower when the amount of training data is smaller, or when the training data is noisy.

*Fig. 2.18a* Network with 2 processing elements in the hidden layer
Fig. 2.18b  Network with 4 processing elements in the hidden layer

Fig. 2.18c  Network with 6 processing elements in the hidden layer
Fig. 2.18d  Network with 10 processing elements in the hidden layer

**Fig. 2.18(a-d): Classification result using one-hidden-layer network**

The results also show that the classification accuracy increases with the number of processing elements in the hidden layer. However as shown in fig. 2.19, the classification performance stops improving when the number of processing element is beyond a certain number.
2.5.2 Two–Hidden–Layer Network

In order to properly design a MLN for approximating an arbitrary function, the significance of the “number” of hidden layer also has to be investigated.

The importance issues as follows:

(a) For a given number of processing elements, does a two–hidden–layer model provide better accuracy than a one–hidden layer model?

(b) What is the optimum distribution of elements between layers, given the same total amount of units for the whole network?

(c) How does two–hidden–layer network perform on a given task when it is compared to one–hidden–layer network with the same number of total units?
Series of numerical simulations carried out in the past with two-hidden-layer show that:

(a) Like one-hidden-layer network, the accuracy of the network for a given amount of training sample improves with increase in the number of total elements.

(b) The smaller networks tend to underfit and converge on local minima more frequently.

(c) Two-hidden-layer requires more computation time than the one-hidden-layer of the same number of elements because it has more connections than one-hidden-layer.

Consider two networks, a one-hidden-layer network with \( m+n \) processing elements in its hidden layer, and a fully connected two-hidden-layer network with \( m \) processing elements in the first hidden layer and \( n \) processing elements in the second hidden layer. The total numerical operation required to forward propagate the one-hidden-layer network, \( TC_1 \) is given by:

\[
TC_1 = 2(m+n)M + (m+m+1)S + (m+n+1)T \quad 2.23
\]

where:

\( M \) = multiplication operation
\( S \) = summation operation
\( T \) = function transferring operation
The corresponding operation count for the two–layer network, TC2, is as follows:

$$TC2 = (m + mn)M + (m + mn + 1)S + (m + n + 1)T \text{ 2.24}$$

From Eqns. (2.23) and (2.24), it is apparent that more numerical operation is required for the two–hidden–layer network. The difference becomes even greater with increasing $m$ and $n$.

In addition to the operational cost, the time required to train the network needs to be considered. The computational time required to train two types of network using the software MATLAB on a 486X (33MHz) machine, is listed in table 2.4. One network has one–hidden–layer network with 80 total units and the other is a two–hidden–layer network with 40 units in each of its hidden layer. The results indicate that the two–hidden–layer network requires about twice as much computation time for the same amount of training cycles.

**Table 2.4:** The training time of a feed forward network

<table>
<thead>
<tr>
<th>Training cycles</th>
<th>Training Time (Secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>One–Hidden–Layer Network</td>
</tr>
<tr>
<td>500</td>
<td>587</td>
</tr>
<tr>
<td>1000</td>
<td>1155</td>
</tr>
</tbody>
</table>

2.5.3 **Optimum Network Architecture**

The data presented in the previous sections, indicates that the performance of the network for a given task is very sensitive to the architecture of the network, and therefore a method of finding the right architecture would be very useful. The cross-validation method is the most popular method (Wahba, 1980 and Liu, 1995). It is also a reliable way to select the right architecture and avoid overtraining, which will be discussed later in this section.

To perform cross-validation, the validation training data is divided into two groups, namely, a training set and a cross-validation set. The initial architecture, which usually, is the architecture with the smallest number of processing elements possible, is then trained with the training set, and then tested with both training set and cross-validation set. The error index of the network based on the training data, is monitored. As shown in fig. 2.20, the training should be stopped when the rate of change of the cross-validation error index with the number of training cycles, reverses sign, even though the training set error index, is still decreasing. More training from this point on, produces a network that is more tuned to the training data set only, instead of the whole data, and hence reduces the ability of
the network to deal with a broader range of inputs. This effect is called “overfitting” or Overtraining” (Ling, 1995)

**Fig. 2.20: Cross-Validation Method**

The cross-validation method should be carried out for every combination of network architecture and training algorithm, and the performance comparison is performed to select the most appropriate network for a particular application (given a set of data). It is also important to note that differently divided data sets (namely training set and cross-validation set) can lead to different performance for the same network. However, the difference is minimal hence the size of the training data set is large.
2.6 Radial Basis Function Network (RBFN)

The RBFN can be considered as a two–layer feedforward network that has a fixed nonlinear transformations with no adjustable parameters in the hidden layer, and linear transformations in the output (Broomhed and Lowe, 1988). As shown in fig. 2.21, the network is a fully connected feed forward network with radial basis functions as transfer functions for the interior units and linear transfer functions at the output units.

![Fig. 2.21: A Typical RBFN](image)

For a RBFN with an input vector, \( x \in \mathbb{R}^n \), and output vector \( y \in \mathbb{R} \), the output \( y \) can be described in the simplest form by the following equation.

\[
y = \sum_{i=1}^{n} c_i G_i(||x - x_i||), \quad y \in \mathbb{R}^n, \quad 2.25
\]

where \( c_i \) is the weighting parameter of the \( i \)th unit; \( x_i \) is the centre of the radial basis function \( G_i \), and \( ||\cdot|| \) is the Euclidean norm on \( \mathbb{R}^n \). The function \( F_i \), o the transfer function of the \( i \)th
unit, is a continuous function from $R^+$ to $R$ that has a maximum value at its centre and drops off rapidly away from the centre. A frequently used class of radial basis function is the Gaussian function,

$$G_i(x) = e^{-||x-x_i||^2}$$  \hspace{1cm} 2.26

Considering the RBFN in fig. 2.21, the output $y$ can also be described by the equation.

$$y = \sum_{i=1}^{n} c_i G_i,$$  \hspace{1cm} 2.27

where:

$$Z_i = G_i(\|x - x_i\|)$$

Least mean square (LMS) approach employs the sum – squared error of all $K$ input – output pairs as the global error function. The function is described by:

$$E = 0.5 \sum_{j=1}^{k} (y_j - o_j)^2$$  \hspace{1cm} 2.28

where:

$o_j$ is the expected output corresponding to $y_j$, which is the output of RBFN due to input vector $x_j$. Given the error measure, $E$, the gradient descent algorithm improves $c_i$ by changing $c_i$ by amount $\Delta c_i$, proportional to the gradient of $E$.

$$\Delta c_i = -lcoef \frac{\partial E}{\partial c_i} = lcoef \sum_{j=1}^{k} Z_i(y_j - o_j)$$  \hspace{1cm} 2.29

where:

$lcoef = a$ constant called “learning coefficient”.
If the change is made individually for each input vector $x_j$,

$$
\Delta C_i = -lcoef \cdot Z_i(y_j - o_j)
$$

which is commonly referred to as least mean square approach, or LMS rule (et.al, 1986).

After the radial basis function and the position of their centres are specified, the only adjustable parameters of the network are the weighting parameter $c_i$. Since the gradient of the error function linear to the weighting parameters, the error function of the output does not have local minima, and the parameters can be adjusted by a linear optimization procedure such as the LMS approach. This leads to optimization procedure that has very fast convergence rate (Bianchini et.al, 1995). This aspect makes RBFN an attractive alternative to the MLN with BP, which requires a time-consuming stochastic optimization procedure.

### 2.6.1 Ability of RBFN to Approximate Arbitrary Function

The ability of RBFN to approximate an arbitrary function can be proved by the regularization theory (Girosi et.al 1987; Wahba 1980; 1990), which relates the radial basis function network to probability and statistics theory. The regularization theory establishes that RBFN can approximate any continuous function within pre-specified error, if the network contains all the radial basis functions needed. However, the types of radial basis
function required for approximating an arbitrary function cannot be predetermined, and a trial and error method is needed to determine the functions.

### 2.6.2 Optimum Network Architecture

General clustering algorithm, such as K–means clustering (Krishnaiah and Kanal, 1982), are usually applied to the input data in order to position the centres of radial basis functions. The type and number of radial basis functions largely depends on the complexity of the function being approximated. The number of radial basis functions usually increases when the function, is more complex, and increases exponentially with the dimension of the input space (Girosi, 1995). Thus RBFN becomes less practical when the dimension of the input space is high. Since the type and number of radial basis functions that are required to approximate a given function cannot be predetermined, cross-validation is usually employed to identify the optimum architecture of the RBFN for a specific task (Liu, 1995). In case the centres of the radial basis functions are not predetermined and reconsidered to be adjustable, more parameters have to be considered in the optimization process. This makes the network much more adaptable, but also makes the gradient of error function non linear to the network.
parameters. In this case, a stochastic learning algorithm has to be employed, and the advantage of RBFN’s sample training vanishes.

**2.7 Performance Comparison Between MLN and RBFN**

The performance of RBFN as compared with MLN in the same investigation of a particular research work shows that:

(a) RBFN clearly requires less training and is more accurate when the appropriate transfer functions and locations of the centres are employed. Since the output of a RBFN is a linear combination of many narrow receptive fields of basis functions, only the parameters that corresponds to the output error, are adjusted during a training cycle. This is called “local training effect”. On the other hand, MLN adjusts all parameters due to an output error (global training effect), and hence reduce the effect of the previous training cycles in the process (Narrendra, 1992). Moreover, all adjustable parameters of RBFN, are linear to the gradient error function, and can be optimized by standard least square techniques, while MLN requires stochastic techniques due to its nonlinearity. Therefore, RBFN is much easier and faster to train.
However, the global effect of parameters in MLN, leads to good generalization ability of the network, which is always required as a trade off with the accuracy (Liu, 1995; Musavi et.al, 1994, Ling, 1995). RBFN may have problem with generalization when the function they approximate is highly discontinuous (Girrosi, 1993, 1995). MLN also performs better when the function is associated with high dimensional input (Narendra, 1992). Having enough a priori knowledge about input data, is crucial for selecting the configuration of RBFN, which directly reflects the performance of the network in approximating arbitrary functions. MLN does not require such knowledge, and hence is preferred when much about the function is not known. If the fixed parameters of RBFN, such as the centres of the radial basis function, become adjustable by the network's learning algorithm, the network will be less dependent on a–priori knowledge. However, the network will require a stochastic learning algorithm, which makes the training characteristic and performance of the network similar to MLN (Girosi, 1993, 1995).

(b) The results of the investigation also confirm that the cross validation method can be employed as a general procedure
for configuring both types of network. The method should be applied specially when there are factors other than the network architecture that affect the network optimization, such as training time, size and quality of training data set, and generalization ability of the training network.

2.8 Probability Framework of Artificial Neural Networks

In the past, artificial neural networks were considered to be mysterious, and lacking a theoretical foundation. However, since the late 1980’s, research has established a relationship between artificial neural network and other field such as approximation theory, and probability and statistics. In this section, feedforward artificial neural networks with back propagation training algorithm and radial basis are discussed from the perspective of these fields. The objective is to provide a better understanding of these artificial neural networks, and thus simplify the development process.

2.8.1 Probability Model of Feedforward Networks

In this subsection, a probabilistic model of a simplified feedforward artificial neural network, is described. The model demonstrates that the maximum likelihood estimation of the parameters of the probabilistic regression model of the function, is equivalent to using a one–layer feedforward network with
linear transfer function to approximate the function (Watanabe et.al, 1995).

Firstly, the basis of the gradient method used in the back propagation learning algorithm, which is normally used for optimizing a multilayer feedforward network, is described. The fig 2.22 illustrates a processing element of artificial neural network.

![Fig. 2.22: Processing Element of Artificial Neural Network](image)

The activation of the processing element with a linear transfer function is given by:

\[
\mu = W^T x
\]

2.31

where:

\( W \) = vector of the weighting parameters of the input connections

\( x \) = input vector

\( \mu \) = output.
Suppose that the set is the data obtained by random sampling of a function $f$, which belongs to some space of function $X$ defined $\mathbb{R}^d$, in the presence of noise. The objective of function approximation is to recover the function $f$, or an estimate of $f$, from the set of data $X$.

$$X = \{(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R}\}_{i=1}^N$$

Suppose a processing element is used to approximation the function, $f$ and the error function of the output is the sum-square error of the output $y$,

$$J(W) = \sum_{i=1}^{N} \frac{1}{2} (y_i - W^T x_i)^2$$

The gradient method was applied to minimize the error function $J(W)$ by finding the gradient of the function.

$$\nabla_W J(W) = -\sum_{i=1}^{N} (y_i - W^T x_i) x_i$$

And, then adjusting the parameter $W \in \mathbb{R}^d$ in the direction opposite to the gradient,

$$\Delta W_i = \rho (y_i - W^T x_i) x_i$$

For the $ith$ training data pair $(x_i, y_i)$ given an appropriate learning rate $\rho$ successive adjusting corresponding to every data pair will provide a set of parameter $W$ that minimizes $J(W)$ and estimates $f$. 
2.8.1.1 Maximum Likelihood Estimation Model

Consider a Gaussian density function

\[ f(x) = \frac{1}{\sqrt{2\pi \sigma}} e^{-\frac{1}{2\sigma^2}(x-\mu)^2}, \]

where:

\[ \mu = \text{the mean} \]

\[ \sigma^2 = \text{the variance, of data set } x. \]

This function can be viewed in different ways, depending on which parameters are considered known or unknown. By assuming \( \mu \) and \( \sigma^2 \) known, Eqn. (3.31) can be considered as:

\[ f(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2\sigma^2}(x-\mu)^2} \]

which is the probability of \( x \), given \( \mu \) and \( \sigma^2 \). Similarly, if the data set \( X \) is known, the same equation is now rewritten as:

\[ L(\mu, \sigma, x) = \frac{1}{\sqrt{2\pi \sigma}} e^{-\frac{1}{2\sigma^2}(x-\mu)^2} \]

which is the likelihood of \( \mu \) and \( \sigma^2 \) given data \( x \).

The maximum likelihood estimate of \( \mu \) and \( \sigma \), by definition, the estimated value \( \hat{\mu} \) and \( \hat{\sigma} \) that maximize \( L(\mu, \sigma; x) \). Intuitively, it corresponds to the value \( \mu \) and \( \sigma \) that best agrees with the actually observed sample.

Fig. 2.23 illustrates a system that has input \( X \in \mathbb{R}^d \), and generates output \( \mu \). Only the real output \( y \), with noise \( E \) and be
observed. Suppose that \( \mu \) can be described by a set of parameter, then

\[
W = \{w_i\}_{i=1}^{d} \quad \text{or} \quad \mu = W^T X
\]

**Fig. 2.23: A simulated system**

The sensory output of the system can be now considered as

\[
y = W^T X + E
\]

Assuming that there is Gaussian noise,

\[
f(E; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2\sigma^2}E^2}
\]

with zero mean, the density function of the output \( y \) can be described by;

\[
f(y; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2\sigma^2}(y-\mu)^2}
\]

\[
= \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2\sigma^2}(y-W^T x)^2}
\]

The objective of function approximation is to estimate the output of the system by performing the regression of the output given the data set.

\[
X = \{(x_i, y_i)\}_{i=1}^{N}
\]

which is generated by the model

If \( \sigma \) is assumed as a known variable, Eqn. (2.41) become the likelihood of \( \mu \) and \( \sigma \), given \( x_i \).
Given that $X$ is an independent, identically distributed data set, the likelihood of $W$ given $X$ is

$$L(W; X) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2\sigma^2}(y_i - W^T x_i)^2}$$  \hspace{1cm} (2.45)$$

To simplify the maximization of Eqn. (2.45), the log likelihood, is maximized instead.

$$\log[L(W; X)] = l(W; X)$$

$$= \left[\log \left(\frac{1}{\sqrt{2\pi\sigma}}\right) \right] \left(\frac{-1}{2\sigma^2}\right) \sum_{i=1}^{N} (y_i - W^T x_i)^2 \hspace{1cm} (2.46)$$

Since log function is monotonic, maximizing log of a function still maximize the function. It is interesting that maximizing the log likelihood of $W$ given $X$ is exactly the same as minimizing the sum–square error of the output,

$$J(W) = \sum_{i=1}^{N} \frac{1}{2} (y_i - W^T x_i)^2$$  \hspace{1cm} (2.47)$$
of a processing element previously described in Section 3.5.1. Using the gradient method for the maximization of Eqn. (2.46), gives:

$$\nabla_w (\sum_{i=1}^{N} (y_i - W^T x_i)^2) = -\sum_{i=1}^{N} (y_i - W^T x_i) x_i$$

$$\Delta W_i = \rho (y_i - W^T x_i) x_i$$  \hspace{1cm} (2.48)$$

where:

$$\rho = \text{learning coefficient.}$$
Comparing Eqn. (2.35) with Eqn. (2.48), shows that the back propagation learning algorithm of a processing element that approximates a function, can be viewed as that effort to find the maximum likelihood estimation of the parameters that associated to the probability model of the function (Ney, 1995).

2.8.1.2 Choice of Transfer Function: A Probabilistic View

Based on the probabilistic approach used in the previous section, this section demonstrates a technique for selecting the types of transfer function of the feed forward networks for different tasks (Watanabe et.al, 1995).

For classification problems, which the output tends to be yes or no, true or false, rather than the real number as in regression problems, it is more rational to model the problem using Bernoulli density function instead of Gaussian. Suppose a system with the input $X \in \mathbb{R}^d$ generates output $\mu$. Only the real output $y$, with some uncertainty, can be observed as shown in fig. 2.24.

![Fig. 2.24: A simulated system](image)

Fig. 2.24: A simulated system
Suppose \( \mu \) can be described by a set of parameter \( W = \{w_i\}_{i=1}^d \) as follow:

\[
\mu = f(W^T X)
\]

\[
X_i W \in \mathbb{R}^d
\]

where:

\( f = \text{an arbitrary function} \)

Using Bernoulli’s probability model, \( \mu \in (0,1) \) can be considered as the probability of success, and hence the probability of success, and hence the probability, \( P \) of the output \( y \), given the probability of success \( \mu \) can be demonstrated as;

\[
P(y; \mu) = \mu^y (1 - \mu)^{1-y}
\]

where:

\( y = 0 \text{ or } 1 \)

For an independent, identically distributed data set;

\[
X = \{(x_i, y_i)\}_{i=1}^N
\]

where:

\( x_i \in \mathbb{R}^d \)

\( y_i = 0 \text{ or } 1 \)

which is generated by the model, the likelihood of \( W \) given \( x_i \) is

\[
L(W; x_i) = \mu_i^{y_i} (1 - \mu_i)^{1-y_i}
\]

where:

\( \mu_i = f(W^T x_i) \)

Therefore, the likelihood of \( W \) given \( X \) is

\[
L(W; X) = \prod_{i=1}^N \mu_i^{y_i} (1 - \mu_i)^{1-y_i}
\]

Consider likelihood,
\[
l(W;X) = \sum_{i=1}^{N} (y_i \log(\mu_i) + (1 - y_i)\log(1 - \mu_i)) \quad 2.55
\]

the term in the summation, which is called “cross entropy”, can be perceived as a measure of closeness between \( y \) and \( \mu \).

The processing element (shown in fig. 2.22) with an arbitrary transfer function, \( f \) can be used to approximate the output of the system. The cross entropy term can be employed as the error function of the feed forward network for classification problems.

\[
J(W) = -\sum_{i=1}^{N} (y_i \log(\mu_i) + (1 - y_i)\log(1 - \mu_i)) \quad 2.56
\]

Applying the gradient method to minimize the error function \( J(W) \) gives

\[
\frac{\partial J}{\partial w_j} = -\left( \sum_{i=1}^{N} \left( \frac{y_i}{\mu_i} \frac{\partial \mu_i}{\partial w_j} - \frac{1-y_i}{1-\mu_i} \frac{\partial \mu_i}{\partial w_j} \right) \right)
\]

\[
= -\left( \sum_{i=1}^{N} \left( \frac{y_i}{\mu_i} - \frac{1-y_i}{1-\mu_i} \frac{\partial \mu_i}{\partial w_j} \right) \right)
\]

\[
= -\left( \sum_{i=1}^{N} \left( \frac{y_i-\mu_i}{\mu_i(1-\mu_i)} \frac{\partial \mu_i}{\partial w_j} \right) \right)
\]

\[
= -\left( \sum_{i=1}^{N} \left( \frac{y_i-\mu_i}{\mu_i(1-\mu_i)} f'(z_i)X_{ij} \right) \right) \quad 2.57
\]

where:

\[
\mu_i = f(W^T X_i) = f(z_i)
\]

Hence, the learning rule of the classification network is as follows:

\[
\Delta W_i = \rho \left( \frac{y_i-\mu_i}{\mu_i(1-\mu_i)} \right) f'(z_i)X_{ij} \quad , \quad 2.58
\]

The Eqn. (2.58) gives the adjustment of the parameters in the direction opposite to the gradient. The choice of the transfer
function, \( f(z) \) should now be the one that its derivation, \( f'(z) \), cancel the variance term of the Bernoulli density function, \( \mu(1 - \mu) \), from the learning rule.

If the network employs the sigmoid function,

\[
\mu = f(z) = \frac{1}{1 + e^{-z}}
\]

the derivative of the function is

\[
f'(z) = \mu (1 - \mu)
\]

Therefore, the learning rule can be reduced to

\[
\Delta W_i = \rho (y - \mu)x_i
\]

This is similar to the learning rule of the feed forward network for regression problem (see Eqn. 2.35). The equation suggests that the type of transfer function used in the output layer of the networks for different tasks, should be carefully selected in order to increase the reliability approximation.

The same probabilistic approach can also be employed to find the proper transfer function for other applications. The step by step procedure of the approach is shown below:

(i.) State the problem

(ii.) Develop a probabilistic model \( P(y|x; \theta) \), where \( y \) is the output, \( x \) is the input and \( \theta \) is the parameter.

(iii.) Formulate the log likelihood, or error functions as follows:

\[
l(\theta; x) = \sum_{i=1}^{N} \log(P(y_i|x_i; \theta))
\]
(iv.) Apply the estimation principle

\[ \hat{\theta} = \arg \max_{\theta} l(\theta, x) \]

(v.) Choose a learning algorithm

Select an optimization procedure for finding the maximum likelihood estimation of parameter \( \theta \).

By following the procedure choices of transfer function for other types of problem can be determined as shown in the following table.

**Table 2.5:** Types of transfer function for different artificial neural network application

<table>
<thead>
<tr>
<th>Types of Problems</th>
<th>Probability model</th>
<th>Error function</th>
<th>Transfer function of the output layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regression</td>
<td>Gaussian</td>
<td>Sum-squared error</td>
<td>Linear</td>
</tr>
<tr>
<td>Classification</td>
<td>Bernoulli (2-way)</td>
<td>Cross entropy</td>
<td>Logistic</td>
</tr>
<tr>
<td></td>
<td>Multinomial (multi-way)</td>
<td>Cross entropy</td>
<td>Soft max</td>
</tr>
<tr>
<td>Counting</td>
<td>Poisson</td>
<td>Cross entropy</td>
<td>Exponential</td>
</tr>
<tr>
<td>Time of failure</td>
<td>Gamma</td>
<td>Cross entropy</td>
<td>Exponential</td>
</tr>
<tr>
<td></td>
<td>Weibull</td>
<td>Cross entropy</td>
<td>Exponential</td>
</tr>
</tbody>
</table>


In order to properly design feedforward artificial neural networks with back propagation learning algorithm, a thorough understanding of how the networks have to solve, are required. Selecting the proper types of transfer function of the output layer is one important decision that can be made, given the knowledge. Although, some improperly designed networks work effectively for some problem, properly designed networks always
perform better. Therefore, it is essential that artificial neural networks users, also understand how they work and apply them properly.

2.8.2 Probabilistic Model of Radial Bases Function Networks

In this section, the relationship between the radial basis function networks (RBFN) and the probability and statistic theory, is described through the regularization theory (Girosi et.al, 1993, 1995; Bertero et.al 1988, Marroquin, 1987; Wahba, 1990). The relationship is informally shown without demonstrating the related mathematical issue.

The set given by Eqn. (2.62) is data obtained by random sampling a function, $f$, which is defined on $R^d$.

$$X = \{(x_i, y_i) \in R^d \times R\}^N_{i=1} \tag{2.62}$$

In case of noisy data, the function, $f$ can be represented as,

$$f(x_i) = y_i + E_i, \quad i = 1, \ldots, N \tag{2.63}$$

where:

$$E_i = \text{a random independent variable with a given distribution}.$$ 

A probabilistic approach is applied in order to recover the function, $f$. The function is considered as a random field with a known a–priori probability distribution. Let’s define;

- $P[f/X]$ as the conditional probability of $f$ given $X$. 

84
\[ P[X/f] \] as the conditional probability of \( X \) given \( f \). In other words, if the function \( f \) correspond to the data, this is the probability that the set of output \( \{y_i\}_{i=1}^N \) is obtained by random sampling of the function \( f \) at the data point \( \{x_i\}_{i=1}^N \).

\[ P[f] \] as the a-priori probability of the random field \( f \). \( P[f] \) covers a-priori knowledge of the function, and can be used to apply constraints on the model by assigning significant probability only to those functions that satisfy those constraints.

Assuming that the probability distributions \( P[f/X] \) and \( P[f] \) are known, the posterior distribution \( P[f/X] \) can be determined by applying Baye’s rule,

\[ P(f/X) \propto P(X/f) P(f) \] \hspace{1cm} 2.64

Assuming the noise \( E \) is normally distributed with variance, \( \sigma \), the probability of \( X \) given \( f \) is given by Eqn. (3.60)

\[ P[X/f] \propto \frac{1}{e^{2\sigma^2}} \sum_{i=1}^{N} (y_i - f(x_i))^2 \] \hspace{1cm} 2.65

where:

\[ \sigma = \text{variance of the noise}. \]

The model for the a-priori probability distribution \( P(f) \) is chosen in correspondence with the discrete case (when function, \( f \) is defined on a finite subset of an n-dimensional lattice) for which the problem can be formalized (Marroquin et al, 1987).
The a–priori probability can be written as:

\[ P[f] \propto e^{-\alpha \Phi(f)} \]  \hspace{1cm} 2.66

where:

\[ \Phi(f) = \text{the smoothness functional which will be explained in detail later} \]

\[ \alpha = \text{a position real number.} \]

This type of probability distribution provides high probability to those functions which has the term \( \Phi(f) \) small, and hence gives a prior knowledge of the system.

According to the Baye’s rule, the posterior probability of function \( f \) can be written as:

\[ P[f/X] \propto e^{\frac{-1}{2\sigma^2} \left[ \sum_{i=1}^{N} (y_i - f(x_i))^2 + 2\alpha \sigma^2 \Phi(f) \right]} \]  \hspace{1cm} 2.67

The function, \( f \) can be estimated from this probability distribution by finding the Maximum A posteriori (MAP) estimate, which considers the function that maximizes the posterior probability \( P[f/X] \), or minimizes the exponent in Eqn. (2.67).

According to (Girosi, 1993, 1995), the MAP estimate of the function \( f \) is actually the function that minimize the functional.

\[ H(f) = \sum_{i=1}^{N} (y_i - f(x_i))^2 + \lambda \Phi(f) \]  \hspace{1cm} 2.68

where:

\[ \lambda = 2\sigma^2 \alpha \]

The first term enforces closeness to the data, while the second term enforces the smoothness. The smoothness is defined by a
smoothness function $\phi(f)$, whose lower value corresponds to smoother functions. The parameter $\lambda$ is called “regularization parameter”, which is used to control the trade-off between the level of noise and the strength of the a-priori assumptions about the solution. From another perspective, the parameter also controls the compromise between the degree of smoothness and the closeness of the solution to the observed data.

The smoothness is actually a measure of the oscillatory behavior of the function. Therefore, within a class of differentiable functions, function is defined to be smoother than another if it has less oscillation. On the other hand, if the functions are considered in the frequency domain, one is smoother than another if it has less energy at high frequency (smaller bandwidth). The high frequency content of a function can be measured by high power (filtering the function, and then measuring the power ($L_2$ norm) of the result (Girosi, 1993, 1995). This leads to defining the smoothness functional as:

$$\phi(f) = \int_{R^d} ds \frac{|f(s)|^2}{\tilde{G}(s)}$$

where $\sim$ verifies the Fourier transform. $\tilde{G}$ is defined as a positive function that drops to zero as $||s|| \to \alpha$, so that $1/\tilde{G}$ is a high-pass filter is a semi-norm with a finite dimensional null space $N$ (Madych and Nelson, 1990, Dyn, 1991). There are several possible choices for the smoothness functional $\phi(f)$ that can be
written in the form of the above equation. If $\tilde{G}$ is also assumed to be symmetric, such that its Fourier transform $G$ is real and symmetric (e.g. Radial Basis functions), it can be proved that the function that minimizes the functional, $H(f)$ has the following form.

$$f(x) = \sum_{i=1}^{N} C_i G(x - x_i) + \sum_{a=1}^{k} d_a \varphi_a(x)$$ \hspace{1cm} 2.70

where:

$\{\varphi_a\}_{a=1}^{k} = $ basis in the $k$-dimensional null space $N$, and the coefficients $d_a$ and $c_i$ satisfy the linear system.

$$(G + \lambda I)c + \psi^T d = y, \quad \psi c = 0$$ \hspace{1cm} 2.71

where:


The existence of the solution of his linear system is guaranteed by the existence of the variation problem (Girosi, 1990).

The approximation procedure of this type can also be shown as network with one hidden layer, and this type of network is called “Regularization Network”, which it smoothness function satisfy the condition.

$$\phi[f(x)] = \phi[f(Rx)]$$ \hspace{1cm} 2.72

For any rotation matrix $R$, is also classified in this category. This choice of smoothness function indicates that the a–priori assumption assumes equal relevancy of all variables, and no privilege directions. There are many radial basis functions that
satisfy these conditions. For example, if the approximation scheme employs the smoothness function of the form.

$$\phi(f) = \int_{R^d} d_s e^{\frac{\|s^2\|^2}{\beta}} |\tilde{f}(s)|^2$$  \hspace{1cm} 2.73

where:

$$\beta = \text{fixed positive number, a Gaussian function}$$

$$\tilde{G}(s) = e^{-\|s^2\| \beta}$$  \hspace{1cm} 2.74

is considered as the basis function of the approximation scheme,

$$f(x) = \sum_{i=1}^{N} C_i G(x - x_i) + \sum_{a=1}^{k} d_a \varphi_a(x)$$  \hspace{1cm} 2.75

Since the Gaussian function is positive definite, Eqn. (2.75) can be reduced to Eqn. (2.76)

$$f(x) = \sum_{i=1}^{N} C_i G(x - x_i)$$  \hspace{1cm} 2.76

which is the mathematical form of the radial basis function network (Poggio and Girosi, 1989; Yuille and Grzywacz, 1988).

2.9 **Artificial Neural Network Method of Optimization**

Artificial Neural Network, from the artificial intelligence family, is a type of information processing system based on modeling the neural system of the human brain (Sathyabalan et.al, 2009). An artificial neural network is an information procession system that has certain performance characteristics in common with biological artificial neural networks (Fausett, 1994). Artificial neural networks have been developed as generalization of
mathematical models of human cognition or neural biology, based on the assumption that;

(i.) Information processing occurs at many simple elements called neurons.

(ii.) Signals are passed between neurons over connected links.

(iii.) Each connected link has an associated weight, which in a typical artificial neural network, multiplies the signal transmitted.

(iv.) Each neuron applies an activation function, (usually non-linear) to its net input (sum of weighted input signal) to determine its output signal.

According to Haykin (2009), ‘Artificial neural network is a massively parallel distributed processor that has a natural propensity for storing experiential knowledge and making it available for use’. It resembles the brain in two respects.

(i.) Knowledge is acquired by the network through a learning process.

(ii.) Inter-neuron connection strengths known as synaptic weights are used to store the knowledge.

In practice, artificial neural networks can compute any computable function i.e they can do everything a normal digital computer can do or perhaps even more, under some
assumptions of doubtful practicality. Artificial neural network (ANN) like people, learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons. This is true of ANN as well.

Artificial neural network is characterized by:

(a) Its pattern of connections between the neurons (called its architecture).

(b) Its method of determining the weights on the connections (called its training or learning algorithm)

(c) Its activation function (usually non-linear).

Artificial neural network consists of a large number of simple processing elements called neurons, units, cells or nodes. Each neuron is connected to other neurons by means of directed communication links, each with an associated weights. The weights represents information being used by the net to solve a problem (Razavi et.al, 2011). Artificial neural network can be applied to a wide variety of problems, such as storing and recalling data or pattern, classifying pattern, performing general mapping from input to output pattern, grouping similar patterns or finding solutions to constrained optimization problem. Each
neuron has an internal state, called its activation or activation level, which is a function of the input it has received. Typically, a neuron sends its activation as a signal to several other neurons. It is important to note that a neuron can send only one signal at a time, although that signal is broadcast to other several other neurons. For example, from the figure below;

![Fig 2.25: A Simple Artificial Neuron](image)

Consider a neuron Y that receives input from neurons $X_1$, $X_2$ and $X_3$ respectively. The weights on the connections from $X_1$, $X_2$ and $X_3$ to neuron Y are $W_1$, $W_2$ and $W_3$ respectively. The net input to neuron $Y(y_{-in})$ is the sum of the weighted signals from neurons $X_1$, $X_2$ and $X_3$ i.e

$$Y_{-in} = W_1X_1 + W_2X_2 + W_3X_3$$  \hspace{1cm} 2.77

The activation of neuron Y given as y will be a function of the sum of the weighted signal from $X_1$, $X_2$ and $X_3$ into Y.

Therefore: $Y = f(y_{-in})$
This function is known as an activation function. Example is the logistic sigmoid function (an s-shaped curve) or any of a number of other activation functions.

\[ f(x) = \frac{1}{1 + \exp(-x)} \]

Now suppose further that neuron Y is connected to neuron \( Z_1 \) and \( Z_2 \), with weight \( V_1 \) and \( V_2 \) respectively as shown below;

![Diagram of a simple artificial neural network](image)

**Fig 2.26: A very simple artificial neural network**

Neuron Y will send \( y \) to each of these units i.e \( Z_1 \) and \( Z_2 \) respectively. However, the values received by neurons \( Z_1 \) and \( Z_2 \) will be different, because each signal is scaled by the appropriate weights, \( V_1 \) and \( V_2 \).

Although, the artificial neural network shown above is very simple, the presence of a hidden unit, together with a non-linear activation function, gives it the ability to solve many more problems than can be solved by a net with only input and output units. On the other hand, it is more difficult to train (i.e find optimal values for weights) a net with hidden units. A trained
artificial neural network is one which presents a large number of examples and from these examples acquire the mapping from input to output (Garrett et.al, 1997).

One does not program artificial neural network as we are used to programming a computer. To program a conventional computer, we write a computer program consisting of a sequence of steps for the computer to follow. Rather to program artificial neural network one presents it with examples of the computation to be performed. These examples consist of collections of input and output patterns, where the patterns are represented as patterns of activation. The programming of artificial neural network is embodied within the connection strengths, the topology of the network, and the mechanism of activation propagation. Artificial neural network has the ability to modify its own weights i.e to self organize, that makes neural computing feasible. For it would be impossible to set the connection strength manually for all but the simplest of artificial neural networks. In addition, self organization leads to the observed artificial neural network characteristics of robustness and ability to generalize. Garrett et.al (1997) stated in their work that by modifying the connection strengths between processors, artificial neural networks can automatically create internal features that;
(1.) Might not be apparent form the data and thus would have defiled the manual setting of connection strength.

(2.) Can be recognized in previously unseen input patterns and used to produce appropriate patterns of output.

2.10 Artificial Neural Networks for Function Approximation

Physical system modeling can be divided into two activities, performing regression to predict the system behavior, and performing classification to identify changes of behavior. Regression involves mapping a numerical domain input to another numerical domain output, while classification maps a numerical or categorical domain input to categorical domain output. Assuming that there is a function $F$ that can perform regression or classification for the purpose of physical system modeling, approximating $F$ can be considered a problem of system modeling.

As shown earlier in the previous chapter, networks that perform well in regression and classification are of the feed forward type such as multilayer feed forward and radial basis function models. Although recurrent networks can also be used for classification, they are not considered here since they are based in unsupervised learning, whereas prediction of elastic modulus
of concrete requires supervising learning. In this chapter, the performance of multilayer feed forward network for function approximation, a popular application civil engineering, is discussed in detail. The performance of radial basis function networks is also investigated for the purpose of providing a comparison. Extensive performance test are presented to show the relative advantages and disadvantages of the multilayer feed forward network versus radial basis function networks, and to identify their limitations for function approximation.

2.11 The Multilayer Feed Forward Network with Back Propagation Learning Algorithm (MLP With BP)

A network consists of an input layer, an output layer. And at least one hidden layer. Each layer is fully connected to its neighboring layers. Fig 2.27 illustrates the connectivity and the notation used to described how a processing element operates (Hertz et.al, 1991).
Fig 2.27: Feed forward network

A superscript in square brackets is used to indicate the layer being considered.

\[ X_j^{[s]} = \text{The current output of the } j^{th} \text{ processing element in layer } s. \]

\[ W_{ji}^{[s]} = \text{The weight for the connection joining the } i^{th} \text{ processing element in layer (s-1) to the } j^{th} \text{ processing element in layer } s. \]
\( I_j^s = \) The weighted summation of the inputs to the \( j \)th processing element in layer \( s \).

For each processing element,

\[
x_j^s = f \left[ \sum_i W_{ji}^s \cdot x_i^{[s-1]} \right]
\]

\[
= f \left( I_j^s \right)
\]

where:

- \( f \) is usually a differentiable monotonic function

For example, \( f \) may be defined as a sigmoid function,

\[
f(z) = (1 + e^{-z})^{-1}
\]

Where:

- \( z = \) variable to be processed in the network

The initial step involves defining a global error function, \( E \), for the network, (Rumelhart et.al, 1986). This function is required to be a continuous function of all the connection weights. It is used to define the local errors at the output layer so that they can be propagated back through the network. The parameter that is propagated back through the layers is defined as:

\[
e_j^s = -\partial E / \partial I_j^s
\]

It will be shown later that \( e_j^s \) can be considered as a measure of the local error at the processing element \( j \) in layer \( s \). Applying the chain rule to evaluate the derivative in Eqn. (2.81), one obtains:
\[
e_j^{[s]} = - \frac{\partial E}{\partial l_k^{[s+1]}} \frac{\partial l_k^{[s+1]}}{\partial f(l_j^{[s]})} \frac{\partial f(l_j^{[s]})}{\partial l_j^{[s]}} \\
= e_k^{[s+1]} \frac{\partial \sum_i (w_{ki}^{[s+1]} \cdot x_i^{[s]})}{\partial x_j^{[s]}} f'(l_j^{[s]}) \\
= f'(l_j^{[s]}) \sum_k [e_k^{[s+1]} \cdot w_{kj}^{[s+1]}] \quad 2.82
\]

This Eqn. (2.82) defines the relationship between the local error at a particular processing element at level \(s\) and the local error at processing elements at level \(s+1\). Note that Eqn. (2.82) only applies to the hidden layers. When \(f\) is taken as the sigmoidal function defined by Eqn. (2.80), its derivative is given by:

\[
f'(z) = f(z) \cdot (1 - f(z)) \quad 2.83
\]

Substituting Eqn. (2.82) into Eqn. (2.83) gives:

\[
e_j^{[s]} = x_j^{[s]} \cdot (1 - x_j^{[s]}) \cdot \sum_k [e_k^{[s+1]} \cdot w_{kj}^{[s+1]}] \quad 2.84
\]

Eqns. (2.79) and (2.84) are the back propagation learning algorithm. The process starts with a forward propagation of the input through the layers to the output layer using is determined, and the error is then propagated back through the network from the output layer using Eqn. (2.79). Next, the error at the output layer determined, and the error is then propagated back through the network from the output layer using Eqn. (2.84), or more generally Eqn. (2.82).

The goal of learning is to minimize the global error, \(E\), of the network by adjusting the weights. Given an initial set of weights,
$W_{ij}^{[s]}$, a procedure for modifying the weights based on a minimization scheme, can be applied. For example, gradient descent leads to the following “correction” equation:

$$\Delta W_{ji}^{[s]} = -lcof \cdot \left( \partial E / \partial W_{ji}^{[s]} \right)$$  \hspace{1cm} \text{2.85}$$

where:

$lcof$ = a “learning coefficient”.

Each weighting parameter is adjusted according to the size and direction of the negative gradient in the error surface. Therefore, successive “update” will lead to the minimum of error surface.

The partial derivations in Eqn. (2.85) can be calculated directly from the local error values previously discussed (see Eqn. (2.81)). Using the chain rule and Eqn. (2.79), Eqn. (2.86) is obtained:

$$\frac{\partial E}{\partial W_{ji}^{[s]}} = \left( \frac{\partial E}{\partial I_{j}^{[s]}} \right) \cdot \left( \frac{\partial I_{j}^{[s]}}{\partial W_{ji}^{[s]}} \right)$$

$$= -e_j^{[s]} \cdot x_i^{[s+1]}$$  \hspace{1cm} \text{2.86}$$

Combining Eqns. (2.85) and (2.86) leads to:

$$\Delta W_{ji}^{[s]} = lcof \cdot e_j^{[s]} \cdot x_i^{[s+1]}$$  \hspace{1cm} \text{2.87}$$

Assuming input vector $\underline{P}$ is presented at the input layer of the network, and the target output $\underline{t}$ is defined and $\underline{o}$ is the actual output vector produced by the network with its current set of weights, then a measure of the global error ‘$E$’ in producing the desired output, can be taken as:
\[ E = 0.5 \sum_k (t_k - O_k)^2 \]  

where:

Subscript \( k \) = index of the component of \( t \) and \( o \).

The Eqn. (2.88) defines the sum–squared error of the network in producing the desired output throughout the range of the input range. From Eqn. (2.88), the local error at each processing element of the output layer can be determined from the following expression:

\[
e_k^{[o]} = -\frac{\partial E}{\partial I_k^{[o]}} = -\frac{\partial E}{\partial O_k} \cdot \frac{\partial O_k}{\partial I_k^{[o]}} = (t_k - o_k) \cdot f'(I_k) \]

The Eqn. (2.89) can then be propagated back to inner layer using Eqn. (2.84). In some cases, this type of learning algorithm requires a very long training time in order to reach the minimum. For example, when the surface corresponding to the cost function has the shape of a valley with steep sides, and a shallow slope on the valley floor, there may be oscillation of the adjusted parameters across the valley during the learning process, and consequently very little movement downward the slope toward the minimum. The addition of a momentum term can make gradient descent avoid this problem and improve the speed of convergence (Plaut et.al, 1986). The strategy is to provide the momentum, so that the adjustment is in the
direction of the average downward force, instead of oscillating back and forth cycle. A layer learning rate can then be used since there is less potential for oscillation. This objective is achieved by including the effect of learning achieved by including the effect of learning from the previous learning cycle in the present learning from the previous learning the effect of learning cycle. The following equation illustrates this process.

$$
\Delta W_{ji}^{(s)}(t + 1) = -lcoef \times \left( \frac{\partial E}{\partial W_{ji}^{(s)}} \right) + \alpha \Delta W_{ji}^{(s)}(t) \tag{2.90}
$$

The momentum parameter $\alpha$ must be between 0 and 1; a value of 0.9 is usually used.

Identifying the appropriated value for the learning coefficient, $lcoef$, for specific application is also a challenge. In most cases, the coefficient is modified as the training progresses. A number of researchers (Cater, 1987; Franzini, 1987; Vogl et.al, 1988; Jacobs, 1988; Baldi, et.al, 1995) have suggested ways to adjust this parameter. The general approach is to observe the direction of the learning path on the error surface and adjust $lcoef$ to improve the rate of convergence. For example, when $n$ successive values of $W_{ji}^{(s)}$ are in the same direction, $lcoef$ in increased. However, when successive values of $W_{ji}^{(s)}$ are in the opposite direction, $lcoef$ is decreased to dampen the oscillation. This process is defined by:
\[
\Delta W_{ji}^{[s]} = -(\text{coef} + \Delta \text{coef}) \cdot (\frac{\partial E}{\partial W_{ji}^{[s]}})
\]

\[
\Delta \text{coef} = +a \cdot \text{coef} \text{ if } \Delta E < 0 \text{ for } n \text{ successive training cycle,}
\]

\[
= -b \cdot \text{coef} \text{ if } \Delta E > 0 \text{ for } m \text{ successive training cycle,}
\]

\[
= 0 \text{ otherwise}
\]

where:

\[
\Delta E = \text{the cost function change, and } a \text{ and } b \text{ are appropriate constants.}
\]
CHAPTER THREE
MATERIALS AND METHODS

3.1 Materials

The materials used for this research work includes:

(i.) Cement
(ii.) Aggregates
(iii.) Water

3.1.1 Cement

Cement may be defined as adhesive substances capable of uniting fragments or masses of solid matter to a compact whole (Ghosh 1983). Cement functions by forming a plastic paste when mixed with water which develops rigidity (sets) and steadily increases in compressive strength (hardness) by chemical reaction with water (hydration).

Dangote brand of Ordinary Portland Cement was used in this research work. It meets the requirements of BS12 (1978). Chemically, its content in mass fraction is given as 20.39 of silicate (SiO₂), 6.03 of alumina (Al₂O₃), 67.62 of lime (CaO), 1.31 of Magnesium oxide (MgO), 2.29 of Iron oxide, 0.84 of potassium oxide (K₂O), 0.20 of Titanium oxide (TiO₂), 2.8 of loss on ignition having a total of 98.98 mass fraction (Yahaya, 2009).
3.1.2 Aggregates

Aggregates used are classified as fine and coarse aggregates. The fine aggregate are those that pass through the 4.75mm BS (British Standard) test sieve, while those that are retained on the 4.75mm BS sieve are referred to as coarse aggregate. Aggregates strongly influence concrete’s freshly mixed and hardened properties, mixture proportions, and economy. Consequently, selection of aggregates is an important process. Although some variation in aggregate properties is expected, characteristics that are considered includes:

- Grading
- Durability
- Particle shape and surface texture
- Unit weights and voids
- Absorption and surface moisture

3.1.3 Fine Aggregate

This was obtained from Otamiri river in Owerri the capital city of Imo State and the particle sizes distribution conforms with the grading requirements of BS 882 part 2 (1973). The fine aggregate was thoroughly cleaned and any deleterious materials likely to affect the properties of concrete removed.
3.1.4 Coarse Aggregate

Crushed stone obtained from quarry site in Ishiagu in Ebonyi State were used as coarse aggregate in the study. The coarse aggregate was graded according to BS 882 part 2 (1973). The aggregate conforms to the requirement of the nominal size of well graded aggregate.

3.1.5 Water

Water is an important constituent in concrete. It chemically reacts with cement (hydration) to produce the desired properties of concrete. Mixing water is the quantity of water that comes in contact with cement, impacts slump of concrete and it is used to determine the water to cementitious material ratio (w/c) of the concrete mixture. Strength and durability of concrete is controlled to a large extent by its (w/c). Mixing water in concrete includes batch water measured and added to the mixer at the batching plant, free moisture on aggregate and water absorbed by aggregates is excluded from mixing water. The water used for mixing of concrete is potable water, fit for human consumption.

3.2 Constituents of Concrete Mix

The proportion of fine aggregate to coarse aggregate is found out by trial, which gives the maximum weight of the combined
aggregate. The density of particle of fine aggregate and coarse aggregate is nearly the same, the mixture which will give the maximum weight will have the maximum solid matter therefore the least voids and surface area. In another approach, the fine aggregate and coarse aggregate are combined in several proportions. The quantity of cement per unit volume of concrete is found to give a certain water-cement ratio for each mixture. The percentage of sand is known as optimum percentage.

Cement required will be more for the same consistency if sand used is more than optimum. The mix will become harsh for low quantity of sand. Optimum percentage of sand is lower for lower water-cement ratio. The water cement ratio theory states that for a given combination of materials and as long as workable consistency is obtained, the strength of concrete at a given age depends on the w/c ratio. The lower the w/c ratio, the higher the concrete strength. Whereas strength depends on w/c ratio, economy depends on the percentage of aggregate present that would give a workable mix. The aim here was to get concrete mixtures of optimum strength at minimum cement content and acceptable workability which will in-turn result to higher modulus of elasticity.
3.2.1 Weighing of Aggregates

Two types of aggregates were weighed, dried and well proportioned before the test, the drying process was by air-drying, a chemical balance scale was used and various tests like specific gravity, bulk density, silt content, and dry density were carried out on aggregates, sieve analysis test was done on both aggregates.

3.2.2 Workability

The behavior of green or fresh concrete from mixing up to compaction depends mainly on the property called “workability of concrete”. Workability of concrete is a term which consists of the following four partial properties of concrete namely:

(i.) Mixability

(ii.) Transportability

(iii.) Mouldability

(iv.) Compactibility

In general terms, workability represents the amount of work which is to be done to compact the concrete in a given mould. The desired workability for a particular mix depends upon the type of compaction adopted and the complicated nature of
reinforcement used in reinforced concrete. A workable mix should not segregate. The partial properties of workability are discussed below:

**3.2.2.1 Mixability**

It is the ability of the mix to produce a homogeneous green concrete from the constituent materials of the batch, under the action of the mixing forces. A less mixable concrete mix requires more time of mixing to produce a homogeneous and uniform mix.

**3.2.2.2 Transportability**

Transportability is the capacity of the concrete mix to keep the homogeneous concrete mix from segregating during a limited time period of transportation of concrete, when forces due to handling operations of limited nature act. Any segregation that is caused during the remaining operations that follow.

**3.2.2.3 Mouldability**

It is the ability of the fresh concrete mix to fill completely the forms or moulds without losing continuity or homogeneity under the available techniques of placing the concrete at a particular
job/ this property is complex, since the behavior of concrete is to be considered under dynamic conditions.

### 3.2.2.4 Compactibility

Compactibility is the ability of concrete mix to be compacted into a dense, compact concrete, with minimum voids, under the existing means of compaction at the site. The best mix from the point of view of compactibility should close the voids to an extent of 99% of the original voids present, when the concrete was placed in the moulds.

### 3.2.2.5 Factors Affecting Workability

Workable concrete is the one which exhibits very little internal friction between particle and particle or which overcomes the frictional resistance offered by the formwork surface or reinforcement contained in the concrete with just the amount of compacting efforts forthcoming.

The factors helping concrete to have more lubricating effect to reduce internal friction for helping easy compaction are given below:

(a) Water content
(b) Size of aggregates
(c) Surface texture of aggregate
(d) Use of admixtures
(e) Mix proportions
(f) Shape of aggregates
(g) Grading of aggregates

3.3 Tests Carried Out

Various tests were carried out during this project on concrete and aggregate. These tests were carried out at Grants Engineering Materials laboratory Port Harcourt, River State. The aim of analysis was to check the adequacy and properties of material used against the recommended standards.

The tests carried out were:

(i.) Bulk density test
(ii.) Slump test
(iii.) Silt content test on fine aggregate
(iv.) Specific gravity of coarse aggregate
(v.) Compressive strength test
(vi.) Sieve test

The details of the tests carried out are explained below:

3.3.1 Bulk Density

Bulk density was carried out for both coarse and fine aggregate and also on hardened concrete. The fine and coarse aggregates
were filled into a mould of known volumes respectively in three layers using a scoop and each layer was compacted 25 time with a tamping rod and the top of the mould was then leveled into the rod. After this, the filled moulds were taken to the weighing machine, weighed and their mass recorded, note, the empty mould was first weighed before filling them with the aggregates. The bulk densities for sand and granite were obtained from the following expression:

\[
\text{Bulk density} = \frac{\text{Weight or mass sand}}{\text{volume of mould}}
\]

3.1

\( W_1 = \text{Weight of empty mould (g)} \)

\( W_2 = \text{Weight of mould + Aggregates (fine or coarse)} \)

\[
\text{Bulk density} = \left( \frac{W_2 - W_1}{V} \right) \times 10^{-6} \text{kg/m}^3
\]

3.2

For dry/hardened concrete,

\[
\text{Bulk density} = \frac{\text{Weight or mass sand}}{\text{volume of mould}} \times 10^{-6} \text{kg/m}^3
\]

3.3

This was done for all the cubes and the results were recorded.

### 3.3.2 Slump Test

Unsupported fresh concrete flows to the sides and sinking in height takes place. This vertical settlement is known as slump. In this test fresh concrete is filled into a mould of specified shape and dimensions, and the settlement or slump is measured when supporting mould is removed. Slump increases as water-content
is increased. For different works different slump values have been recommended.

The slump is a measure indicating the consistency or workability of cement concrete. It gives an idea of water content needed for concrete to be used for different works. A concrete is said to be workable if it can be easily mixed, placed, compacted and finished. A workable concrete should not shown any segregation or bleeding. Segregation is said to occur when coarse aggregate tries to separate out from the finer material and a concentration of coarse aggregate at one place occurs. This results in large voids, less durability and strength.

Bleeding of concrete is said to occur when excess water comes up at the surface of concrete. This causes small pores through the mass of concrete and is undesirable. By this test we can determine the water content to give specified slump value. In this test water content is varied and in each case slump value is measured till we arrive at water content giving the required slump value.

**Apparatus:**

Iron pan to mix concrete, slump cone, spatula, trowels, tamping rod and graduated cylinder.
Fig 3.1: Slump test apparatus

Procedure:

Four mixes were prepared with water-cement ratio (by mass) of 0.50, 0.60, 0.70 and 0.80, respectively, and for each mix take 10 kg of coarse aggregates, 5kg of sand and 2.5kg of cement with each mix proceed as follows:

(a) Mix the dry constituents thoroughly to get a uniform colour and then add water

(b) Place the mixed concrete in the cleaned slump cone mould in 4 layers, each approximately ¼ of the height of the mould. Tamp each layer 25 times with tamping rod distributing the strokes in a uniform manner over the cross-section of the mould. For the second and subsequent layers the tamping rod should penetrate in to the underlying layer.

(c) Strike off the top with a trowel or tamping rod so that the mould is exactly filled.
(d) Remove the cone immediately, raising it slowly and carefully in the vertical direction.

(e) As soon as the concrete settlement comes to a stop, measure the subsidence of concrete in mm which will give the slump.

A slump cone of 300mm height was placed on a tray, with the cone pressed upon the tray the concrete mix is loaded into it, using the scoop. The mix was placed in four layers, with each layer compacted (with tamping rod of 16mm diameter and 350 mm height) 25 times. After compaction, it was allowed to overfull the slump cone and then the tamping rod was used to level it. A stop watch was started, and immediately the cone was removed from the concrete by raising it slowly and carefully in a vertical direction. In 10 seconds. This allowed the concrete to subside thus showing the difference in altitude between the height of the mould and that of the highest point on the subsided concrete. The slump which is the subsidence of the concrete was measured. Using a 30cm (300mm) steel rule, and then recorded. The type of slump was also taken into account. This procedure was repeated for all the other mixes.
3.3.3 Silt Content Test on Fine Aggregate

The process entails adding 1 percent solution of common salt (NaCl) in water. Place 50ml of the solution in 250ml mercury cylinder. Add sand gradually until it levels to 100ml and solution added until the total volume of the mixture in the cylinder is 150ml. Cover the cylinder with Pam and shake vigorously repeatedly turned upside down and then allow to stand for about 3 hours, the clay and silt content will settle above the sand and the height of this layer in mm can be expressed as the percentage of the sand below:

\[
\frac{\text{Thickness of silt/clay}}{\text{Height of sand below}} \times 100\% \quad 3.4
\]

3.3.4 Specific Gravity Determination and Coarse Aggregate

**Apparatus:**

Specific gravity bottle

Weighing balance

**Procedure:**

According to BS1881, part 103, six specific gravity bottles were used for the experiment, three for coarse aggregate and the other three for fine aggregate sample. The mass of the specific gravity bottles were measured for the fine and coarse aggregate sample. Their masses were recorded as \( M_1 \), the bottles for coarse
aggregate samples were filled with 200g sample of the coarse aggregates. The same was done for the fine aggregate and their masses were measured and recorded as $M_2$. The bottles already containing the various aggregate samples were filled with water to the brim and their masses measured and recorded as $M_3$. The bottles were then emptied and filled with water alone and their masses were measured and recorded as $M_4$.

The volume of the aggregate was calculated as:

$$(M_4 - M_1) - (M_3 - M_2)$$

Finally the specific gravity $GS$ of the samples was calculated as follows:

$$GS = \frac{M_2 - M_1}{(M_4 - M_1) - (M_3 - M_2)}$$

For each sample of the two aggregates and then their average was taken as the specific gravity of fine and coarse aggregates respectively.

**3.3.5 Compressive Strength Test**

The concrete cubes, after it has been removed from the curing tank and allowed to dry, were weighed and then taken to the compressive strength machine for crushing. The cubes were loaded one after the other into the machine and then it was
crushed at a constant load rate. At the end the stress in N/mm$^2$ and load at failure were recorded.

![Diagram](image)

**Fig 3.2:** The making of the test specimens and the gauge length $i$

### 3.3.6 Sieve Test

An air dried sample of the aggregate was prepared by quartering, and an appropriate weight of 1000g and 500g of coarse and fine aggregates were weighed. This material was poured into stack of sieves of appropriate aperture starting from the largest down to pan and was shaken. The mass of coarse and fine aggregate retained in each sieve at the end of the operation was weighed and recorded. The cumulative percentage by weight passing each of the sieves was calculated to the nearest whole number and the percentage by weight of total sample passing one sieve calculated to the nearest whole number.
3.4 Concrete Mix Design

The determination of the mix ingredients and their proportions is referred to as mix design, the selection of mix proportion is simply, the process of choosing suitable ingredients of concrete and determining their quantities with the aim of producing as economically as possible, concrete with certain minimum properties including compressive strength, flexural strength, durability and required consistency. Mix design tries to find appropriate proportions of cement, fine and coarse aggregate of particular water-cement (W/C) ratio. Majid (1974) stated that compressive strength is the most convenient property to measure among the properties of concrete. Concrete is known to be made up of cement, Water, fine and coarse aggregates. Sometime a fifth ingredient is added to enhance the concrete properties for better performance.

3.5 Laboratory Tests

3.5.1 Specimen for Determination of Elastic Modulus of Concrete

Following the provision of BS EN 1352:1997, the modulus of elasticity (E-modulus) is determined on cylindrical test specimen taken from prefabricated components. It is calculated from the difference of longitudinal compressive strains corresponding to
the increase of longitudinal compressive stress from the basic test stress $\sigma_1$ (approximately 5% of the declared compressive strength of the concrete) to the upper test stress $\sigma_2$ (in general one-third of the declared compressive strength of the concrete).

In all, a total of 15 cylindrical specimens measuring 150mm in diameter and 300mm in length were crushed to obtain values of elastic modulus used in this work.

**Apparatus:**

A steel mould measuring 150mm diameter and 300mm height with joints of the mould coated with wax and inside surface oiled to facilitate quick release of concrete specimen.

(a) *Any saw*, suitable for cutting reinforced concrete component.

(b) *Callipers*, capable of reading the dimensions of the test specimens to an accuracy of 0.1mm;

(c) *A straight-edge*.

(d) *A balance*, capable of determining the mass of the test specimens to an accuracy of 0.1%;

(e) *A compression testing machine*, which meets the requirements of ISO 4012:1978, capable of applying the required load at the specified rate and maintaining it at the required level for at least 60 seconds.
(f) *Equalizing layers of soft fibreboard*, with a thickness of (12±2) mm and a density of (250 to 400) kg/m³ was inserted between the load bearing surfaces of the test specimens and the platens of the compression testing machine. A *ventilated drying oven*, capable of maintaining a temperature of (105 ± 5)°C (see note);

(g) *Gauges*, for determining the longitudinal deformations or strains with a gauge length suitable to determine the strains to an accuracy of 5 X 10⁻⁶ (e.g. inductive displacement transducers, dial gauges, mirror extensometers, strain gauges etc.).

### 3.5.2 Test Specimens

#### 3.5.2.1 Preparation of Specimens

(a) A predetermined proportions of concrete materials were measured out and mixed manually using shovel. Sharp sand was deposited on an impermeable surface before cement was added. These two materials were thoroughly mixed to homogenous texture before granite was added. Measured quantity of water was added to make the concrete mix workable.

(b) The steel mould was lightly oiled to facilitate easy removal of concrete specimen.
(c) The mixed concrete was introduced into the mould in four layers and a steel rod 25mm diameter and 500mm high was used to temper each layer of concrete for 25 times.

(d) Trowel was then used to level off the top of mould and each sample was distinguished with identification mark.

(e) The test specimens were de-molded after 24 hours and stored in water curing tank at room temperature and left to cure for 28 days. The test specimens were removed from water not earlier than 30 mins prior to test and the surfaces was allowed to dry.

3.5.3 Testing Procedure

3.5.3.1 Deformation Meter

A strain gauge for the determination of changes in length of the specimen under load with a gauge length L was attached to the specimen on two opposite sides as shown in fig. 3.2. The accuracy of measurement was within ±25x10^-6.

3.5.4 Compressive Strength of Concrete Specimen

Mean compressive strength (fcm) of 14 specimens made from same concrete mix used for the preparation of samples for elastic modulus determination was obtained by loading the specimens to failure point.
3.5.5 Measurement of Modulus of Concrete

The test specimen with deformation meter was placed centrically in the compressive testing machine within an accuracy of ±1mm. The test specimen was then loaded with basic stress of $\sigma_0 = 0.5\text{MPa}$ and a reading was taken on the deformation meter. Incremental stress was continuously applied in a controlled manner at the rate of $0.8\pm0.2\text{MPa/s}$ until stress of $\sigma_1 = 0.5f_{cm}$ is reached and it is maintained for 60 seconds where after the deformation meter was again read during the following 30s. The strain $\varepsilon_{01}$ from the stress $\sigma_0$ to $\sigma_1$ was calculated.

The test specimen was re-loaded until $\sigma_2 = f_{cm}/3$ is reached, and it was then off-loaded at the same rate as was used earlier until the basic stress of $\sigma_0$ is reached while $\sigma_2$ and $\sigma_0$ are kept constant during 60 seconds. The process was repeated once. The deformation meter was now read when the stress $\sigma_2$ and $\sigma_{02} = 0.5\text{MPa}$ after a pause of 60 seconds for each respective load. The strain $\varepsilon_{02}$ resulting from the stress $\sigma_2$ to $\sigma_{02}$ was calculated.

3.6 Test Results

The modulus of elasticity is calculated according to equation (3.6):

$$E_c = \frac{\sigma_2 - \sigma_1}{\varepsilon_{02} - \varepsilon_{01}}$$

where:

- $E_c$ is the modulus of elasticity, in newtons per square millimetre;
\( \sigma_2 \) is the upper test stress in the last loading cycle, in newtons per square millimeter;
\( \sigma_1 \) is the basic test stress before the last loading cycle, in newtons per square millimetre;
\( \varepsilon_{02} \) is the mean strain under the upper test stress \( \sigma_2 \) in the last loading cycle;
\( \varepsilon_{01} \) is the mean strain under the basic test stress \( \sigma_1 \) before the last loading cycle.

### 3.7 Prediction of Elastic Modulus of Concrete

More data (i.e. elastic modulus and their corresponding mix proportions were generated by use of mixture equation derived by Egbulonu (2011) shown in Eqn (3.7)

\[
Y = 41.31X_1 + 50.04X_2 + 25.21X_3 + 19.24X_4 + 95.58X_1X_2 - 28.64X_1X_3 + 22.10X_1X_4 - 55.76X_2X_3 - 25.96X_2X_4 + 14.26X_3X_4
\] 3.7

The equation was used to generate 800 data, which was supplied to the artificial neural network for testing, training and validating respectively. The results are shown in chapter four.
Table 3.1: Mixture Proportion For Concrete specimens (150mmdiax300mm long)

<table>
<thead>
<tr>
<th>S/N</th>
<th>Mix proportion</th>
<th>Water-cement ratio</th>
<th>Mixture label</th>
<th>Water (kg)</th>
<th>Cement (kg)</th>
<th>Sand (kg)</th>
<th>Granite (kg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>1:2:4</td>
<td>0.55</td>
<td>S₁</td>
<td>2.56</td>
<td>4.63</td>
<td>9.26</td>
<td>18.51</td>
</tr>
<tr>
<td>2.</td>
<td>1:2.5:6</td>
<td>0.50</td>
<td>S₂</td>
<td>1.71</td>
<td>3.41</td>
<td>8.53</td>
<td>20.46</td>
</tr>
<tr>
<td>3.</td>
<td>1:1.5:3.5</td>
<td>0.60</td>
<td>S₃</td>
<td>3.24</td>
<td>5.40</td>
<td>8.10</td>
<td>18.90</td>
</tr>
<tr>
<td>4.</td>
<td>1:2.25:5</td>
<td>0.52</td>
<td>S₄</td>
<td>2.06</td>
<td>3.93</td>
<td>8.84</td>
<td>19.64</td>
</tr>
<tr>
<td>5.</td>
<td>1:2.5:4.75</td>
<td>0.49</td>
<td>S₅</td>
<td>1.96</td>
<td>3.93</td>
<td>9.82</td>
<td>18.65</td>
</tr>
<tr>
<td>6.</td>
<td>1:1.75:3.75</td>
<td>0.58</td>
<td>S₆</td>
<td>2.87</td>
<td>4.98</td>
<td>8.72</td>
<td>18.69</td>
</tr>
<tr>
<td>7.</td>
<td>1:2.75:5.75</td>
<td>0.48</td>
<td>S₇</td>
<td>1.62</td>
<td>3.41</td>
<td>9.38</td>
<td>19.61</td>
</tr>
<tr>
<td>8.</td>
<td>1:2:4.75</td>
<td>0.55</td>
<td>S₈</td>
<td>2.30</td>
<td>4.18</td>
<td>8.36</td>
<td>19.86</td>
</tr>
<tr>
<td>9.</td>
<td>1:2.25:4.5</td>
<td>0.52</td>
<td>S₉</td>
<td>2.19</td>
<td>4.18</td>
<td>9.41</td>
<td>18.81</td>
</tr>
<tr>
<td>10.</td>
<td>1:2.375:4.875</td>
<td>0.51</td>
<td>S₁₀</td>
<td>2.01</td>
<td>3.93</td>
<td>9.33</td>
<td>19.14</td>
</tr>
<tr>
<td>11.</td>
<td>1:2.25:4.74</td>
<td>0.53</td>
<td>S₁₁</td>
<td>2.13</td>
<td>4.05</td>
<td>9.11</td>
<td>19.24</td>
</tr>
<tr>
<td>12.</td>
<td>1:2.125:4.625</td>
<td>0.54</td>
<td>S₁₂</td>
<td>2.25</td>
<td>4.18</td>
<td>8.88</td>
<td>19.33</td>
</tr>
<tr>
<td>13.</td>
<td>1:1.75:4.125</td>
<td>0.58</td>
<td>S₁₃</td>
<td>2.71</td>
<td>4.71</td>
<td>8.25</td>
<td>19.44</td>
</tr>
<tr>
<td>14.</td>
<td>1:2.25:4.375</td>
<td>0.52</td>
<td>S₁₄</td>
<td>2.23</td>
<td>4.25</td>
<td>9.56</td>
<td>18.59</td>
</tr>
<tr>
<td>15.</td>
<td>1:2.25:9.675</td>
<td>0.53</td>
<td>S₁₅</td>
<td>2.23</td>
<td>4.35</td>
<td>9.71</td>
<td>18.8</td>
</tr>
</tbody>
</table>

3.8 Work Flow for the Design of the Artificial Neural Network Process

The work flow for the general artificial neural network design process has seven primary steps:

(1.) Collection and preparation of Data

(2.) Creating the network

(3.) Configuration of the network

(4.) Initializing the weights and Biases

(5.) Training the network

(6.) Validating the network

(7.) Using the network
3.8.1 Collection and Preparation of Data

Data used for the learning, training, and validating of the artificial neural network were obtained from previous work on similar topic by Egbulonu (2011). The concrete mix ratios are the training inputs while their modulus of elasticity is the training targets. These input and output are stored in a matrix form in the network.

The inputs and output values were processed using the “mapminmax” and “removeconstantrows” processing and output values to fall within a bipolar range i.e [-1, 1], while the “removeconstantrows” helps to remove inputs/targets that are constant. All of these help to prevent the network from being ‘saturated’. Saturation helps in a small network gradient which leads to a very slow training process. Therefore the standard is to normalize the data before applying them to the network.

The data is then divided into three subsets. The first subset is the training set, which is used for computing the gradient and updating the network weights and biases. The second set is the validation set that is used in monitoring the training process. The third subset is the test set. It is not used in monitoring the training process rather it is used to compare different models and plot the test set error during training. The processing
function used for dividing these data to their various subsets is the ‘dividerand’. It divides the data randomly.

3.8.2 Creating the Network

After the data has been collected and prepared, the next step is to create the network. To achieve this, we need a feed forward artificial neural network with four input neurons (representing the water-cement ratio, cement, sand and granite respectively) and one output neuron (representing the modulus of rupture). This is achieved by using the network command ‘newff’. For purpose of illustration, to create a feedforward network, we type the command line; net=newff(p,t,10);

newff = Creates a feedforward network
p = Matrix of the inputs
t = Matrix of the outputs
10 = No. of neurons in the hidden layer

3.8.3 Configuration of the Network

Configuration of the network has to do with specifying the network parameter i.e the network’s processing functions e.g the activation function for the different layers, the training algorithms etc. the network input and output process functions can be override by adjusting the network properties after the network is created. To see a cell array list of processing functions assigned to the input and the output of a network, use the
command line; ‘net.inputs{1}.processFcns’ and ‘net.outputs{2}.processFcns’ respectively.

3.8.4 Initializing the Weights and Biases

Weights are values associated with a connection path between two processing elements in artificial neural network. They contain fundamental information concerning the problem being solved. Biases are same as weights only that their output is always 1.

Before training a feedforward network, these weights and biases must be in initialized. The configure command automatically initializes the weights but one can reinitialize them using the command line: ‘net=init (net)’. Each time a feedforward network is initialized; the network parameters are different and might produce different solutions.

3.8.5 Training the Network

Once network weights and biases are initialized, the network is ready for training. The multilayer feedforward network can be trained for function approximation (non-linear regression) or pattern recognition. The training process requires a set of examples of proper network behavior i.e network inputs ‘p’ and target output ‘t’.
The process of training required tuning the values of the weights and biases of the network to optimize network performance using the mean square error (mse). The average mean square error between network outputs ‘o’ and target outputs ‘t’ is defined as:

\[
\text{mse} = \frac{1}{N} \sum_{k=1}^{N} (t_k - o_k)^2 \tag{3.8}
\]

In this work, in order to improve generalization, the performance function was modified by adding a term that consists of the mean of the sum of squares of the network weights and biases (msw). Therefore, the mean square with regularization ‘msereg’ can be defined as:

\[
\text{msereg} = k\text{mse} + (1 - k)\text{msw} \tag{3.9}
\]

where,

\[
K \text{ - Performance ratio}
\]

\[
\text{msw} = \frac{1}{n} \sum_{j=1}^{n} w_j^2 \tag{3.10}
\]

where:

\[
w = \text{weights and biases}
\]

This causes the network to have smaller weights and biases, forcing the network response to be small and less likely to overfit.
The training is implemented using the batch made. Here, all the inputs in the training set are applied to the network before the weights are updated. This gives a significantly faster training and produces smaller errors.

The gradient of the network performance with respect to the weights is the optimization method used. Let us consider the simplest optimization algorithm—gradient descent or Jacobian of the network error with respect to the weights. It updates the network weights and biases in the direction in which the performance function decreases most rapidly, i.e., the negative of the gradient. One iteration of this algorithm can be written as:

\[ x_{k-1} = x_k - \alpha_k g_k \]  

where:
\[ x_k \] = Vector of current weights and biases  
\[ g_k \] = Current gradient of network  
\[ \alpha_k \] = Learning rate

This equation is iterated until the network converges. The training algorithm adopted in this work is the gradient descent with momentum constant.

3.8.6 Validating the Network

When the training is complete, one will want to check the network performance and determine if any changes need to be
made to the training process, the network architecture or the date sets. The first thing to do is to check the training records using the command line ‘tr=’. This function keeps track of several variables during the course of training, such as the value of the performance function, magnitude of the gradient, the regression plot etc.

3.8.7 Using the Network

After the network is trained and validated, the network object can be used to calculate the network response to any input.

Table 3.2: Artificial Neural Network (ANN) information

<table>
<thead>
<tr>
<th>No.</th>
<th>Ann Information</th>
<th>Database</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>The number of used data in network</td>
<td>571 data were selected from 800 results</td>
</tr>
<tr>
<td>2.</td>
<td>Data number used for training</td>
<td>400</td>
</tr>
<tr>
<td>3.</td>
<td>Data number used for testing</td>
<td>89</td>
</tr>
<tr>
<td>4.</td>
<td>Data number used for validating</td>
<td>82</td>
</tr>
<tr>
<td>5.</td>
<td>Input data</td>
<td>Water, cement, sand and granite</td>
</tr>
<tr>
<td>6.</td>
<td>Output</td>
<td>Modulus of Elasticity of concrete</td>
</tr>
</tbody>
</table>
4.1 Presentation of Results

In this work, various experiments were carried out. These include:

(i.) Test on physical properties of the Dangote cement
(ii.) Test on chemical properties of the Dangote cement
(iii.) Grain size distribution analysts of sand.
(iv.) Sieve analysis of crushed rock aggregate.
(v.) Modulus of elasticity test on concrete.

The results of the tests are presented in Tables 4.1 to 4.7.

4.1.1 Physical Properties of the Dangote Cement

The results of the test carried out on physical properties of Dangote cement by Yahaya (2009) according to 8SI (1978) specifications are presented in Table 4.1 below.

Table 4.1: Physical Properties of Dangote Cement

<table>
<thead>
<tr>
<th>Physical Properties</th>
<th>Average quantities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Moisture content</td>
<td>0.003</td>
</tr>
<tr>
<td>Specific gravity</td>
<td>3.15</td>
</tr>
<tr>
<td>Fitness</td>
<td>190 plus</td>
</tr>
<tr>
<td>PH value</td>
<td>9.2</td>
</tr>
</tbody>
</table>

Source: Yahaya, (2009)
4.1.2 Chemical Properties of the Dangote Cement

A test on chemical properties of Dangote Cement was also carried out in accordance to BSI (1978) specifications by Yahaya (2009). The results are presented in Table 4.2.

Table 4.2: Chemical Properties of Dangote Cement

<table>
<thead>
<tr>
<th>S/N</th>
<th>Oxides</th>
<th>Quantities (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Lime (CaO)</td>
<td>51.67</td>
</tr>
<tr>
<td>2.</td>
<td>Alumina (Al₂O₃)</td>
<td>1.25</td>
</tr>
<tr>
<td>3.</td>
<td>SiO₂</td>
<td>18.02</td>
</tr>
<tr>
<td>4.</td>
<td>Iron Oxide Fe₂O₃</td>
<td>10.5</td>
</tr>
<tr>
<td>5.</td>
<td>Potassium Oxide (K₂O)</td>
<td>-</td>
</tr>
<tr>
<td>6.</td>
<td>Magnesium Oxide (MgO)</td>
<td>1.45</td>
</tr>
<tr>
<td>7.</td>
<td>Sodium Oxide (Na₂O)</td>
<td>-</td>
</tr>
<tr>
<td>8.</td>
<td>Titanium Oxide (TiO₂)</td>
<td>-</td>
</tr>
<tr>
<td>9.</td>
<td>Loss in Ignition</td>
<td>3.27</td>
</tr>
<tr>
<td>10.</td>
<td>Insoluble Residue</td>
<td>2.03</td>
</tr>
<tr>
<td>11.</td>
<td>Free Cao</td>
<td>2.15</td>
</tr>
<tr>
<td>12.</td>
<td>S₀₃</td>
<td>1.40</td>
</tr>
</tbody>
</table>

Source: Yahaya, (2009)

4.1.3 Grain Size Distribution Analysis of Sand

The results of the grain size distribution analysis of sand carried out in accordance with BS 882 as explained earlier in section 3.1.3, are presented in Table 4.3: From the grain size distribution, the grading curve for the sand is obtained shown in appendix A.
Table 4.3: Grain Size Distribution of Sand

<table>
<thead>
<tr>
<th>S/N</th>
<th>Sieve size (mm)</th>
<th>Sieve No</th>
<th>Wt of Sieve (g)</th>
<th>Wt of sand + Sieve (g)</th>
<th>Wt of Sand Retained (Only) (g)</th>
<th>Wt Passing (g)</th>
<th>Cum% Passing</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.83mm</td>
<td>7</td>
<td>474</td>
<td>475</td>
<td>1</td>
<td>499</td>
<td>99.8</td>
</tr>
<tr>
<td>2</td>
<td>1.41mm</td>
<td>14</td>
<td>447</td>
<td>454</td>
<td>7</td>
<td>492</td>
<td>98.4</td>
</tr>
<tr>
<td>3</td>
<td>707µm</td>
<td>25</td>
<td>364</td>
<td>674</td>
<td>310</td>
<td>182</td>
<td>36.4</td>
</tr>
<tr>
<td>4</td>
<td>297µm</td>
<td>50</td>
<td>326</td>
<td>494</td>
<td>168</td>
<td>14</td>
<td>12.8</td>
</tr>
<tr>
<td>5</td>
<td>149µm</td>
<td>100</td>
<td>317</td>
<td>319</td>
<td>2</td>
<td>12</td>
<td>2.8</td>
</tr>
<tr>
<td>6</td>
<td>Pan</td>
<td>Pan</td>
<td>433</td>
<td>445</td>
<td>12</td>
<td>-</td>
<td>2.4</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>500</td>
<td>500</td>
</tr>
</tbody>
</table>

where:

\[ Wt = \text{weight} \]
\[ \text{Cum} = \text{cumulative} \]

4.1.4 Sieve Analysis of Crushed Rock Aggregate

The results of the sieve analysis of rushed rock aggregate are presented in Table 4.4. The results are used in obtaining the grain size distribution and grading curve shown in appendix B.

Table 4.4: Results of Sieve Analysis of Crushed Rock Aggregate

<table>
<thead>
<tr>
<th>S/N</th>
<th>Sieve size</th>
<th>Weight of Aggregate (g)</th>
<th>Cumulative % passing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retained</td>
<td>Passing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>25mm</td>
<td>0</td>
<td>2000</td>
</tr>
<tr>
<td>2</td>
<td>19.5mm</td>
<td>249</td>
<td>1751</td>
</tr>
<tr>
<td>3</td>
<td>12mm</td>
<td>1569</td>
<td>182</td>
</tr>
<tr>
<td>4</td>
<td>9.5mm</td>
<td>155</td>
<td>27</td>
</tr>
<tr>
<td>5</td>
<td>4.76mm</td>
<td>27</td>
<td>0</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>2000</td>
<td>2000</td>
</tr>
</tbody>
</table>

4.1.5 Results of Modulus of Elasticity Test

The values of the modulus of elasticity, (i.e E-value) is the ratio between stress (load/area) and strain (deformation or change of length/original length).
The modulus of elasticity, $E$, is calculated using Eqns. (4.1)-(4.4), and the results presented in Table 4.6.

\[\Delta l = (l_a - l_b) \times 10^{-3}\text{mm}\]

\[\Delta \sigma = \frac{p_a - p_b}{\pi r^2}\]

\[\Delta \varepsilon = \frac{\Delta l}{l}\]

\[\Sigma = \frac{\Delta \sigma}{\Delta \varepsilon} (N/\text{mm}^2)\]

where:

- $\Delta l$ = Reduction in length of cylindrical specimen due to compression, measured in mm
- $l_a$ = Final gauge reading measured at approximately $1/3$ of compressive strength ($X 10^{-3}mm$)
- $l_b$ = Initial gauge reading measured in mm $X 10^{-3}$
- $p_a$ = Load at final gauge reading (i.e. at $l_a$)
- $p_b$ = Load at initial gauge reading (i.e. at $l_b$)
- $r$ = Radius of cylindrical sample = $\frac{150mm}{2}$
- $\sigma_a$ = Upper stress ($N/\text{mm}^2$)
- $\sigma_b$ = Initial stress ($N/\text{mm}^2$)
- $l$ = Length of cylindrical sample in mm = $200mm$
- $E$ = Static Modulus of Elasticity ($N/\text{mm}^2$)
- $\Delta \sigma$ = Stress difference ($N/\text{mm}^2$)
- $\Delta \varepsilon$ = Strain
**Table 4.5:** Experimental Results of Modulus of Elasticity, E.

<table>
<thead>
<tr>
<th>S/N</th>
<th>Replicates</th>
<th>Mass kg</th>
<th>Loads</th>
<th>Gauge length</th>
<th>Modulus</th>
<th>Mean Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Pb</td>
<td>Pa</td>
<td>Lb</td>
<td>La</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>P</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.</td>
<td>A</td>
<td>4.00</td>
<td>0.5</td>
<td>4.5</td>
<td>53</td>
<td>104.4</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>4.00</td>
<td>0.5</td>
<td>4.5</td>
<td>120</td>
<td>168.8</td>
</tr>
<tr>
<td>2.</td>
<td>A</td>
<td>4.20</td>
<td>0.5</td>
<td>4.5</td>
<td>125</td>
<td>173</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>4.20</td>
<td>0.5</td>
<td>4.5</td>
<td>127</td>
<td>170.4</td>
</tr>
<tr>
<td>3.</td>
<td>A</td>
<td>4.00</td>
<td>0.5</td>
<td>4.5</td>
<td>170</td>
<td>215</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>4.20</td>
<td>0.5</td>
<td>4.5</td>
<td>250</td>
<td>292</td>
</tr>
<tr>
<td>4.</td>
<td>A</td>
<td>4.20</td>
<td>0.5</td>
<td>4.5</td>
<td>80</td>
<td>126</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>4.20</td>
<td>0.5</td>
<td>4.5</td>
<td>230</td>
<td>270.8</td>
</tr>
<tr>
<td>5.</td>
<td>A</td>
<td>4.30</td>
<td>0.5</td>
<td>4.5</td>
<td>160</td>
<td>212.2</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>4.20</td>
<td>0.5</td>
<td>4.5</td>
<td>200</td>
<td>249.2</td>
</tr>
<tr>
<td>6.</td>
<td>A</td>
<td>4.20</td>
<td>0.5</td>
<td>4.5</td>
<td>310</td>
<td>340.8</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>4.20</td>
<td>0.5</td>
<td>4.5</td>
<td>215</td>
<td>244</td>
</tr>
<tr>
<td>7.</td>
<td>A</td>
<td>4.20</td>
<td>0.5</td>
<td>4.5</td>
<td>330</td>
<td>359.6</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>4.20</td>
<td>0.5</td>
<td>4.5</td>
<td>420</td>
<td>448.4</td>
</tr>
<tr>
<td>8.</td>
<td>A</td>
<td>4.10</td>
<td>0.5</td>
<td>4.5</td>
<td>480</td>
<td>528.6</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>4.10</td>
<td>0.5</td>
<td>4.5</td>
<td>535</td>
<td>579</td>
</tr>
<tr>
<td>9.</td>
<td>A</td>
<td>4.10</td>
<td>0.5</td>
<td>4.5</td>
<td>200</td>
<td>231.6</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>4.10</td>
<td>0.5</td>
<td>4.5</td>
<td>220</td>
<td>250.6</td>
</tr>
<tr>
<td>10.</td>
<td>A</td>
<td>4.20</td>
<td>0.5</td>
<td>4.5</td>
<td>160</td>
<td>187.2</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>4.20</td>
<td>0.5</td>
<td>4.5</td>
<td>210</td>
<td>236.8</td>
</tr>
<tr>
<td>11.</td>
<td>A</td>
<td>4.10</td>
<td>0.5</td>
<td>4.5</td>
<td>410</td>
<td>439.6</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>4.10</td>
<td>0.5</td>
<td>4.5</td>
<td>320</td>
<td>347.8</td>
</tr>
<tr>
<td>12.</td>
<td>A</td>
<td>4.20</td>
<td>0.5</td>
<td>6.0</td>
<td>100</td>
<td>138.8</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>4.20</td>
<td>0.5</td>
<td>6.0</td>
<td>130</td>
<td>166.6</td>
</tr>
<tr>
<td>13.</td>
<td>A</td>
<td>4.120</td>
<td>0.5</td>
<td>6.0</td>
<td>95</td>
<td>142</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>4.10</td>
<td>0.5</td>
<td>6.0</td>
<td>200</td>
<td>246.2</td>
</tr>
<tr>
<td>14.</td>
<td>A</td>
<td>4.20</td>
<td>0.5</td>
<td>6.0</td>
<td>150</td>
<td>195.4</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>4.20</td>
<td>0.5</td>
<td>6.0</td>
<td>200</td>
<td>244.8</td>
</tr>
<tr>
<td>15.</td>
<td>A</td>
<td>4.2</td>
<td>0.5</td>
<td>6.0</td>
<td>215</td>
<td>288.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>230</td>
<td>302.6</td>
</tr>
</tbody>
</table>

MOE: Modulus of Elasticity
4.2 Analysis of Results

4.2.1 Artificial Neural Network-Based Modeling of Modulus of Elasticity of Concrete

The ability of the ANN to map the relationship between the input data and their corresponding output data, was used in this work for predicting the modulus of elasticity of normal concrete. The ‘mapminmax’ function in the artificial neural network toolbar in the MATLAB software was used in normalizing the values of input and output data within the range -1 and +1.

4.2.1.1 Learning Rates and Momentum Constant

The learning rate and momentum constants were kept constant at 0.2 and 0.9 respectively.

4.2.1.2 Architecture of the Artificial Neural Network

The artificial neural network used in the training consists of three layers, namely, one input layer, one hidden layer and one output layer. The input layer consists of 4 neurons each representing one of the inputs namely, cement, sand, crushed aggregate and water. The hidden layer consists of neurons and the outer layer has one neuron, which represents the modulus of elasticity, E.

And the activation function which helps the network to learn the non-linear relationship between the input and output was
tangent sigmoid transfer function.

4.2.1.3 Selection of the Training Data

In selecting the data for training the Artificial Neural Network, effort was made to select adequate data that cover the range of inputs for which the network will be used and yield high level of accuracy without overwhelming the training algorithm.

In all, a total of 800 data was supplied to the network. Out of which, 571 data was selected randomly by the network. From the selected data, 400 were used for training the network, 89 for testing and 82 for validating network. These proportions were achieved using the ‘dividend’ function and the network objects.
Fig 4.1: Regression plot Network Outputs and Network Targets

4.2.1.4 Validation of the Network Performance

Fig 4.3 shows a regression plot between network outputs and network targets. The ‘R’ value is an indication of the relationship between the outputs and targets. If R = 1, this indicates that
there is an exact linear relationship between the output and targets. (Beale et al, 2010). For this study, the training data set has an ‘R’ value of 0.95237, the validation data set has an ‘R’ value of 0.91905 and the test data set has an ‘R’ value of 0.93521. All of these results have an ‘R’ value of 0.93731 which is greater than 0.9. Therefore we can conclude that data used for training the network have a good fit.

**Fig 4.2:**  *Bar Chart showing the Plot of Artificial Neural Network Values against Experimental Values*
**Fig 4.3:** (a) Plot of Network against number of Epoch

(b) Plot of validation against Number of Epoch

The gradient at 1000 epoch is 0.059929. There was no network validation during training because the training of the network stopped when the maximum epoch was reached.
4.2.1.5 Comparison of Predicted and Experimental Values

The Artificial Neural Network predicted values of Modulus of Elasticity of Concrete and those of the experimental values, can be seen in Table 4.6. From the comparison, the maximum percentage difference between the predicted artificial neural network values and experimental values of the modulus of elasticity of concrete, MOE is 1.49%. This shows that the Artificial Neural Network model can accurately be used to predict MOE of concrete.

Table 4.6: Experimental Result compared with Artificial Neural Network Prediction

<table>
<thead>
<tr>
<th>Mix label</th>
<th>Experimental Result (KN/m²)</th>
<th>Artificial Neural Network Prediction (KN/m²)</th>
<th>Percentage difference of Experimental and ANN values</th>
</tr>
</thead>
<tbody>
<tr>
<td>E1</td>
<td>20.000</td>
<td>20.101</td>
<td>-0.505</td>
</tr>
<tr>
<td>E2</td>
<td>22.000</td>
<td>21.672</td>
<td>1.49</td>
</tr>
<tr>
<td>E3</td>
<td>23.100</td>
<td>23.00</td>
<td>0.43</td>
</tr>
<tr>
<td>E4</td>
<td>23.200</td>
<td>23.310</td>
<td>-0.47</td>
</tr>
<tr>
<td>E5</td>
<td>19.800</td>
<td>19.652</td>
<td>0.75</td>
</tr>
<tr>
<td>E6</td>
<td>33.500</td>
<td>33.401</td>
<td>0.30</td>
</tr>
<tr>
<td>E7</td>
<td>34.500</td>
<td>34.700</td>
<td>-0.58</td>
</tr>
<tr>
<td>E8</td>
<td>21.700</td>
<td>21.750</td>
<td>-0.23</td>
</tr>
<tr>
<td>E9</td>
<td>32.200</td>
<td>32.201</td>
<td>0.00</td>
</tr>
<tr>
<td>E10</td>
<td>37.100</td>
<td>37.00</td>
<td>0.27</td>
</tr>
<tr>
<td>E11</td>
<td>35.000</td>
<td>34.988</td>
<td>0.03</td>
</tr>
<tr>
<td>E12</td>
<td>37.000</td>
<td>37.120</td>
<td>-0.32</td>
</tr>
<tr>
<td>E13</td>
<td>30.000</td>
<td>30.086</td>
<td>-0.29</td>
</tr>
<tr>
<td>E14</td>
<td>31.000</td>
<td>30.987</td>
<td>0.04</td>
</tr>
<tr>
<td>E15</td>
<td>19.100</td>
<td>19.186</td>
<td>-0.45</td>
</tr>
</tbody>
</table>
The network output values are plotted against the experimental values in the form of a bar chart shown in fig 4.2. A cursory look shows that the predicted values approximate the experimental value.

4.3 **Test of Adequacy of the Artificial Neural Network Model**

The artificial neural network will be tested for adequacy the experimental results.

4.3.1 **Statistical Hypothesis**

(a) **Null Hypothesis (H$_0$):** There is no significant difference between the experimental and artificial neural network results at 0.05 significance level.

(b) **Alternative Hypothesis (H):** There is a significant difference between the experimental and artificial neural network results at 0.05 significance level.
Using student’s t-test

Table 4.7: Check of Adequacy of the Network

<table>
<thead>
<tr>
<th>S/N</th>
<th>Experimental Result (KN/m²)</th>
<th>Artificial Neural Network Prediction (KN/m²)</th>
<th>Dᵢ=EX-NP</th>
<th>Dₐ-Dᵢ</th>
<th>(Dₐ-Dᵢ)²</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>20.00</td>
<td>20.101</td>
<td>-0.101</td>
<td>0.1193</td>
<td>0.01423</td>
</tr>
<tr>
<td>2.</td>
<td>22.00</td>
<td>21.672</td>
<td>0.328</td>
<td>-0.3097</td>
<td>0.09591</td>
</tr>
<tr>
<td>3.</td>
<td>23.10</td>
<td>23.00</td>
<td>0.10</td>
<td>-0.0817</td>
<td>0.0067</td>
</tr>
<tr>
<td>4.</td>
<td>23.20</td>
<td>23.310</td>
<td>0.11</td>
<td>-0.0917</td>
<td>0.084</td>
</tr>
<tr>
<td>5.</td>
<td>19.80</td>
<td>19.652</td>
<td>0.148</td>
<td>-0.1297</td>
<td>0.0168</td>
</tr>
<tr>
<td>6.</td>
<td>33.50</td>
<td>33.401</td>
<td>0.099</td>
<td>-0.0807</td>
<td>0.0065</td>
</tr>
<tr>
<td>7.</td>
<td>34.50</td>
<td>34.700</td>
<td>-0.200</td>
<td>0.2183</td>
<td>0.0477</td>
</tr>
<tr>
<td>8.</td>
<td>21.70</td>
<td>21.750</td>
<td>-0.05</td>
<td>0.0683</td>
<td>0.0047</td>
</tr>
<tr>
<td>9.</td>
<td>32.20</td>
<td>32.201</td>
<td>0.001</td>
<td>0.0173</td>
<td>0.00299</td>
</tr>
<tr>
<td>10.</td>
<td>37.10</td>
<td>37.00</td>
<td>0.10</td>
<td>-0.0817</td>
<td>0.0067</td>
</tr>
<tr>
<td>11.</td>
<td>35.00</td>
<td>34.988</td>
<td>0.012</td>
<td>0.0063</td>
<td>0.0004</td>
</tr>
<tr>
<td>12.</td>
<td>37.00</td>
<td>37.120</td>
<td>-0.120</td>
<td>0.1383</td>
<td>0.0191</td>
</tr>
<tr>
<td>13.</td>
<td>30.00</td>
<td>30.086</td>
<td>-0.86</td>
<td>0.1043</td>
<td>0.0108</td>
</tr>
<tr>
<td>14.</td>
<td>31.00</td>
<td>30.987</td>
<td>0.013</td>
<td>0.0053</td>
<td>0.0003</td>
</tr>
<tr>
<td>15.</td>
<td>19.10</td>
<td>19.180</td>
<td>-0.08</td>
<td>0.0983</td>
<td>0.0097</td>
</tr>
</tbody>
</table>

\[ \sum Dᵢ = 0.274 \]

\[ Dₐ = \frac{\sum Dᵢ}{N} = \frac{0.274}{15} = 0.0183 \]

\[ S^2 = \frac{\sum (Dₐ-Dᵢ)^2}{N-1} = \frac{0.2503}{14} = 0.01788 \]

\[ S = \sqrt{S} = \sqrt{0.01788} \]
\[ S = 0.1337 \]
\[ t = \frac{D_A \cdot N^{0.5}}{S} = \frac{0.0183 \cdot 15^{0.5}}{0.1337} \]
\[ t = 0.530109 \]

For two-tail test

Allowable variation in t-test

\[ = t (0.05 , \ N - 1) \]
\[ = t (0.05 , 14) \]
\[ = 2.14 \]

\[ T_{\text{calculated}} = 0.530109 \]

Since \( t_{\text{allowed}} > t_{\text{calculated}} \)

The artificial neural network model is adequate. Thus null hypothesis is accepted.

Here:

\( N_p = \) Artificial neural network responses

\( Ex = \) Experimental results

\( D_i = \) Difference of \( Ex \) and \( N_p \)

\( D_A = \) Mean of \( D_i \)

\[ S = \sqrt{\frac{(D_A - D_i)^2}{N-1}} \]
CHAPTER FIVE
CONCLUSIONS, RECOMMENDATIONS AND CONTRIBUTIONS TO KNOWLEDGE

5.1 Conclusions

The following conclusion can be drawn from this research:

(i.) Literature was reviewed. It was discovered from the literature that ANN is an artificial intelligence method approach that mimic, in a very simplistic way, the cognition capability of human brain. It learns by examples of data inputs and outputs presented to it so that the subtle functional relationships among the data are captured, even if the underlying relationships are unknown or the physical meaning is difficult to understand.

(ii.) Values of modulus of elasticity (MOE) were obtained experimentally from various mix designs. And the maximum and minimum values are 37 N/mm and 20 N/m respectively.

(iii.) The artificial neural network model was established using MATLAB version 7.5. In the establishment of the model, 800 mixes were supplied to the network. Out of which, 571 were selected by the network. From the selected data, 400 were used to train the network, 89 for testing and 82 for validating the network. The coefficient of correlation, R
values of the data used for training, testing and validating
the network were respectively greater than 0.90. This
shows that the data used in the network have good fit.

(iv.) The elastic modulus of concrete of different mixes was
predicted by the ANN. The results obtained from the
network were compared with the experimental values and
found to be very close. The maximum and minimum
percentage differences between the experimental elastic
modulus and ANN elastic modulus of concrete were 1.49%
and 0.00% respectively.

(v.) The results obtained from the network elastic modulus
were compared with the regression, i.e., Scheffe (5,2)
simplex equation values and found to be very close.

5.2 Recommendations

(i.) The artificial neural network model can be used in
optimization of the Modulus of Elasticity of concrete
prepared manually in the laboratory.

(ii.) Research work could still be carried out using other
artificial intelligent method of optimization like genetic
algorithm, Fuzzy Logic method and probabilistic networks.

(iii.) Artificial neural network could still be used to predict other
properties of concrete, like the modulus of Rupture,
modulus of rigidity, workability, poison ratio, etc.

(iv.) Accurate experimental data should be sort for or generated, and used for the training of the network as the accuracy of any network is dependent on the accuracy of the training data sets.

5.3 Contributions to Knowledge

This research work, Artificial Neural Network Model for the optimization of the Modulus of Elasticity of concrete contributed the following to knowledge.

(i.) Provide artificial neural network model for prediction of modulus of elasticity of concrete.

(ii.) Provide model for estimation of elastic modulus of concrete without limit unlike in the case of regression model.
APPENDIX C

COMMAND LINES FOR THE CREATION OF THE GRAPHICAL USER INTERFACE AND THE GRADIENT DESCENT BACKPROPAGATION ARTIFICIAL NEURAL NETWORK

Qui_singleton=1;
Qui_State = struct ('gui_Name', mfilenamce, ......
    'gui_Singleton', 'gui_Singleton, ......
    'gui_openingFcn', @flexuralStrength_OpeningFcn, ...'gui_OpeningFcn', @flexuralStrength_OpeningFcn, ...
    'gui_LayoutFcn', [],.....
    'gui_Callback', []; ...

If nargin && ischar (varargin [1])
    gui_State.gui_Callback = str2func (varargin [1]);
end

If nargout
    {varargout [1:nargout]} = gui_mainfncn (gui_State, varargin {;});
else
    gui_mainfncn(gui_State, varargin {;});
end
% End initialization code – DO NOT EDIT

% --- Executes just before flexuralStrength is made visible,
function flexuralStrength_OpeningFcn (hobject, eventdata, handles, varargin)
handles.output = hobject;

& Update handles structure
guidata (hobject, handles);
function varargout = flexuralStrength_Output Fcn (hObject, eventdata, handles)
varargout {1} =handles.output;
% --- Executes on button press in radiobutton3.
function radiobutton3_Callback (hobject, eventdata, handles)
If (get (hobject, 'value'), = get (hobject, 'Max')
    %Radio button is selected, take appropriate action
    set (handles.waterv, 'enable', 'on');
    set (handles.sanddv, 'enable', 'on');
    set (handles.clear2, 'enable', 'on');
    set (handles.granitev, 'enable', 'on');
    set (handles.cementv, 'enable', 'on');
set (handles.calc, 'enable', 'on');

else
    set (handles.waterv, 'enable', 'inactive');
    set (handles.sanddv, 'enable', 'inactive');
    set (handles.clear2, 'enable', 'inactive');
    set (handles.granitev, 'enable', 'inactive');
    set (handles.cementv, 'enable', 'inactive');
    set (handles.calc, 'enable', 'inactive');
end

function waterv_Callback (hobject, eventdata, handles)

function waterv_createFcn (hobject, eventdata, handles)
    if ispc && isequal (get (hobject, 'BackgroundColor'),
                    get (0, 'defaultUnicontrolBackgoundColor'))
        set (hobject, 'BackgroundColor', 'white');
    end

function cementv_Callback (hobject, eventdata, handles)

function cementv_createFcn (hobject, eventdata, handles)
    if ispc && isequal (get (hobject, 'BackgroundColor'),
                    get (0, 'defaultUnicontrolBackgoundColor'))
        set (hobject, 'BackgroundColor', 'white');
    end

function sanddv_Callback (hobject, eventdata, handles)
    % --- Executes during object creation, after setting all properties.
    function sanddv_CreateFcn (hobject, eventdata, handles)
        if ispc && isequal (get (hobject, 'BackgroundColor'),
                            get (0, 'defaultUnicontrolBackgoundColor'))
            set (hobject, 'BackgroundColor', 'white');
        end
    end

function granitev_Callback (hobject, eventdata, handles)

function granitev_createFcn (hobject, eventdata, handles)
    if ispc && isequal (get (hobject, 'BackgroundColor'),
                        get (0, 'defaultUnicontrolBackgoundColor'))
    end
set (hobject, 'BackgroundColor', 'white');
end

% --- Executes during object creation, after setting all properties
% on button press in Calc.

function sanddv_CreateFcn (hobject, eventdata, handles)

global user_text;
global networkAlu;
global g;
global nettiq;
%user_text = str2double (get (hobject, 'string'))

a = str2double (get (handles, waterv, 'string'));
b = str2double (get (handles, cementv, 'string'));
d = str2double (get (handles, sanddv, 'string'));
e = str2double (get (handles, granitev, 'string'));
if (a~=0) && (b~=0) && (e~=0) && (d~=0)
simval = [a;b;d;e]
c = sim(networkAlu, simval);
h = num2str(c);
f = strcat ('The flexuralstrength is : ', h);
msgbox (f, 'value');
else
    msgbox ('the values you enter is invalid', 'Invalid data');
end

% --- Executes on button press in clear2.
function clear2_Callback (hobject, eventdata, handles)

set (handles.waterv, 'string', ');
set (handles.sanddv, 'string', ');
set (handles.cementv, 'string', ');
set (handles.granitev, 'string', ');
set (handles.calc, 'string', ');

% --- Executes on button press in radiobutton2.
function radiobutton2_Callback (hobject, eventdata, handles)

% hObject handles to radiobutton2 (see GCBO)
% eventdata reserved – to be defined in a failure version of
% MATLAB
% handles structure with handles and user data (see GUIDATA)
if (get (hobject, ‘value’) == get (hobject, ‘Max’))
    % Radio button is selected, take appropriate action
    set (handles.from, ‘enable’, ‘on’);
    set (handles.to, ‘enable’, ‘on’);
    set (handles.gen, ‘enable’, ‘on’);
    set (handles.clear2, ‘enable’, ‘on’);
else
    set (handles.from, ‘enable’, ‘inactive’);
    set (handles.to, ‘enable’, ‘inactive’);
    set (handles.gen, ‘enable’, ‘inactive’);
    set (handles.clear2, ‘enable’, ‘inactive’);
end

function from_Callback (hobject, eventdata, handles)

function from_CreateFcn (hobject, eventdata, handles)

    if ispc && isequal (get (hobject, ‘BackgroundColor’),
                        get (0, ‘defaultUnicontrolBackgoundColor’))
        set (hobject, ‘BackgroundColor’, ‘white’);
    end

end

function from_Callback (hobject, eventdata, handles)

function from_CreateFcn (hobject, eventdata, handles)

    if ispc && isequal (get (hobject, ‘BackgroundColor’),
                        get (0, ‘defaultUnicontrolBackgoundColor’))
        set (hobject, ‘BackgroundColor’, ‘white’);
    end

end

% --- Executes on button press in gen.
function gen_Callback (hobject, eventdata, handles)

global dat;
global networkAluRev;
global data1;

    f = (get (handles, from, ‘string’));
    t = (get (handles, to, ‘string’));
    from = str2double (f);
    to = str2double (t);
    diff = to-from;
dat = from
data1 = [];
from k = 0.1: 0.1: diff
    sval = sim (networkAluRev, from);
    dat = [dat; sval];
d = dat’
k;
from = from + 0.1;
end
set (handles, t1, ‘data’, data1);

function fs_Callback (hobject, eventdata, handles)

function fs_CreateFcn (hobject, eventdata, handles)
if ispc & & isequal (get (hobject, ‘BackgroundColor’),
    get (0, ‘defaultUnicontrolBackgoundColor’)
    set (hobject, ‘BackgroundColor’, ‘white’);
end

% --- Executes on button press in radiobutton1.
function radiobutton1_Callback (hobject, eventdata, handles)
If (get (hobject, ‘value’), = get (hobject, ‘Max’)
    % Radio button is selected, take appropriate action 
    set (handles.fs, ‘enable’, ‘on’);
    set (handles.generate, ‘enable’, ‘on’);
    set (handles.clear2, ‘enable’, ‘on’);
else
    set (handles.fs, ‘enable’, ‘inactive’);
    set (handles.generate, ‘enable’, ‘inactive’);
    set (handles.clear2, ‘enable’, ‘inactive’);
end

% --- Executes on button press in generate.
function generate_Callback (hobject, eventdata, handles)
global simval;
global inputValue;
global networkAluRev;
    inputValue = str2double (get (handles, fs, ‘string’));
    simval = sim (networkAluRev, inputValue);
set (handles.watv, 'string', 'simval (1));
set (handles.cemv, 'string', 'simval (2));
set (handles.sanv, 'string', 'simval (3));
set (handles.grav, 'string', 'simval (4));

% ---------------------------------------------------------------------------
function Untitled_1_Callback (hobject, eventdata, handles)

% ---------------------------------------------------------------------------
function mnuNetwork_Callback (hobject, eventdata, handles)

global in;
global out;
global inv;
global outv;
global networkAlu;
global networkAluRev;
out =
4.579; 4.558; 4.539; 4.503; 4.486; 4.471; 4.458; 4.445; 4.445; 4.434; 4.432;....
4.414; 4.407; 4.407; 4.414; 4.423; 4.434; 4.445; 4.458; 4.471; 4.486; 4.503;....
4.52; 4.539; 4.558; 4.579; 4.593; 4.58; 4.568; 4.557; 4.547; 4.538; 4.529;....
4.522; 4.516; 4.51; 4.506; 4.502; 4.50; 4.498; 4.498; 4.50; 4.502; 4.505;....
4.51; 4.515; 4.521; 4.528; 4.536; 4.545; 4.555; 4.566; 4.578; 4.591; 4.59; 4.564;....
4.539; 4.515; 4.493; 4.471; 4.451; 4.432; 4.414; 4.403; 4.42; 4.438; 4.458; 4.478;....
4.50; 4.523; 4.547; 4.573; 4.599; 4.89; 4.855; 4.822; 4.791; 476; 4.31; 4.702;....
4.702; 4.731; 4.76; 4.791; 4.822; 4.855; 4.89; 4.876; 4.85; 4; 4.824; 4.80; 4.799;....
4.754; 4.732; 4.711; 4.708; 4.728; 4.75; 4.772; 4.796; 4.82; 4.846; 4.872; 4.899;....
4.896; 4.886; 4.877; 4.869; 4.861; 4.854; 4.848; 4.842; 4.837; 4.833; 4.829; 4.826;....
4.824; 4.822; 4.821; 4.825; 4.827; 4.829; 4.822; 4.824; 4.827; 4.83; 4.834; 4.838;....
4.844; 4.85; 4.856; 4.863; 4.871; 4.879; 4.888; 4.898; 4.88; 4.843; 4.807; 4.773;....
4.739; 4.707; 4.718; 4.75; 4.784; 4.819; 4.855; 4.893; 4.897; 4.885; 4.874; 4.865;....
4.857; 4.85; 4.844; 4.839; 4.836; 4.834; 4.833; 4.833; 4.834; 4.837; 4.841; 4.846;....
4.852; 4.859; 4.868; 4.878; 4.889; 4.89; 4.872; 4.855; 4.839; 4.825; 4.811; 4.799;....
4.788; 4.778; 4.77; 4.762; 4.756; 4.751; 4.747; 4.745; 4.743; 4.743; 4.744; 4.746;....
4.75; 4.754; 4.76; 4.767; 4.775; 4.785; 4.795; 4.807; 4.82; 4.834; 4.85; 4.866;....
REFERENCES


---
