Springer Geophysics

Alireza Hajian
Peter Styles

Application of Soft Computing and Intelligent Methods in Geophysics

Springer

Springer Geophysics

The Springer Geophysics series seeks to publish a broad portfolio of scientific books, aiming at researchers, students, and everyone interested in geophysics. The series includes peer-reviewed monographs, edited volumes, textbooks, and conference proceedings. It covers the entire research area including, but not limited to, applied geophysics, computational geophysics, electrical and electromagnetic geophysics, geodesy, geodynamics, geomagnetism, gravity, lithosphere research, paleomagnetism, planetology, tectonophysics, thermal geophysics, and seismology.

More information about this series at http://www.springer.com/series/10173

Alireza Hajian · Peter Styles

Application of Soft Computing and Intelligent Methods in Geophysics

Springer

Alireza Hajian
Department of Physics
Najafabad Branch, Islamic Azad University
Najafabad
Iran

Peter Styles
Applied & Environmental Geophysics
Research Group, School of Physical and
Geographical Sciences
Keele University
Keele
UK

ISSN 2364-9127 ISSN 2364-9119 (electronic)
Springer Geophysics
ISBN 978-3-319-66531-3 ISBN 978-3-319-66532-0 (eBook)
https://doi.org/10.1007/978-3-319-66532-0

Library of Congress Control Number: 2018938772

Printed on acid-free paper

This Springer imprint is published by the registered company Springer International Publishing AG
part of Springer Nature
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

Inferences regarding the interior structure of the Earth, ranging from over 6000 km to just the first few hundred meters of depth, have to be made to assess the potential for geohazards as well as to explore for and extract Earth's many invaluable resources and even to evaluate the possibility of safely storing materials underground which we wish to remove from the atmosphere (carbon dioxide) or the surface (radioactive or hazardous wastes). These inferences are based on the measurement, analysis, and interpretation of external geophysical fields. These include potential fields such as gravity and magnetic data which are intrinsically nonunique, i.e., there are many interpretations, each with their inherent measurement uncertainties, which can be derived from a single set of measurements, and electrical, electromagnetic, or seismic data which similarly offer multiple applicable models. It is understood intuitively and intellectually by practitioners within the discipline that the interpretations are intrinsically "FUZZY". In some cases, it is possible to drill down to quite considerable depths to "prove" an interpretation, even if the answer is only actually applicable to a very small region of subsurface space. However, this is very costly and in some cases, the invasive nature is not appropriate, e.g., where the act of drilling may compromise the integrity of a structure such as a radioactive waste repository or quite simply the target may be too deep; here, geophysical methods are invaluable.

Geophysical research over the past few decades has witnessed a flurry of activity especially related to soft computing and intelligent methods. Working in the interdisciplinary field of soft computing and intelligent methods applications in geophysics and geotechnical aspects for the past 10 years has motivated us to publish a textbook on this important area of multi-interdisciplinary applied science. When Prof. Peter Styles was writing his book on the application of geophysics to environmental and civil engineering, I proposed that he added a section within a chapter or a chapter about the application of neural networks and fuzzy logic in gravity interpretation. However, upon reflection, it was felt that this subject was too advanced to be included. He then encouraged me to develop a specialist book on this topic. We agreed to collaborate on a draft for a book on the application of soft computing and intelligent methods in geophysics combining my expertise in the

application of these mathematical tools and his wide-ranging expertise across the fields of applied, environmental, and engineering geophysics. We finally developed the blueprint which is applied in this book.

During the more than a decade that we have worked on the application of neural networks, fuzzy logic, and neuro-fuzzy aspects, there have been many graduate students who were very eager to apply soft computing and intelligent methods (SCIMs) to their own geophysical problems but the books to introduce them to the topic and guide them in its application, either weren't available, didn't explain the applications to engineering geophysics or for the main part focused on applications in oil exploration with a principal emphasis on seismic reflection interpretation. Our book, in contrast, tries to cover the application of SCIM to a broad spectrum of geophysical methods. In addition, we provide the tools to design and test SCIM applications using MATLAB software, and these are presented as simply as possible, so that the reader can apply the MATLAB routines themselves and therefore learn the necessary skills in a practical manner.

The book has four main parts: Neural Networks, Fuzzy Logic, Combination of Neural Networks and Fuzzy Logic, and Genetic Algorithms.

In Part I, Chap. 1 outlines the principles of neural networks (NNs) and their design in MATLAB with practical examples, and in Chap. 2, the application of NNs to geophysical applications is presented with many varied examples.

In Part II, Chap. 3 develops the principles of fuzzy logic with the related fuzzy arithmetic and provides various examples in order to practically train the reader to grasp this new fuzzy viewpoint. In Chap. 4, we investigate the application of fuzzy logic to various geophysical methods.

In Part III, Chap. 5, we explain the application of the combination of NNs with Fuzzy logic as Neuro-Fuzzy Methods, with instructions for using the MATLAB ANFIS Toolbox through practical examples. In Chap. 6, the application of these Neuro-Fuzzy methods to geophysical analysis and interpretation is presented.

In Part IV, Prof. Mrinal Sen and Prof. Mallick, the additional contributors to this book, present the Genetic Algorithm and its applications in geophysics with many varied and interesting practical examples.

Readers of the book will find chapters dealing with preliminary aspects as well as advanced features of SCIMs. With a unique focus toward geophysical applications but with applicability to other physical and measurement sciences, this book will serve as a valuable reference for graduate students, research workers, and professionals interested in learning about these useful computing tools. The goal of this book is to help SCIMs find greater acceptability among researchers and practicing geophysicists alike.

It gives us both great pleasure to express our appreciation and thanks to a number of individuals who have helped us, either directly or indirectly, toward starting and eventually completing this book. I am grateful to past Ph.D. students especially R. Kimiaefar (who graduated in 2015 and is now a board member of Physics Department at IAUN) for help with MATLAB codes for NN design to attenuate seismic noise, and also my present Ph.D. student M. Rezazadeh Anbarani, who helped develop the MATLAB guides in Chapters 1 and 5. My colleague

Dr. Kh. Soleimani (board member of Mathematics Department at IAUN) provided invaluable help in the utilization of Fuzzy arithmetic. I will never forget my advisers for my Masters and Ph.D. degrees, Prof. V. E. Ardestani who first motivated me to use ANN in gravity interpretations in my M.S. thesis and Prof. H. Zomorrodian for his kindness and encouragement in combining neural and fuzzy methods in my Ph.D. thesis to interpret microgravity data. A very special mention is due to Prof. Caro Lucas (previously a board member of the Electrical Engineering Department at the University of Tehran) who is not now among us in this mortal world having passed away about a year before I defended my Ph.D. I am especially appreciative of having such a mentor who always encouraged me to go deeper into multi-interdisciplinary researches in soft computing.

I would also like give to a very special thanks to Dr. Pasandi, Assistant Professor of Geology, University of Isfahan and to record my sincere appreciation to Dr. A. Bohlooli, Assistant Professor in the Faculty of Computer Engineering at the University of Isfahan for his key guidance in arranging the book chapters.

Peter Styles would like to thank, en masse, the legions of Ph.D. and M.Sc. students from Swansea, Liverpool, and Keele Universities who worked with him in collecting a vast array of geophysical data, from which we select unique examples for this book, and who gave him such pleasure in their company, often despite adverse conditions and difficult circumstances. He is enormously grateful to his long-suffering wife Roslyn, who once again has had to put up with his chosen, strongly focused, writing modus operandi!

I especially want to give my thanks and express my deep appreciation of my wife Mohaddeseh and my daughter Elina for their support, over the course of more than a year, in writing this book and their understanding; therefore, that I could not spend all of my spare time with them.

Last, but not least, I am indebted to my parents: Mohammad Hassan and Ozra, for the continued appreciation and support that I have received in all my educational pursuits. I will never forget my Mother's endeavors in training me how to think deeply about the philosophy of work and life.

Najafabad, Iran

Alireza Hajian
Department of Physics
Najafabad Branch, Islamic Azad University

Keele, UK

Peter Styles
Applied & Environmental Geophysics Research Group
School of Physical and Geographical Sciences
Keele University

Contents

Part III Combination of Neural Networks and Fuzzy Logic

Part I
Neural Networks

Chapter 1
Artificial Neural Networks

– Principles of neural networks
– Design and test of neural networks
– Neural Networks toolbox in Matlab: NNTOOL

1.1 Introduction

Artificial neural networks are perhaps the most common method amongst intelligent methods in geophysics and are becoming increasingly popular. Because they are universal approximations, these tools can approximate any continuous function with any arbitrary precision.

Neural networks are increasingly being used in prediction, estimation, pattern recognition and optimization problems (Bohlooli et al. 2011) (Fig. 1.1). Neural networks have gained popularity in geophysics this last decade (Gret et al. 2000). Elavadi et al. (2003), Hajian (2008) and Hajian et al. (2011a) used a Hopfield neural network in order to obtain depth estimates of subsurface cavities. Osman et al. (2007) used forced neural networks for forward modeling of gravity anomaly profiles. Styles and Hajian (2012) used Generalized Regression Neural Networks (GRNN) for cavity depth estimation using microgravity data. Hajian and Shirazi (2015) used GRNN for depth estimation of salt dome using gravity data.

In the geophysical domain, neural networks have been used for waveform recognition and first-peak picking (Murat and Rudman 1992; McCormack et al. 1993); for electromagnetic (Poulton et al. 1992), magneto telluric (Zhang and paulson 1997), and seismic inversion purposes (Röth and Tarantola 1994; Langer et al. 1996; Calderón-Macías et al. 1998); neural networks (Elawadi 2001; Osman et al. 2007, Hajian et al. 2012); multi-adaptive neuro—fuzzy interference systems (Hajian et al. 2011b).

The fundamental important step is the mapping of a geophysical problem onto a neural network solution which can have various applications in geophysics for both non-potential and potential methods.

A. Hajian and P. Styles, *Application of Soft Computing and Intelligent Methods in Geophysics*, Springer Geophysics,
https://doi.org/10.1007/978-3-319-66532-0_1

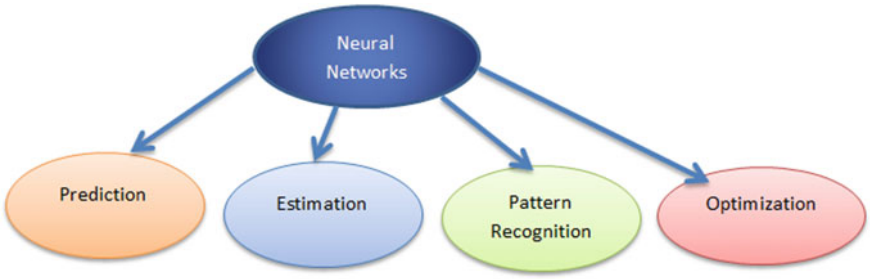


Fig. 1.1 Various fields of neural networks applications

1.2 A Brief Review of ANN Applications in Geophysics

A comprehensive look at all geophysical application for ANNs is impractical. However, a glance at the popular publications in journal books and conference proceedings provides illustrative examples. Geophysical applications include various disciplines which are short listed below:

- Petrophysics:
  - Modeling permeability in tight gas sands using well-log responses
  - predicting permeability from petrographic
  - Porosity estimation, lithofacies mapping,
- Seismic data processing:
  - Wave Form Recognition
  - Picking Arrival Times
  - Trace Editing
  - Velocity Analysis
  - Elimination of Multiples
  - Deconvolution
  - Inversion Of Velocity Model (Moya and Irikura 2010)
  - Random Noise Attenuation
  - Joint Inversion
  - Tracking Horizons And Classifying Seismic Traces
  - Initial Impedance, Impedance Model Estimation
  - Scatter Distribution (Neural Wavelets)
  - 4D Filtering (Fu 1999)
  - Seismic attribute analysis
- Gas detection from absorption and amplitude measurements versus offset (Clifford and Aminzadeh (2011)
- Cross-borehole geological interpretation model based on geotomography (Kumar et al. 2002)

- Geological pattern recognition and modeling from well logs and seismic data processing (Huang and Williamson 1994)
- Classification of seismic windows (Diersen et al. 2011)
- Rock Mass and Reservoir characterization
  - Horizon tracking and facies maps, time lapse interpretation
  - Predicting log properties, Rock/Reservoir characterization
  - Assessing Enhanced Oil Recovery (EOR) methods for reservoirs
- Seismology:
- Regional seismic event classification (Dysart and Pulli 1990)
  - Seismic discrimination
  - Non-linear Dynamic behavior of earthquakes
  - Automatic classification of seismic signals (Scarpetta et al. 2005)
  - Earthquake predictions (Reyes et al. 2013)
  - Prediction of seismicity cycles (Sharma and Arora 2005)

    Identification of earthquake phases under increased noise level conditions (Fernando et al. 2010)
    Earthquake magnitude prediction using artificial neural network (Alarifi et al. 2012)
  - Volcanology:
    - Predicting eruptions
    - Classification of eruptions
    - EM:
      - Detection of cavities and tunnels from magnetic anomaly data (Elawadi 2001)
      - Interpretation of electromagnetic elasticity soundings for near—surface objects (Poulton 1992)
      - Extracting IP parameters from TEM data (Poulton 1992)
      - Magnetic anomaly separation using cellular Neural Networks (Albora 2001)
      - Detection of Airborne Electromagnetic Method (AEM) Anomalies corresponding to dike structures
      - Interpretation of Geophysical surveys of Archeological sites
      - Piecewise Half-space interpretation
      - Inverse modeling of EM data with neural networks
      - Mineral potential mapping using EM, Gravity & geological data
      - Pattern recognition of subsurface EM images (Poulton 1992)
      - Forecasting solar activities (Uwamahoro et al. 2009)
      - Automatic detection of UXO from Airborne Magnetic Data (Salem et al. 2001)
- Solar Cycle prediction (Petrovay 2010)

- Gravity:
  - Depth estimation of cavities using microgravity data
  - Shape factor estimation of gravity anomalies
  - Prediction of linear trends
  - Adaptive learning 3D gravity inversion for salt-body imaging 4D gravity time series prediction
  - Attenuation of the effect of atmospheric temperature and pressure (as noise) on microgravity continuously record by gravimeters
  - Boundary detection for iron ore deposits (Wavelet Cellular Neural Networks)
- Geodesy:
  - Optimal Interpretation of the Gravity of the earth
  - Predicting vertical displacement of structures
  - Orbit propagation for small satellite missions
  - Sea-level prediction using satellite altimetry
  - Determination of structure parameters (Kaftan and Salk 2009)
- Resistivity:
  - Layer boundary picking, locating layer boundaries with unfocused resistivity tools
  - Inversion of DC resistivity data (EL-Qady and Ushijima 2001)
  - Obtaining formation resistivity and layer thickness from vertical electrical sounding (VES)

The most common Neural Networks used in geophysics are:

- Back-propagation
- SVM (Support Vector Machine)
- GRNN (General Regression Neural Networks)
- Cellular Neural Networks, Wavelet Cellular neural networks (CNN)
- Modular Neural Networks (MNN)
- Forced Neural Networks (FNN)
- Radial Basis Function Neural Networks (RBF)

In this chapter we first explain neural network concepts and introduce some of the important types of neural networks which are most common in geophysical problems.

1.3 Natural Neural Networks

The human brain is a complex organism with great powers of learning and generalization from specific data. The human brain is one of the great mysteries of our time and scientists have not reached a consensus on exactly how it works. Two theories of the brain exist namely: the “grandmother cell theory” and the

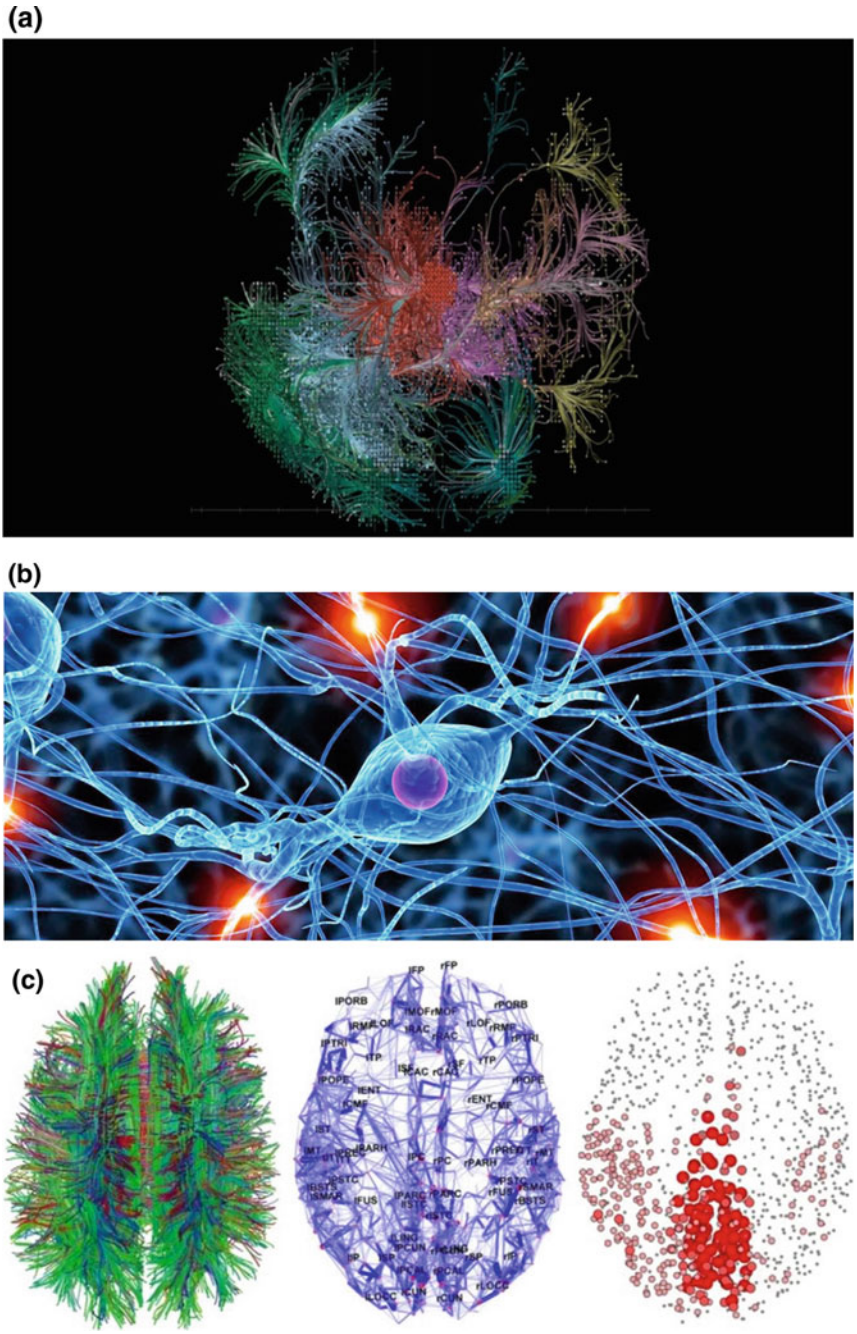
"distributed representation theory". The first theory asserts that individual neurons have high information capacity and are capable of representing complex concepts such as 'your grandmother' or even 'Jennifer Aniston'. The second theory asserts that neurons are much simpler and the representation of complex objects is distributed across many neurons. Artificial neural networks are loosely inspired by the second theory. Up to now much research has been done by neuroscientists to explore the nature of the human brain operation and structure. These scientists have explored the map of a mouse brain which is shown in Fig. 1.2a. This is the First Detailed Map of a Mammal's Neural Network; if this looks like an incredibly complex wiring diagram to you, it's because it is: you're looking at the Allen Mouse Brain Connectivity Atlas, the first detailed map of any mammal's neural network. It's not a full connectome—the name given to maps of every single interconnection between neurons in a brain—but it's the most detailed rendering of interconnections in any mammalian brain yet. It traces connections between tiny cubes, called voxels, of brain tissue containing between 100 and 500 neurons. Hongkui Zeng and colleagues at the Allen Institute for Brain Science in Seattle, Washington, injected the brains of 469 mice with a virus that introduced a fluorescent protein into the neural network. Because each animal was injected at a slightly different location, when taken together, the fluorescing proteins gave a snapshot of the network's shape. Next, the team diced up each brain into 500,000 pieces each measuring 100 cubic micrometers. Based on the strength of fluorescence in the cubes, they generated a 3D map of how each of the 469 different signals spread through the brain's thoroughfares and quieter by roads (www.gizmodo.com.au).

Human brains have a lot of cells namely "neurons" (Fig. 1.2b, c) and the number of these elements is approximately 10^{11} neurons with perhaps 10^{15} interconnections over transmission paths that may range a metre or more.

1.4 Definition of Artificial Neural Network (ANN)

> A single neuron in the brain is an incredibly complex machine that even today we don't understand. A single "neuron" in a neural network is an incredibly simple mathematical function that captures a minuscule fraction of the complexity of a biological neuron. So to say neural networks mimic the brain, that is true at the level of loose inspiration, but really artificial neural networks are nothing like what the biological brain does.—Andrew Nigrin.

An artificial neural network is a processing method which is inspired by how the human brain and the nervous system are interconnected with neurons. Azoff (1994) states that, "A neural network may be considered as a data processing technique that maps, or relates, some type of input streams of information to an output stream of data". Nigrin (1993) described a neural network based on circuit concepts as" a circuit composed of a very large number of simple processing elements that are


(a)
(b)
(c)

◂**Fig. 1.2** **a** The first detailed map of a mammal's neural network (*source* http://www.gizmodo.com.au/2014/04/this-is-the-first-detailed-map-of-a-mammals-neural-network/). **b** Schematic of neurons in human brains (*source* http://detechter.com/10-interesting-facts-about-the-human-brain). **c** Some very interesting views of the brain as created by state of the art brain imaging techniques (*source* http://www.turingfinance.com/misconceptions-about-neural-networks/#comment-10376)

neutrally based. Each element operates only on local information. Furthermore, each element operates asynchronously; thus there is no overall system clock".

There are numerous alternative definitions of artificial neural networks. One of the commonest is: "Neural networks are composed of simple elements operating in parallel. These elements are inspired by biological nervous systems. As in nature, the network function is determined largely by the connections between elements. We can train a neural network to perform a particular function by adjusting the values of the connections (weights) between elements" (Schalkoff 2011).

An artificial neural network is composed of a number of interconnected units (artificial neurons) each unit has an input/output (I/O) characteristic and implements a local computation or function. The output of any unit is determined by its I/O characteristics, its interconnections to other units, and (possibly) external inputs. Although "hand crafting" of the network is possible, the network usually develops an overall functionally through one or more forms of training. It is necessary to mention that numerous alternative definitions about ANN exist and the one we select above is a somewhat generic definition.

Commonly, neural networks are adjusted, or trained, so that a particular input leads to a specific target output. Such a situation is shown in Fig. 1.3. There, the network is adjusted, based on comparison of the output and the target, until the network output matches the target. Typically many such input/target pairs are used, in this supervised learning, to train a network.

The overall computing model of an ANN consists of a reconfigurable interconnection of simple elements namely "neurons". The neuron model will be described in detail in the next section. Corresponding to the interconnection

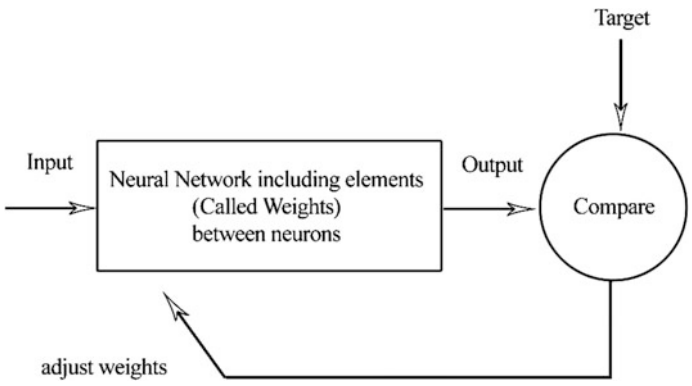


Fig. 1.3 Diagram of a neural network operation

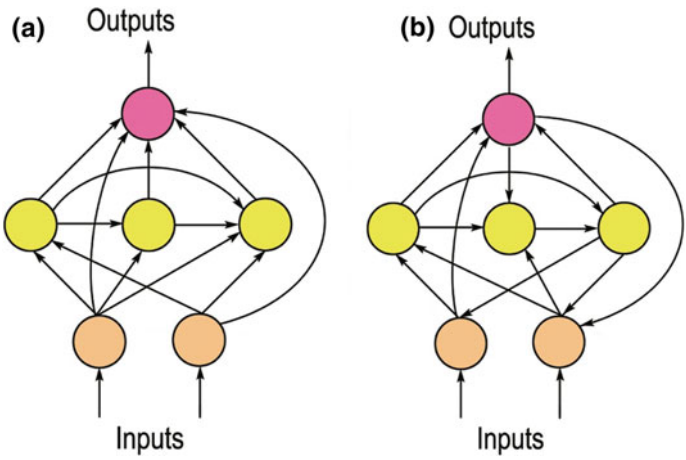


Fig. 1.4 **a** Non-recurrent, **b** recurrent neural networks (Hajian et al. 2012)

direction ANNs can be placed into one of three classes based on their feedback link connection structure ANN's topology: non-recurrent (e.g. feed forward) recurrent (global feedback connections). Local recurrent structure (local feedback connections, e.g. cellular NN).

The non-recurrent ANN contains no closed interconnection paths but in the recurrent ANN the interconnection has arbitrary interconnection flexibility which allows closed-loop (feedback) paths to exist. This allows the network to exhibit far more complex temporal dynamics compared with the (open-loop) strategy, illustrated in Fig. 1.4a, b.

1.5 From Natural Neuron to a Mathematical Model of an Artificial Neuron

Artificial neural networks are a very much simplified model of the natural neural networks of the human nervous system. As we mentioned in the last sections the biological nervous system is built of cells called neurons. Each neuron shares many characteristics with the other cells in the human body and has the capability to receive process and transmit electrochemical signals over the neural pathways that comprise the brain's communication system. The schematic drawing of a Biological Neuron is shown in Fig. 1.5.

1. Dendrites: extend from the cell body out to other neurons where they receive signals at a connection point called "Synapse".
2. The axon is a single long fiber that carries the signal from the cell body out to other neurons.

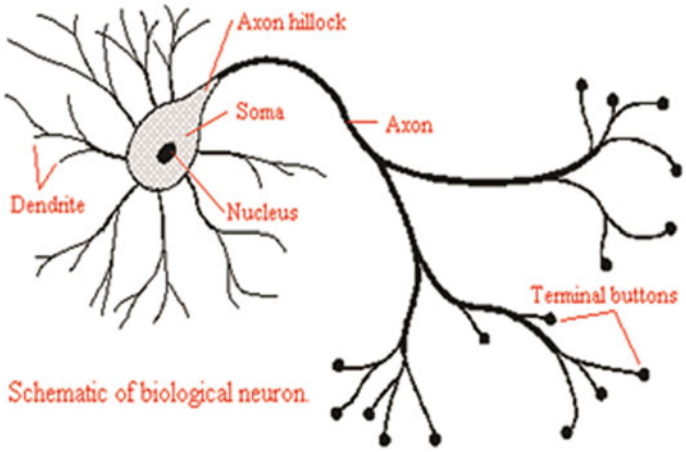


Fig. 1.5 Schematic of a biological neuron. Redrawn from Van der Baan and Jutten (2000)

On the receiving side of the synapse, these inputs are conducted through the cell body out to other neurons. There they are summed; some inputs tend to excite the cell, others tend to inhibit its firing.

When the cumulative excitation in the cell body exceeds a threshold, the cell fires, sending a signal through the axon to other neurons. This basic functional outline has many complexities and exceptions; nevertheless, most Artificial Neural Networks contain only these simple characteristics (Gret and Klingele 1998). Artificial neural networks comprise a set of highly interconnected but simple processing units called nodes which are such neurons. The nodes are arranged in a series of layers that are interconnected through functional linkages. The mathematical model of a simple neuron or 'perception' is presented in Fig. 1.6.

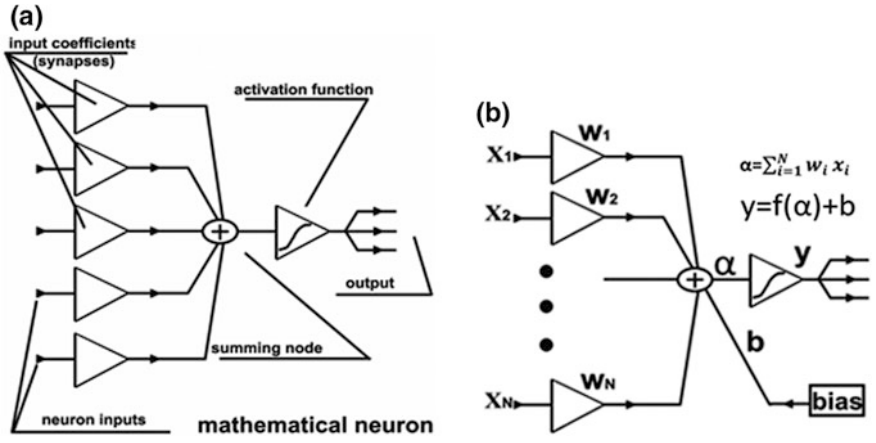


Fig. 1.6 **a** Mathematical model of a 'perception' or 'neuron' **b** the weighted sum of the inputs is rescaled by an activation function. Redrawn from Van der Baan and Jutten (2000)

As shown in Fig. 1.6 the neuron consists of a set of inputs:

$$X = [x(1),\ x(2), \ldots,\ x(n)]$$

and one output. Functionally each input x(i) is weighted or multiplied by a weight function W(i) and summed in a process equivalent to the mathematical process known as Convolution. This is called net input of the neuron:

$$net = \sum_{i=1}^{N} [x(i)w(i)]$$

Then an activation function is computed. This simulates the firing of the nerve cells by inputs from other never cells. The Full diagram of steps from a biological neuron to its mathematical model is shown in Fig. 1.7.

The activation function can be a: step function (hard-limiter), arc-tangent sigmoidal function, hyperbolic tangent sigmoid etc. (see Fig. 1.8). The various kind of activation functions or transfer functions are listed in Table 1.1 with their related MATLAB commands.

The 'sigmoid' activation function is very commonly used for geophysical applications because many applications require a continuous valued output rather than the binary output produced by the hard-limit. In addition, it is particularly well-suited to pattern recognition problems because it produces an output between 0 and 1 that can often be interpreted as a probability estimate (Richard and Lippmann 1991).

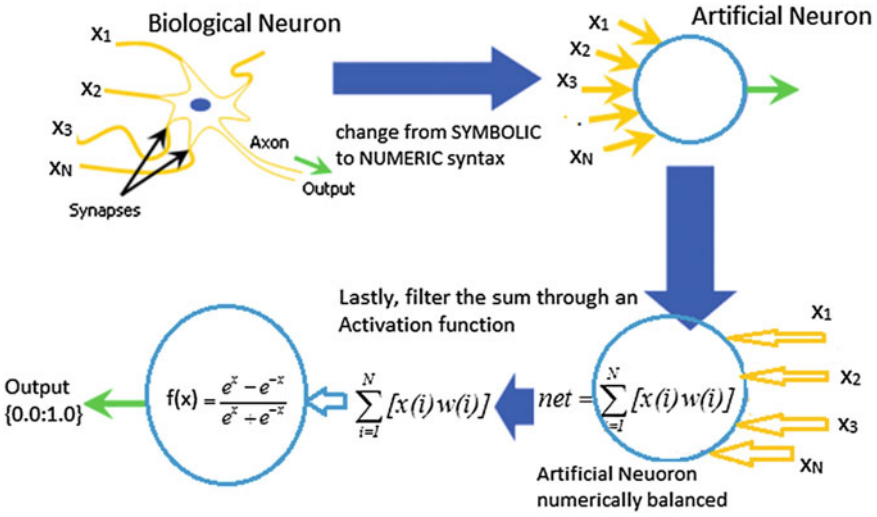


Fig. 1.7 Full diagram of steps from a biological neuron to its mathematical model (here, the activation function is a tangent sigmoid, see Table 1.1)

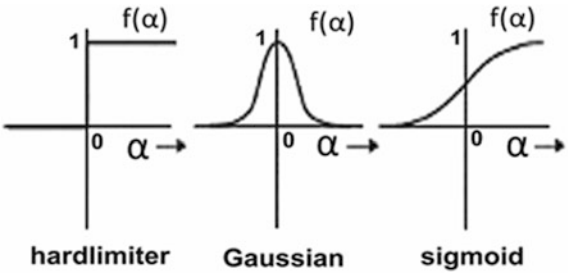


Fig. 1.8 Some of common activation functions

Table 1.1 Some of activation functions with related relations and Matlab commands

Function	Equation		Matlab command
Hard limit	$y = \begin{cases} 0 \\ 1 \end{cases}$	$x<0$ $x \geq 0$	Hardlim
Symmetrical hard limit	$y = \begin{cases} -1 \\ 1 \end{cases}$	$x<0$ $x \geq 0$	hardlims
Saturating linear	$y = \begin{cases} 0 \\ x \\ 1 \end{cases}$	$x<0$ $0<x<1$ $x \geq 1$	–
Symmetrical saturating linear	$y = \begin{cases} -1 \\ x \\ 1 \end{cases}$	$x<-1$ $-1<x \leq 1$ $x>1$	elliotsig
Log-sigmoid	$y = \frac{1}{1+e^{-x}}$		logsig
Hyperbolic tangent sigmoid	$y = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$		tansig
Linear	$y = x$		purelin
Positive linear	$y = \begin{cases} 0 \\ x \end{cases}$	$x<0$ $x \geq 0$	–
Competitive	$y = \begin{cases} 1 \\ 0 \end{cases}$	Neuron with Max X other neurons	–

Before the activation function is computed, a bias value (v) is added which serves as a threshold for the activation function. For example if the activation function is a sigmoidal function the output of the neuron will be:

$$b(j) = \frac{1}{\left\{1 + exp\left[-\frac{net+v}{d}\right]\right\}} \tag{1.1}$$

where "d" is a sigmoid shaping factor, "v" is the bias factor and net is the weighted average of neuron inputs. If "d" is a high number then the sigmoid will be a gently

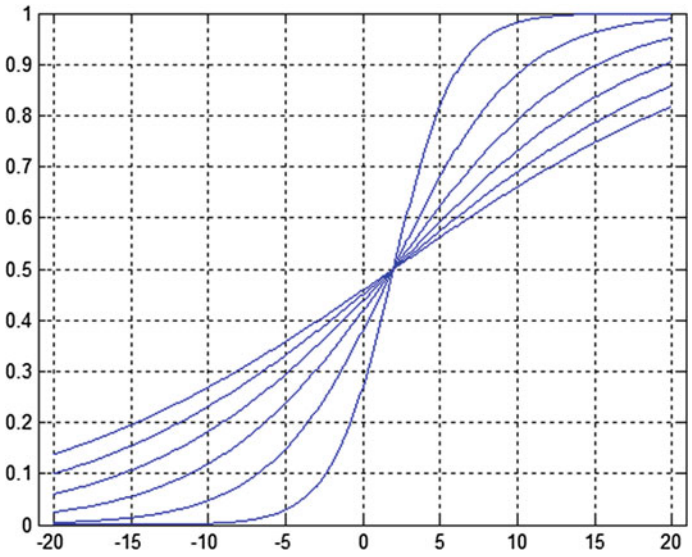


Fig. 1.9 Sigmoid activation functions for different values of shaping factor (as it can be seen the bias factor is fixed at 2)

varying function. For lower values of the shape factor the sigmoid will take a steeper form. To plot the neuron output in MATLAB run below codes the results are illustrated in Fig. 1.9.

```
*****Matlab codes to generate sigmoid activation functions with different
shape factors**************************************


x=[-20:0.1:20];
bias=2;
for j=1:6
   d=[2 4 6 8 10 12];
        f=-(x-bias)/(d(j));
ff=exp(f)+1;
     b=ff.^(-1);
axis([-21 21 0 1])
     plot(x,b)
   hold on
end
grid
```

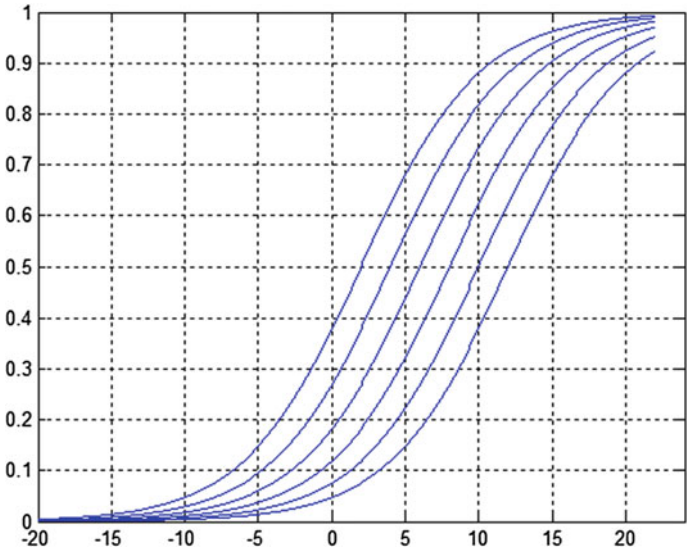


Fig. 1.10 Sigmoid activation function for different values of biasing (shape factor is fixed; d = 4)

It is necessary to mention that the bias included in 'net', offsets the origin of the activation function, for example if the transfer function the 'logistic' or 'sigmoid' (meaning s-shaped) is chosen then the function is plotted for higher and lower values of bias in MATLAB. By running the MATLAB codes below this can obviously be seen (Fig. 1.10).

```
****Matlab code to draw sigmoids with different biases****
x=[-20:0.1:22];
d=4;
bias=[2 4 6 8 10 12];
for j=1:6
           f=x-bias(j))/(d);
ff=exp(f)+1;
     b=ff.^(-1);
axis([-20 24 0 1])
     plot(x,b)
   hold on
end
grid
```

The purpose of a non-linear activation function is to increase the computational power of a multi-layer network because if we use a linear activation function for a multi-layer net then an equivalent one-layer network can be replaced. So, the non-linear activation functions are vital to expansion of the network's capability beyond that of the single-layer network. Training a neuron means computing the values of weights: $w(i)$ and biases so that it will correctly image the inputs into true-outputs.

1.6 Classification into Two Groups as an Example

Suppose that we want to classify the seismic signals of earthquake events into two classes: class A as hazardous conditions and class (B) as safe conditions.

This is not a complete or comprehensive classification of the seismic events but as a simple example of classification the perception output is "+1" if X belongs to class A and is "0" if X belongs to class B, on the other hand:

$$X.W = \begin{cases} +1 & if \quad X \in A \\ 0 & if \quad X \in B \end{cases} \tag{1.2}$$

If the input data are noisy, which is generally the case, we will never achieve exactly 0 and 1, and the output will be somewhere in-between. So it is necessary to include some subjective threshold value to decide where the classification of "A" ends and classification of "B" begins.

For example; if the output is 0.5 which class will the event be ascribed to: A or B? It is obvious that this depends on the nature and type of the classification. Here we design a neuron, which, when trained can perform as a linear discriminator, similar to the well-known single or multi-channel wiener operator (Taner 1995).

The single neuron or perceptron was developed by Widrow (1962) and he named it the 'linear perceptron'.

Let $X = \{x(1), x(2), \ldots, x(n)\}$ represents the seismic event feature vector as an input training pattern. To classify the event into class A or B we must solve the vector equation:

$$X.W = b \tag{1.3}$$

where W is the column vector of weights with M elements and $b = 1$ specifies that the corresponding X set belongs to the class A, and $b = 0$ specifies that the corresponding X set belong to the class B. Matrix X has N columns and M rows because:

Pattern 1:

$$\begin{array}{l} X_1 = \{x(1,1) \quad x(1,2) \quad \ldots. \quad x(1,N)\} \\ X_2 = \{x(2,1) \quad x(2,2) \quad \ldots. \quad x(2,N) \\ . \\ . \\ . \\ . \\ X_M = \{x(M,1) \quad x(M,2) \quad \ldots. \quad x(M,N) \\ \overset{train}{x} = \begin{pmatrix} x_{11} & x_{12} & \ldots & x_{1N} \\ x_{21} & x_{22} & \ldots & x_{2N} \\ \vdots & & & \vdots \\ x_{M1} & x_{M2} & \ldots & x_{MN} \end{pmatrix}_{M \times N} \end{array} \tag{1.4}$$

If $M < N$, then we can solve this matrix equation by the classical least mean errors squares method. We can obtain the normal equation square matrix by pre-multiplying both sides by transpose of the X matrix:

$$X^{\mathrm{T}}.X.W = X^{\mathrm{T}}b \tag{1.5}$$

and solve for W:

$$W = (X^{\mathrm{T}}X)^{-1}.X^{\mathrm{T}}.b \Rightarrow W = \hat{x}.b \tag{1.6}$$

where $\hat{x}$ is the pseudo- inverse of the original rectangular $(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}$ matrix:

$$\hat{x} = (X^{\mathrm{T}}X)^{-1}.X^{\mathrm{T}} \tag{1.7}$$

In theory, this inverse can be computed directly. However we may get some non-realistic values which do not represent the real situation. The inverse is computed by an iterative procedure called the linear perceptron algorithm which leads us to the Delta rule.

Training the network means computing a single set of weights (W) to yield the correct set of outputs 'b' for all input patterns (x(p): input training data). In the Delta rule an arbitrary set of values $W^{(h)}(i)$ is first randomly considered to update by the following rule:

$$W^{(h+1)}(j) = W^h(j) + \rho[b(j) - W^h(j)x(j)].x(j) \tag{1.8}$$

The updating procedure is continued until all the patterns are classified correctly, at which time $\left[b(j) - W^h(j)x(j)\right]$ becomes zero or very small.

In some practical cases this cannot be reached, hence the iteration should be stopped when the sum of the squares of errors (SSE) or (RMSE): Rout Mean Square Error reaches some prescribed threshold value.

1.7 Extracting the Delta-Rule as the Basis of Learning Algorithms

The error function, as indicated by the name least mean squares, is the summed squared error which is the total error E as defined to be:

$$E = \sum_p E^p = \frac{1}{2}\sum_p (d^p - y^p)^2 \tag{1.9}$$

where the index P ranges over the set input patterns and E^p represent the error on pattern P. The *LMS* procedure finds the values of all the weights that minimize the error function by a method called gradient descent. The idea is to make a change in the weight proportional to the negative of the derivative of the error as measured on the current pattern in respect to each weight:

$$\Delta_p W_j = -\gamma \frac{\partial E^p}{\partial W_j} \tag{1.10}$$

where γ is a constant of proportionality. The derivative is:

$$\frac{\partial E^p}{\partial W_j} = \frac{\partial E^p}{\partial y_p} \cdot \frac{\partial y^p}{\partial W_j} \tag{1.11}$$

Because of the linear unites:

$$\frac{\partial y^p}{\partial W_j} = x_j \tag{1.12}$$

Such that:

$$\Delta_p W_j = -\gamma \delta^p x_j \tag{1.13}$$

where $\delta^p = d^p - y^p$ is the difference between the target output and the actual output for pattern P_0 and:

x_j^p = the j-th element of the p-th input pattern vector

Expression 9 can be written in the form:

$$\Delta(W) = \eta.\delta.\chi \tag{1.14}$$

where δ is the difference between the desired output and the computed actual output produced by the perceptron. This is the "delta-rule" expression which states that the change in the weight vector should be proportional to the delta (the error) and to the input pattern.

Using this simple equation, the **Generalized Delta Rule** is finally derived for updating the weights in the multilayer reception neural networks.

This is the common rule that uses the difference between the actual and desired activation for adjusting the weights:

$$\Delta W_{jh} = \gamma y_j (d_h - y_h) \tag{1.15}$$

where W_{jh} is the weight connection from unit j to unit h and γ is a positive constant of proportionality representing the learning rate, in which d_h is the desired activation provided by a teacher. This is often called the *Widrow-Hoff* rule or the *delta* rule.

1.8 Momentum and Learning Rate

True gradient descent requires that infinitesimal steps are taken. The constant of proportionality is the learning rate γ. For practical purposes we choose a learning rate which is as large as possible without leading to oscillation. One way to avoid oscillation at large γ is to make the weight change dependent on the past weight change by adding a momentum term:

$$\Delta W_{jh}(t+1) = \gamma \delta_h^p y_j^p + \alpha \,.\, \Delta W_{jh}(t) \tag{1.16}$$

where $t + 1$ is the step after step t, and α is a constant which determines the effect of the previous weight change.

The role of the momentum term is shown in Fig. 1.11. When no momentum term is used it takes a long time before the minimum is reached with a low learning rate, whereas for high learning rates the minimum is never reached because of oscillation. When adding the momentum term, the minimum is reached faster (Kröse and Smagt 1996).

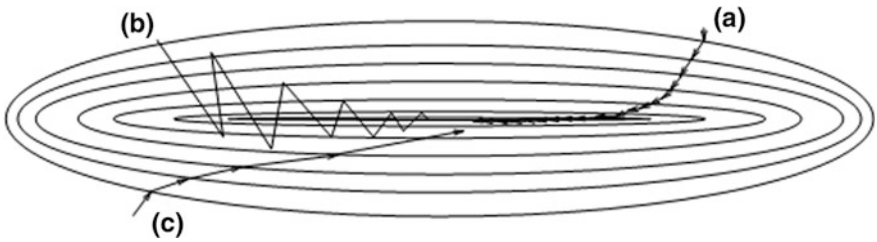


Fig. 1.11 The descent in weight space: **a** for small learning rate **b** for large learning rate: note the oscillations and **c** with large learning rate and momentum term added (Kröse and Smagt 1996)

1.9 Statistical Indexes as a Measure of Learning Error

To finalize the procedure of updating weights there are several error indexes which are listed in Table 1.2 with the related equation for each one. The weight updating usually continues until the minimum value for the error index is very close to zero (or even zero in very especial problems). The most common error index used in this way is the MSE and RMSE. It's obvious that when the MSE is a minimum the RMSE (which is the root of MSE) will also be a minimum.

1.10 Feed-Forward Back-Propagation Neural Networks

A Feed-Forward Back-Propagation (FFBP) neural network is a very common type of supervised neural network which is shown in Fig. 1.12. As mentioned, for a neuron, all inputs and nodes are collected at each hidden node after being multiplied by

Table 1.2 Various statistical indexes to measure the learning error of a neural network

Error index	Full name	Equation
SSE	Sum squared error	$\sum (x - \hat{x})^2$
MSE	Mean squared error	$\sum \frac{(x-\hat{x})^2}{N}$
RMSE	Root mean squared error	$\sqrt{\sum \frac{(x-\hat{x})^2}{N}}$
MAE	Mean absolute error	$\frac{\sum \lvert x-\hat{x} \rvert}{N}$

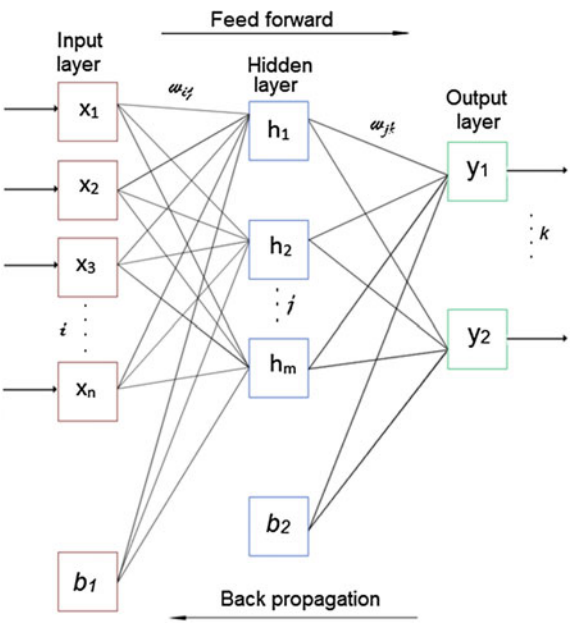


Fig. 1.12 Schematic diagram of an FFBP neural network structure (Hajian et al. 2012)

weights. Then, a bias is attached to this sum, and it is transformed through a certain function, and transferred to the next layer.

There are various types of initial transfer function that can be used as activation functions such as linear functions, sigmoid and hyperbolic tangents. Historically, the Heaviside or hard-limiting function was used. However, as explained previously this particular activation function gives only a binary output (i.e., 1 or 0, meaning yes or no). Moreover, the optimum weights are very difficult to estimate since this particular function is not continuously differentiable. As mentioned in the previous sections, in FFBP neural networks the sigmoid function is mostly used as the activation function for all the neurons in the layers most of the times. The main reason is that the sigmoid function is a continuously differentiable and monotonically increasing function that can best be described as a smooth step function. Equation (1.17) describes the sigmoid activation function as expressed by (Van der Baan and Jutten 2000):

$$f_s(\alpha) = (1 + e^{-\alpha})^{-1} \tag{1.17}$$

As the forward processing step terminates at the output layer, the difference between the errors of the network output and those of the desired output can be calculated simply. The error at the output layer propagates backward to the input layer through the hidden layer in the network to obtain the final desired outputs (Günaydin and Günaydin 2008). While the backward procedure is running, all the weights are adjusted in accordance with an error-correction rule (Haykin 1999, see Eq. 4). Figure 1.13 shows a flowchart of the feed forward back propagation algorithm (Hajian et al. 2012).

In a multi-layer perceptron neural network with one hidden layer and an architecture of (n, m, L) there is one input layer having n neurons; one hidden layer having m neurons, and one output layer having L neurons $y_1, y_2, \ldots, y_L$; Each node

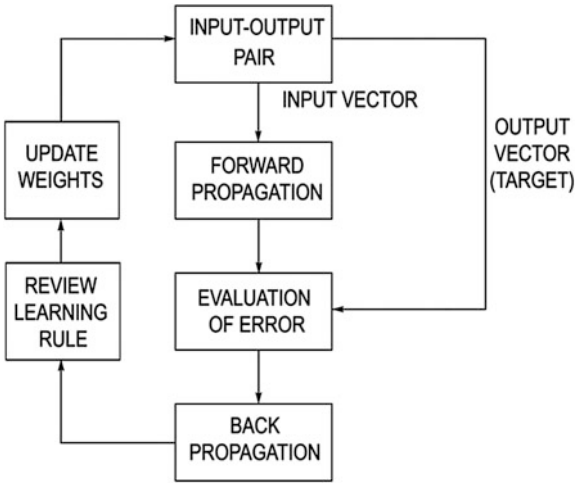


Fig. 1.13 Flowchart of general feed forward back propagation algorithm (Hajian et al. 2012)

in a layer is connected to that of the next layer by weights w, so that the neuron p is connected to neuron r in the next layer by weight w_{pr}. Consequently the weights of the Input-to-Hidden layer and that of the Hidden-to-Output layer are w_{ij} and w_{jk}, respectively. Mathematically the k-th output of network y_k is computed by the following equation:

$$y_k = \tilde{\mathbf{f}}\left[\sum_{j=1}^{m} w_{jk}.\mathbf{f}\left[\sum_{i=1}^{n}(w_{ij}.x_i + b_1) + b_2\right], \quad k = 1, 2, \ldots, L \tag{1.18}$$

where b_1 is the bias of the first layer, b_2 is that of the second layer, f is the activation function between input and hidden layers, and $\tilde{\mathbf{f}}$ is the activation function between the hidden and output layers. The tangent sigmoid (tan-sigmoid), logarithmic sigmoid) log-sigmoid) and linear activation functions (Haykin 1999) are the most common functions used for both **f** and $\tilde{\mathbf{f}}$ to obtain the best performance.

The back propagation network (BPN) is used to train a MLP neural network and was proposed by Rumelhart et al. (1986). "Training" of a neural network model means tuning the weights of the neuron connections in all the layers so that the outputs of the neural network will be as close as possible to the observed targets. To measure how closely the outputs correspond to the observed targets a global error function is defined, so the aim of training a supervised neural network is to iteratively minimize the global error or Mean Squared Error (MSE). The global error E will be defined generally as:

$$E = \frac{1}{2p}\sum_{p=1}^{P}\sum_{k=1}^{1}\left(T_{pk} - y_{pk}\right)^2, \quad p = 1, 2, \ldots, P \tag{1.19}$$

where T_{pk} is the target output (observed) at the k-th neuron of the p-th pattern, y_{pk} is the predicted output at k-th output neuron of the p-th pattern and P is the total number of training patterns.

The global error (E) at the output layer propagates backward from the output to the hidden layer in order to adjust the weights in each layer of the network during the iteration. The iterations are continued until a specified convergence is reached, or a given number of iterations are over.

Each step in the learning process is called a Learning Epoch. There are various types of learning algorithms such as:

- LM: Levenberg-Marquardt
- BFG: BFGs quasi-Newton back propagation
- BR: Bayesian regulation back propagation
- CGB: Conjugate gradient with Powell-Beale restarts
- CGF: Conjugate gradient back propagation with Fletcher-Reeves updates
- CGP: Conjugate gradient back propagation with Polak-Ribiere updates
- GD: Gradient descent back propagation
- GDM: Gradient descent with momentum back propagation

– GDA: Gradient descent with adaptive learning rate back propagation
– GDX: Gradient descent with momentum and adaptive back propagation
– OSS: One Step Secant back propagation

Among these above learning algorithms, the Levenberg-Marquardt (L-M) algorithm (Levenberg 1944; Marquardt 1963) is one the most commonly used algorithms. As the book would be excessively long if we explained all these learning algorithms, in this section we explain the L-M learning algorithm, and the reader is referred to neural networks book texts (listed in the end of chapter) and the Matlab guide for the other algorithms.

The L-M algorithm minimizes E while trying to keep the step between the old weights configuration (w_{new}) and the updated one (w_{old}) 'small enough'. This algorithm can be written as follows (Günaydun and Günaydun 2008):

$$w_{new} = w_{old} - [J^T J + \gamma I]^{-1} J^T E(w_{old}) \tag{1.20}$$

where J is the Jacobian function of the error E, I is the identity matrix and γ is the parameter used to define the iteration step value (Panizzo and Briganti 2007).

For $\gamma = 0$ it converges to become the Gauss-Newton method. For very large γ the L-M algorithm becomes the steepest decent secure convergence. The adaptive learning rate, which changes dynamically during the training stage, is used here. The learning rate lies in the range between 0 and 1. For each epoch, if performance decreases toward the goal, then the learning rate is increased by the learning factor increment. If performance increases, the learning rate is adjusted by the learning factor decrement. When training with the L-M method, the increment of weights Δw can be obtained from equation above as follows (Hajian et al. 2012):

$$\Delta w = [J^T J + \gamma I]^{-1} J^T E(w_{old}) \tag{1.21}$$

The sum of the squared errors between the target outputs and the network's simulated outputs (see Table 1.2), defined as the global error: E is minimized. The global error for each iteration with updated weight is depicted with E (w). Throughout all the FFBP simulations the performance goal is assumed to be very near to zero (or even zero). In particular, γ is multiplied by the decay rate β ($0 < \beta < 1$) whenever E (w) decreases, whereas γ is divided by β whenever E(w) increases in a new step.

The standard L-M training algorithm can be performed using the following pseudo-codes (Hajian et al. 2012):

****

1. Initialize the weights and parameter γ (γ is appropriate).
2. Compute the sum of the squared errors over all inputs E (w)
3. Solve (7) to obtain the increment of weights Δw
4. Recomputed the sum of squared errors F (w) Using $w + \Delta w$ as the trial w, and judge

IF trial E (w) < E (w) in step 2 THEN
$W = w + \Delta w$
$\Upsilon = \Upsilon . \beta \quad (\beta = 0.1)$
Go back to step 2
ELSE
$\Upsilon = \frac{\Upsilon}{\beta}$
Go back to step 4
END IF
****

1.11 A Guidance Checklist for Step-by-Step Design of a Neural Network

There are 9 main steps in designing a neural network estimator model which are described in Table 1.3.

1.12 Important Factors in Designing a MLP Neural Network

The most important factors which should be determined for a multi-layer perceptron neural network are:

- the number of hidden layers
- the number of hidden neurons
- the number of training data

Table 1.3 Seven steps in designing an ANN estimator model

Step 1:	Variable selection
Step 2:	Data collection and processing (preparing training data)
Step 3:	Assemble training and test data
Step 4:	ANN paradigm – Number of hidden layers – Number of hidden neurons – Number of output neurons – Transfer function
Step 5:	Evaluation criteria
Step 6:	ANN training Number of iterations for training
Step 7:	Testing ANN with new synthetic inputs without noise
Step 8:	Testing ANN with new synthetic inputs with noise
Step 9:	Testing ANN with new real data

1.12.1 Determining the Number of Hidden Layers

Determining the number of hidden layers and the number of neurons in each hidden layer is a considerable task. The number of hidden layers is a critical step which is usually determined first.

It is obvious that the number of hidden layers depends on the complexity of the relationship between the input parameters and the output value. If the relationship between the inputs and output is linear the network does not need more than one hidden layer, however, most problems only require one hidden layer.

It is unlikely that any practical problem will require more than two hidden layers. The more complex the problem the greater the probability of requiring more than one hidden layer. Cybenko (1989) suggested that one hidden layer is enough to classify input patterns into different groups.

Chester (1990) implied that a two hidden layer system should perform better than a one hidden layer network. More than one hidden layer systems can be useful for certain architectures, such as cascade correlation (Fahlman and Lebiere 1990). The main reason why multi-layer networks can sometimes provide better training with a lower generalization error is that the extra degrees of extra parameters can decrease the chance of becoming stuck in local minima or on a "plateau". The most commonly used training methods for back propagation networks are based on gradient descent; that is, error is reduced until a minimum is reached whether this is a local minimum or a global. However, there isn't any clear theory which can inform you as to how many hidden layers are needed to approximate any given function.

The rule of thumb in deciding the number of hidden layers is normally to start with one hidden layer (Lawrence 1994). If one hidden layer does not train well, then you increase the number of neurons but adding more hidden layers should be a last resort (Park 2011).

1.12.2 Determination of the Number of Hidden Neurons

Selecting the number of hidden neurons is problem-dependent. For example, any network that requires data compression must have a hidden layer with a smaller number of hidden neurons than the input layer (Swingler 1996).

A conservative approach is to select a number within the range but smaller than the number of output neurons. In Table 1.4 some rules of thumb to select the number of neurons in hidden layer are listed.

It can be seen from this table that the general wisdom concerning selection of initial number of hidden neurons is somewhat contradictory.

A good rule of thumb is to start with the number of hidden neurons equal to half of the number of input neurons and then either add neurons if the training error remains above the training error tolerance, or reduce neurons if the training error quickly drops to the training error tolerance (Park 2011).

Table 1.4 Rule of thumbs to select the number of neurons in hidden layer (Redrawn after Park 2011)

Formula	Comments
$h = 2i + 1$	Hecht-Nelson (1987) used Kolmogorov's theorem which and function of i variables may be presented by the superposition of set of 2i + 1 univariate functions-to derive the upper bound for the required number of hidden neurons
$H = (i + o)/2$ $\frac{N}{10} - i - o \leq h \leq \frac{N}{2} - i - o$	Lawrence and Fredrickson (1998) suggested that a best estimation for the number of hidden neurons is to half the sum of inputs and outputs. Moreover, they proposed the range of number of hidden neurons
$H = i \log_2 P$	Mirchandani and Cao (1989) proposed an equation for best number of hidden neurons

1.13 How Good Are Multi-layer Per Feed-Forward Networks?

It is clear that the approximation results of a MLP are not perfect. Kröse and Smagt (1996) suggest that the resulting approximation error is influenced by:

1. The learning algorithm and the number of iterations determine how well the error on the training set is minimized.
2. The number of learning samples; this determines how well the training samples represent the actual function.
3. The number of hidden units determines the 'expressive power' of the network. For 'smooth' functions only a few hidden units are needed; for wildly fluctuating functions more hidden units will be needed.

Let us first define an adequate error measure. All neural network training algorithms try to minimize the error of the set of learning samples which are available for training the network. The average error per learning sample is defined as the 'learning error rate':

$$E_{learning} = \frac{1}{P^{learning}} \sum_{P=1}^{P_{learning}} E^P \tag{1.22}$$

where E^P is the difference between the desired output value and the actual network output for the learning samples:

$$E^P = \frac{1}{2} \sum_{o=1}^{N_o} (d_0^P - y_0^P)^2 \tag{1.23}$$

This is the error which is measurable during the training process.

The actual error of the network is different from the training samples. The difference between the desired output value and the actual network output should be integrated over the entire input domain to give a more realistic error measure. The integral can be determined if a large set of samples is available as the 'test' set and the '*test error rate*' is defined as the average error of the test set:

$$E_{test} = \frac{1}{P^{test}} \sum_{P=1}^{P_{test}} E^P \tag{1.24}$$

1.14 Under Training and Over Fitting

In order to train the neural network well, the number of data sets must be carefully chosen. The major problem during the training process of the neural network is the possible over-fitting of the training data. That is, during a certain training period, the network no longer improves in its ability to solve the problem. In this case the training becomes trapped in a local minimum, leading to ineffective results and indicating a poor fit of the model, namely over-fitting.

An over-fitted model could approximate the training data well but function poorly for the validation data set. Also if the number of training data is low then the trained neural network will be an under-fitted model which generalizes to the validation data set well but approximates the training data poorly. In this way we explain in the next subsection the importance and effect of the learning data set size on the performance of training.

Underfitting occurs when a statistical model or machine learning algorithm cannot capture the underlying trend of the data. It occurs when the model or algorithm does not fit the data enough. Underfitting occurs if the model or algorithm shows low variance but high bias (to contrast the opposite, overfitting from high variance and low bias) is often a result of an excessively simple model (Frost 2015).

1.15 To Stop or not to Stop, that Is the Question! (When Should Training Be Stopped?!)

One important question that arises when we are running a training algorithm is when should the training be stopped? It seems to be a good idea to stop training when a local minimum is attained or when the convergence rate has become very small, i.e., improvement from iteration to iteration is zero or minimal. However, Geman et al. (1992) show that this leads to overtraining, i.e., memorizing of the training set (while you want network to generalize not memorize): now the noise is being fitted, not the global trend. Hence, the weight distribution obtained may be optimal for the training samples, but it will result in bad performance in general.

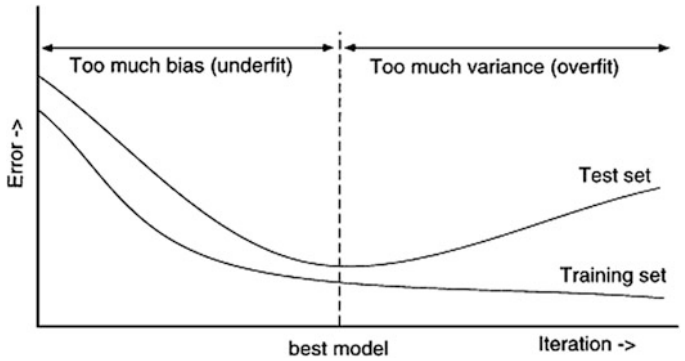


Fig. 1.14 Generalization versus training error. Adapted from Moody (1994) and redrawn after Van der Baan and Jutten (2000)

A similar phenomenon occurs in tomography problems, where it is known as over-fitting (Scales and Snieder 1998; Van der Baan and Jutten 2000).

The classical solution to this dilemma is to use a split set of examples. One part is used for training; the other part is used as a reference set to quantify the general performance (Fig. 1.14). Training is stopped when the misfit of the reference set reaches a minimum. This method is known as 'holdout' cross validation.

1.16 The Effect of the Number of Learning Samples

When the number of learning samples is low the $E_{learning}$ will be small but E_{test} will be large which means that the test error of the network is large. If we increase the learning set size and plot the average learning and test error rates as a function of the learning set size we will see that a low learning error on the (small) learning set in no way guarantees a good network performance. With an increasing number of learning samples the two error rates converge to the same value. This value depends on the representational power of the network: given the optimal weights, how good is the approximation? This error depends on the number of hidden units and the activation function (Fig. 1.15). If the learning error rate does not converge to the test error rate the learning procedure has not found a global minimum error rate.

In order to avoid over-fitting and under-fitting the optimal number of training observations should be determined. Until the date of writing this book no general guidelines have been available to achieve this Holy Grail. However, Lawrence and Frederickson (1998) suggested the following rule of thumb:

$$2(i+h+o) \leq N \leq 10(i+h+o) \tag{1.25}$$

where 'i' is the number of inputs, 'h' is the number of hidden layers and 'o' is the number of outputs.

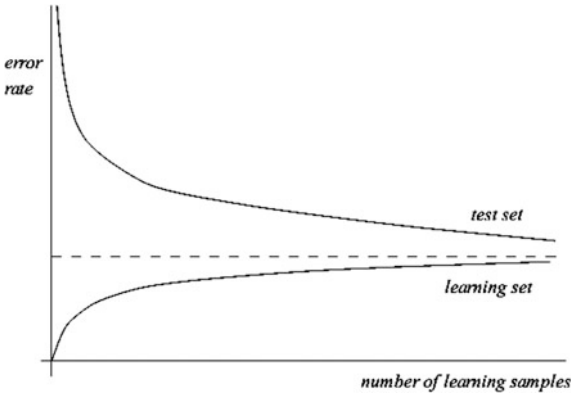


Fig. 1.15 Effect of the learning set size on the error rate. The average error rate and the average test error rate as a function of the number of learning samples (Krose and Smagt 1996)

1.17 The Effect of the Number of Hidden Units

If the number of hidden units is large then the network may fit the learning samples exactly, but because of the large number of hidden units the output which is achieved by the network is far more variable than the actual values of output, this is called over-training or over-fit, where we have too much variance (see Fig. 1.14). Particularly for the case of learning samples which contain a certain amount of noise (which all real-world data have), the network will begin to 'fit the noise' of the learning samples instead of making a smooth approximation.

A large number of hidden units leads to a small error for the training set but not necessarily to a small error on the test set. Adding hidden units always leads to a reduction of the $E_{learning}$. However, adding hidden units will first lead to a reduction of the E_{test}, but then eventually lead to an increase in E_{test}. This effect is called the 'peaking effect'. The average learning and test error rates are plotted as a function of learning set size in Fig. 1.16.

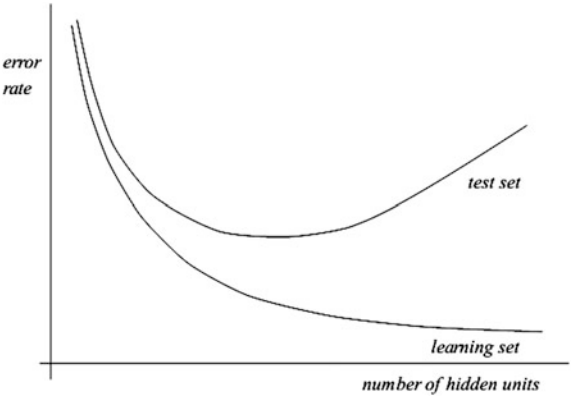


Fig. 1.16 The average learning error rate and the average test error rate as a function of the number of hidden units (Krose and Smagt 1996)

1.18 The Optimum Number of Hidden Neurons

One way to find the optimum number of neurons in the hidden layer is by running the neural network with different number of neurons and calculating the mean square error of training each time. Plotting the MSE versus the number of hidden neurons, we can select the one with the minimum MSE as the optimum number of hidden neurons. The flowchart of this algorithm is shown in Fig. 1.17.

1.19 The Multi-start Approach

If a multi-layer perception neural network training process is repeated N times with the same training data set and the same neurons the MSE error of test/train set will be different. Because the initial value of weights for training are selected randomly the curve of the training errors will consequently be different for different runs. It is better to run the training process for different values of initial weights to get the

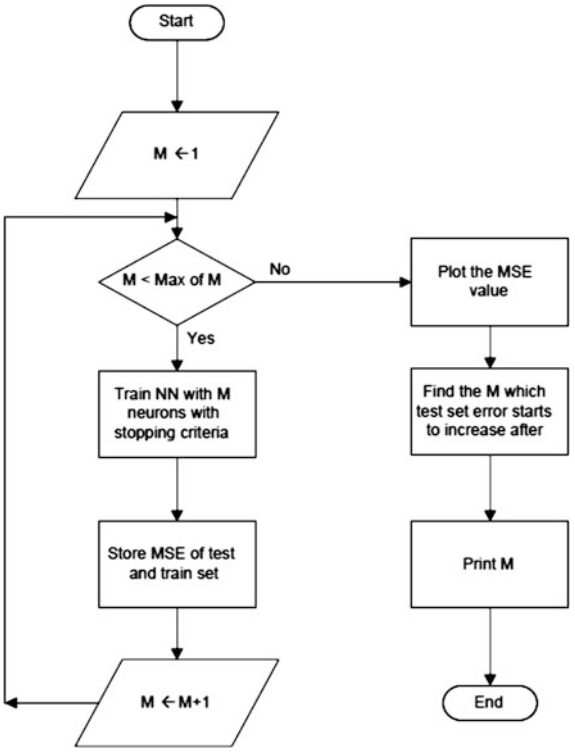


Fig. 1.17 Flowchart of finding the optimum number of hidden neurons (Hajian et al. 2012)

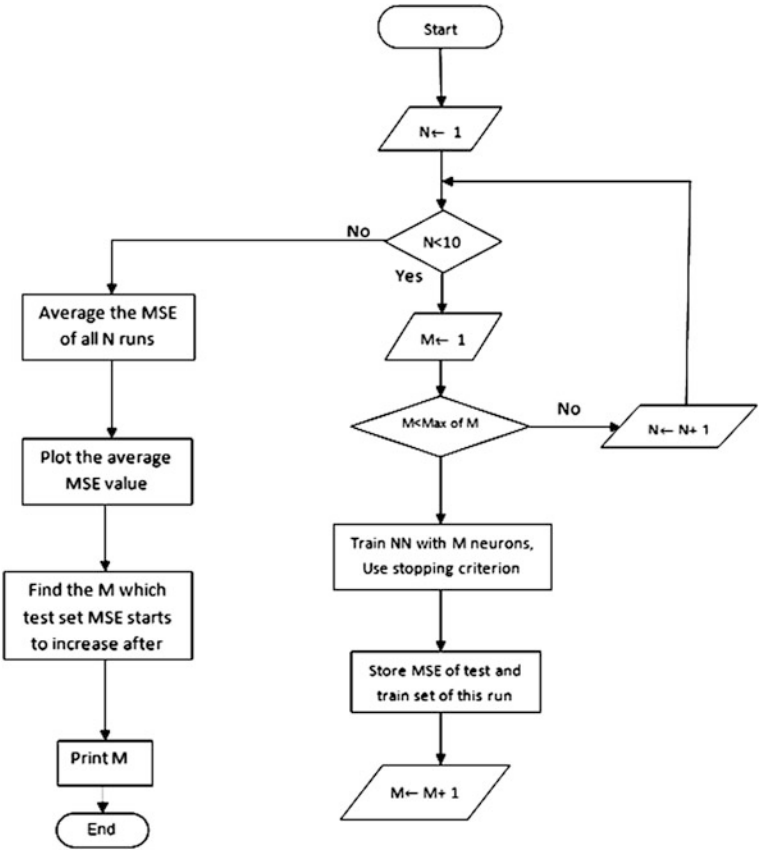


Fig. 1.18 Flowchart to find the optimum number of neurons in the hidden layer with a multi-start approach (Hajian et al. 2012)

optimum values of the final weights with the least training/testing error. This method is named 'Multi-start approach'. The flowchart of this approach is depicted in Fig. 1.18.

Hajian et al. (2012) use a novel algorithm for finding the optimum number of hidden neurons using the multi-start approach with different number of neurons. The flowchart of this algorithm is shown in Fig. 1.18.

In this method for each number of neurons in the hidden layer, the network is run with N different values for initial weights. The initial values of weights are chosen randomly for different N runs.

Finally, the optimum number of neurons is selected on the basis of average values of MSE of the test set over N runs. It means that the point at which the averaged MSE for N runs is a minimum shows optimum number for the hidden neurons.

1.20 Test of a Trained Neural Network

After the training phase of the network is successfully completed, the performance of the trained model is tested with a data set other than the training data; the so-called test data. On the other hand, after the errors are minimized, the model with all the parameters including the connection weights is tested with a separate set of "testing" data that is not used in the training phase. At the end of the training, the neural network represents a model that should be able to predict the target value given the input pattern. One issue when fitting a model is how well the newly-created model behaves when applied to new data. To address this issue, the data set can be divided into multiple partitions: a training partition used to create the model, a validation partition to test the performance of the model, and a third test partition. Partitioning is performed randomly to avoid the possibility of a biased partition; according to proportions specified by the user, or according to rules concerning the data set type. For example, when creating a time series forecast, data is partitioned by chronological order (http://www.solver.com/partition-data). The training set is used to fit the models; the validation set is used to estimate prediction error for model selection; the test set is used for assessment of the generalization error of the final chosen model. Ideally, the test set should be kept in a "vault," and be brought out only at the end of the data analysis (Hastie et al. 2009).

1.20.1 The Training Set

The training set is used to train or build a model. For example, for a linear regression, the training set is used to fit the linear regression model (i.e., to compute the regression coefficients). In a neural network model, the training set is used to obtain the network weights. After fitting the model to the training set, the performance of the model should be tested on the validation set.

1.20.2 The Validation Set

Once a model has been built using the training set, the performance of the model must be validated using new data. If the training set itself was utilized to compute the accuracy of the model fit, the result will be an overly optimistic estimate of the accuracy of the model. This is because the training or model fitting process ensures that the accuracy of the model for the training data is as high as possible, and that the model is specifically suited to the training data. To obtain a more realistic estimate of how the model would perform with unseen data, we must set aside a part of the original data and not include this set in the training process. This data set is known as the Validation Set. To validate the performance of the model, we can

measure the discrepancy between the actual observed values and the predicted value of the observation. This discrepancy is known as the error in prediction, and is used to measure the overall accuracy of the model which was mentioned before in Table 1.2.

1.20.3 The Test Set

The Validation Set is often used to fine-tune models. For example, you might try out neural network models with various architectures and test the accuracy of each on the Validation Set to choose the best performer among the competing architectures. When a model is chosen, its accuracy with the Test Set is still an optimistic estimate of how it will eventually perform with unseen data. This is because the final model has come out as the winner among the competing models based on the fact that its accuracy with the Validation Set is highest. As a result, it is a good idea to set aside another portion of data that is used in either training or in validation. This set is known as the Test Set. The accuracy of the model on the test data gives a realistic estimate of the performance of the model on completely unseen data. There are two approaches to standard partitioning: random partitioning and user-defined partitioning.

1.20.4 Random Partitioning

In simple random sampling, every observation in the main data set has equal probability of being selected for the partition data set. For example, if you specify 60% for the Training Set, then 60% of the total observations are randomly selected for the training set. In other words, each observation has about 60% (or 70% in some cases) chance of being selected. Random partitioning uses the system clock as a default to initialize the random number seed. Alternatively, a random seed can be manually set, resulting in the same observations being chosen for the Training/Validation/Test Sets each time a standard partition is created.

1.20.5 User-Defined Partitioning

In user-defined partitioning, the partition variable specified is used to partition the data set. This is useful when you have already pre-determined the observations to be used in the Training, Validation, or Test Sets. This partition variable takes the value: t for training, v for validation and s for test. Rows with any other values in the Partition Variable column are ignored. The partition variable serves as a flag for writing each observation to the appropriate partition(s).

1.20.6 Partition with Oversampling

This method of partitioning is used when the percentage of successes in the output variable is very low (i.e., callers who opt into a short survey at the end of a customer service call). Typically, the number of people who finish the survey is very low, so information connected with these callers is minimal. As a result, it would be almost impossible to formulate a model based on these callers. In these types of cases, we must use Oversampling (also called weighted sampling). Oversampling can be used when there are only two classes, one of much greater importance than the other (i.e., callers who finish the survey as compared to callers who simply hang up) (http://www.solver.com/partition-data).

1.20.7 Data Partition to Test Neural Networks for Geophysical Approaches

In geophysical problems, depending on the nature of data, when the pattern set is prepared, it is usually randomly partitioned into two parts: a large part which is used as the training set and a small part which is used as the test set. The percentage of testing/validation is done experimentally, which varies from 60 to 75%. Hajian et al. (2012) used 75% of the pattern set considered as the training set and the remained part (25% of pattern set) was considered as test/validation data (Fig. 1.19).

Most supervised data mining algorithms in geophysical approaches follow these three steps:

1. The training set is used to build the model. This contains a set of data that has pre-classified target and predictor variables.
2. Typically a hold-out dataset or test set is used to evaluate how well the model works with data outside the training set. The test set contains the pre-classified results data but they are not used when the test set data is run through the model

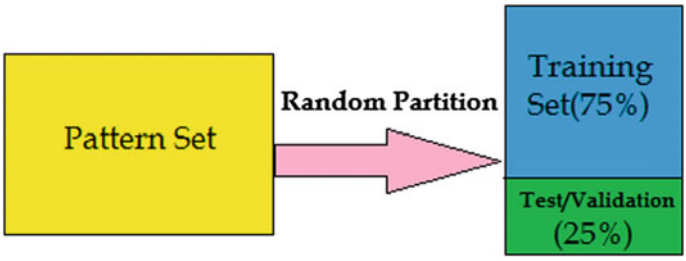


Fig. 1.19 Schematic of random partitioning of pattern set into training set and test/validation set (Hajian et al. 2012)

until the end, when the pre-classified data are compared against the model results. The model is adjusted to minimize error on the test set.
3. Another hold-out dataset or validation set is used to evaluate the adjusted model in step #2 where, again, the validation set data is run against the adjusted model and results compared to the unused pre-classified data.

The partitioning of pattern set into these three parts is important for testing and validating of the network training process.

The concept of 'Training/Cross-Validation/Test Data Sets is as simple as this. When you have a large of geophysical data set (collected or simulated), it is recommended that it is split it into 3 parts:

- Training set (usually 60% of the original data set): This is used to build up our prediction algorithm. Our algorithm tries to tune itself to the quirks of the training data sets. In this phase we usually create multiple algorithms in order to compare their performances during the Cross-Validation Phase.
- Cross-Validation set (usually 20% of the original data set): This data set is used to compare the performances of the prediction algorithms that were created based on the training set. We choose the algorithm that has the best performance.
- Test set (usually 20% of the original data set): Now we have chosen our preferred prediction algorithm but we don't know yet how it's going to perform on completely unseen real-world data. So, we apply our chosen prediction algorithm on our test set in order to see how it is going to perform so we can have some idea of the quality of the algorithm's performance on unseen data.

It's very important to keep in mind that skipping the test phase is not recommended, because the algorithm that performed well during the cross-validation phase isn't necessarily the best, because the algorithms are compared based on the cross-validation set and its quirks and noise. During the Test Phase, the purpose is to see how our final model is going to behave in the wild, so that in case its performance is very poor we may have to repeat the whole process starting from the Training Phase.

1.21 The General Procedure for Testing of a Designed Neural Network in Geophysical Applications

Usually, to test a designed neural network for geophysical applications, after passing the training steps successfully, another three steps are followed. In the first step the NN is tested for synthetic data with different level of noise.

Synthetic data means the data are produced (for a suitable domain of input) through a modeling process, for example the gravity effect of a sphere with no added-noise or the seismic wave velocity in a homogenous media, or the electrical response of a three layered model of resistivity, etc. It is obvious that a synthetic

Table 1.5 Different levels of noise for testing neural networks

Noise type	Level of noise (%)
Low	5
Medium	10
High	25

model does not exactly reproduce the real conditions under which environmental noise affects the main geophysical signal and furthermore the simplification of the models cause the outputs of models to be far from the real world.

So, the successful behaviour of a neural network for synthetic data is not sufficient and essentially doesn't mean it will work as well for real data. To ensure the network performance for real conditions is sufficient, a level of desired Gaussian noise is usually added to synthetic geophysical data. Then, to test the robustness of the network to natural noise, the level of artificial noise is increased and again the network is tested.

Through increasing the level of noise and testing the NN outputs error, one can estimate the level of robustness of the NN model to the noise. Hajian et al. (2012) used three different level of noise to test the network for depth estimation of cavities: low, medium and high, listed in Table 1.5.

After testing the neural network for both synthetics without and with noise with different levels, the designed and trained neural network is tested for real data. An optimized neural network is one which has passed all testing stages for synthetic without noise, noisy synthetic data and real data. In each of the testing stages if the performance of the neural network is not as desired it is modified by changing the architecture (number of hidden layers, number of neurons in the hidden layer, type of activation function) and/or the training procedure (changing training algorithm or training algorithm factors,…) and/or the partitioning of data set into training, testing and validation such that the desired performance is finally achieved. Discovering the way to improve the designed neural network performance is very dependent on the designing expert and it will be possible to do this much faster if they have considerable experience in this field.

1.22 Competitive Networks—The Kohonen Self-organising Map

Competitive neural networks represent a type of ANN model in which neurons in the output layer compete with each other to determine a winner. The winner indicates which prototype pattern is most representative of (most similar to) the input pattern.

The competition among output layer neurons is implemented by *lateral inhibition*—a set of negative connections between the neurons. The most well-known among this paradigm of ANNs is the Self-organising Map (SOM), also known as the Kohonen net (Beale and Jackson 1990).

1.22.1 Learning in Biological Systems—The Self-organising Paradigm

The type of learning utilised in multilayer perceptrons requires the correct response to be provided during training (supervised training).

Biological systems display this type of learning, but they are also capable of learning by themselves—without a supervisor showing them the correct response (unsupervised learning). A neural network with a similar capability is called a *self-organising system* because during training, the network changes its weights to learn appropriate associations, without any right answers being provided. The propagation of biological neural activation via axons can be modelled using a Mexican hat function[1] (Fig. 1.20).

Cells close to the active cell have excitatory links. The strengths of the links drop off with distance and then turn inhibitory. The Kohonen neural network also uses only locally connected neurons and restricts the adjustment of weight values to localised "neighbourhoods".

1.22.2 The Architecture of the Kohonen Network

The Kohonen network consists of an input layer, which distributes the inputs to each node in a second layer, the so-called *competitive layer*. Each of the nodes on this layer acts as an output node.

Each neuron in the competitive layer is connected to other neurons in its neighbourhood and feedback is restricted to neighbours through these lateral connections (Fig. 1.21).

Neurons in the competitive layer have excitatory connections to immediate neighbours and inhibitory connections to more distant neurons. All neurons in the competitive layer receive a mixture of excitatory and inhibitory signals from the input layer neurons and from other competitive layer neurons.

1.22.3 The Kohonen Network in Operation

As an input pattern is presented, some of the neurons are sufficiently activated to produce outputs which are fed back to other neurons in their neighbourhoods. The node with the weight vector closest to the input pattern vector (the so-called

[1]The function $f(x) = A(x^4 - 3x^2)$; $A > 0$; is the Mexican Hat function. It has the appearance of a Mexican Hat, hence its name. It is symmetric with respect to the y-axis. Its two minima produce two stable points when viewed as a potential function in physics and engineering.

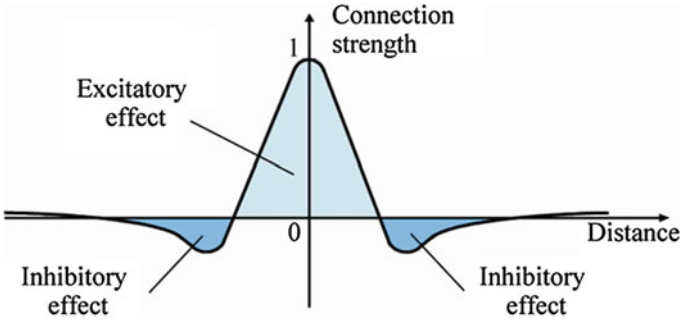


Fig. 1.20 Mexican hat function (Negnevitsky 2001)

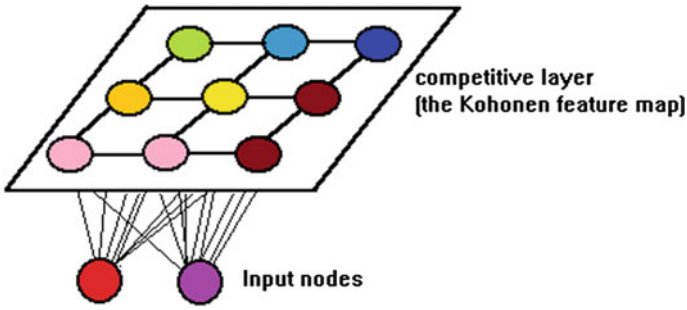


Fig. 1.21 Kohonen structure of layers. Redrawn after Beale and Jackson (1990)

"winning node") produces the largest output. During training, input weights of the winning neuron and its neighbours are adjusted to make them resemble the input pattern even more closely. At the completion of training, the winning node ends up with its weight vector aligned with the input pattern and produces the strongest output whenever that particular pattern is presented (Beale and Jackson 1990).

The nodes in the winning node's neighbourhood also have their weights modified to settle down to an average representation of that pattern class. As a result, unseen patterns belonging to that class are also classified correctly (generalisation). The *m* neighbourhoods, corresponding to the *m* possible pattern classes are said to form a *topological map* representing the patterns. The initial size of the neighbourhood mentioned above and the fixed values of excitatory (positive) and inhibitory (negative) weights to neurons in the neighbourhood are among the design decisions to be made (Beale and Jackson 1990).

1.22.4 Derivation of the Learning Rule for the Kohonen Net

The summed squared error for pattern p for all output layer neurons can be written as:

$$E_p = \frac{1}{2}\sum_j (w_{ij} - x_j^p)^2 \tag{1.26}$$

where x_j^p is the i-th component of pattern p for neuron j. The summation is done over all j neurons.

Any change Δw_{ij} in the weight is expected to cause a reduction in error E_p. Now E_p is a function of all the weights, so its rate of change with respect to any one weight value w_{ij} has to be measured by calculating its partial derivative with respect to w_{ij} (That is why we have the small delta δ, instead of d in the following equation for the derivative).

$$\Delta_p w_{ij} = -\eta \frac{\delta E_p}{\partial w_{ij}} \tag{1.27}$$

where η is a constant of proportionality.

Now we have to calculate the partial derivatives of E_p. Using (1.27):

$$\frac{\delta E_p}{\partial w_{ij}} = w_{ij} - x_j^p \tag{1.28}$$

Combining (1.27) and (1.28), we get

$$\Delta_p w_{ij} = -\eta \frac{\delta E_p}{\partial w_{ij}} = -\eta(w_{ij} - x_j^p) = \eta(x_j^p - w_{ij}) \tag{1.29}$$

1.22.5 Training the Kohonen Network

1.22.5.1 The Kohonen Algorithm

1. **Initialise weights**
   Initialise weights from N inputs to the nodes to small random values. Set the initial radius of the neighbourhood.
2. **Present new input** $x_0(t)$, $x_1(t)$, $x_2(t)$, …, $x_{n-1}(t)$, where $x_i(t)$ is the input to node i at time t.

3. **Compute distances to all nodes**
   Compute distances *dj* between the input and each output node *j using:*

$$d_j = \sum_{i}^{N-1} \left(x_i(t) - w_{ij}(t)\right)^2 \tag{1.30}$$

   Where $x_i(t)$ is the input to node *i* at time *t* and $w_{ij}(t)$ is the weight from input node *i* to output node *j* at time *t.*
4. **Select output node with minimum distance**
   Select output node *j** as the output node with minimum *dj.*
5. **Update weights to node *j* and neighbours***
   Weights updated for node *j** and all nodes in the neighbourhood defined by *Nj** (*t*). New weights are

$$w_{ij}(t+1) = w_{ij}(t) + \eta(t)\left(x_i(t) - w_{ij}(t)\right) \tag{1.31}$$

   for *j* in *Nj**, $0 \leq i \leq N-1$.
   The term $\eta(t)$ is a gain term $0 \leq \eta \leq 1$. Both η and $Nj^*(t)$ decrease with time (Beale and Jackson 1990).
6. **Repeat by going to step 2**

1.22.5.2 Learning Vector Quantisation (LVQ)

A supervised learning technique for optimising the performance of Kohonen networks, e.g., when new vectors are to be added.

Weight adjustments—For a correctly classified input (Beale and Jackson 1990).

$$nw(t+1) = nw(t) + \eta(t)[x(t) - nw(t)] \tag{1.32}$$

For incorrect classification:

$$nw(t+1) = nw(t) - \eta(t)[x(t) - nw(t)] \tag{1.33}$$

1.22.6 Training Issues in Kohonen Neural Nets

1.22.6.1 Vector Normalisation

To make vector comparison independent of magnitudes and dependent on orientation only, the vectors are normalised by dividing them by their magnitudes. This also helps to reduce training time.

1.22.6.2 Weight Initialisation

A random distribution of initial weights may not be optimal, resulting in sparsely populated trainable nodes and poor classification performance.

Possible remedies:

a. Initialisation of weights to the same value and lumping of input vectors with similar orientation. This increases the likelihood of all nodes being closer to the pattern vector. Inputs are slowly returned to original orientation with training.
b. Addition of random noise to inputs to distribute vectors over a larger pattern space.
c. Using a large initial neighbourhood changing slowly (Beale and Jackson 1990).

1.22.6.3 Reducing Neighbourhood Size

The neighbourhood radius should be decreasing linearly with time (iterations). Neighbourhood shape may vary to suit the application—e.g., circular or hexagonal instead of rectangular.

1.22.7 Application of the Kohonen Network in Speech Processing—Kohonen's Phonetic Typewrite

The Kohonen network is widely used in speech and image processing and has the potential for statistical and database applications. Speaker-independent, unlimited vocabulary continuous speech recognition remains to be achieved with conventional techniques. The problem is made difficult by the same word being spoken with different pronunciations, levels of loudness, emphases and background noise. Apart from analysing individual units of sound (phonemes), the human brain uses stored speech patterns, context and other clues to recognise speech successfully (Beale and Jackson 1990).

Kohonen's Phonetic Typewriter combines digital signal processing techniques and the use of a rule base with a Kohonen network to achieve 92–97% accuracy with multiple speaker unlimited vocabulary Finnish and Japanese speech(Beale and Jackson 1990) (Fig. 1.22).

1.23 Hopfield Network

The Hopfield neural network is a type of unsupervised neural network which doesn't need to be trained and hence no training samples are needed. The Hopfield model is a single layer feedback neural network. This means that the flow of data is not only from one direction and this network incorporates feedback into neuron from all neuron expect itself.

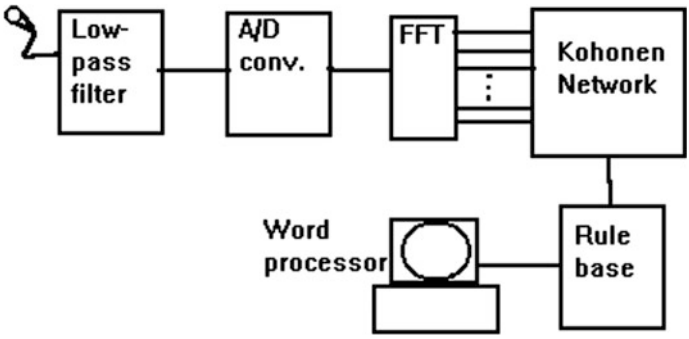


Fig. 1.22 Kohonen's Phonetic Typewriter block diagram. Beale and Jackson (1990), *Source* ftp.it.murdoch.edu.au/units/ICT482/Notes/Kohonen%20net.doc

On the other hand, in this type of neural network the neurons are all connected to each other and the weights of the forward connections are the same as its reverse connection, so the data entered in the network has the same effects in all of the neurons available. The weight of the neurons connections is fixed and can be calculated by Hopfield methodology. As shown in the Fig. 1.23, this is a 4-neuron Hopfield network diagram. The sum square error of the Hopfield network outputs is the energy function which must be minimized to get the best results.

So in a Hopfield network we have an energy function which may be defined as:

$$\mathrm{E(v)} = \frac{1}{2}\sum_{i=1}^{n}\sum_{j\neq i=1}^{n} W_{ij}V_iV_j - \sum_{i=1}^{n} I_iV_i \tag{1.34}$$

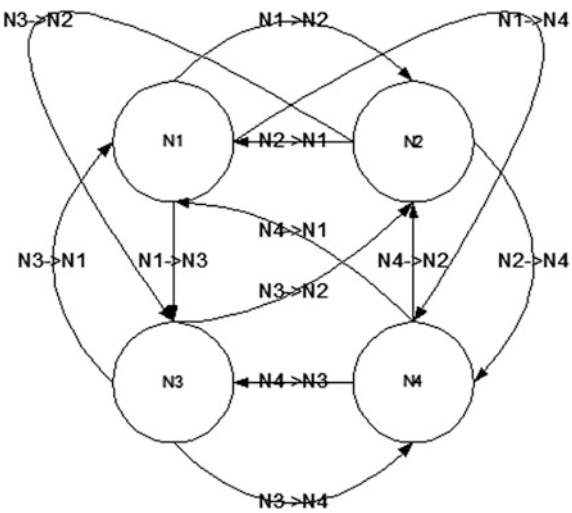


Fig. 1.23 Hopfield neural network with 4 neurons (Hajian et al. 2011a)

where:

E(v)	The energy function of Hopfield network
n	Total number of neurons
W_{ij}	The weight of connection from neuron i to neuron j
V_i	The i element of input vector V
Vj	The j element of input vector V
l_i	The i'th input also known as threshold

If weight W_{ij} is a symmetric matrix, the energy function E(v) will never increase as the state of the neurons(V_1, V_2, …, V_n) changes (Hopfield 1984; Hopfield and Tank 1985; Tank and Hopfield 1986). This means that the network will converge to a state at which the energy function E(v) is locally minimized (Wang and Mendel 1992). Accordingly, the Hopfield network can invert any set of measured geophysical data to another set of model parameters if a cost function can be formulated between the measured data and those theoretically calculated based on the model parameters The Hopfield network can be used in geophysical applications analogous to the conventional inversion techniques such as least squares method. However, inversion of geophysical data in general is not a simple task. The main fundamental difficulties are the problem of non-uniqueness and instability in the solutions (Li and Oldenberg 1996).

The Hopfield neural network has proven to be a powerful tool for solving a wide variety of optimization problems (Hopfield and Tank 1985; Tank and Hopfield 1986). The key step in applying the Hopfield neural network to an optimization problem is to relate a suitable cost function of the optimization problem to the Hopfield energy function. Once this relationship is formulated, the network changes from its initial state to final state. The final state constitutes the solution of the problem where the energy function is minimized. The error function plays the role as energy cost function which should be minimized (Hajian et al. 2011a).

1.24 Generalized Regression Neural Network (GRNN)

1.24.1 GRNN Architecture

The Generalized Regression Neural Network is a neural network architecture that can solve any functional approximation problem. The learning process is equivalent to finding a surface in a multidimensional space that provides a best fit to the training data, with the criterion for the "best fit" being measured in some statistical sense. The generalization is equivalent to the use of this multidimensional surface to interpolate the test data.

As can be seen from Fig. 1.24, the Generalized Regression Network consists of three layers of nodes with entirely different roles:

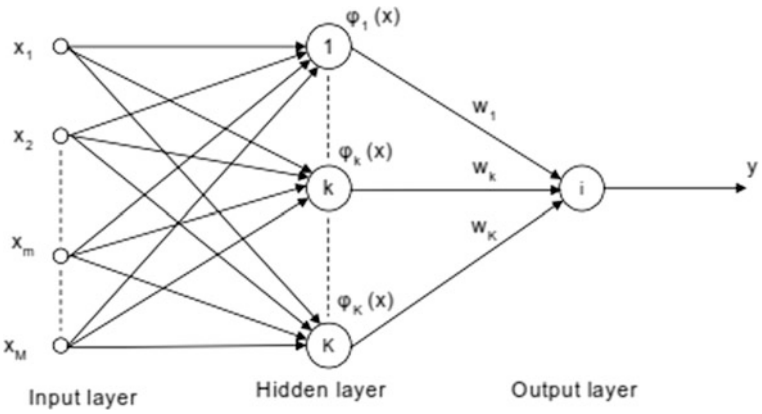


Fig. 1.24 Generalized regression neural network architecture

- The input layer, where the inputs are applied,
- The hidden layer, where a nonlinear transformation is applied on the data from the input space to the hidden space; in most applications the hidden space is of high dimensionality,
- The linear output layer, where the outputs are produced.

GRNN, as proposed by Specht (1991) falls into the category of probabilistic neural networks. This neural network, like other probabilistic neural networks, needs only a fraction of the training samples which a back propagation neural network would need (Specht 1991). The data available from measurements of an operating system is generally never enough for a back propagation neural network (Specht 1990). Therefore, the use of a probabilistic neural network is especially advantageous due to its ability to converge to the underlying function of the data with only few training samples available. The additional knowledge needed to get the fit in a satisfactory way is relatively small and can be done without additional input by the user. This makes GRNN a very useful tool to perform predictions and comparisons of system performance in practice.

1.24.2 Algorithm for Training of a GRNN

The probability density function used in GRNN is the Normal Distribution. Each training, sample, X_i, is used as the mean of a Normal Distribution.

$$Y(X) = \frac{\sum_{i=1}^{n} Y_i \exp(-D_i^2/2\sigma^2)}{\sum_{i=1}^{n} \exp(-D_i^2/2\sigma^2)} \tag{1.35}$$

$$D_i^2 = (X - X_i)^T.(X - X_i) \tag{1.36}$$

where Y(x) is the predicted output for input vector x, n indicates the number of training patterns and σ is the standard deviation. The distance, D_j, between the training sample and the point of prediction, is used as a measure of how well the each training sample can represent the position of prediction, X. If the distance, D_j, between the training sample and the point of prediction is small, $\exp(-D_j^2/2\ \sigma^{2)}$, becomes big. For $D_j = 0$, $\exp(-D_j^2/2\ \sigma^{2)}$ becomes one and the point of evaluation is represented best by this training sample. The distance to all the other training samples is bigger. A bigger distance, D_j, causes the term $\exp(-D_j^2/2\ \sigma^{2)}$ to become smaller and therefore the contribution of the other training samples to the prediction is relatively small. The term $Y_j^*\ \exp(-D_j^2/2\ \sigma^2)$ for j-th training sample is the biggest one and contributes very much to the prediction. The standard deviation or the smoothness parameter, σ, as it is defined in Specht (1991), is subject to a search. For a bigger smoothness parameter, the possible representation of the point of evaluation by the training sample is possible for a wider range of X. For a small value of the smoothness parameter the representation is limited to a narrow range of X, respectively.

Using Eq. (1.35) it is possible to

- predict the behavior of systems based on a few training samples
- Predict smooth multi-dimensional curves
- Interpolate between training samples.

1.24.3 GRNN Compared to MLP

GRNN networks have advantages and disadvantages compared to Multi-layer Perceptron (MLP) networks:

- It is usually faster to train a GRNN network than a MLP network.
- GRNN networks are often more accurate than MLP network.
- GRNN networks often have good performance in noisy environments.
- Less data is required to train a GRNN compared to MLP.
- GRNN networks are relatively insensitive to outliers (wild points.

1.25 Radial Basis Function (RBF) Neural Networks

1.25.1 Radial Functions

Radial functions are a special class of function. Their characteristic feature is that their response decreases (or increases) monotonically with distance from a central point. The center the distance scale and the precise shape of the radial function are all fixed parameters of the model if it is linear.

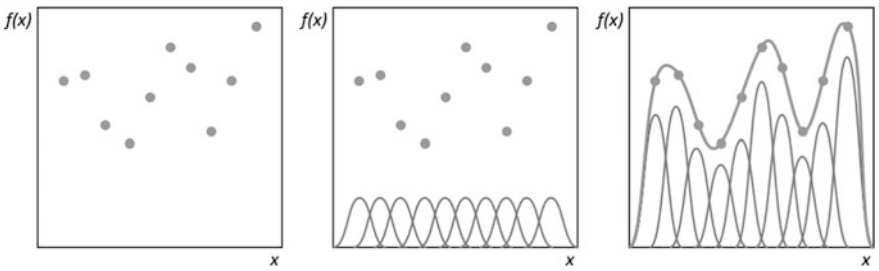


Fig. 1.25 The diagram of how curve fitting can be done using radial basis functions

A typical radial function is the Gaussian which in the case of a scalar input is:

$$y = e^{\frac{-(x-c)^2}{r^2}} \tag{1.37}$$

Its parameters are its center "c" and its radius "r".

A Gaussian RBF monotonically decreases with distance from the centre In contrast a multi-quadric RBF which in the case of scalar input is:

$$h(x) = \frac{\sqrt{r^2 + (x-c)^2}}{r} \tag{1.38}$$

Multi-quadratic RBF monotonically increase with distance from the centre, see Fig. 1.25. Gaussian like RBFs are local (give a significant response only in a neighborhood near the centre) and are more commonly used than multi-quadric type RBFs which have a global response. They are also more biologically plausible because their response is infinite.

1.25.2 RBF Neural Networks Architecture

Radial basis function neural networks although not a different type of architecture in the sense of perceptrons and connections, make use of radial basis functions as their activation functions; these are real valued functions whose output depends on the distance from a particular point. The most commonly used radial basis function is the Gaussian distribution. Because radial basis functions can take on much more complex forms, they were originally used for performing function interpolation. As such, a radial basis function neural network can have a much higher information capacity. Radial basis functions are also used in the kernel of a Support Vector Machine (SVM).

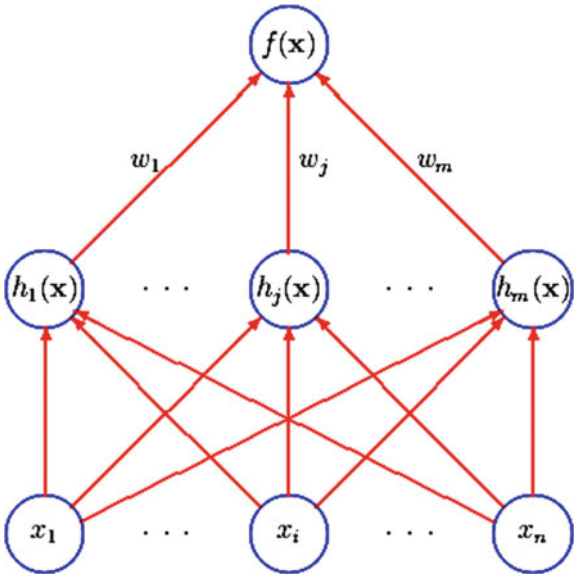


Fig. 1.26 The traditional radial basis function network, each of n components of the input vector feeds forward to m basis functions whose outputs are linearly combined with weights into the network outputs (Orr 1996)

Radial functions are simply a class of functions, in principle they could be employed in any sort of model, linear or nonlinear, and any sort of network single layer or multi-layer. However since Broomhead and Lowe's seminal paper, radial basis function networks, RBF networks, have traditionally been associated with radial functions in a single layer network such as shown in Fig. 1.26. An RBF network is nonlinear if the basic functions can move or change size or if there is more than one hidden layer (Orr 1996).

RBF neural networks can be represented as a three layer feed forward structure (Fig. 1.27) in which:

1. The input layer serves only as input distributer to the hidden layer.
2. Each node in the hidden layer is a function, its dimensionality being the same as the dimensionality of the input data.
3. The output is calculated by a linear combination. i.e. weighted sum of the radial basis functions plus the bias, according to (Orr 1996):

$$\mathrm{y}(\mathrm{x_i}) = \sum_{j=1:k} w_j \phi\left(\left\|x_i - c_{ji}\right\|\right) \tag{1.39}$$

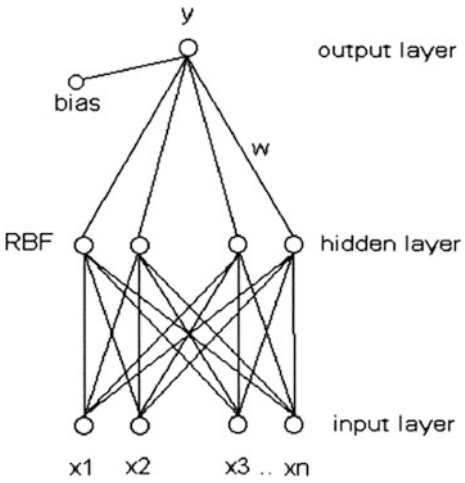


Fig. 1.27 Three layer feed forward structure of RBF neural network

In matrix notation:

$$Y = \phi.W \tag{1.40}$$

$W = [w_1\ w_2\ \ldots\ w_k\ w_0]^T$ and ϕ is:

$$\begin{array}{c} \\ \text{object}_1 \\ \\ \boldsymbol{\Phi} = \\ \\ \\ \end{array} \begin{array}{c} \begin{array}{cccc} \text{Nod}_1 & \text{Nod}_2 & \text{Nod}_k & \text{bias} \end{array} \\ \begin{bmatrix} \phi_{11} & \phi_{21} & \phi_{k1} & 1 \\ \phi_{12} & \phi_{22} & \phi_{k2} & 1 \\ \phi_{13} & \phi_{23} \ldots & \phi_{k3} & 1 \\ \ldots & & & \\ \phi_{1m} & \phi_{2m} & \phi_{km} & 1 \end{bmatrix} \end{array} = [\phi_1 \quad \phi_2 \quad \phi_k \quad 1] \tag{1.41}$$

1.26 Modular Neural Networks

A modular neural network (MNN) as defined by Haykin (1994) is one in which the computation performed by the network can be decomposed into a group of modules (local experts) that operate on distinct inputs without communicating with each other (Fig. 1.28).

For instance, the functional relationship between stress and strain for a solid material solid is likely to be quite different for low and high strains (Fig. 1.29).

Fully-connected networks require a great deal more effort in such instances during the training process, in addition to a vast data set containing all combinations of examples. When faced with a new example during training, they tend to alter all their

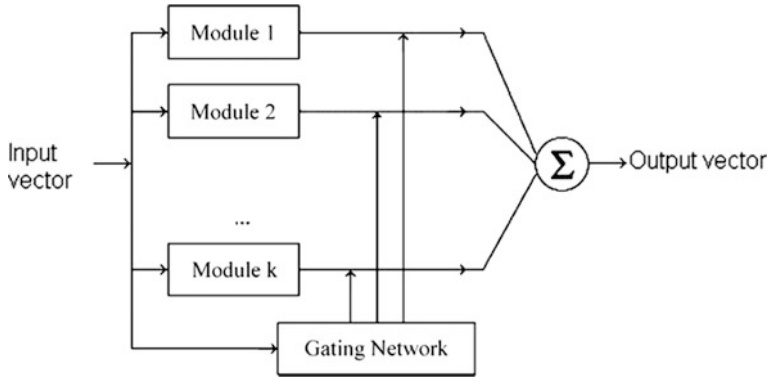


Fig. 1.28 Block diagram of the general modular neural network architecture (Haykin 1994)

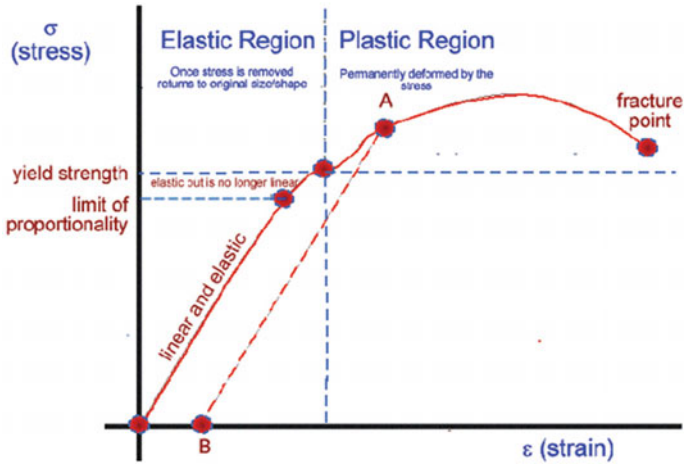


Fig. 1.29 Stress Strain curves and material behavior (after Cyberphysics.co.uk). *Source* Styles (2012)

weights and times hold values, rather than selecting a small, but relevant, portion of the network. For a singular network to be successful in such cases, the training data set needs to be very long and the training process is very slow as the network attempts to learn all the nuances of the input space in a global fashion continuing with the example of the events, a comprehensive training data set may contain blocks of seemingly incompatible data pairs, as is quite often the case with data obtained from real measurements. A singular network tends to find this data confusing, and may sometimes dismiss it as spurious noise, subsequently influencing the prediction accuracy. In a well-constructed modular neural network this can be handled with different blocks of data being assigned to different modules of the network (Eshaghzadeh and Hajian 2018).

1.27 Neural Network Design and Testing in MATLAB

There are two ways to implement Neural Networks in MATLAB, one is using MATLAB special commands to prepare related suitable M-files.

The other is NN tool commands to run the neural networks toolbox; this toolbox is a user friendly toolbox window in which the most common types of neural networks can be designed, and tested, rapidly. In this section we introduce nntools with a simple example.

To open the neural network toolbox in Matlab, first type "nntool" in the command window which means neural networks toolbox.

A toolbox window like Fig. 1.30 will appear.

In this window, the user can enter input data and target data as the training data set.

To do this step, first, select the "Import" icon in the left-button of the box, and then a window named "Import to network/data manager" will open, illustrated in Fig. 1.31.

Here, you can input both input and target data from your source whether from MATLAB work space or disk file.

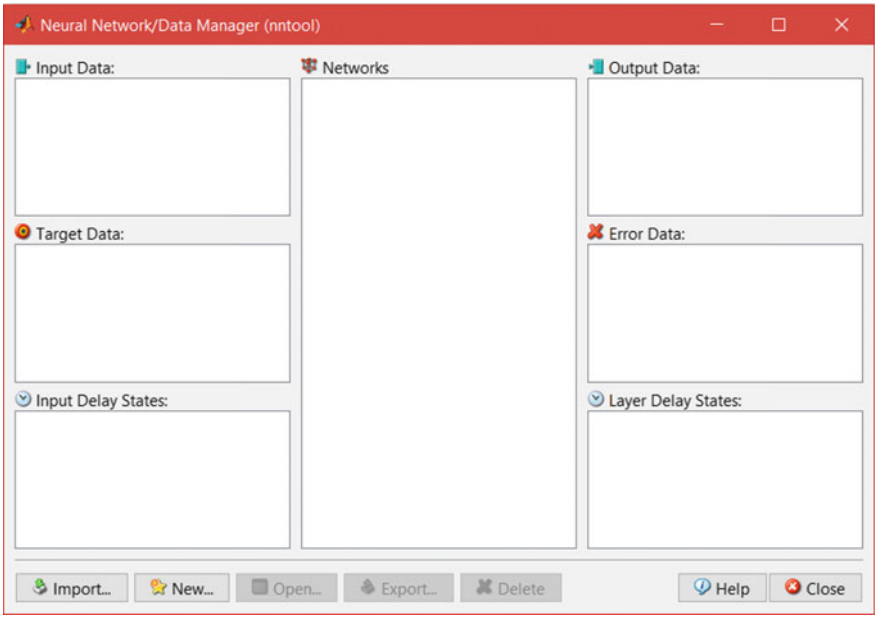


Fig. 1.30 Neural network toolbox

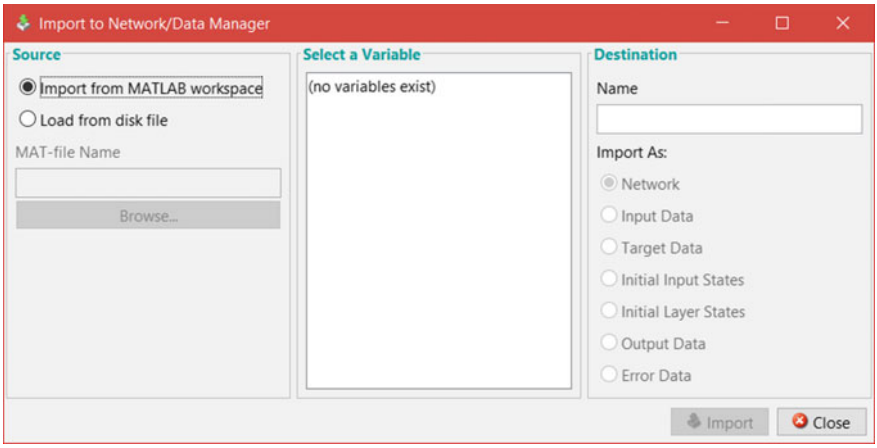


Fig. 1.31 Window of importing to Network/data manager in nntool

If the training data is not available in the workspace of MATLAB you should load it from a disk file. As a simple example suppose we want to design and train a network for:

$$\boldsymbol{y} = \mathbf{cos}(2\boldsymbol{x}_1)/\mathbf{exp}(\boldsymbol{x}_2).$$

First we produce the training data in the command window:

$$\gg \mathbf{x1} = \mathbf{0}{:}\mathbf{0.5}{:}\mathbf{14.5};$$
$$\gg \mathbf{x2} = -\mathbf{1}{:}\mathbf{0.1}{:}\mathbf{1.9};$$
$$\gg \mathbf{y} = \mathbf{cos}(\mathbf{2} * \mathbf{x1})./\mathbf{exp}(\mathbf{x2});$$
$$\gg \mathbf{x} = [\mathbf{x1};\ \mathbf{x2}];$$

Note that the whole inputs of a neural network is entered as a matrix in nntool and each of its rows are the vectors of each of inputs.

If you select the MATLAB workspace as your source in the 'select a variable' sub-box, both x and y will be considered as input and target data of the neural network, so, in the destination sub-window, select 'input data' choice and click the import icon in the right button of this window. A message box will be sent from MATLAB "Variable" 'x' has been imported as input data into the network/data manager, click 'ok' in this case.

Now, go back to 'Select variable sub-box' and select 'y' as a variable, and in 'destination' sub-box select 'target data', and click import button at the right button, again a message from MATLAB will be shown "Variable" 'y' has been imported as target data into the network/data manager, click 'on' again.

Now set $\{x, y\}$ has been imported into network/data manager space (Fig. 1.32) and are ready to be used for training of the Neural Network when it will be designed (Fig. 1.33).

To design a neural network, select 'New' in Network/Data manager toolbox, then the "Create Network or Data" window (Fig. 1.33).

A schematic chart of the procedure for the design of an MLP neural network through nntool has been shown in Fig. 1.34.

Here there are two main parts:

Part 1 Name: enter the name (label) of the network

Part 2 Network properties: enter the properties of the network: network type, input data, target data, training function, adapting learning function, perform function, number of layers and properties on each of the layers which means the number of neurons and the transfer function for each layer.

In this example enter 'simulator' as the name of the network.

The network type menu there are several types of neural networks as below:

- Cascade-forward back propagation
- Competitive
- Elman Back propagation
- Feed-forward propagation
- Feed-forward distributed time delay
- Feed-forward time-delay
- Generalized regression

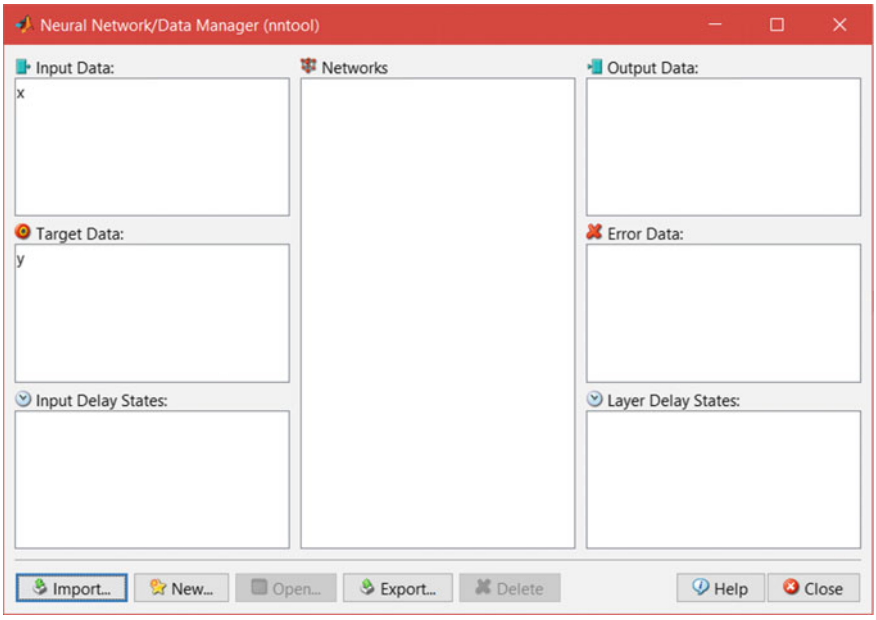


Fig. 1.32 Neural network/data manager window of nntool

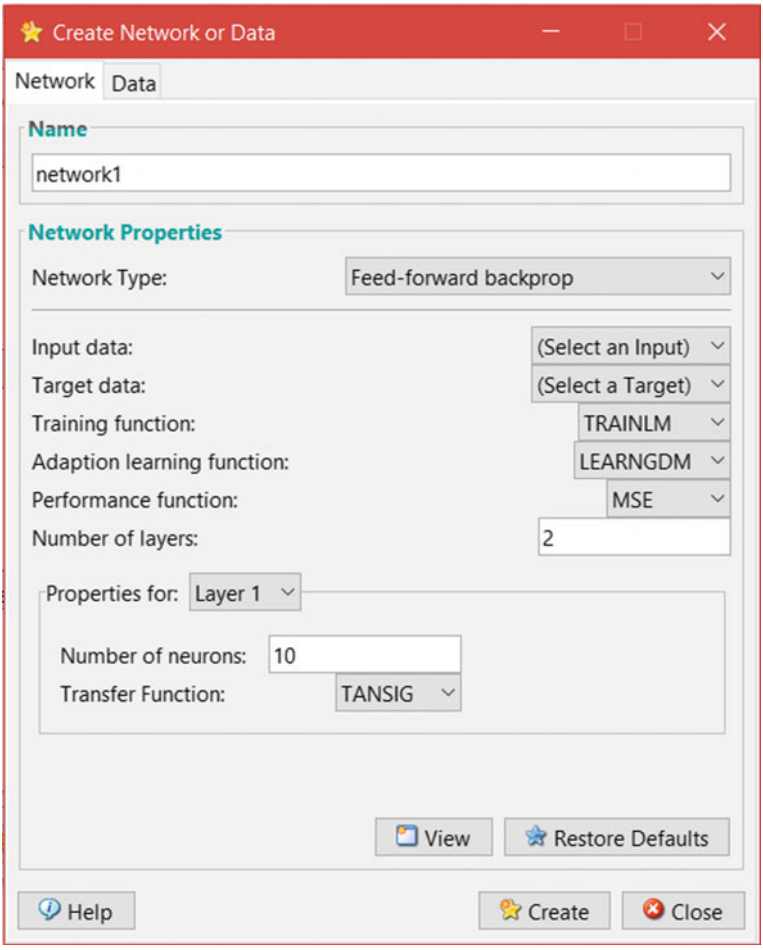


Fig. 1.33 Window of creating network/data in nntool

- Hopfield
- Layer recurrent
- Linear layer (design)
- Linear layer (train)
- LVQ: Learning vector quantization
- NARX: Nonlinear autoregressive exogenous model
- NARX Series Parallel
- Perception
- Probabilistic
- Radial basis (exact fit)
- Radial basis (fewer neurons)
- Self-organizing map

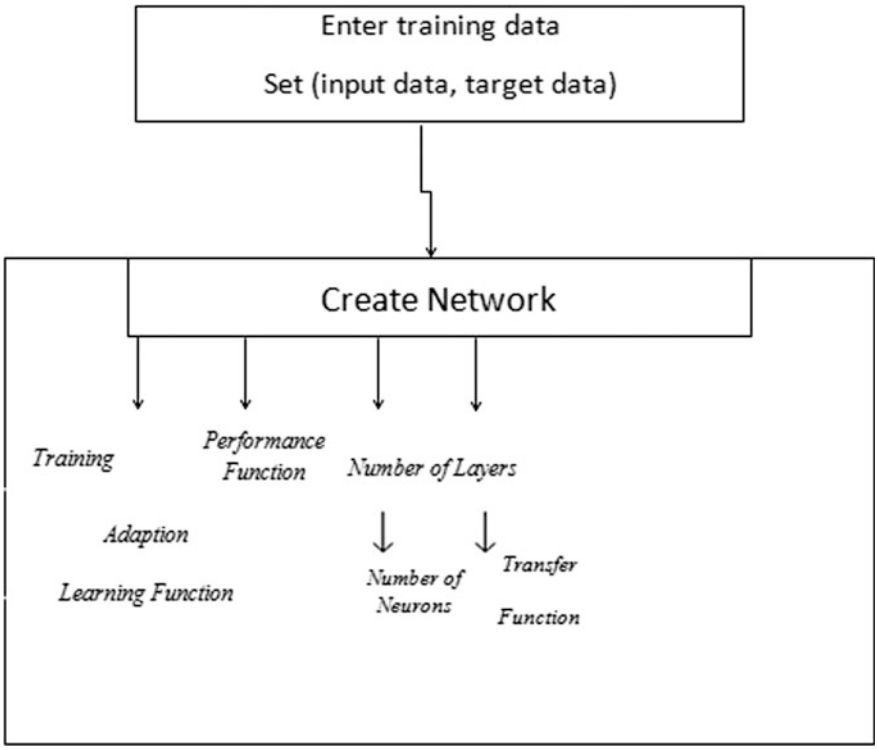


Fig. 1.34 Procedure for design of an MLP neural network through nntool

Here we select 'Feed-Forward back prop' as Network type, 'x' as input data and 'y' as target date. It is necessary to mention that in the icon 'input data' each of variables that were imported before in the network/data manager (NDM) space outputs appear and also the same for icon. Target data', each of variable that were imported in NDM space appear and can be selected as target data.

In the training function menu, the below functions are available (to be selected); select TRAIN LM.

- TRAIN LM (default): Levenberg Marquardt
- TRAIN BFG: BFGs quasi-Newton back prop.
- TRAIN BR: Bayesian regulation back propagation
- TRAIN CGB: Conjugate gradient back prop.
- TRAIN CGF: Conjugate gradient back prop. With Polak-Ribiere updates
- TRAIN CGP: Conjugate gradient back prop. With Fletcher-Reeves updates
- TRAIN GD: Gradient descent back propagation
- TRAIN GDM: Gradient descent with momentum back propagation
- TRAIN GDA: Gradient descent with adaptive learning rate back propagation
- TRAIN GDX: Gradient descent with momentum and adaptive back propagation

- TRAIN OSS: One Step Secant back propagation
- TRAIN R:Random order incremental training with learning functions
- TRAIN RP: Resilient Back propagation
- TRAIN SCG: Scaled conjugate gradient back propagation

TRAIN LM is the Levenberg-Marquardt function which is a famous and common algorithm for training which was explained in section 1-10, of this chapter.

In the part 'Adaption learning function' two choices are selectable:

- LEARN GD
- LEARN GM

Select LEARN GD

Also, in 'performance function' menu bar the choices are:

- MSE: Mean square Error
- MSEREG: Mean squared error with regularization performance function; it is weight sum of two factors: the mean squared error and the mean squared weight and bias values.
- SSE: Sum Square Error

Performance function means the Global Error which will be used to measure how closely the outputs correspond to the observed targets, and the aim of training for a supervised neural network is to iteratively minimize this error.

For this example, we select 'MSE'

- 'Number of layers' menu: select the number of layers of the network. It is necessary to mention that in MATLAB the input layer is not connected as a layer. It is perhaps because of the fired '2' weight for the inputs

In this example for considered function a 2-4-1 network is selected which means 2 neuron as input, 4 neurons as hidden layer and 1 neuron as output so Number of layers is 2 in this example.

Properties for layers

In this mean part, the properties of each layer is configurable for layer 1, number of neurons is set to '4' for the first run and 'TANSIG' is selected as transfer function (Fig. 1.35).

The transfer function menu choices are:

- TANSIG
- LOGSIG
- PURELIN

To check the architecture of the designed neural network selects the 'view' icon in the visit bottom of the window. Selecting 'view' neural network viewer will be illustrated as Fig. 1.36.

Here you can check to ensure if the general structure of the network is true or not. If there are any mistakes you can go back to create network/data menus and

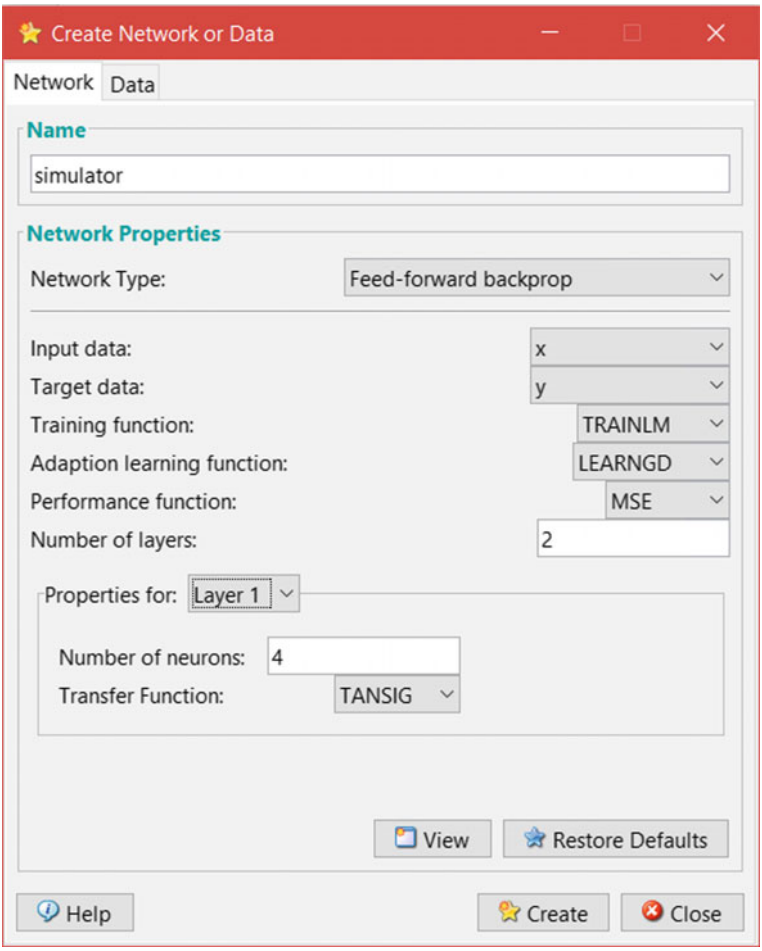


Fig. 1.35 Window of creating network/data; here create network is shown

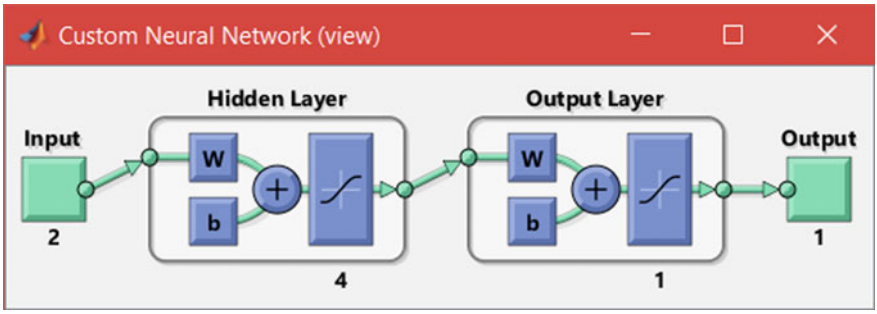


Fig. 1.36 Window to view the custom neural network

change any of the parts needed in this way. After checking the network in viewer window, select 'create' in the right bottom of 'create network data' window. The message box will show as in Fig. 1.37.

Select 'ok' and go to "Network/data manager" window in the part 'networks' the designed network appears as its chosen name, here as we named before the 'simulator' is appeared (Fig. 1.38).

Double-click on the network name and the new window 'Network: simulator' is opened (Fig. 1.39).

At top of this window there are six menu bars. First choose 'train' menu bar to start the training process.

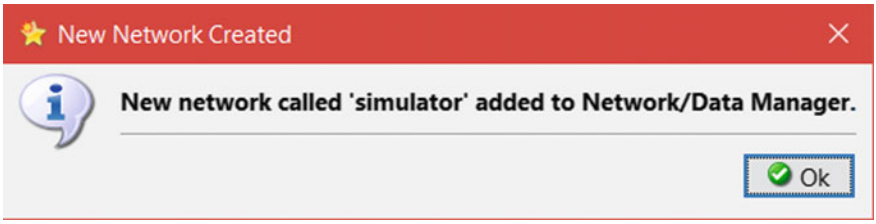


Fig. 1.37 Message box shows that new network is created

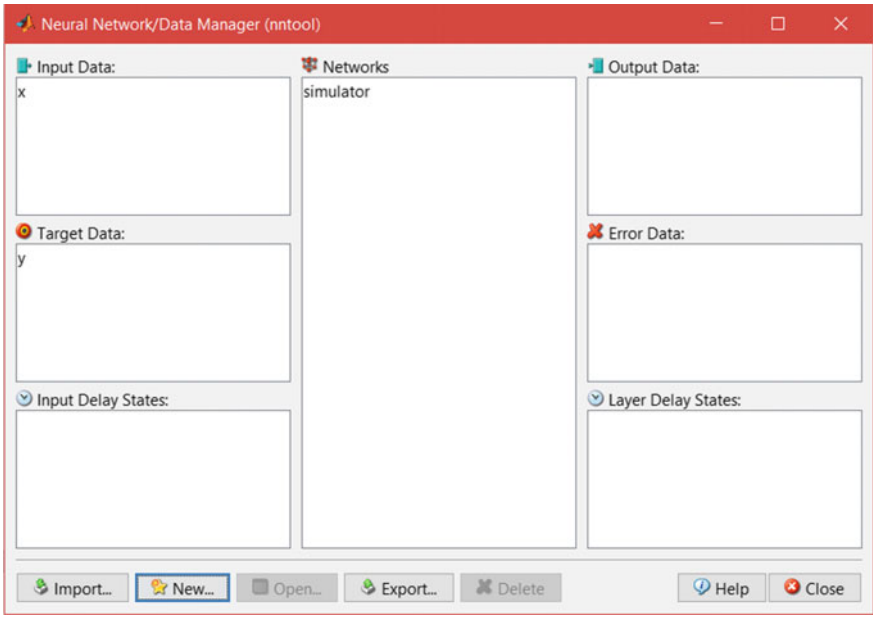


Fig. 1.38 "Network/data manager" window

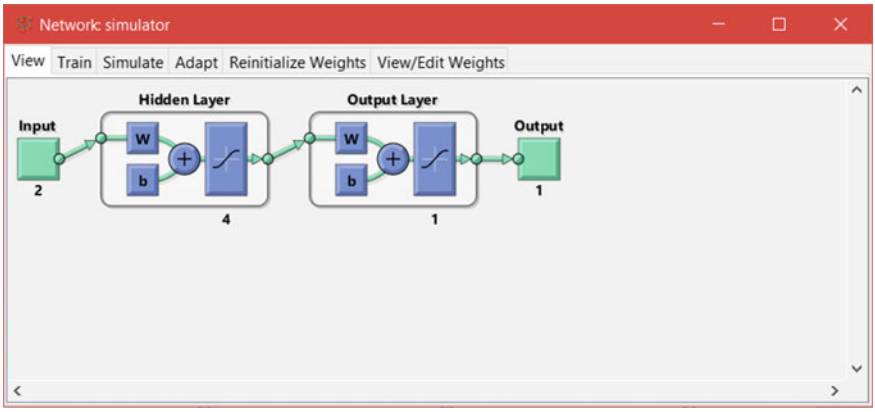


Fig. 1.39 Network simulator window for designed network in nntool

In this menu there are two parts:

– Training information as 'Training Info' (Fig. 1.40)

and Training Parameters (Fig. 1.41).

In the menu training information, you can select your own network inputs and targets and also assign names for outputs and errors of the network in the 'Training Results' part.

Here as the name of the network was defined as 'simulator' the defaults for output name is 'simulator_outputs' and for error name is 'simulator_errors' these names also can be changed through typing the new name in the related menu to a new name.

In the next sub-menu (of the 'train' menu) 'training parameters' menu is accessible, the parameters and their defaults are shown in Fig. 1.41.

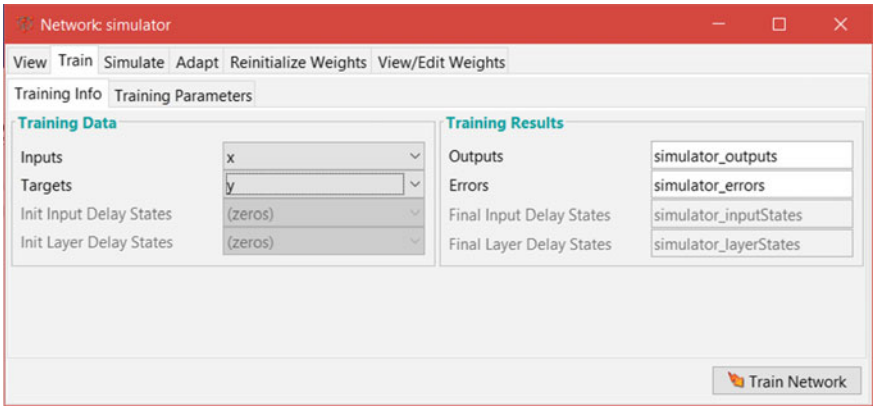


Fig. 1.40 Training information sub-menu in network simulator

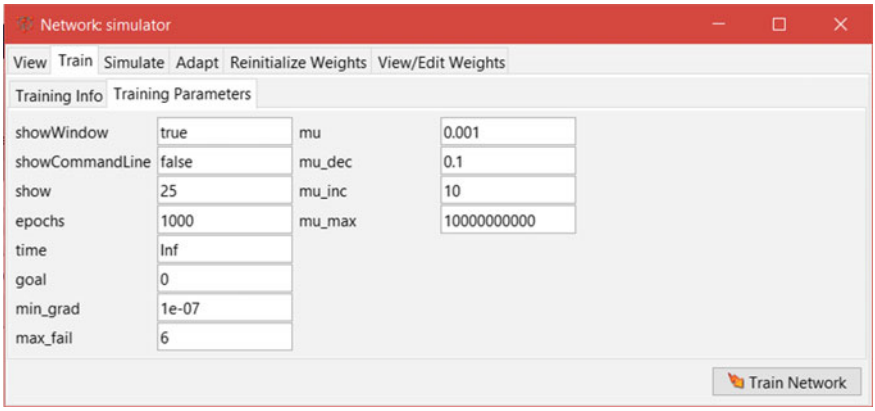


Fig. 1.41 Training parameters sub-menu in network simulator

Now, click on the icon "Train Network" to start training the designed ANN with the training information and parameters determined in the last step. In this case, the "Neural Network Training (un train tool)" window opens (Fig. 1.42).

This window displays the training progress and allows the user to interrupt training at any point by clicking 'stop training'.

In this window there are three main parts:

Neural Network: Here the schematic of the designed ANN is shown.

Algorithms: Here, the training algorithm, the performance function (and data division) selected before by the user, are shown.

The default for data division method is random data division.

In this method of division, the input data is divided randomly so that 60% of the samples are assigned to the training set, 20% to the validation set and 20% to the test set.

The philosophy of dividing of data into these three parts is to improve generalization by stopping learning before the global minimum of the training data set error (SSE); i.e. before the idiosyncrasies of the data set are learnt. The training data set is used to adapt the weights in the network and the test data set is used to determine when to stop training. The validation data set plays no role in training and is used to assess the generalization performance of the trained network.

The below information is shown per epoch:

Progress: in this part the number of epochs, time: elapsed time for training, performance: performance value change during training process, Gradient: the gradient of the propagates in every epoch.

Mu: the momentum,

validation checks: The numbers of samples used as validation check are displayed. For this example, the training stopped when the validation error increased for 6 iterations which occurred at iteration 12.

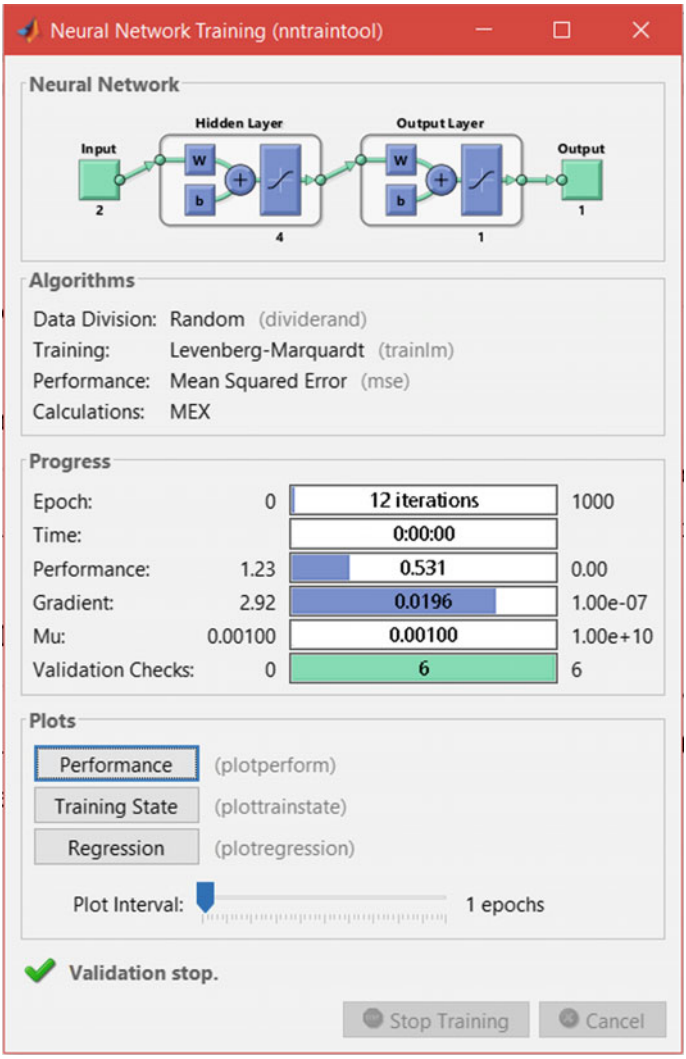


Fig. 1.42 Window of neural network training in nntool

Plots

In this part the plots of performance, training state and regression between outputs and target can be selected to be shown.

Click on 'performance', for this example, the Figure below will be displayed in a new window namely 'performance (plotperform)' (Fig. 1.43).

As it is shown in Fig. 1.43 the mean square Error (MSE) is plotted versus each of the epochs for training set, validation set and test data. To achieve the best number of epochs for training in which the over training is prevented while the MSE error for validation is at its minimum value.

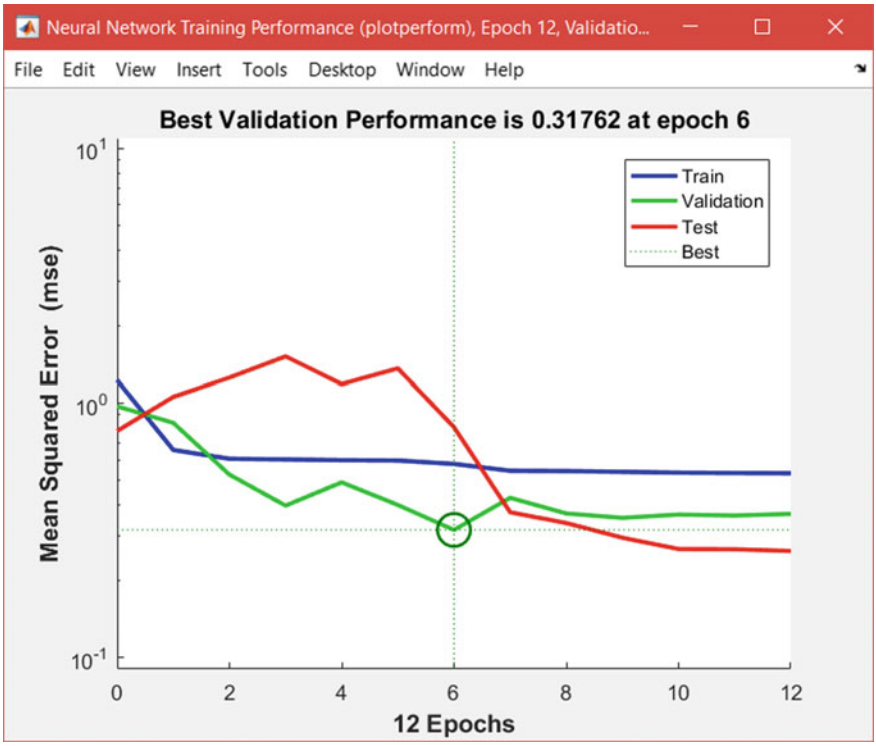


Fig. 1.43 Plot for neural network training performance in nntool

As is shown in Fig. 1.43, the best validation performance is achieved (0.31762) at epoch 6.

Clicking on the 'training state' bottom the 'plot train state' window will be opened. In this window the gradient, Mu, and validation checks are plotted versus (any of the) each of epochs.

For this example, as it is shown in Fig. 1.44, the final value for gradient, Mu and validation checks, in the last iteration (epoch 12) converges to 0.019621, 0.001 and 6.

Clicking on the 'Regression' four regression plots are drawn. They show the regression between target and output data with respect to each of training validation and test data separately and also the total response is shown as 'all data' plot (Fig. 1.45).

The regression plots help the user to know how much the output of the designed neural network are close to the target values by performing a linear regression, for this example the regression values are listed in Table 1.6 on the other hand these plots show how well the output tracks the Table 1.6 targets for training, testing and validation.

The regression plots also help the user to compare a designed ANN with another ANN with the same training data set. From the R^2 value point view very high values

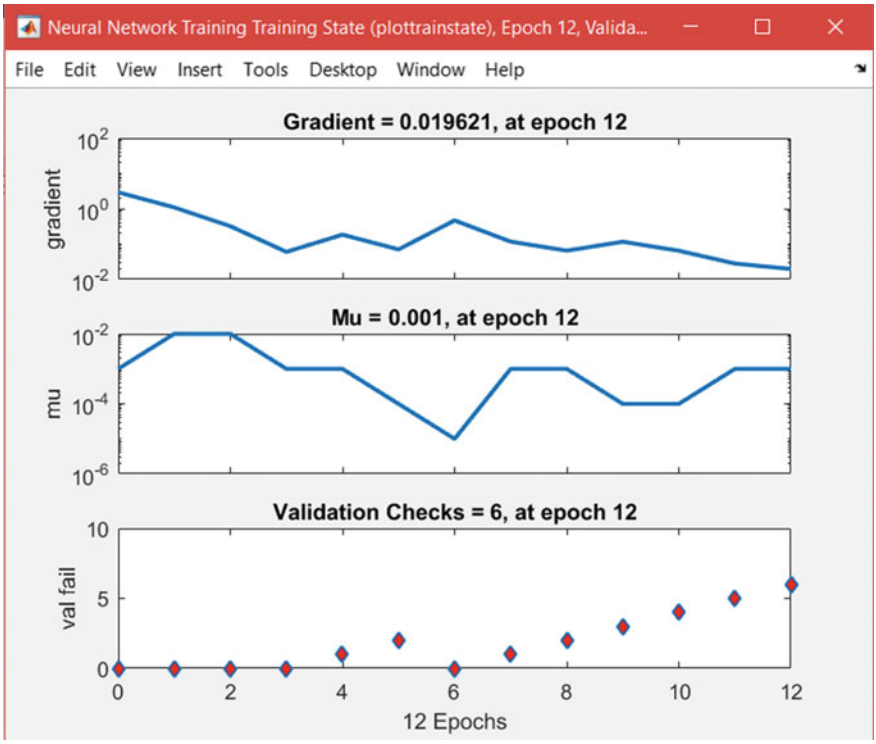


Fig. 1.44 Plot of neural network training state in nntool

of R^2 or very near to 1 means that over fitting is most probably happening which means that the number of training patterns might be over greater than that required.

It is necessary to mention that the results will be reasonable if:

- The final mean-square error is small.
- The test set error and the validation set error have similar characteristics.
- No significant over fitting has occurred by the iteration where the best validation performance occurs.

In this example, for a 2-4-1 ANN the results are not good enough, because:

1. In the performance window the MSE for training and testing are not small enough and are very much more that the error value for the validation data.
2. As it can be obviously seen in regression window (Fig. 1.45).

Some of data for training are far from the line Y = T especially for the initial values (between −1 and 0) located on the target axis.

This means the higher level of error for this domain of data train and test are very much larger than the error value for the validation data.

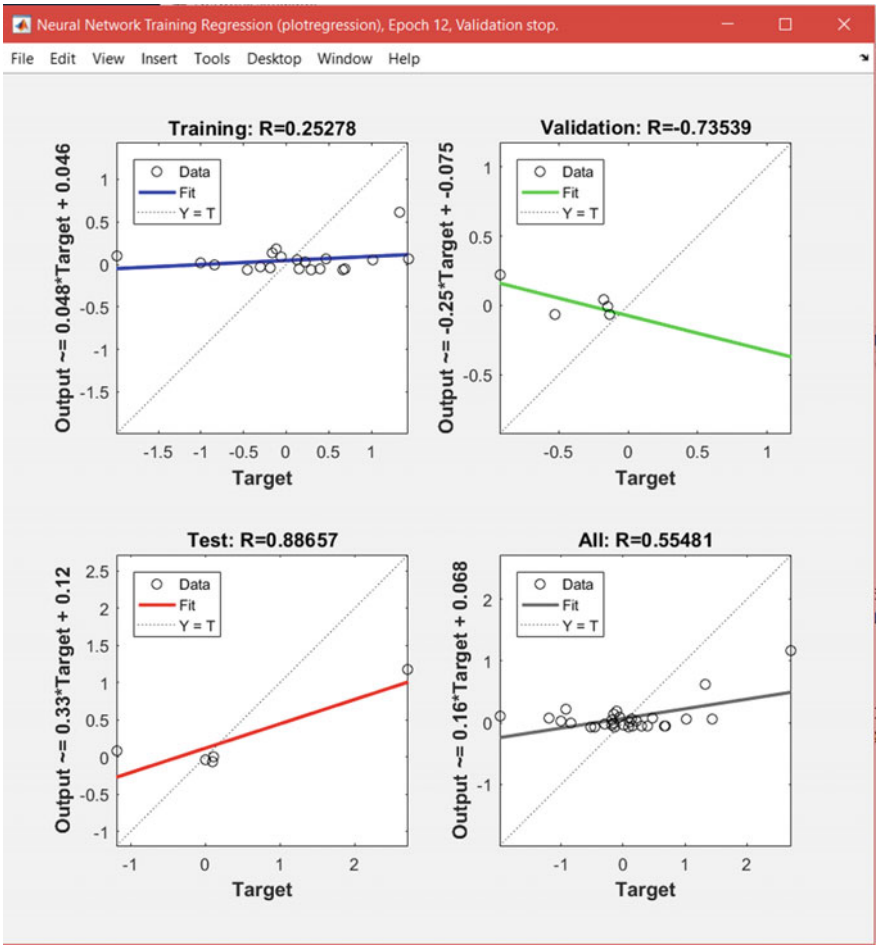


Fig. 1.45 Plot of the neural network training regression in nntool for a 2-4-1 MLP network

Table 1.6 Regression values of training for designed neural network

Case	R
Training	0.25278
Validation	−0.73539
Test	0.88657
All	0.55481

Furthermore, the test set error and validations set error do not have similar characteristics.

So, we increase the number of neurons in the hidden layer to 6 and test it again. The results for a 2-6-1 ANN are shown in Fig. 1.46.

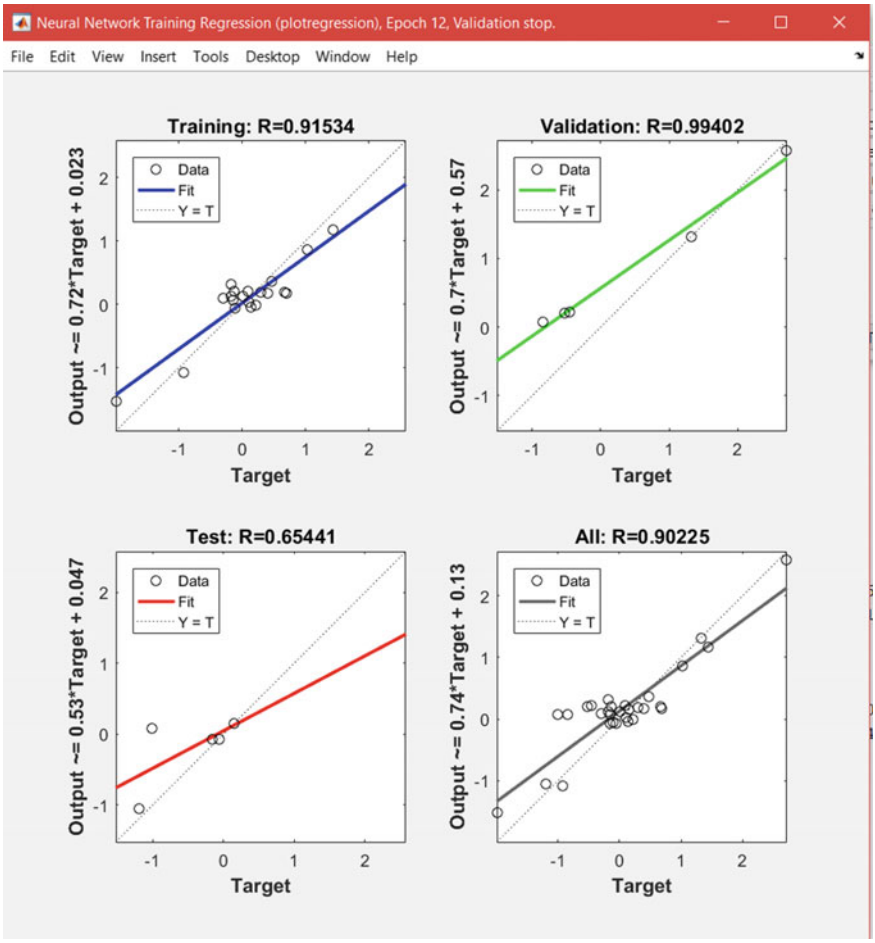


Fig. 1.46 Training results plot regression for a 2-6-1 ANN

In this state the output trains the targets very well for training, testing and validation and the R-value is over 0.90 also in this case the test set error and the validation set error have similar characteristics, so the ANN response is satisfactory.

If the designed ANN is not accurate enough and more accurate results are required, any of these approaches can be tried (followed):

- Reset the initial network weights and biases to new values referring to window (Fig. 1.47) and train again
- Increase the number of hidden neurons
- Increase the number of training vectors
- Increase the number of input values, if more relevant information is available.
- Try a different training algorithm.

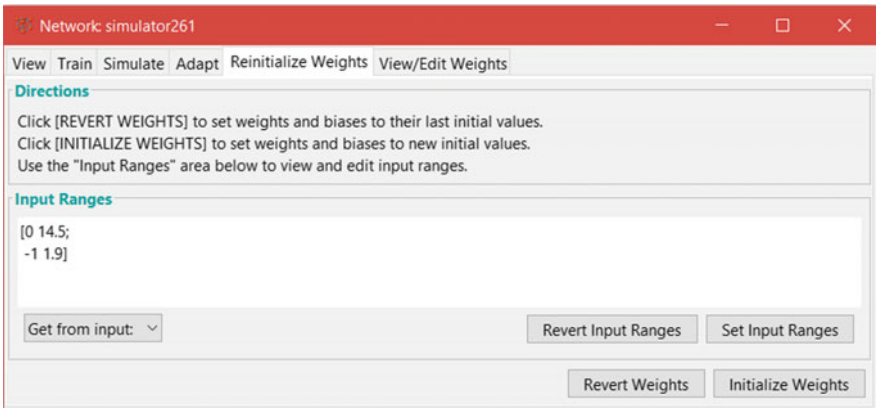


Fig. 1.47 View/edit weights menu in network simulator

Usually, more neurons require more computation but they allow the network to solve more complicated problems. More layers require more computation but their use might result in the network solving complex problems more efficiently. Of course more neurons don't necessarily mean more accuracy and may cause over-fitting as when the number of the neurons in s MLP network hidden layer is high/very high the number of modifiable weights will be high in relation to the number of patterns in the training data set and training is continued beyond the stage at which the general trends in the data are learnt, and the network begins to learn the noise in the data set. This is harmful to the ability of the network to generalize with new data sets. It depends on the degree of complexity of the problem to achievable the optimum value for the number of hidden neurons one way is to run ANN with different number of neurons and calculate the total MSE for each and compare the results via plot the test/train error MSE versus to select the optimum value. Hajian et al. (2012) used this method for choosing epochs.

The optimum value of hidden neurons which was designed for depth estimation of cavities, trained the neural network with the below algorithm starting from M = 1, until the stopping criterion was reached and the resultant error of the test and training data set stored.

This iterative process continued until it converged at a predefined maximum number for M. In the case of divergence, it is obvious that the number of epochs should be increased.

– **How good are multi-layer per feed-forward networks?**

It is clear that the approximation result of a MLP is not perfect. Kröse and Smagt (1996) noted that the resulting approximation error is influenced by:

1. The learning algorithm and number of iterations which determines how well the error on the training set is minimized.

2. The number of learning samples; this determines how well the training samples represent the actual function.
3. The number of hidden units. This determines the 'expressive power' of the network. For 'smooth' functions only a few number of hidden units are needed, for wildly fluctuating functions more hidden units will be needed.

References

Albora A. M., Özmen A., Ucan O. N., 2001, Residual Separation of Magnetic Fields Using a Cellular Neural Network Approach, Pure and Applied Geophysics, 158(9),1797–818. https://doi.org/10.1007/pl00001244.

Frost J., 2015, The Danger of Overfitting Regression Models, 3 September, http://blog.minitab.com/blog/adventures-in-statistics-2/the-danger-of-overfitting-regression-models.

Lawrence J., 1994, Introduction to Neural Networks: Design, Theory, and Applications, California Scientific Software, ISBN1883157005, 9781883157005.

Alarifi Nassir A. S.N., Alarifi N.S.N., Al-Humidan S., 2012, Earthquakes magnitude prediction using artificial neural network in northern Red Sea Area, Journal of king Saud University science, 2012, Vol. 24 (4): 301–313 https://doi.org/10.1016/j.jksus.2011.05.002.

Azoff, E.M., 1994, Neural Network time series forecasting of financial markets, Chi Chester: John Wiley and Sons.

Beale R. and Jackson T., 1990, Neural Computing: An introduction, IOP Publishing Ltd, ISBN 0-85274-262-2.

Bohlooli, A. and Jamshidi, K., 2011, "A GPS-Free Method for Vehicle Future Movement Directions Prediction using SOM for VANET", Applied Intelligence, 36(3), 685–697.

Calderón-Macías, C., Sen M.K., and Stoffa P.L., 1998, Automatic NMO correction and velocity estimation by a feedforward neural network, Geophysics 63(5), 1696–1707, https://doi.org/10.1190/1.1444465.

Chester, D.L., 1990. Why two hidden layers are better than one? Proceedings of the International Joint Conference on Neural Networks, 1265–1268.

Clifford A. and Aminzadeh F. 2011, Gas detection from absorption attributes and amplitude versus offset with artificial neural networks in grand by field, SEG expanded abstracts, Vol. 30, A Vol on New methods & case studies.

Cybenko, G., 1989, Approximations by superpositions of sigmoidal functions, Mathematics of Control, Signals, and Systems, 2 (4), 303–314.

Diersen S., En-jui Lee, Diana Spears, Po chen, Liqiang Wang, 2011, Classification of seismic windows using artificial neural networks, International Conference on Computational Science, ICCS 2011, Procedia Computer Science 4 (2011) 1572–1581.

Dysart P.S., J.J. Pulli, 1990, Regional seismic event classification at the noress array: seismological measurements and the use of trained neural networks, Bulletin of the seismological society of America 80 (68) (1990), 1910–1933.

Elawadi, E., A. Salem, and K. Ushijima, 2001, Detection of cavities and tunnels from gravity data using a neural network, Exploration Geophys. 32, 4, 204–208, https://doi.org/10.1071/eg01204.

El-Qady G. and Ushijima K., 2001, Inversion of DC resistivity data using neural networks, Geophysical Prospecting 49(4), 417 – 430. July 2001, https://doi.org/10.1046/j.1365-2478.2001.00267.x.

Eshaghzadeh A., Hajian A., 2018, "2D Inverse modeling of residual gravity anomalies from simple geometric shapes using modular feed-forward Neural Network", Annals of Geophysics, 61(1), SE115(1-5), https://doi.org/10.4401/ag-7540

Fahlman S. E. and C. Lebiere, 1990, The cascade-correlation learning architecture, CMU-CS-90-100 School of Computer Science, Carnegie Mellon University Publisher, 524–532.

Fernando J.S., Silva Dias, Valéria C.F. Barbosa and João B.C. Silva, 2010, Identification of earthquake phases in increased noise level conditions, Geophysical Research Abstracts, EGU 2010, 24–34. ftp.it.murdoch.edu.au/units/ICT482/Notes/Kohonen%20net.doc.

Fu, L. Y., 1999, A neuron filtering model and its neural network for space and time-varying signal processing: 3rd International Conference on Cognitive and Neural Systems, Boston University, Paper B03.

Geman, S., Bienenstock, E., and Doursat, R., 1992, Neural networks and the bias/variance dilemma: Neural Comp., 4, 1–58.

Grêt A.A., Klingelé E.E., and Kahle H.-G., 2000, Application of artificial neural networks for gravity interpretation in two dimensions: a test study, Bollettino Digeofisica teorica Edapplicata, 41(1), 1–20.

Gret, A. A. and Klingele, E. E., 1998, Application of Artificial Neural Networks for Gravity Interpretation in Two Dimension, Report No.279, Institute of Geodesy and Photogrammetery, Swiss Federal Institute of Technology, Zurich.

Günaydın, K. and Günaydın A., 2008, Peak ground acceleration prediction by artificial neural networks for Northwestern Turkey, Math. Probl. Eng. 2008, https://doi.org/10.1155/2008/919420.

Hajian A., Styles P., Zomorrodian H., 2011b, "Depth Estimation of Cavities from microgravity Data through Multi Adaptive Neuro Fuzzy Interference System", Near Surface 2011 – 17th European Meeting of Environmental and Engineering Geophysics, Leicester, UK.

Hajian A., Zomorrodian H., Styles P., 2012, Simultaneous Estimation of Shape Factor and Depth of Subsurface Cavities from Residual Gravity Anomalies using Feed-Forward Back-Propagation Neural Networks, Acta Geophysica, 60(4), 317–336. https://doi.org/10.2478/s11600-012-0049-1.

Hajian A., 2008, Depth estimation of gravity anomalies by Hopfield Network, proceeding of 5th Annual meeting, AOGS: Asia Oceania Geosciences Society Busan, Korea, 16–20 Jun.

Hajian A., Ardestani, E. V. and Lucas C., 2011a, Depth estimation of gravity anomalies using Hopfield Neural Networks, Journal of the Earth & Space Physics, 37(2), 1–9.

Hajian A., M Shirazi, 2015, Depth estimation of salt domes using gravity data through General Regression Neural Networks, case study: Mors Salt dome Denmark, Journal of the Earth and Space Physics, 41(3), 425–438(published in Persian Language with English abstract).

Hastie T., Tibshirani R., Friedman J., 2009, The Elements of Statistical Learning Data Mining, Inference, and Prediction, Springer Series in Statistics, Second Edition, Springer.

Haykin S., 1994, Neural Networks: A Comprehensive Foundation, 1st Edition, Prentice Hall PTR Upper Saddle River, NJ, USA ©1994, ISBN: 0023527617.

Haykin, S., 1999, Neural Networks: A Comprehensive Foundation, 2nd ed., Prentice-Hall Inc., Englewood Cliffs.

Hecht-Nelson, R., 1987, Kolmogorov's mapping neural network existence theorem, Proc., 1st IEEE Annual Conf. on Neural Networks, IEEE, Piscataway, NJ.

Hopfield J.J. and Tank DW., 1985, Neural computation of decisions in optimization problems, Biological Cybernetics, 52, 141–152.

Hopfield, J. J., 1984, Neural networks and physical systems with emergent collective computational abilities, Proceeding of the National Academy of Sciences, U.S.A.,79, 2554–2558. http://www.solver.com/partition-data.

Huang Z. and Mark A. Williamson, 1994, Geological pattern recognition and modeling with a general regression neural network, Canadian journal of Exploration geophysics, 30(1), 60–68.

Kaftan, I., and Salk, M., 2009, Determination Of Structure Parameters On Gravity Method By Using Radial Basis Functions Networks Case Study : Seferihisar Geothermal Area (Western Turkey). Society of Exploration Geophysicists. SEG Annual Meeting, 25–30 October, Houston, Texas.

Kröse and Smagt, 1996, An introduction to Neural Networks, Eighth edition, November 1996, The University of Amsterdam.

Kumar K., Janak I. and Konno M., 2002, Cross-borehole geological interpretation model based on a fuzzy neural network and geotomography, Geophysics, 67(4), 1177–1183, Technical papers, seismic interpretation (p-wave); https://doi.org/10.1190/1.1500379.

Langer, H., Nunnari G., and Occhipinti L., 1996, Estimation of seismic waveform governing parameters with neural networks, Journal of Geophys. Res. 101, B9, 20109–20118, https://doi.org/10.1029/96jb00948.

Lawrence, J. and Fredrickson, J., 1998, Brain Maker User's Guide and Reference Manual, 7th Ed., Nevada City, CA: California Scientific Software.

Levenberg, K., 1944, A method for the solution of certain nonlinear problems in least squares, Quart. Appl. Math. 2, 164–168.

Li, Y. and Oldenburg, D. W., 1996, 3D inversion of magnetic data, Geophysics, 61, 394–408. http://stats.stackexchange.com/questions/19048/what-is-the-difference-between-test-set-and-validation-set.

Marquardt, D.W., 1963, An algorithm for least-squares estimation of nonlinear parameters, SIAM J. Appl. Math. 11, 2, 431–441, https://doi.org/10.1137/0111030.

McCormack, M.D., Zaucha D.E., and Dushek D.W., 1993, First-break refraction event picking and seismic data trace editing using neural networks, Geophysics 58, 1, 67–78, DOI:0.1190/1.1443352.

Mirchandani, G., Cao, W., 1989, On hidden nodes for neural nets. IEEE Trans. Circuits Syst., 36, 661–664.

Moya A. and Irikura K., 2010, Inversion of a velocity model using artificial neural networks, Computers & Geosciences, 36, 1474–1483.

Mukat L. Sharma and Manoj K. Arora, 2005, Prediction of seismicity cycles in the Himalayas using artificial neural network, Acta Geophysica Polonica, 53(3), 299–309.

Murat, M.E., and A.J. Rudman, 1992, Automated first arrival picking: A neural network approach, Geophysical Prospecting, 40(6), 587–604, https://doi.org/10.1111/j.1365-2478.1992.tb00543.x.

Negnevitsky M., 2001 Artificial Intelligence Systems: A Guide to Intelligent Systems, 3rd Edition, Pearson Education Limited, Harlow.

Nigrin A., 1993, Neural networks for pattern recognition, Cambridge MA: The MIT Press.

Orr J.L., 1996, Introduction to Radial Basis Function networks, Centre for Cognitive Science University of Edinburgh; source: www.cc.gatech.edu/∼isbell/tutorials/rbf-intro.pdf.

Osman O., Albora A.M. and Ucan O.N., 2007, Forward modeling with forced Neural Network for Gravity Anomaly Profile, Mathematical Geology, 39, 593–605. https://doi.org/10.1016/j.coastaleng.2007.01.001.

Osman O., Albora A.M. and Ucan O.N., 2007, Forward modeling with forced Neural Network for Gravity Anomaly Profile, Mathematical Geology 39, 593–605. https://doi.org/10.1007/s11004-007-9114-8.

Park, H. I., 2011, Study for Application of Artificial Neural Networks in Geotechnical Problems. INTECH Open Access Publisher, ISBN 9533071885, 9789533071886.

Petrovay K.,2010, Solar Cycle Prediction, 2010, Living Rev. Solar Phys. 7, http://www.livingreviews.org/lrsp-2010-6, https://doi.org/10.12942/lrsp-2010-6.

Poulton, M.M., B.K. Sternberg, and C.E. Glass, 1992, Location of subsurface targets in geophysical data using neural networks, Geophysics 57, 12, 1534–1544, https://doi.org/10.1190/1.1443221.

Reyes J., Morales-Esteban A., Martínez-Álvarez F., 2013, Neural networks to predict earthquakes in Chile, Applied Soft Computing 13,1314–1328.

Richard M.P. and Lippmann R.P, neural network classifiers estimate Bayesian a posteriori probabilities, Neural computation, 3 (4), 461–483, 1991.

Roth, G., and A. Tarantola, 1994, Neural networks and inversion of seismic data, J. Geophys. Res. 99, B4, 6753–6768, https://doi.org/10.1029/93jb01563.

Rumelhart, D.E., G.E. Hinton, and R.J. Williams, 1986, learning internal representation by error back propagation. In: D.E. Rumelhart and J.L. Mc Clelland (eds.), Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 1. Foundations, MIT Press, Cambridge, USA, 318–362.

Salem A. and Ushijima K., Jeffrey Gamey T., Ravat D., 2001, Automatic detection of UXO from airborne magnetic data using a neural network, Subsurface Sensing Technologies and Applications Vol. 2, No. 3, 191–213.

Scales, J. A., and Snieder, R., 1998, What is noise?, Geophysics, 63, 1122–1124.

Scarpetta S., Giudicepietro F., Ezin E.C., Petrosino S., Del Pezzo E., Martini M., and Marinaro M., 2005, Automatic Classification of Seismic Signals at Mt. Vesuvius Volcano, Italy, Using Neural Networks, Bulletin of the Seismological Society of America, 95(1), 185–196, https://doi.org/10.1785/0120030075.

Schalkoff R. J., 2011, Artificial Neural Networks, Tata McGraw-Hill, "Neural Computing".

Specht D. F., 1991 A general regression neural network, IEEE Transactions on Neural Networks, 2 (6), 568–576, November 1991, https://doi.org/10.1109/72.97934.

Specht D.F., 1990, Probabilistic neural networks, Neural Networks, 3(1), 109–118, https://doi.org/10.1016/0893-6080(90)90049-q.

Styles P. and Hajian A., 2012, Generalized Regression Neural Networks for Cavities Depth Estimation using Microgravity Data, Case Study: Kalgorlie Gold, Near Surface Geoscience – 18th European Meeting of Environmental and Engineering Geophysics, DOI: https://doi.org/10.3997/2214-4609.20143479.

Styles P., 2012, Environmental Geophysics: Everything You Ever Wanted (needed!) to Know But Were Afraid to Ask, EAGE Publications BV, ISBN 10: 9073834333 - ISBN 13: 9789073834330.

Swingler K., 1996, Applying neural networks, A practical guide; 3rd edition, Morgan Kaufmann, ISBN-13: 978-0126791709, ISBN-10: 0126791708.

Tank, D. W. and Hopfield, J. J., 1986, Simple neural network optimization networks: An A/D converter, signal decision circuit, and a linear programming circuitries Transaction on Circuits and Systems, 33, 535–541.

Turhan Taner M., 1995, Neural networks and computation of neural networks and biases by the generalized delta rule and beck-propagation of errors, 1995 Proceeding of IEEE, Special Issue on Neural networks, Neural networks and their supervised learning, 1–11.

Uwamahoro J., McKinnell, Lee-Anne, Cilliers, Pierre. J., 2009, Forecasting solar cycle 24 using neural networks, Journal of Atmospheric and Solar-Terrestrial Physics, 71(5), 569–57.

Van der Baan, M., and Jutten C., 2000, Neural networks in geophysical applications, Geophysics 65(4), 1032–1047, https://doi.org/10.1190/1.1444797.

Wang, L. X. and Mendel, J. M., 1992, Adaptive minimum prediction-error deconvolution and source wavelet estimation using Hopfield neural networks, Geophysics, 57, 670–679.

Widrow B., 1962, Generalization and Information Storage in Networks of Adaline 'Neurons', in Self-Organizing Systems, symposium proceedings, M.C. Yovitz, G.T. Jacobi, and G. Goldstein, eds., 435–461, Spartan Books, Washington, DC.

Zhang, Y., and K.V. Paulson, 1997, Magnetotelluric inversion using regularized Hopfield neural networks, Geophys. Prospect. 45(5), 725–743, https://doi.org/10.1046/j.1365-2478.1997.660299.x.

Chapter 2
Prior Applications of Neural Networks in Geophysics

2.1 Introduction

As mentioned previously in the primary sections of chapter one, there are several, and varied applications of neural networks in geophysics; here we give a short list of some of these applications:

- Modeling of crustal velocity using Artificial Neural Networks (case study: Iran Geodynamic GPS Network (Ghaffari and Mohammadzadeh 2015)
- Crustal velocity field modeling with neural network and polynomials, SIDERIS, M.G. (Ed.), Moghtased-Azar and Zaletnyik (2009)
- Artificial neural network pruning approach for modular neural networks (Yilmaz 2013)
- Modular neural networks for seismic tomography (Barráez et al. 2002)
- Estimating one-dimensional models from frequency domain measurements
- Quantifying sand fraction from seismic attributes using modular artificial neural networks
- Borehole electrical resistivity modeling using neural networks
- Determination of facies from well logs using modular neural networks (Bhatt and Helle 2002)
- Inversion of self-potential anomalies caused by 2D-inclined sheets (Hesham 2009).
- 2D inverse modeling of residual gravity anomalies from Simple geometric shapes using Modular Feed-forward Neural Network (Eshaghzadeh and Hajian 2018)

Most applications involve training the network with a finite amount of data. While this may be adequate for approximating a well-behaved relationship between the input and output variables, it may not be suitable when the training data are fragmented or are a discontinuous representation of a mapping that has significant variation over the input parameter space.

A. Hajian and P. Styles, *Application of Soft Computing and Intelligent Methods in Geophysics*, Springer Geophysics,
https://doi.org/10.1007/978-3-319-66532-0_2

2.2 Application of Neural Networks in Gravity

One way to decrease the dependency of the interpretation method on the expert interpreter is to use intelligent methods in which the experience of the interpreter can be taught to a neural network. In the recent decade efforts have been focused on using neural networks for gravity anomalies with significant success. In recent years, Hajian et al. (2012) presented a neural-network based method for simultaneous estimation of the shape factor and depth of microgravity anomalies to locate subsurface cavities. The probable depth to cavities is of critical input in engineering design in many parts of the world especially in the southern US, Central America and the Middle East where extensive areas are underlain by great thicknesses of limestones which are often karstic (Hajian et al. 2012). Most of the research into the application of neural networks for the gravity method is associated with the interpretation of gravity anomalies. Various types of ANN, which have been used in gravity exploration, are listed in Table 2.1. However a combination of neural networks and fuzzy logic is very useful for gravity signal processing especially for noise attenuation of microgravity time series (repeated gravity observations at a fixed station), Negro et al. (2008) at INGV, Catania used a neuro-fuzzy system to eliminate the effect of pressure and temperature as noise sources on microgravity data from the gravity stations over Etna volcano.

Table 2.1 Various types of ANN used in gravity exploration

Researcher(s)/ year	Neural-net based method	Target
Salem et al. (2003a)	Hopfield neural network	Depth estimation of subsurface cavities in medford site
Hajian et al. (2007)	Radial basis function (RBF) neural network	Detection of subsurface sinkholes triggered by earthquakes
Osman et al. (2007)	Forced neural network	Forward modeling of gravity anomaly profiles
Hajian (2004, 2008)	Hopfield/MLP	Depth estimation of Buried Qanats
Hajian (2009)	Multi-layer perceptron	Estimation of dangerous subsidence of areas associated with earthquakes
Styles and Hajian (2012)	General regression neural networks	Cavities depth estimation using microgravity data
Hajian and Shirazi (2015)	General regression neural networks/MLP	Depth estimation of salt domes
Albora et al. (2001)	Cellular NN	Separation of regional/residual anomalies
Albora et al. (2007)	Cellular neural network	Tectonic modeling
Tierra and Freitas (2005)	MLP	Predicting gravity anomaly from sparse data
Sarzeaud et al. (2009)	Modified Kohonen neural network	Optimal Interpolation of gravity maps

2.2.1 *Depth Estimation of Buried Qanats Using a Hopfield Network*

Hajian et al. (2011a, b) presented a Hopfield Neural Network for depth estimation of gravity anomalies which was tested for both synthetic and real data. The case study on real data was an ancient buried Qanat located under the surface of the north entrance of institute of geophysics at the University of Tehran. The designed Hopfield neural model result was very close to the real depths (one 2.9 m and the other 9.25 m).

2.2.1.1 Extraction of Cost Function for Hopfield Neural Network

The gravity effect of an object, at an observation station (x, z = 0) caused by simple bodies such as a sphere or a horizontal cylinder (at x = 0 and depth = Z) (Fig. 2.1) is generally calculated using equation (2.1) (Abdelrahman et al. 2001):

$$g = \frac{A}{(x^2 + z^2)^q} \tag{2.1}$$

where q is the shape factor of the buried object:

(a) q = 1 for a horizontal cylinder
(b) q = 1.5 for a sphere
and A is the amplitude factor.

The value of 'q' lies between 1 and 1.5 for spherical and cylindrical shapes. From Eq. 2.1 it is obvious that the horizontal location of the object (x) can be estimated from the raw Bouger anomaly contours and even better from the residual anomaly contours. As an example in Fig. 2.2 the horizontal location of the object is clearly close to where the Bouger color of the contours tend to dark blue, which indicates the minimum value of the Bouger anomaly.

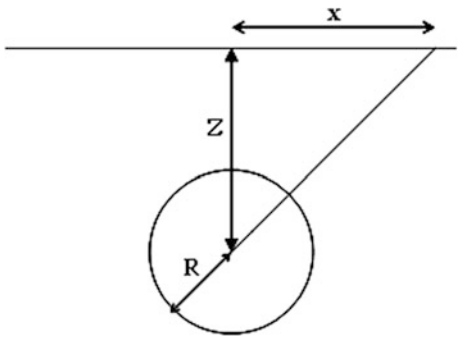


Fig. 2.1 Geometrical specifications of a simple body model

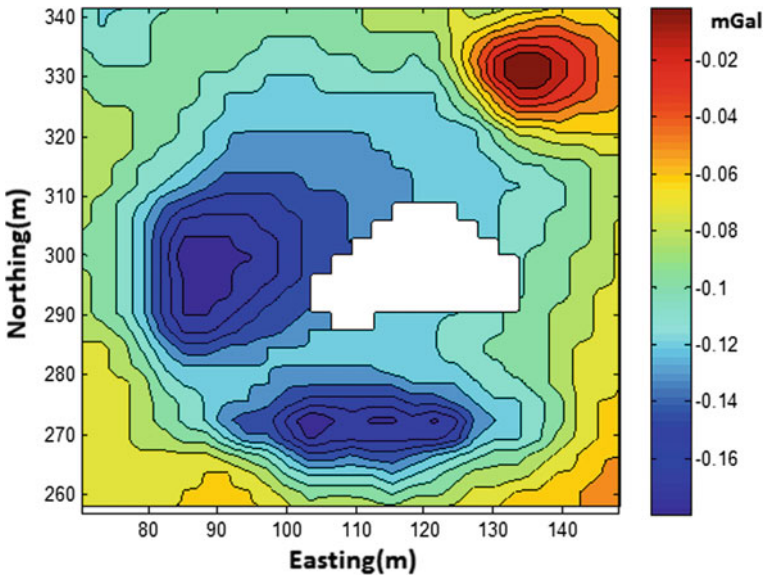


Fig. 2.2 Bouger anomaly map over the Buried Qanat

Suppose we have measured M gravity data on a principal profile over a buried object with unknown amplitude factor, 'A' which is located at the position (l). For estimating the amplitude factor, a cost function between the measured and calculated gravity anomaly is defined (see Eq. 2.3). The ideal gravity anomaly at station k can be written:

$$g_k^c = G_{lk}A \tag{2.2}$$

where G_{lk} represents the geometrical relationship between the position (i) and the observation point, k and A is the amplitude factor.

There is a problem here in that if we have noisy data, how can we calculate the amplitude factor q? This can be approximated in a least squares fashion as the solution that minimizes the cost function between the measured and calculated data. We define a cost function C, in terms of the sums of the squares of the differences between the measured and calculated data (Salem et al. 2003a):

$$C = \frac{1}{2}\sum_{k=1}^{M}\left(g_k - g_k^c\right)^2 = \frac{1}{2}\sum_{k=1}^{M}\left[g_k - G_{lk}A\right]^2 \tag{2.3}$$

where g_k represents the measured gravity data.

For Hopfield nets, the units have a binary threshold. This means that the units can take on only two values which are either 0 or 1. When the unit's input exceeds the threshold it output is '1' and otherwise it is '0'. Hence, the amplitude factor A

should be consistent with the output of the Hopfield network, where a typical bit is 1 or 0. With reference to binary digits rules the amplitude factor can be expressed as:

$$A = \sum_{i=1}^{n=D+U+1} 2^{i-D-1} b_i \tag{2.4}$$

where:

D the number of down bits
U is the number of up bits
b is a binary digit (0 or 1). Obviously the values of D and U depend on the precision and amplitude, respectively. For example if A = $(100.11)_2$ then U = 3 and D = 2 so A = $(4.75)_{10}$

Substituting Eq. (2.3) into Eq. (2.4) gives the connection weights and the initial values of inputs as:

$$W_{lij} = -\sum_{k=1}^{M} 2^{(i+j-2D-2)} (G_{lk})^2 \tag{2.5}$$

$$I_{ij} = \frac{1}{2} \sum_{k=1}^{M} (2^{(i+j-2D-2)} G_{LK})^2 b_i + \sum_{k=1}^{M} 2^{(i-D-1)} G_{lk} g_k \tag{2.6}$$

Consequently the energy of the Hopfield net for estimating the amplitude factor at location (l) is as below:

$$E_l(b) = -\frac{1}{2} \sum_{i=1}^{n} \sum_{j \neq i=1}^{n} W_{ij} b_j b_i - \sum_{i=1}^{n} I_{li} b_i \tag{2.7}$$

We selected 9 neurons for the Hopfield neural network because the accuracy of gravity data was 10 microGal and so 9 bits are needed to present the amplitude factor of gravity value in binary digits. (As mentioned before with reference to Eq. (2.4) the binary value of the amplitude factor of gravity data consists of D+U+1 digit.

We then applied different values of depth (Z) and calculated the amplitude factor and final minimized cost function for each depth by the Hopfield network.

The depth value at which the cost function has the minimum value is assumed to be the nearest value to the real depth of the gravity source.

2.2.1.2 Synthetic Data and the Hopfield Network Estimator in Practical Cases

To initialize a Hopfield Network, the values of the units are set to the desired start pattern. Then the network is sequentially updated until it converges to an attractor pattern. The convergence in a Hopfield is achieved when it gets to a state where the

pattern can't change after the next updating. Therefore, in the context of Hopfield Networks, an attractor pattern is a final stable state, a pattern that cannot change any value within it under updating. To clarify this concept, suppose there is a topographic area with valleys and peaks along which a ball runs (Fig. 2.3). It is obvious that the ball will stop in one of the valleys but the actual stop point among the available valleys depends on its momentum. If the ball has enough momentum at the starting point, it will stop in the valley with the minimum depth. So we run a Hopfield neural network with different initial values of depth to find the stable state where the energy of the network achieves a steady state. The output which has the least energy indicates the optimum estimate of the depth of the anomaly.

In order to examine the network behavior, the gravity effect of a cylinder with noise levels of 5%, and 10% was used. When the Hopfield net achieved its stable state, the optimum depth was estimated using the procedure described in the last paragraph. The results are shown in Table 2.2.

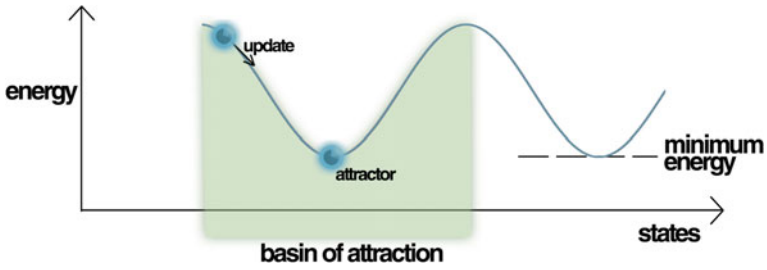


Fig. 2.3 Energy variation during the updating process in a Hopfield neural network. *Source* https://en.wikipedia.org/wiki/Hopfield_network#/media/File:Energy_landscape.png

Table 2.2 Outputs of present of Hopfield neural network in 10% noise

Training values for R, Z		Outputs of MLP (3,5,2) in present of 10% noise			
Horizontal cylinder	Sphere or vertical cylinder	Horizontal cylinder		Sphere or vertical cylinder	
R(m)	Z(m)	R(m)	Z(m)	R(m)	Z(m)
1	2	1.17	2.22	1.12	2.22
1	3	1.22	3.45	1.08	3.45
2	4	2.15	4.25	2.09	4.25
2	5	2.18	4.28	2.14	4.28
3	6	3.25	6.53	3.17	6.53
4	8	4.17	8.76	4.28	8.76
5	13	5.65	13.45	5.30	13.45
6	14	6.25	13.4	6.31	13.40
6	15	6.25	14.65	6.35	14.65

This shows that the ability of Hopfield net is good for the depth estimation of shallow objects in the presence of noise sources such as tunnels or massive buildings near the gravity observed stations. Explicit design of a suitable filter to attenuate environmental noise, for use with gravity data, is difficult but neural networks are commonly less sensitive to noise. As is shown in Table 2.2, the network was tested for synthetic gravity data for cylinder and sphere models in the presence of noise and the values of depth (Z) and amplitude factor (A) were calculated. It is clear from Table 2.2 that the depth estimates are close to their training value, seven, in the presence of 10% noise.

In the next stage the network was tested for a set of gravity data measured over a subterranean canal (Qanat) located in the north of the Geophysics Institute at the University of Tehran. We first select a principal profile from the gravity network profiles (Fig. 2.4).

The principal profile data is perpendicular to the extension of the anomaly. So we selected the profiles shown in Fig. 2.5 (lines A and B).

In the next stage a simple spherical or cylindrical body was assumed and the values of G and g_k in Eqs. 2.6 and 2.7) were extracted from the principal profiles. Then the weights W_{ij} and thresholds I_{ij} are calculated via Eqs. 2.5 and 2.6 to

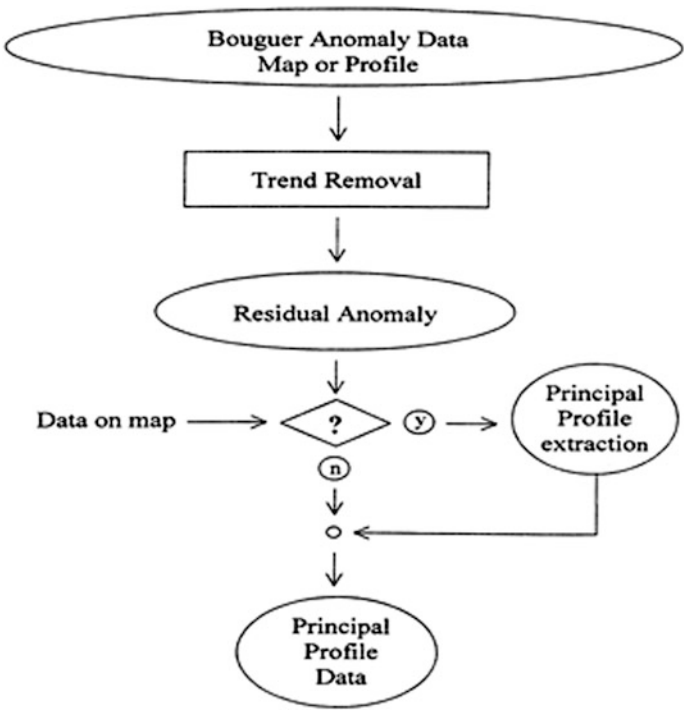


Fig. 2.4 Flowchart of selecting principal profile data. Redrawn after GRÊT et al. (2000)

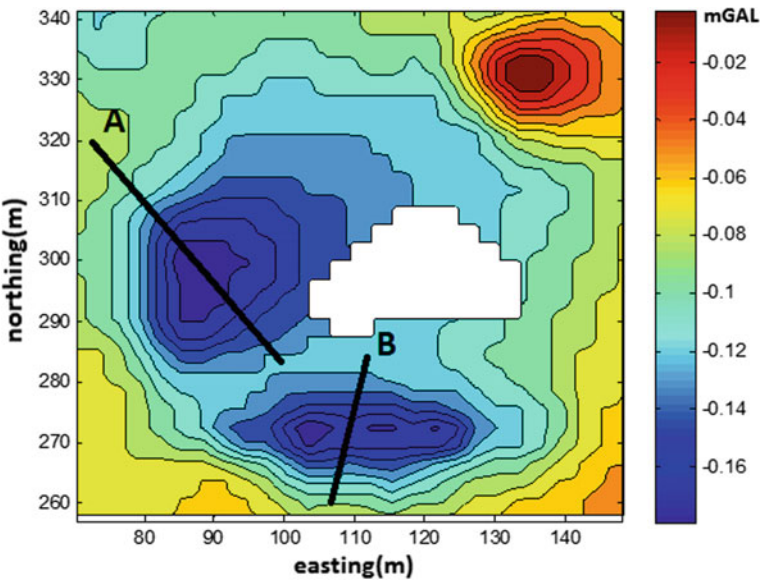


Fig. 2.5 The principal profiles selected for interpretation with Hopfield NN

minimize the energy function. Then different random initial values of amplitude factor ('A' in Eq. 2.1) were applied to the Hopfield network.

As the accuracy of gravity data was 0.01 mGal; 9 bits are needed to present the amplitude factor of gravity value in binary digits. So we used a 9-neuron Hopfield neural network. The network updates are repeated until the new output is equal to the output of the last updating stage. In this condition the network has reached its stable state which means that the energy function has been minimized. The energy function for each initial value is calculated using Eq. 2.7 and the output of the network, which has the minimum energy function, indicates the real amplitude factor. The result from the Hopfield neural network was 2.5 m for profile A and 9.5 m for profile B, which when compared to the real depths which were respectively 2.9 and 9.25 m and so this method has a good depth estimation accuracy.

2.2.1.3 Conclusions

It this section a Hopfield neural network was presented for depth estimation of gravity anomalies caused by simple objects with geometries close to cylindrical or spherical The designed network was tested on noisy synthetic data and was also examined for two principal profiles of a real case study and the results showed the good accuracy of the network as the Qanat depth estimated by the network (2.5 m) was very close to its real value (2.9 m).

2.2.2 *Depth Estimation of Salt Domes Using Gravity Anomalies Through General Regression Neural Networks*

In this section, as an example we describe how it is possible to estimate salt dome depth through the use of General Regression Neural Network (GRNN). The results of this method were compared to the most common neural networks; Multi-Layer Perceptron (MLP) and both GRNN and MLP models were tested. The model's appropriateness for application to genuine cases, was degraded by adding Gaussian noise to the information to reproduce a few levels of vulnerability and the outcomes were compared to a traditional strategy, in particular the standardized full gradient method.

The results showed that both the GRNN and MLP are robust to noise but that GRNN is more precise than either the least-squares minimization or MLP method.

The main purpose of the interpretation of gravity data is essentially the estimation of the location and depth of the causative sources. Although initial simple models may be geologically unrealistic, they are typically adequate to permit the analysis of the sources of many isolated anomalies (Nettleton 1976; Abdelrahman and El-Araby 1993). A few strategies have been described for the interpretation of gravity anomalies. Such methods include graphical methods (Nettelton 1976), Fourier transformation (Sharma and Geldart 1968), Mellin transforms techniques (Mohan et al. 1986), ratio techniques (Bowin et al. 1986; Abdelrahman et al. 1989), least-squares minimization approaches (Gupta 1983; Lines and Treitel 1984; Abdelrahman et al. 1991; Salem et al. 2003a), Euler deconvolution (Thompson 1982; Reid et al. 1990), neural networks (Elawadi et al. 2001; Osman et al. 2007), 3D analytic signal amplitude (Li 2006), continuous wavelet transforms (Chamoli et al. 2006), eigenvector analysis of gravity tensor (Beiki and Pedersen 2010), gravity gradient tensor invariants and vertical component analysis (Oruç 2010) and finally multi-adaptive neuro fuzzy interference systems (Hajian et al. 2011a, b).

Neural networks have become increasingly popular in geophysics during this last decade (Grêt et al. 2000) since they are generalized approximators which can approximate any continuous function to any desired accuracy. In the geophysical domain, neural networks have been used for waveform recognition and first-peak picking (Murat and Rudman 1992; McCormack et al. 1993); for electromagnetic interpretation (Poulton et al. 1992; Al-Garni 2009), magneto telluric (Zhang and Paulson 1997), and seismic inversion purposes (Roth and Tarantola 1994; Langer et al. 1996); neural networks (Elawadi et al. 2001; Osman et al. 2007; Styles and Hajian 2012); multi-adaptive neuro–fuzzy interference systems (Hajian et al. 2011a, b).

Salem et al. (2003b) used a Hopfield network for depth estimation of cavities. Osman et al. (2007) used forced neural networks for forward modeling of gravity anomaly profiles. Styles and Hajian (2012) used Generalized Regression Neural Networks for cavity depth estimation.

2.2.2.1 GRNN and MLP Design and Test

To design GRNN and MLP neural networks for depth estimation of salt domes, we select some features, which are characteristic of the gravity data instead of utilising all the gravity points of an observed profile to diminish the complexity of the designed neural network.

The features, we use as inputs, were selected based on the features which Grêt et al. (2000) have previously used successfully for gravity depth estimation. The main reason to choose these inputs is to cover the important features from the observed gravity anomaly. The features F1, F2, F3 and F4 which were used as inputs are defined as below (Fig. 2.6):

$$F_1 = g_{Max}, F_2 = X_{g50\%}, F_3 = X_{g75\%}, F_4 = X_{gMax} \tag{2.8}$$

- F1: The maximum gravity value ($F_1 = g_{Max}$).
- F_2: The distance over which the value of the gravity anomaly falls to 50% of the maximum value, which is known as the half-width ($F_2 = X_{g50\ \%}$).
- F_3: The horizontal distance over which the value of the gravity anomaly falls to 75% of the maximum value.
- F_4: The horizontal distance of the maximum gravity value (X_{gMax}).

The process of training and design of the GRNN/MLP neural network is shown in Fig. 2.7.

The flow Chart for the GRNN training procedure is shown in Fig. 2.8. In order to test the designed GRNN we produced 50 pairs of synthetic gravity data by selecting 50 random depths for the three geometrical shapes: sphere, horizontal cylinder and vertical cylinder. Finally the RMSE (Root Mean Square Error) was selected as an index to compare the methods, and the estimation error for each model and for both near surface and deep objects was evaluated. The results are listed in Tables 2.3 and 2.4.

After utilizing the synthetic data for training, the application to a nearly-real case was tested by adding Gaussian noise to the data. To simulate a few levels of vulnerability, we performed two different levels of noise addition: 5% Gaussian noise and 10% Gaussian noise. The results of the GRNN in the presence of noise were compared with the MLP and Non-linear least square method to assess the

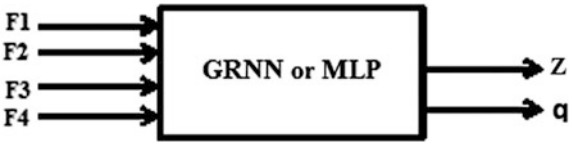


Fig. 2.6 Input-output structure of GRNN/MLP

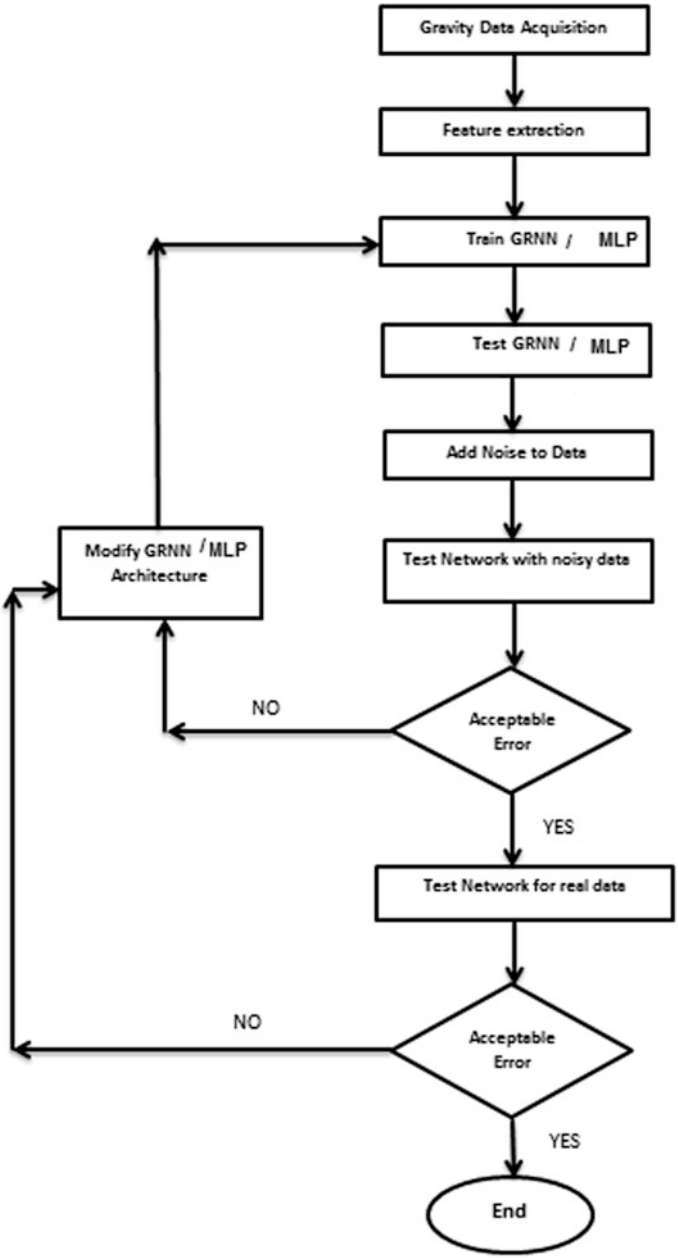


Fig. 2.7 Flowchart of GRNN/MLP neural network training and design

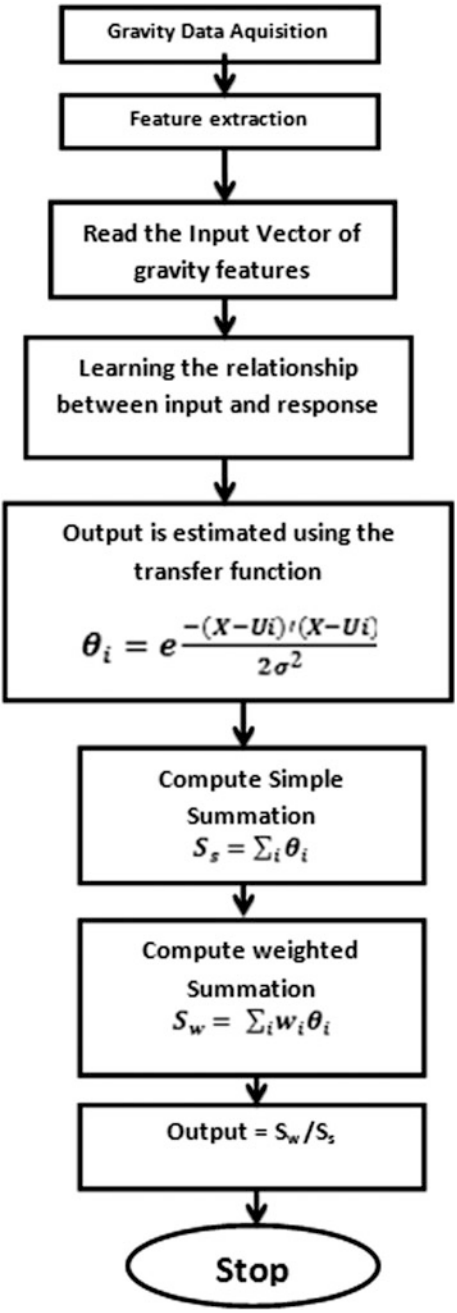


Fig. 2.8 Flow chart for GRNN training procedure

Table 2.3 Root Mean Square Error (RMSE) of the GRNN and MLP network and Non-Linear Least Squares Minimization (NLLSM) method for near surface object synthetic data without noise, note that the depths (Z) are in meters and q is the geometric factor

	Root Mean Square Error (RMSE)					
Model	Oil		Oil-water		Oil-gas	
Parameters	Q	Z	Q	Z	q	Z
MLP	0.06	0.24	0.15	0.45	0.07	0.35
GRNN	0.02	0.13	0.10	0.32	0.04	0.18

Table 2.4 Root Mean Square Error (RMSE) of the GRNN and MLP network and Non-Linear Least Squares Minimization (NLLSM) method for deep object synthetic data without noise, note that the depths (Z) are in kilometers

	Root Mean Square Error (RMSE)					
Method	Sphere		Vertical cylinder		Horizontal cylinder	
Parameters	Z	q	Z	q	Z	Q
MLP	0.045	0.09	0.061	0.23	0.084	0.30
GRNN	0.033	0.04	0.042	0.16	0.067	0.20
NLLSM	0.057	0.05	0.069	0.19	0.090	0.23

Table 2.5 Root Mean Square Error (RMSE) value of the GRNN, MLP network and Non-Linear Least Square Minimization (NLLSM) method for near surface objects and synthetic data in the presence of several level of Noise to Signal (N/S)

	Root Mean Square Error (RMSE)						
Model	N/S (%)	Sphere		Vertical cylinder		Horizontal cylinder	
Parameters	—	Q	Z	Q	Z	Q	Z
MLP	5	0.12	0.33	0.19	0.25	0.17	0.15
GRNN	5	0.09	0.32	0.15	0.22	0.13	0.12
NLLSM	5	0.15	0.28	0.26	0.32	0.18	0.19
MLP	10	0.26	0.50	0.24	0.63	0.21	0.56
GRNN	10	0.19	0.37	0.20	0.49	0.16	0.45
NLLSM	10	0.36	0.65	0.25	0.88	0.19	0.53

The depth values are in meters

model robustness. The RMS error value of each method was calculated and listed in Tables 2.5 and 2.6. The results showed that the GRNN model is more robust in the presence of noise as compared to the MLP and Non-Linear Least Square Minimization approaches.

Table 2.6 Root Mean Square Error (RMSE) value of the GRNN, MLP network and Non-Linear Least Square Minimization (NLLSM) method for deep objects synthetic data in the presence of several level of Noise to Signal (N/S)

	Root Mean Square Error (RMSE)						
Model	N/S (%)	Sphere		Vertical cylinder		Horizontal cylinder	
Parameters	—	q	Z	q	Z	q	Z
MLP	5	0.15	0.45	0.20	0.38	0.19	0.55
GRNN	5	0.15	0.38	0.18	0.30	0.26	0.40
NLLSM	5	0.18	0.47	0.150	0.32	0.22	0.45
MLP	10	0.25	0.55	0.34	0.63	0.24	1.03
GRNN	10	0.22	0.49	0.25	0.48	0.20	0.88
NLLSM	10	0.35	0.65	0.30	0.67	0.31	1.14

The depth values are in kilometers

2.2.2.2 Location of Field Data

The case study was located in Denmark, Mors. The investigation was confined to the island of Mors, which is located in the northwestern piece of Jutland, Denmark (Fig. 2.9). The island covers a zone of around 360 km^2. It is 10–15 km wide and around 35 km long with a SSW-NNE alignment.

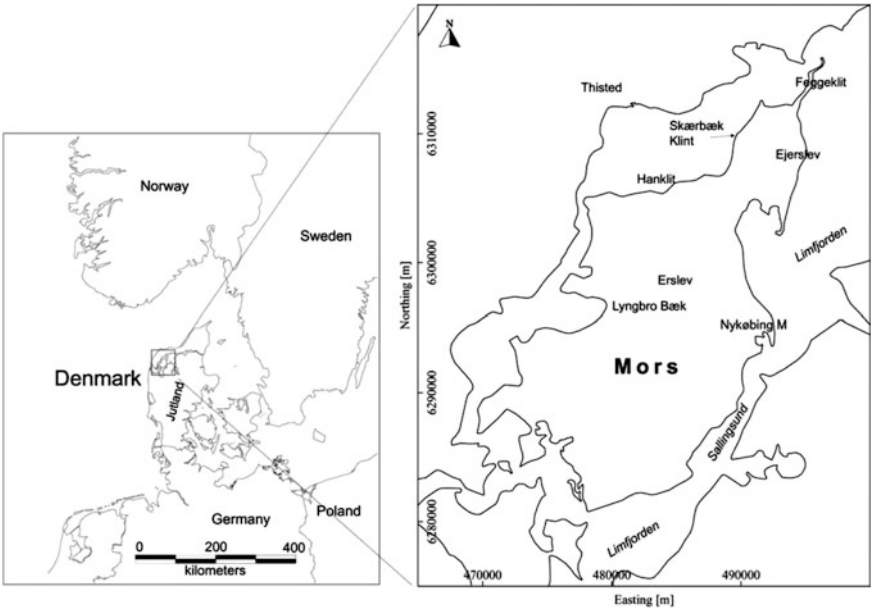


Fig. 2.9 Location map of Mors (Jorgensen et al. 2005)

2.2.2.3 Geological Setting of Mors

The structural contour lines at 25 m intervals show the elevation of the pre-Quaternary surface. The dome structure on Mors is made of chalk that covers the highest point of the Erslev salt diaper. The boundaries between glaciotectonic complexes and their foreland on northern Mors are also shown in Figs. 2.10 and 2.11 (Jorgensen et al. 2005).

The sub-Quaternary strata comprise Upper Cretaceous white chalk (Maastrichtian) and Paleocene limestone (Danian) covered by clays and diatomite (Paleocene–Eocene) in the Fur Formation (Pedersen and Surlyk 1983). These layers are followed by Oligocene micaceous clay and, to the south, also by Miocene clay, silt and sand (Gravesen 1990). The Paleogene sediments are in general 50–250 m thick, yet, locally they thinner or even missing. They have normally been subjected to deformation during Quaternary glaciations, yet the most significant effect on the Paleogene topography has been due to the down-cutting of a series of incised Quaternary valleys. These valleys are generally filled with glacial deposits of various ages but otherwise the surface is generally covered by a thin veneer of glacial sediments. The general structure of the Tertiary and Quaternary deposits on Mors is controlled by (1) The Mors salt diapir, (2) glaciotectonic events during the Quaternary glaciations and (3) Extensive networks of incised valleys.

2.2.2.4 Results and Discussion

Data Analysis

The data used in this research were organized into inputs as described previously

$$(\mathrm{F}_1 = \mathrm{g}_{\mathrm{Max}}, \mathrm{F}_2 = \mathrm{Xg}_{50\%}, \mathrm{F}_3 = \mathrm{Xg}_{75\%}, \mathrm{F}_4 = \mathrm{Xg}_{\mathrm{Max}}) \text{ and } \mathrm{output}(\mathrm{z}).$$

We used Grav2dc for modeling and surfur11 for neural network (NN) processing and additional MATLAB (R2012a) built-in codes. Three separate subsets were selected for training, validation and testing. It is worth noting that the data were first modeled in a conventional manner as a salt dome, and then the main data from "Mors" applied.

Modeling

Grav2dc is a well-known geophysical software package for gravity data modeling and inversion. Each body making up the model has its own individual density. The current salt dome model contains 9 bodies with specific densities. The maximum horizontal extent of this model is 60 km.

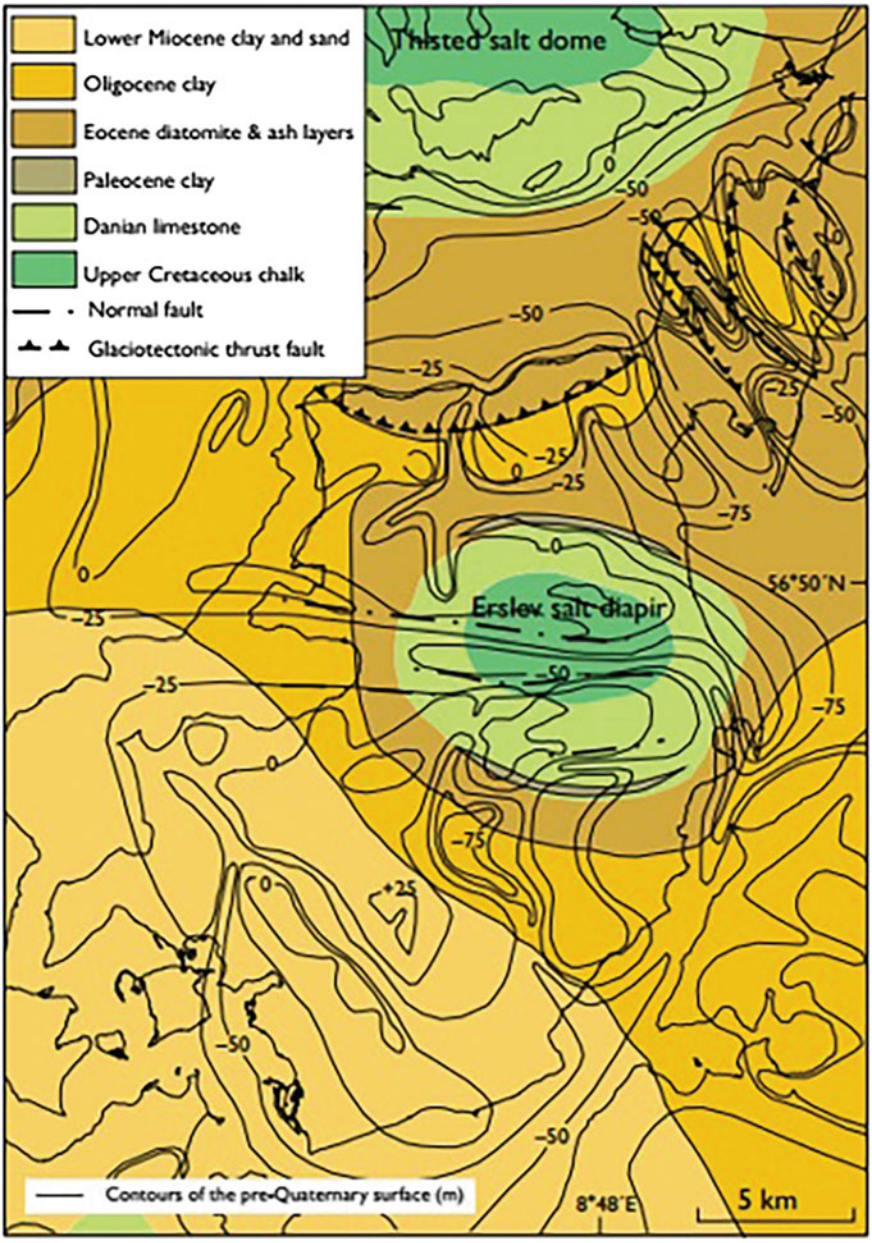


Fig. 2.10 Bedrock map of Mors (Jorgensen et al. 2005)

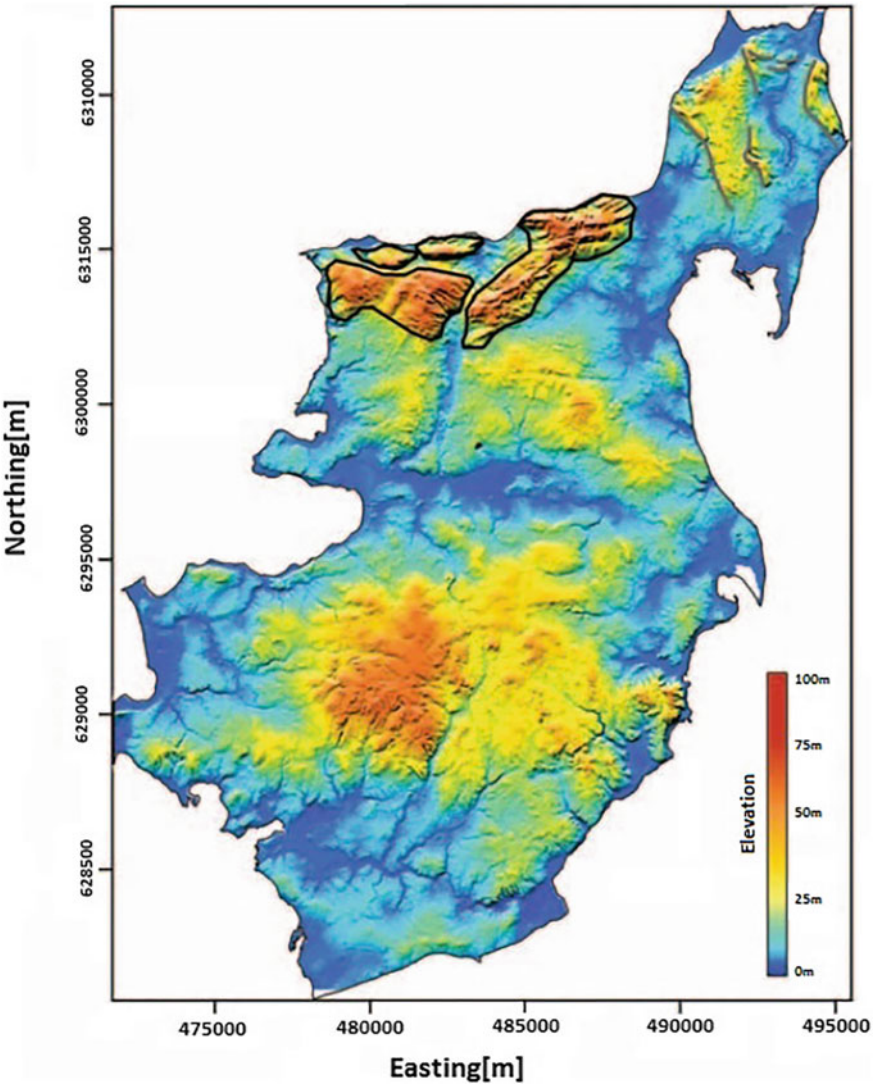


Fig. 2.11 Topographic map of Mors (Jorgensen et al. 2005)

A typical set of modeled bodies is shown in Fig. 2.12 with the synthetic gravity anomaly from this model shown in Fig. 2.13.

After modeling, synthetic data created by Grav2dc was imported to Excel and all data prepared for trained by the neural networks as described in the next section.

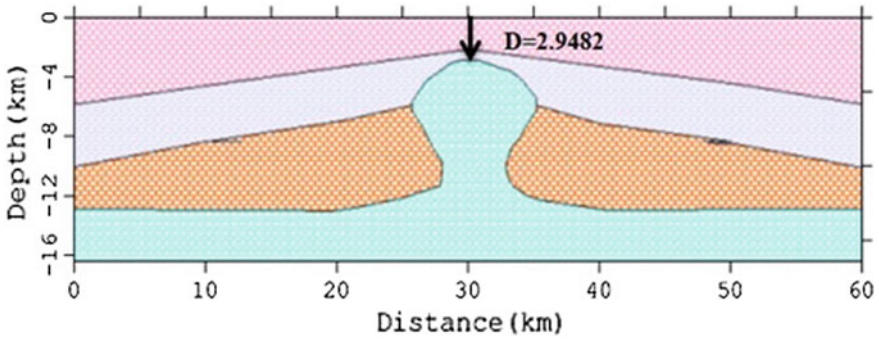


Fig. 2.12 Geometrical model of a salt dome, depth = 2.93 km (Hajian and Shirazi 2015)

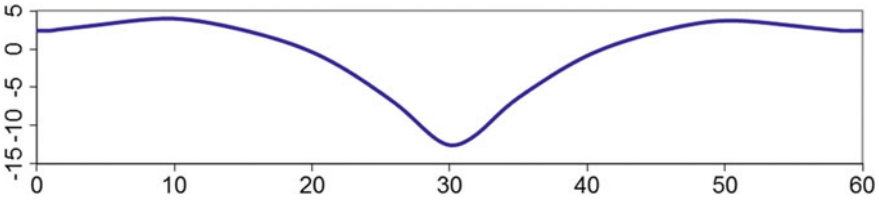


Fig. 2.13 Gravity anomaly of the salt dome in Fig. 2.12, modeled by Grav2dc (Hajian and Shirazi 2015)

Training and Testing of GRNN

As shown in Fig. 2.14 in order to prepare M pairs of training data, Hajian and Shirazi (2016) divided the primary training models into three separate groups of salt domes with differing internal fluid structure, including: Oil–Gas, Oil–Water, Only Oil and salt dome alone which are shown in Figs. 2.15, 2.16 and 2.17. The figures contain the parameters of the salt dome, depth, geological layers and location of anomalies. The gravity anomaly is calculated for these models and the features described previously are extracted and used in GRNN codes as inputs and output (see flowchart in Fig. 2.14).

The specific features used from the gravity anomalies include

$$F_1 = \mathbf{g_{Max}}, F_2 = Xg_{50\%}, F_3 = Xg_{75\%}, F_4 = Xg_{Max} \text{and output}(z).$$

The GRNN architecture used in this research is shown in Fig. 2.18. We used these four models in order to have four training sets of synthetic data. It is important to mention that all models have different depth of anomalies. The GRNN code should be trained for two different parallel codes, so we applied the GRNN for data

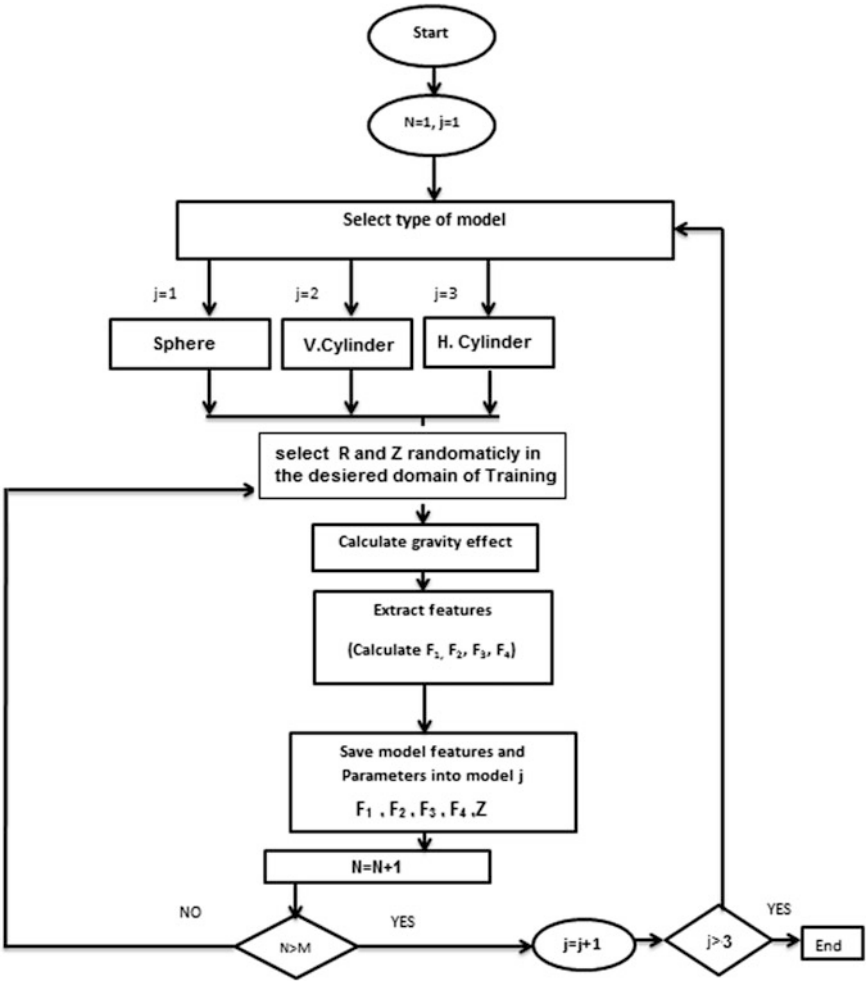


Fig. 2.14 Flowchart of preparing M pairs of training data for an MLP/GRNN neural network

with high depths and low depths. All data were imported to GRNN and the network tested with different MSE to select the best value, which is 0.45 (Fig. 2.19).

The network was tested on a typical gravity anomaly profile selected from the real gravity data over Mors (Figs. 2.20 and 2.21); the results are shown in Table 2.7 and were then compared to the normalized full gradient method which was used by Aghajani et al. (2009). The GRNN estimation for salt dome depth is near to the depth estimated by NFG method.

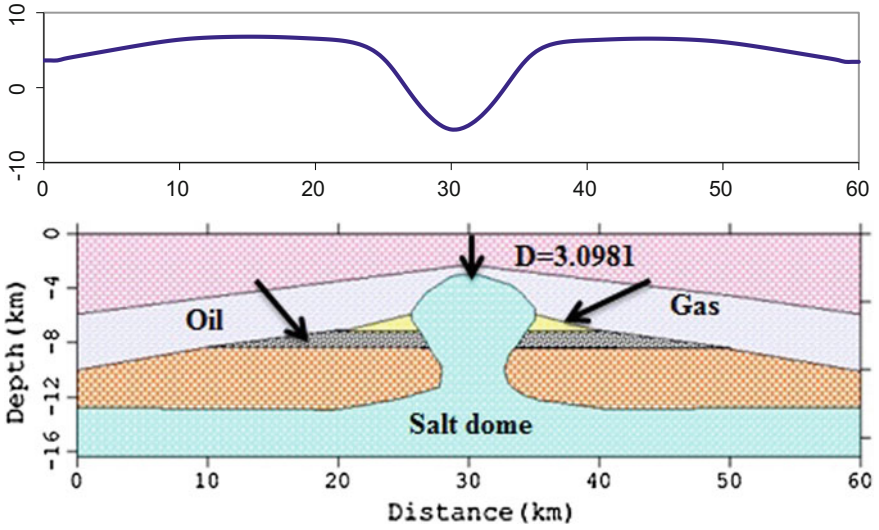


Fig. 2.15 Primary training model, Gas–Oil model (Hajian and Shirazi 2015)

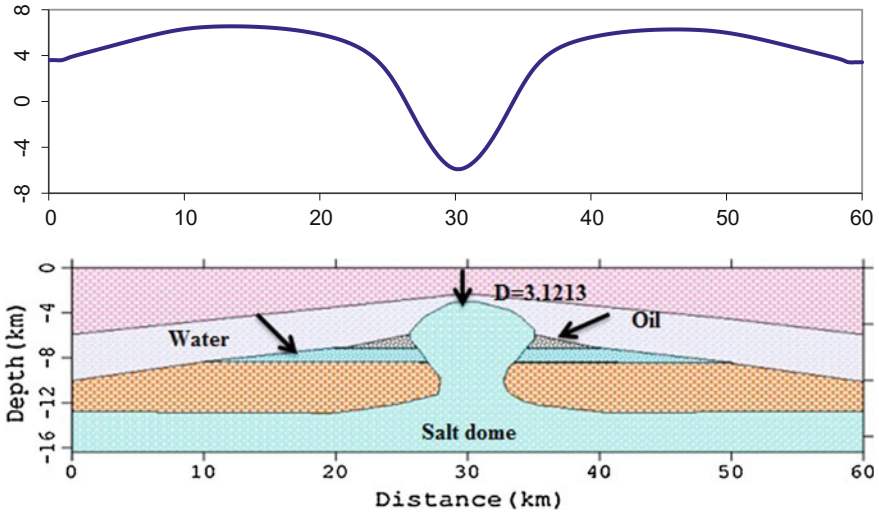


Fig. 2.16 Primary training model, Oil–Water model (Hajian and Shirazi 2015)

Training and Testing of MLP

The data modeled in the previous section was also used to train the MLP with the same inputs and output as used for GRNN training. The flowchart of the MLP training procedure is shown in Fig. 2.22.

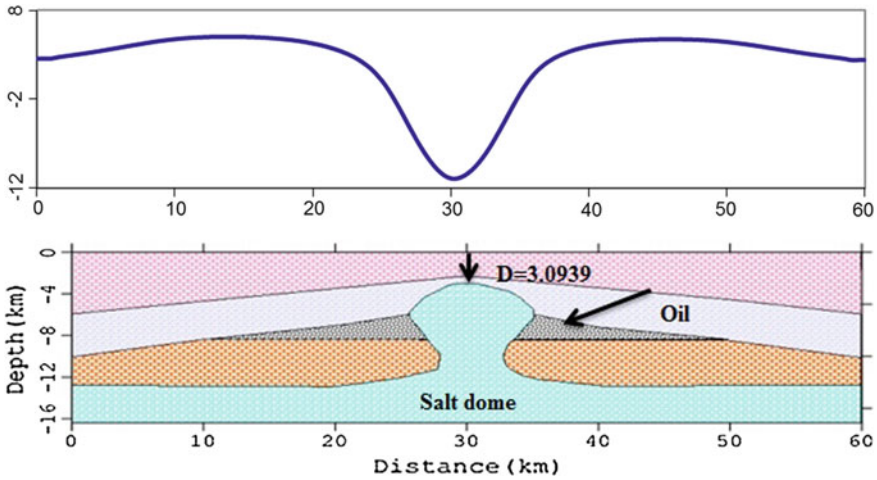


Fig. 2.17 Primary training model, Oil–Water model (Hajian and Shirazi 2015)

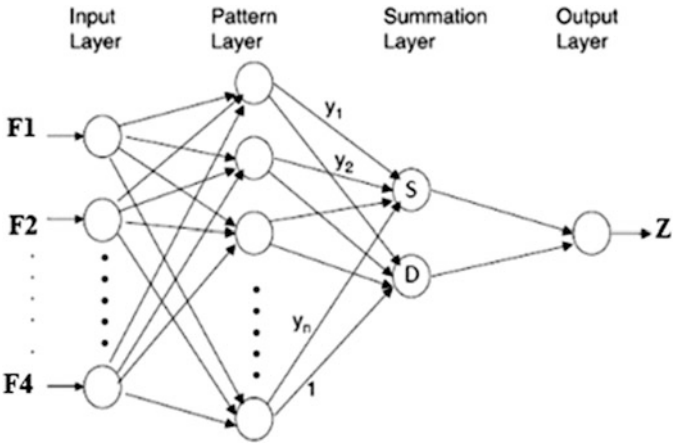


Fig. 2.18 Detailed Structure of GRNN with inputs and output (Styles and Hajian 2012)

In order to assess the optimal number of neurons the network was tested with synthetic data with different number of neurons in the hidden layer, and the MSE error was calculated (Fig. 2.23). This showed that the optimum number of neurons in the hidden layer where the MSE reaches its minimum value is 10. In order to compare the results with GRNN, we tested this network for the same synthetic and real gravity data. As shown in Fig. 2.24, the best validation performance at epoch 50 for real data is 0.0429. The estimate the depth to the Mors salt dome is shown in Table 2.8 and is 4.7 km.

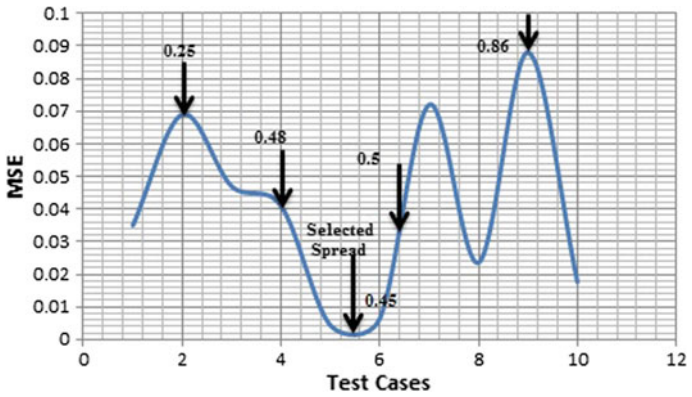


Fig. 2.19 Selecting the optimum value for spread (Hajian and Shirazi 2015)

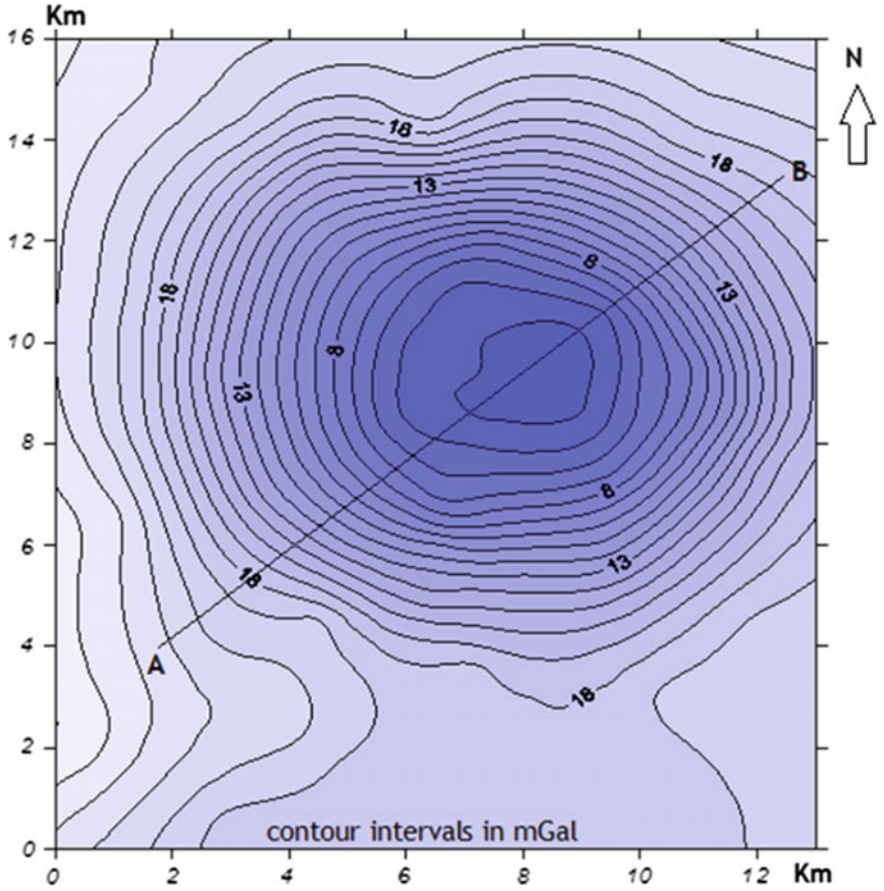


Fig. 2.20 Gravity anomaly over Mors salt dome (Hajian and Shirazi 2015)

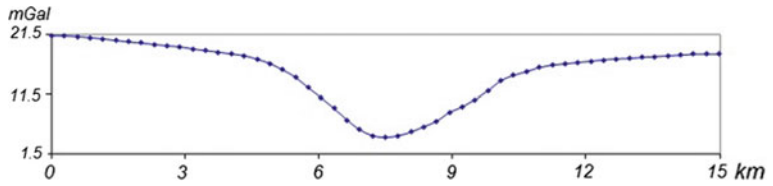


Fig. 2.21 Gravity along a selected principal profile over the Mors salt dome (Hajian and Shirazi 2015)

Table 2.7 The result of GRNN applied for real data

Value of spread	MSE Of learning	Estimated depth by GRNN	Estimated by NFG Method Aghajani et al. (2009)
0.45	0.0009	4.82 km	4.7 km

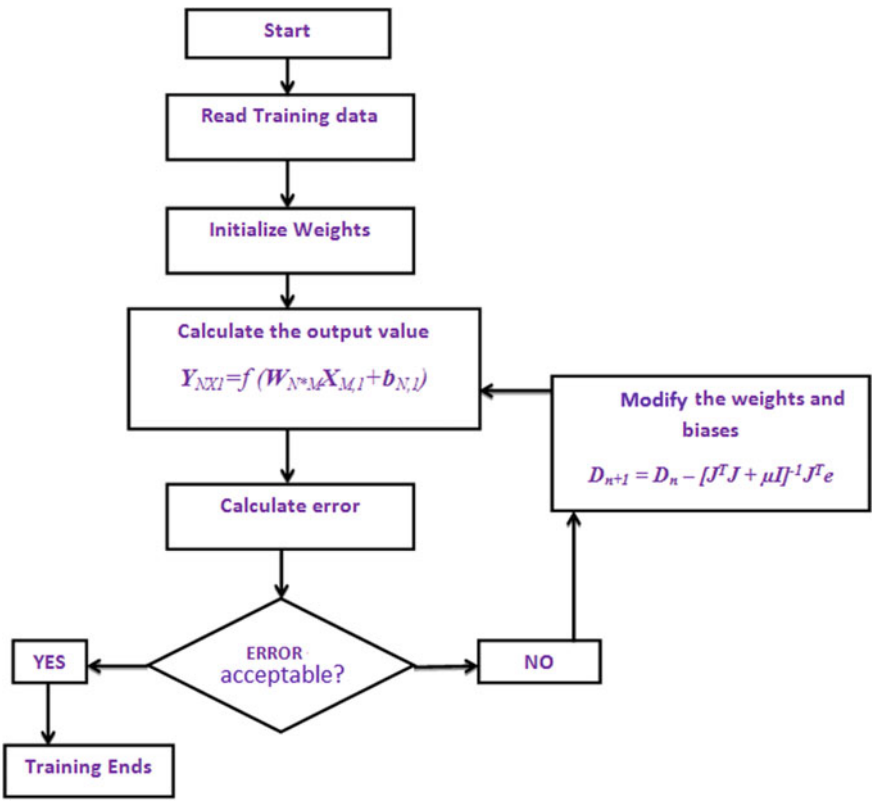


Fig. 2.22 Flow chart for MLP training procedure

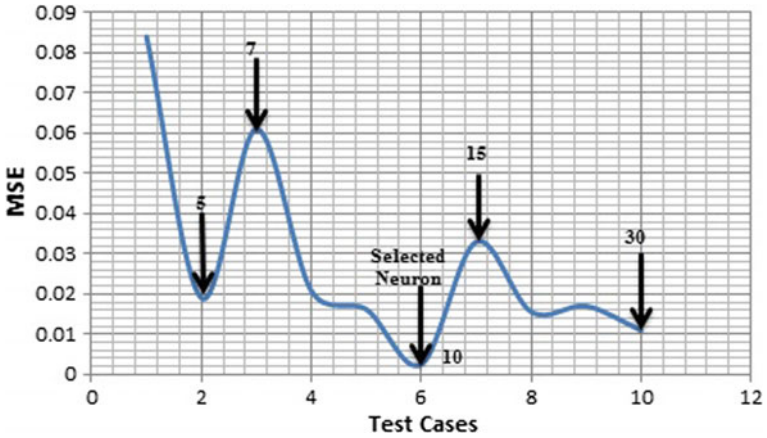


Fig. 2.23 Selecting the optimum value for the number of neurons in the hidden layer for a MLP network (Hajian and Shirazi 2015)

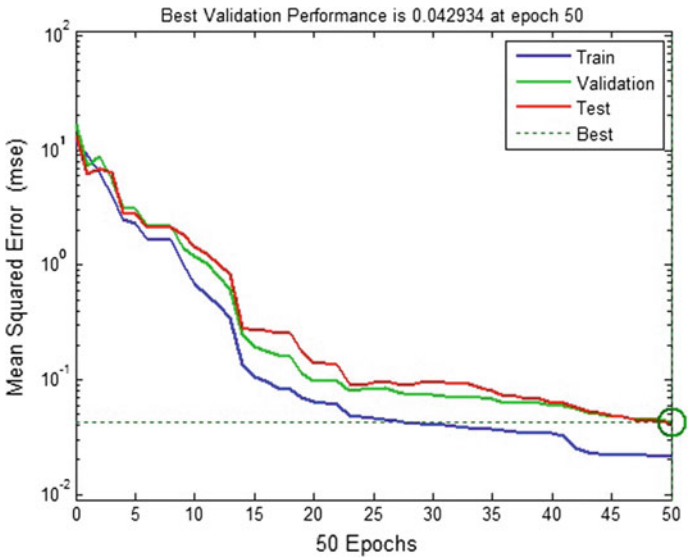


Fig. 2.24 The best validation performance at epoch 50 for real data. MSE = 0.04293, and the estimated depth = 4.98 km. Hajian and Shirazi (2015)

Table 2.8 The result of MLP applied to the principal gravity anomaly profile over the Mors salt dome (Hajian and Shirazi 2015)

Number of neurons in hidden layer	MSE	Estimated Depth by MLP	Estimated depth by NFG Method Aghajani et al. (2009)
10	0.042934	4.98 km	4.7 km

2.2.2.5 Conclusions

In this section, a novel method using an artificial neural network via GRNN and MLP was used to estimate the depth of a salt dome from gravity data. Results for synthetic data and real data via GRNN show that this network can perform reasonably well for noisy gravity data. The MLP network was applied for various numbers of neurons in the hidden layer and the GRNN network applied for several numbers of spreads, the result for MLP is 10 neurons in the hidden layer and for GRNN the spread factor is 0.45. Both GRNN and MLP methods were tested on real data for the Mors salt dome in Denmark. The results of calculating the depth of salt dome showed that GRNN is more accurate than MLP for estimating the depth of the salt dome and needs less data for training. According to the results, the possible depth of Mors salt dome is 4.82 km for training data. According to the NFG method, the depth of salt dome is estimated accurately. We have developed this method for gravity data interpretation in order to simulate and estimate the depth of a salt dome anomaly as an example, but this method is applicable to other complex problems, as well.

2.2.3 Simultaneous Estimation of Depth and Shape Factor of Subsurface Cavities

Hajian et al. (2012) developed a feed-forward back-propagation (FFBP) neural network to simultaneously estimate the shape factor and depth of subsurface cavities from residual gravity anomalies. Unlike the common classical methods, no pre-assumptions were made about source shape and the designed FFBP neural network estimated both the depth and shape factor of buried objects using both the gravity and the prior geological information of the area from the real data.

There are various methods for depth estimation of gravity anomalies. Most of the conventional interpretation methods like Euler devolution (Thompson 1982) least-squares minimization (Abdelrahman et al. 2001), Fourier transform (Sharma and Geldart 1968), continuous wavelet transform (Chamoli et al. 2006), Eigen vector analysis of gravity tensor (Beiki and Pederson 2010), Mellin transforms (Mohan et al. 1986) depend crucially on the experience of the interpreter to assign some of the selectable parameters and almost most of the methods are not robust in the presence of noise. For example the analytical signal method can only detect the edges of the object, the Euler method produces different responses for differing window size and/or structural index and how good the structural index is assigned to the model is particularly dependent on the expertise of the interpreter (Hajian et al. 2011a, b).

An intelligent method should not depend on the experience of the interpreter and we believe that this is available through artificial neural networks, because they are able to learn from training data. They can estimate the output for a new input, which

is not used as the training data, but lies within the domain of the training set (Hajian et al. 2012). The advantage of using FFBP neural networks in this way is that they are more robust in the presence of the noise which is always present in gravity data. Furthermore, once the neural network is trained it performs satisfactorily for any new observations in the training space without requiring re-calculation of the initial parameters. Hajian et al. (2012) used a novel multi-start algorithm to optimize the number of neurons in the hidden layer. The related flowchart is depicted in Fig. 2.25.

The inputs of the neural network that Hajian et al. (2012) used for simultaneous estimation of depth and shape factor are the normalized residual gravity of the complete set of gravity stations along the selected principal profile, so the number

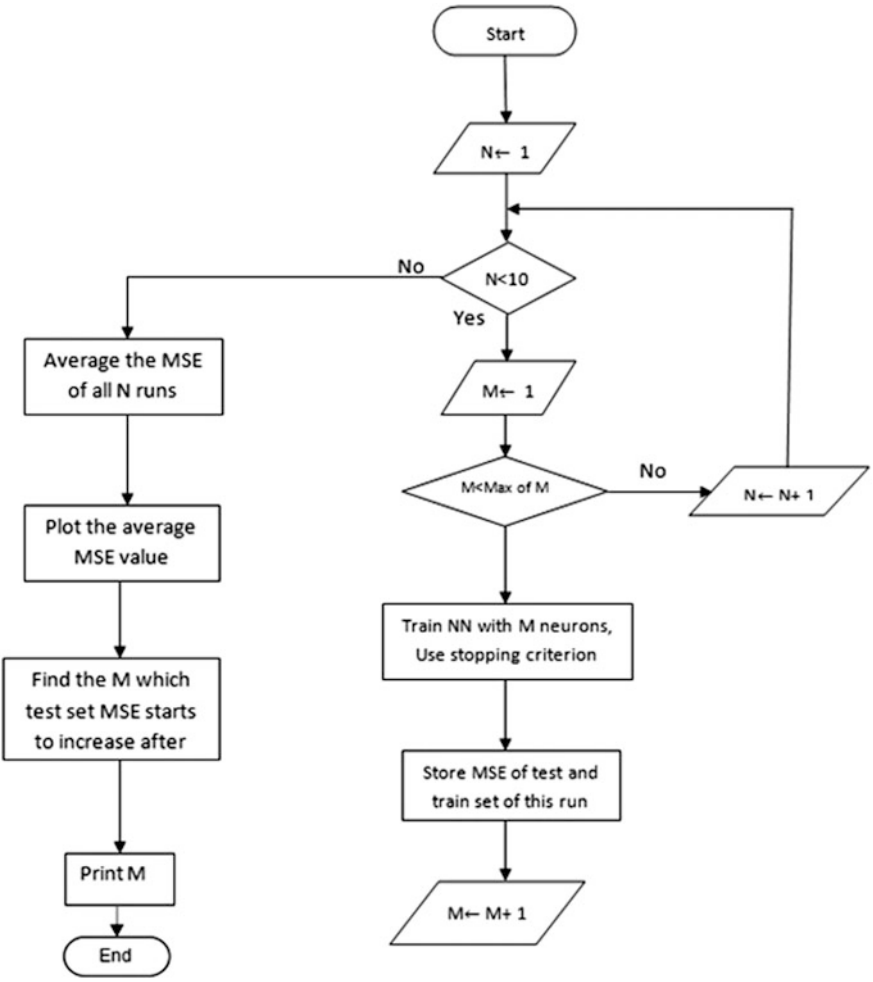


Fig. 2.25 Flowchart of multi-start approach to find the optimum number of neurons in the hidden layer (Hajian et al. 2012)

of inputs is the same as the number of measured gravity points along the selected gravity profile. To normalize the real gravity data the equation below was used:

$$g^{i}_{normalized} = \frac{g^{i} - g_{\min}}{g_{\max} - g_{\min}} \tag{2.9}$$

It is obvious that the outputs of the FFBP neural network for this work are Z (depth to cavity) and q (shape factor).

The mathematical equation for the perceptrons is given below:

$$\begin{aligned} q &= \tilde{f}\left\{\sum_{j=1}^{m} w_{ji} f\left[\sum_{i=1}^{n}\left(w_{ji}\, g_i + b_1\right) + b_2\right]\right\} \\ Z &= \tilde{f}\left\{\sum_{j=1}^{m} w_{jz} f\left[\sum_{i=1}^{n}\left(w_{ji}\, g_i + b_1\right) + b_2\right]\right\} \end{aligned} \tag{2.10}$$

where b_1 is the basis of the first layer, b_2 is that of the second layer, $f(0)$ is the activation function between input and hidden layers and $\tilde{f}(0)$ is the activation function between hidden and output layers. The FFBP that Hajian et al. (2012) used is illustrated in Fig. 2.26.

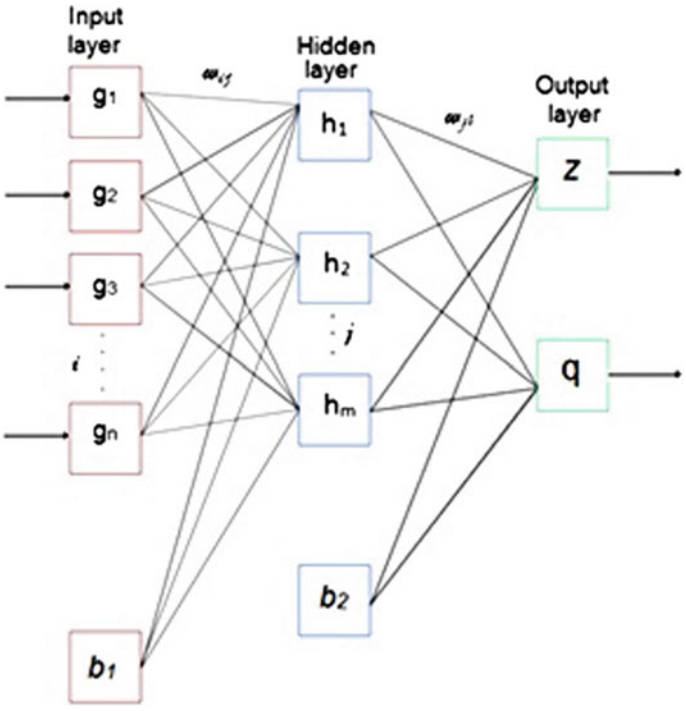


Fig. 2.26 Schematic of FFBP used for depth and shape factor estimation (Hajian et al. 2012)

To prepare the training set Hajian et al. (2012) postulated that the cavity's shape is one of the simple models of sphere, vertical cylinder or horizontal cylinder because most cavities, to a first approximation can be considered to lie close to these shapes and most natural cavities can be simulated with these models however, most sedimentary basins approximate quite well to slabs or wedges.

The gravity effect of the proposed models was calculated using the analytical equations presented by Abdelrahman et al. (2001) as follows:

$$g(n_i) = \frac{A}{(X_i^2 + Z^2)^q} \tag{2.11}$$

where Z is the depth to the center of the cavity, x is the horizontal distance, A is the amplitude factor and q is the shape factor. The amplitude factor depends on both the size and the density contrast of the object while q depends only on the shape factor of the object (Table 2.9). If we normalize the gravity values by dividing each by the maximum residual gravity amplitude (which occurs at $x = 0$) then the normalized residual gravity values will only depend on the depth and shape factor of the object (Hajian et al. 2012).

The normalized gravity is as follows (Eq. 2.12):

$$g_n(x) = \frac{g(x)}{g(0)} = \frac{g(x)}{g(x)|x = 0} = \left(\frac{Z^2}{x^2 + Z^2}\right)^q \tag{2.12}$$

Hajian et al. (2012) generated 410 pairs of sample data points for training purposes as $\{Z_n, q_n\{g_{n1}, g_{n2} \ldots, g_{nk}\}\}$, 106 data pairs as the test/validation set and k is the total number of measured gravity points on the profile.

Z_n and q_n are the normalized depth and shape factor, respectively. This scaling is derived via the equations:

$$\begin{aligned} q_n &= q_{normalized} = \frac{q - q_{\min}}{q_{\max} - q_{\min}} \\ Z_n &= Z_{normalized} = \frac{Z - Z_{\min}}{Z_{\max} - Z_{\min}} \end{aligned} \tag{2.13}$$

Table 2.9 The values of amplitude factor (A) and shape factor (q) in relation 4, for different shapes of objects (where R is the radius of the object as shown in Fig. 2.2, G is the universal gravity constant and ρ is the density contrast) (Hajian et al. 2012)

Shape	q (Shape factor)	A (Amplitude factor)
Sphere	1.5	$\frac{4}{3}\pi G\rho R^3$
Horizontal cylinder	1	$2\pi G\rho R^3$
Vertical cylinder	0.5	$\pi G\rho R^3$

Consequently the FFBP neural network outputs are the normalized shape factor and the normalized depth and should be converted back to their real values using the following equations:

$$\begin{aligned} q &= q_n = (q_{\max} - q_{\min}) + q_{\min} \\ Z &= Z_n = (Z_{\max} - Z_{\min}) + Z_{\min} \end{aligned} \tag{2.14}$$

Figure 2.27 shows the flowchart for generating N pairs of training sets for the FFBP neural system.

Hajian et al. (2012) used a cross-validation algorithm through the K-fold cross-validation approach. Cross validation is a technique for evaluating generalization error through a process of "resampling" (Weiss and Kulikowski 1991; Efron and Tibshirahi 1993). The resulting estimates of error are often used for making a choice between models using different neural network architectures. In the K-overlap cross validation method the information is separated into K subsets of about the same size. The system is trained K times, and each time one of the subsets is omitted to compute the relevant error. The process of K-fold cross validation is depicted in Fig. 2.28.

The FFBP neural network parameters are shown in Table 2.10. The result of testing the designed FFBP with different values for number of hidden neurons (N) is shown in Fig. 2.29 showing that the optimum number for N is 22.

As the range of probable cavities was within the range from 0.5 to 30 m, testing the designed FFBP neural network was performed for the synthetic data in this training space and, the RMS error is shown in Table 2.11.

Furthermore the network was tested for its adaption to robustness for realistic cases by adding Gaussian noise and simulating a few levels of uncertainty as 5 and 10%. of noise were added, The results showed that the FFBP model is significantly more precise and robust for noisy synthetic data as compared to nonlinear least square minimization.

As a next stage, to evaluate the efficiency of the designed FFBP network in practical situations Hajian et al. (2012) investigated two real cases: the field microgravity data measured over a major container terminal at Freeport Grand Bahama (Fig. 2.30) and the gravity for a long profile over the Medford cavity site located in Florida, USA. The gravity anomaly guide of the Bahama site shown in Fig. 2.31.

For the Bahamas case study, five principal profiles were selected (I–V) and the depth and shapes were estimated and compared with the results of NLLSM (Table 2.12). The results of the FFBP technique agreed well with the drilling data, despite the differences between the basic model and the real cavity shapes. While both methods give acceptable agreement with the borehole data the FFBP results are closer to the observed values.

This study showed that the FBPP technique is a robust estimator even in the presence of significant noise and that it can give reliable estimates of both cavity shape and depth using microgravity measurements.

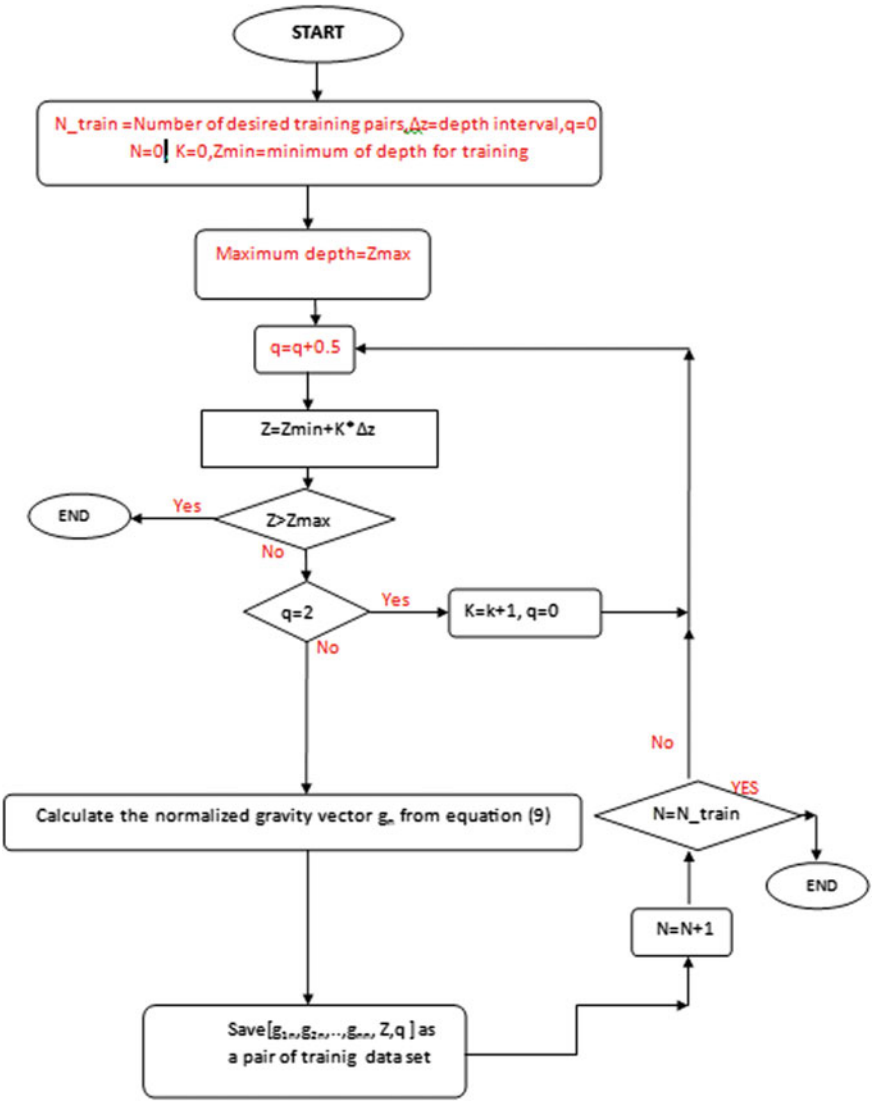


Fig. 2.27 Flowchart of preparing N pairs of training sets for FFBP neural network (Hajian et al. 2012)

Hajian et al. (2012) also tested the ANN method for the residual gravity profile for a cavity anomaly located at the Medford site, Florida, USA (Butler 1984) which is shown in Fig. 2.32. Both the Bahamas and Florida sites are cavities within karstic limestone geological structures and in both sites the selected principal profiles have 15 points of residual data as inputs of the FFBP neural network (the same number of inputs for the neural network) and also the domain of cavity depth for the second

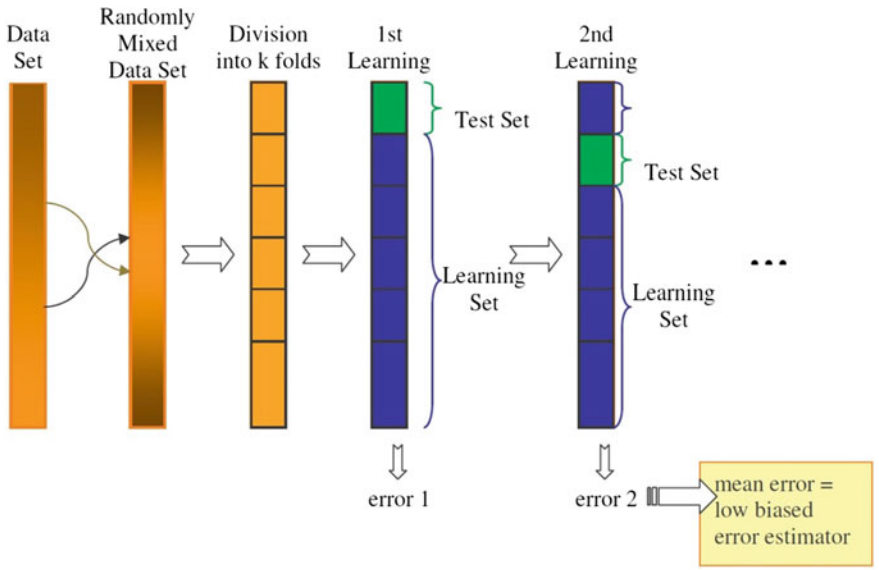


Fig. 2.28 The process of cross-validation for neural networks learning (Hajian et al. 2012)

Table 2.10 FFBP parameters designed for estimation of the shape factor and the depth of cavities Hajian et al. (2012)

Parameter	Value
Number of inputs	Equal to number of selected gravity points along the profile (in this study 15)
Number of outputs	2
Number of hidden neurons (M)	Calculated adaptively (see Fig. 4)
Number of runs of multi-start (N)	10
Transfer function of hidden layer neurons	Sigmoid
Transfer function of output layer neurons	Sigmoid
Train algorithm	Levenberg-Marquardt
Maximum number of hidden neurons	42
Maximum number of epochs	2000

site is a subset of the Bahama site cavity depth domain. Furthermore, from available geological information it was known that the shapes of available cavities in both sites are close to either a sphere, or a vertical or horizontal cylinder. But the strength of the designed FFBP neural network method is that it can estimate the shape and depth of whichever best approximates the cavity.

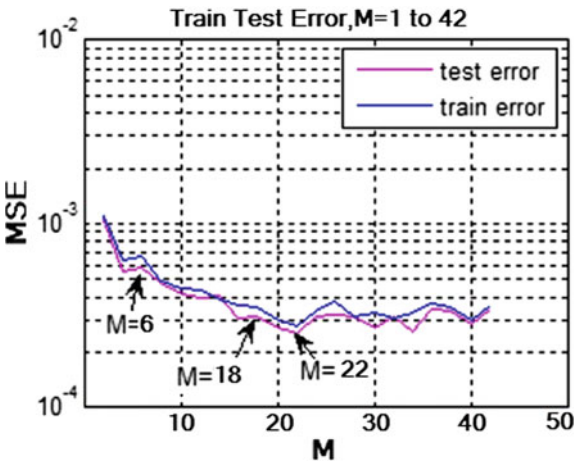


Fig. 2.29 This plot shows the changes of test/train MSE versus epochs used to select the best M values (number of hidden neurons) (Hajian et al. 2012)

Table 2.11 Root Mean Square Error (RMSE) values of FFBP network and Non-Linear Least Square Minimization (NLLSM) method for synthetic data with different levels of Noise to Signal (N/S) The depth values are in meters (Hajian et al. 2012)

	Root Mean Square Error (RMSE)						
Model	N/S (%)	Sphere		Vertical cylinder			
Horizontal-cylinder							
Parameters	—	q	Z	q	Z	q	Z
FFBP	5	0.031	0.13	0.045	0.20	0.034	0.15
NLLSM	5	0.036	0.18	0.050	0.32	0.037	0.19
FFBP	10	0.057	0.50	0.074	0.63	0.074	0.56
NLLSM	10	0.085	0.65	0.087	0.84	0.085	0.81
FFBP	20	0.168	0.98	0.182	1.04	0.146	1.12
NLLSM	20	0.204	1.18	0.195	1.16	0.153	1.23

The residual anomaly data were interpolated at 1 m intervals and so the number of residual gravity points is 15 (from g(x = −7) to g(x = +7); see Fig. 2.32) exactly equals the number of inputs in the designed FFBP neural network for the Bahamas site. In this study Hajian et al. (2012) first trained the designed FFBP model for a maximum depth of 30 m for Bahamas field data and found that this also works equally well for the Medford cavity site because the depth range of cavities located under the surface of this site is less than or equal 10 m (Butler 1984) and so lies within the training range. As the set = {z|z less than or equal to 10 m} is obviously a subset of set = {z|z less than 30 m}, then the designed FFBP neural network can

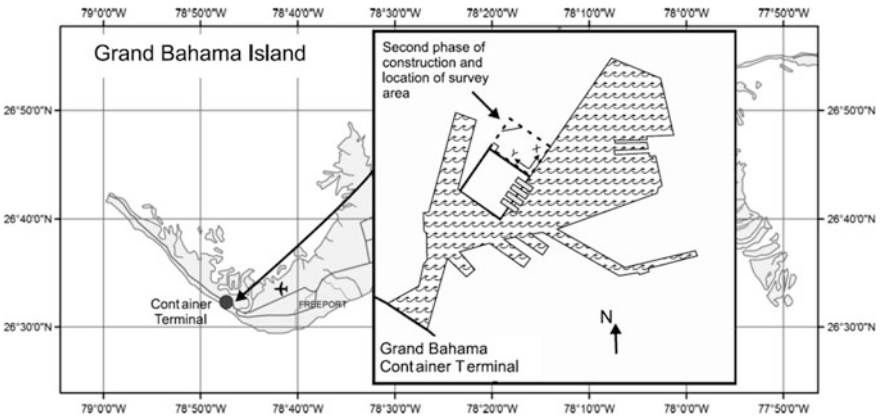


Fig. 2.30 Map of microgravity study zone at the Bahamas container terminal, Freeport, Grand Bahama, Bahamas. Redrawn from Styles et al. (2005)

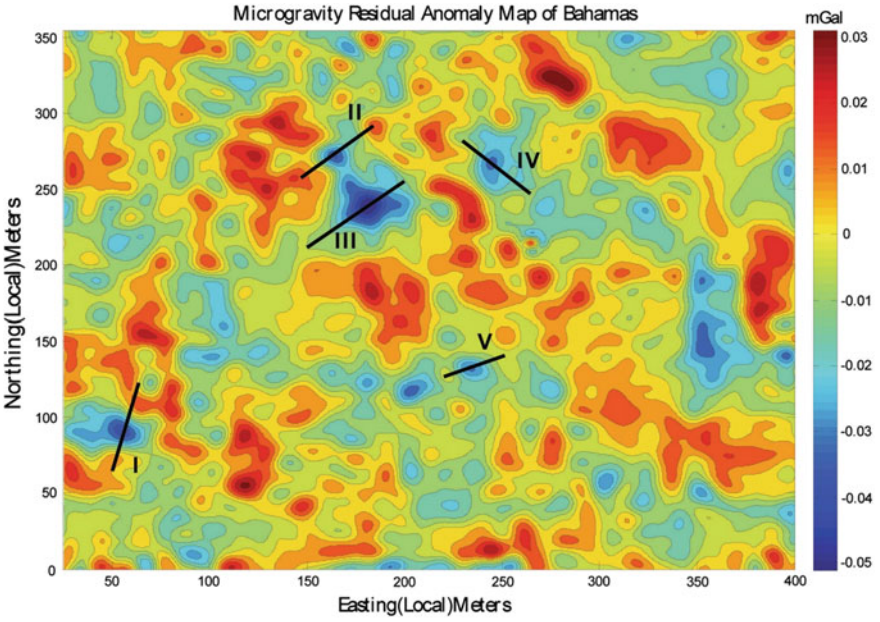


Fig. 2.31 Residual gravity anomaly map for the Freeport container terminal survey, showing the location of profiles I–V selected as test example (Hajian et al. 2012)

be used for depth and shape estimation of subsurface cavities of the Medford site or indeed, probably any site within similar geological formations without any need for a new training process.

Table 2.12 Interpreted cavity shape factor and depth calculated with the FFBP method for the selected principal profiles of Grand Bahama site, in comparison with borehole results and NLLSM method, for absence of any drillings near profile IV, FFBP and NLLSM results are compared with Euler method result (Styles et al. 2005; Hajian et al. 2012)

Selected principal profile	Borehole results		Results of FFBP		Results of NL LSM	
	Shape (Near to)	Depth(m)	q	Depth(m)	q	Depth(m)
Profile I	H.Cylinder	4.26	0.87	4.55	0.70	4.92
Profile II	Sphere	7.00	1.45	6.72	1.34	6.53
Profile III	Sphere	15.24	1.61	15.43	1.30	15.75
Profile IV	H.Cylinder	14.50	0.93	14.20	0.82	15.06
Profile V	H.Cylinder	14.50	0.95	14.36	0.84	15.10

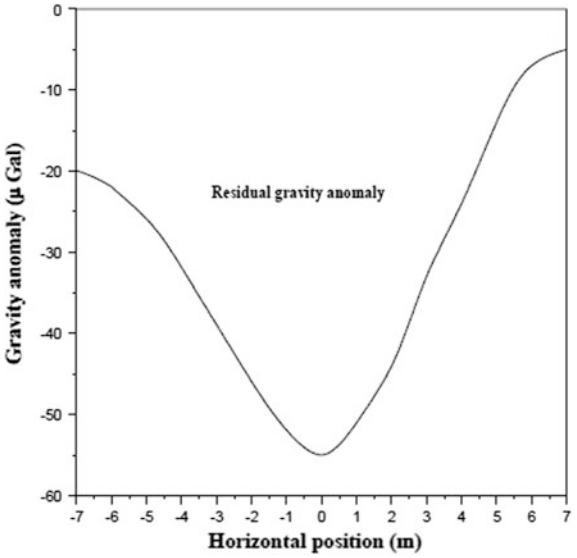


Fig. 2.32 Residual gravity profile over a cavity at the Medford test site (after Butler 1984)

The outputs achieved by the FFBP method were q = 0.95 and Z = 3.86 (m). The shape factor obtained suggests that the cavity could reasonably be approximated by a horizontal cylinder model (see Table 2.12). Additionally, the result for depth demonstrates that there is an acceptable correspondence between the depth obtained from this technique (3.86 m) and that obtained from drilling (3.57 m) (Hajian et al. 2012).

2.2.4 *Modeling Anticlinal Structures Through Neural Networks Using Residual Gravity Data*

Eshaghzadeh and Kalantari (2015) used feed forward neural networks for modeling anticlinal structures using the residual gravity anomaly. Anticlines are one of the most common targets for hydrocarbon exploration. Generally inversion of gravity anomalies is non-unique, in the sense that the observed gravity anomalies may be generated by a variety of density distributions and it would be helpful to be able to reduce this level of uncertainty by homing in on preferred geometries and density contrasts. Eshaghzadeh and Kalantari (2015) proposed a simple geometrical model composed of two adjoining right-angled triangles as a simple geometrical analog of anticlinal structures. The geometry of the anticline structure they used for preparing the training data is shown in Fig. 2.33. Where d_1 is the depth to top and d_2 is the depth to bottom of the model, i and j are angles of the fold limbs of the anticlinal structures.

Modeled gravity effects were calculated for different values of d_1, d_2, $\Delta\rho$, i and j, and then some suitable "features" were extracted from the gravity signal. An important challenge in training the neural network with gravity is that if all the data is applied as inputs to the network, this will require many inner connections (weights) to be tuned which is very time is consuming for the training process.

To overcome this problem, some features are first selected from the gravity signal. These features should be independent and must possess a relationship or

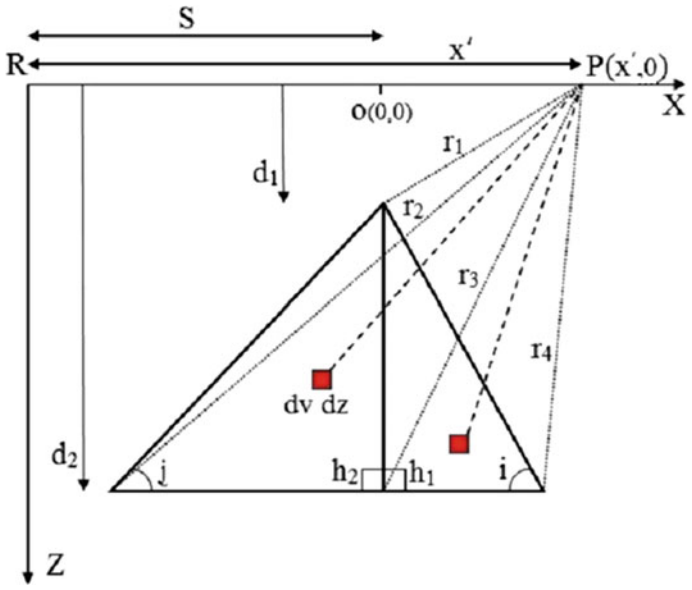


Fig. 2.33 Geometrical model of anticlinal structure (Eshaghzadeh and Kalantari 2015)

correlation with the geometrical parameters of the anticline. The parameters that were used in this way are

P_1: the maximum gravity amplitude
P_2: The 75% level of the maximum gravity amplitude
P_3: The 50% level of the maximum gravity amplitude
P_4: The 25% level of the maximum gravity amplitude
P_5: width of gravity curve between the coordinates with 75% of g_{max}
P_6: width of gravity curve between the coordinates with 50% of g_{max}
P_7: width of gravity curve between the coordinates with 25% of g_{max}

These parameters are shown in Fig. 2.34 and were used as inputs to the neural network and the outputs are d_1, d_2, $\Delta\rho$, i and j (see Fig. 2.35).

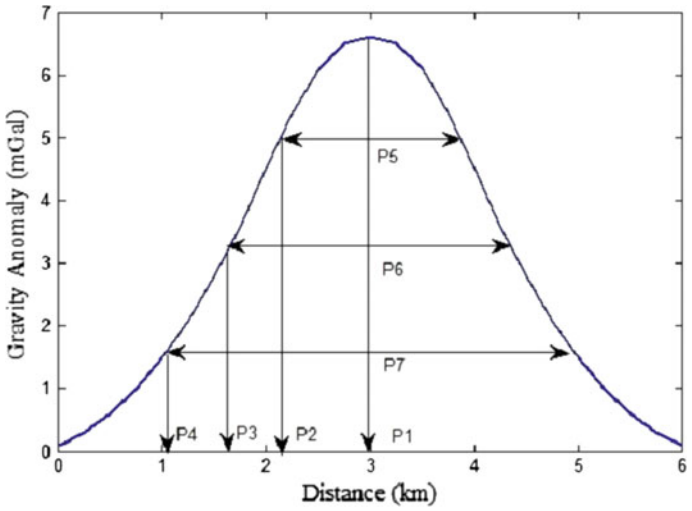


Fig. 2.34 Definition of input parameters (features) for an anticlinal structure gravity anomaly (Eshaghzadeh and Kalantari 2015)

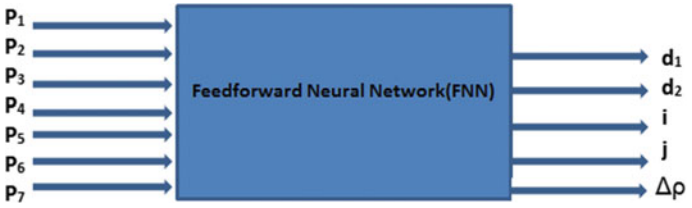


Fig. 2.35 Neural input/output architecture for anticlinal structure estimation

The gravity anomaly of a 2D anticline structure at any point P (n', o) on the X-axis can be calculated from the fundamental equation for a gravity anomaly due to a 2D source with cross-sectional area S (Rao and Murthy 1978):

$$G(x) = 2g \int_{s} \frac{\Delta\rho . Z dv dz}{r^2} \tag{2.15}$$

where 'G'; is the universal gravitational constant, 'dv dz' is the cross-sectional area of a line, Δρ is the density contrast and r is the radial distance from the element dvdz to the observed point P(x', o).

Substituting limits for the gravity anomalies due to each right triangle, this can be expressed as:

$$g(x) = 2G\Delta\rho \int_{z=d_1}^{d_2} \int_{v=0}^{(d_2-d_1)cos\varphi} \frac{z d_{sd_z}}{z^2 + (x' - s)^2} \tag{2.16}$$

where φ = i and j are the right-hand triangle and left-hand triangles respectively and x = x' − s, s is the distance between the origin of the model and the reference point (R) (see Fig. 2.33).

They produced a data set of 12,800 points to train and test the neural network. 8960 points from this set (about 70% of the total data) were randomly selected as training data and the remaining 30% of the data was used as test data.

The network which was designed was tested on both synthetic and real data. Before testing the network for real data they evaluated the validity of the method for both noise-free and noise-corrupted synthetic models and obtained satisfactory results (see Table 2.13).

Table 2.13 The initial parameters of the scalene triangle synthetic model and estimated parameters using FNN inversion (Eshaghzadeh 2011)

	Parameters					
	ZI (km)	Z2 (km)	I (deg)	J (deg)	Δρ (kg/m^3)	RMSE
Initial values	3	5	20	30	150	
Free-noise gravity data	2.98	5.01	20.13	30	148.7	0.152
Error %	0.7	0.3	0.65	0	0.87	
With 5% noise	3.05	4.92	19.36	31.62	156.4	0.423
Error %	1.7	1.6	3.2	5.4	4.3	
With 10% noise	3.09	5.12	18.57	32.27	158.1	0.437
Error %	3	2.4	7.15	7.6	5.4	

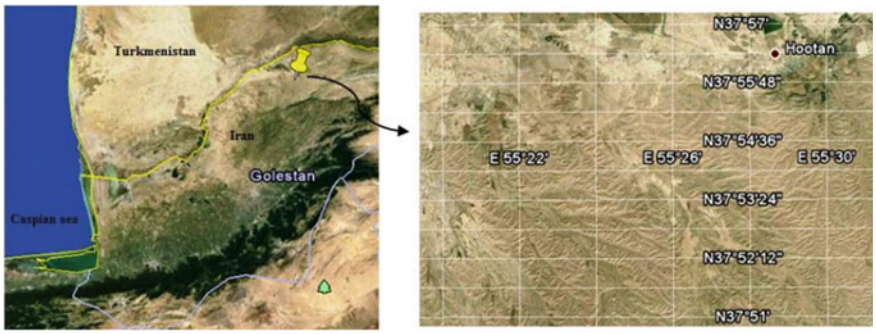


Fig. 2.36 The satellite image from the area under consideration location, namely Korand (yellow cursor in left-hand side picture). The rightmost side picture shows the morphology of Korand (Eshaghzadeh 2011)

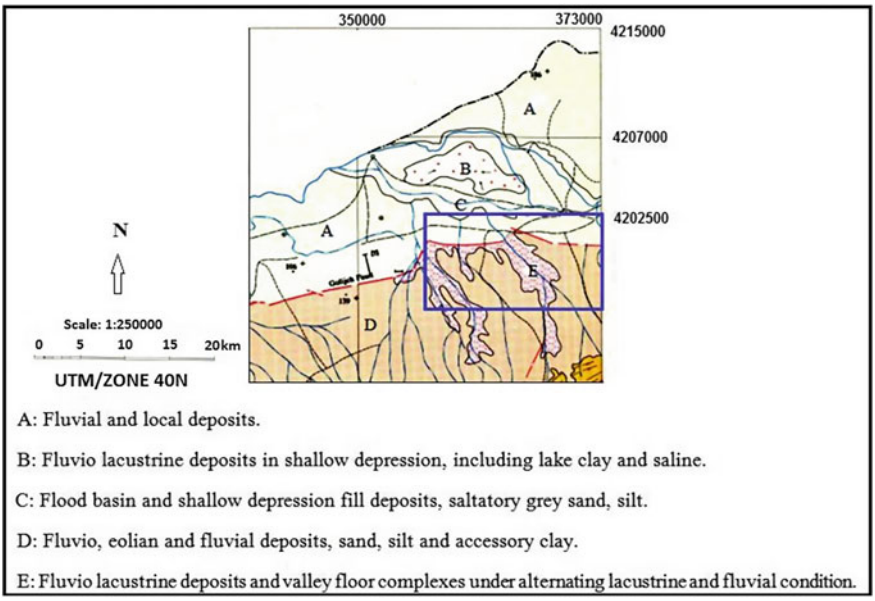


Fig. 2.37 The geological map of Korand; the blue rectangular shows the study area limits (Eshaghzadeh 2011)

The case study using real data was from the gravity field in the northeast of Iran, analyzed based on 2D modeling of the gravity anomalies of selected principal profiles. The Korand region of the study is located in Golestan province, Northeastern Iran. The yellow marker pin in Fig. 2.36 shows the situation of Korand region. The geological map of Korand is shown in Fig. 2.37.

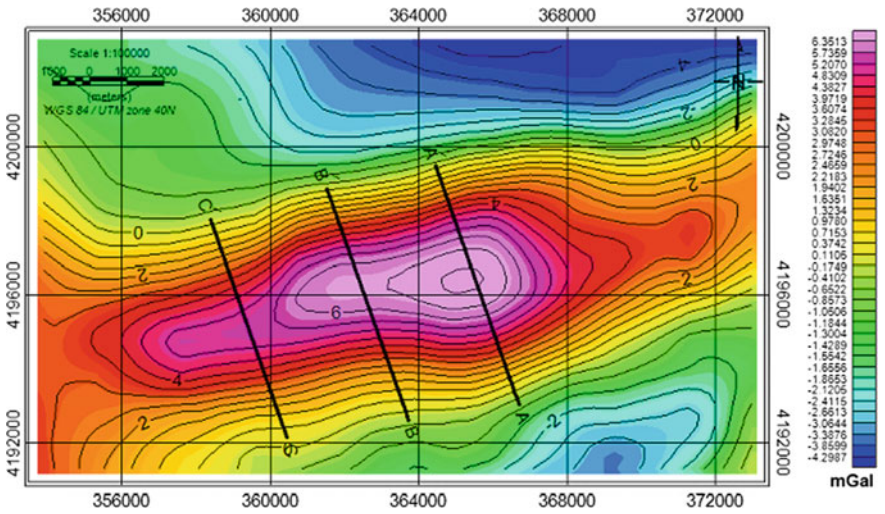


Fig. 2.38 The residual gravity anomaly map showing the profiles A–A′, B–B′ and C–C′ (Eshaghzadeh 2011)

Table 2.14 Evaluated parameters using the FNN inversion for the profiles A–A′, B–B′ and C–C′ (Eshaghzadeh 2011)

Profile	Parameters					
	Z1 (km)	Z2 (km)	I (deg.)	J (deg.)	Δρ (kg/m^3)	RMSE
A−A′	3.43	5.16	24.93	22.36	372.8	0.741
B−B′	3.75	5.3	18.04	31	383.4	0.264
C−C′	3.86	5.37	23.85	27.2	391.7	0.502

The residual gravity map of the region is shown in Fig. 2.38 where three principal profiles namely A–A′, B–B′ and C–C′ were selected to attempt to estimate the depth to the top (Z_1), the depth to bottom (Z_2) and angles I, J. The parameters obtained using this neural network and the error estimates are tabulated in Table 2.14.

The Root Mean Square Error (RMSE) was calculated for each of the interpreted values and show that the inversion results are acceptable.

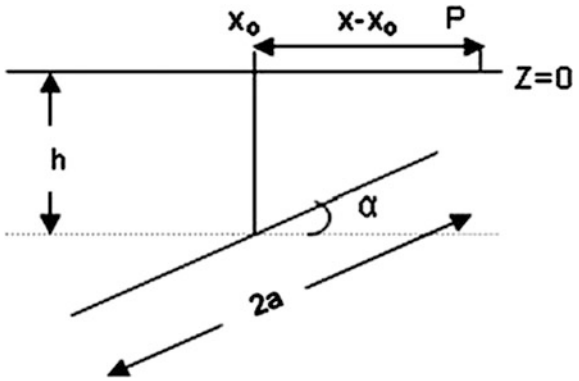


Fig. 2.39 Inclined sheet geometry of infinite horizontal extent (El-Kaliouby and Al-Garni 2009)

2.3 Application of ANN for Inversion of Self-potential Anomalies

El-Kaliouby and Al-Grani (2009) utilized modular neural networks (MNN) to invert self-potential (SP) data from sloping infinite slabs. The input of the MNN was the SP data along a principal profile and the output is:

Depth to the center of the sheet (h), the half-width of the sheet (a), the angle of inclination (α) zero distance from origin (x_0) and the polarization amplitude k (Fig. 2.39). N.B. A principal profile lies perpendicular to the strike of the sheet.

The training data is prepared by forward modeling of the SP potential effect on the surface profile perpendicular to the strike of an 2*D* inclined sheet of infinite horizontal extent which is given by (Murthy and Haricharan 1985; Sundarajan et al. 1998):

$$V(x) = K\,In\left\{\frac{[(x-x_0)-a\,\cos\alpha]^2+[h-a\,\sin\alpha]^2}{[(x-x_0)-a\,\cos\alpha]^2+[h-a\,\sin\alpha]^2}\right\} \tag{2.17}$$

where K the polarization amplitude is $\left(K=\frac{IP}{2\pi}\right)$, P is the resistivity of the medium and I is the current density (current per unit area of the medium).

To evaluate the validity of MNN for inverse modeling of SP for a 2*D* sheet El-Kaliouby and Al-Garni (2009) first tested it on a synthetic example and then applied it to two field examples from the Surda area of Rakha mines, India and the Kalava fault zone, India.

The synthetic example data used samples of 81 points of input data over a 160 m profile with a 2 m interval (Fig. 2.40). They used MNN to invert the SP data using 81 points for the input with 81 nodes in the hidden layer.

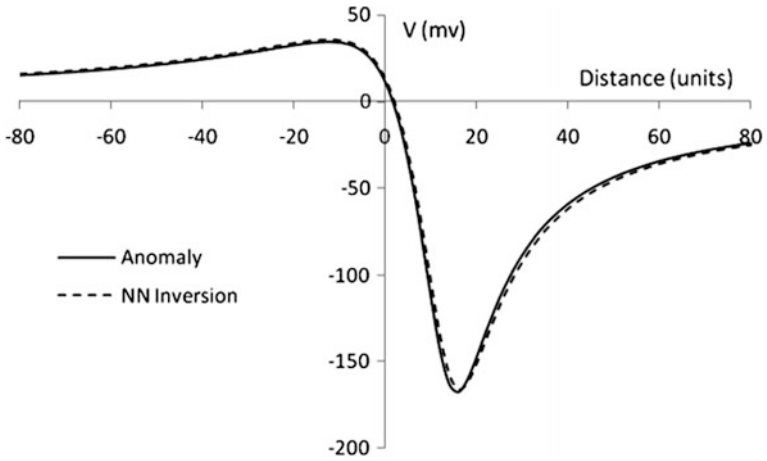


Fig. 2.40 Synthetic SP anomaly profile over a sheet-like body and its NN inversion response. The sheet parameters are h = 10 m, a = 5 m, $\alpha = 40°$, x_o = 10 m and k = 100 mV (El-Kaliouby and Al-Garni 2009)

Table 2.15 Parameter range and number of points used for MNN training

Parameter	Range	Number of points in the range
Depth	$5 \leq h \leq 15$ m	5
Half-width	$2 \leq a \leq 8$ m	5
Inclination	$20° \leq \alpha \leq 60°$	7
Polarization amplitude	$7 \leq K \leq 130$ mV	7
Origin location	$-15 \leq x_0 \leq 30$ m	5

In the MNN that Al-Garni (2009) designed, five local experts with seven processing elements were used and the selected activation function was a hyperbolic tangent. They assumed that the parameters of the target for the synthetic data were:

$$h = 10 \text{ m}, a = 5 \text{ m}, \alpha = 40°, x_0 = 10 \text{ m and } k = 100 \text{ mV}.$$

giving the response shown in Fig. 2.40.

To cover the range of the parameter they used 6152 training models. The parameters they used for training the MNN are listed in Table 2.15.

The parameter range is mainly dependent on the measured field data (voltage) response, for example the zero distance from the origin range is selected centered around the anomaly minimum (El-Kaliouby and Al-Garni 2009). Selecting a suitable range for the parameters used for training is important and so initially a coarse range is selected with a small number of points for each parameter. In the event that any of the parameters falls outside the chosen range the NN reports this fact and the range is extended. Additionally, the learning error of each parameter is computed

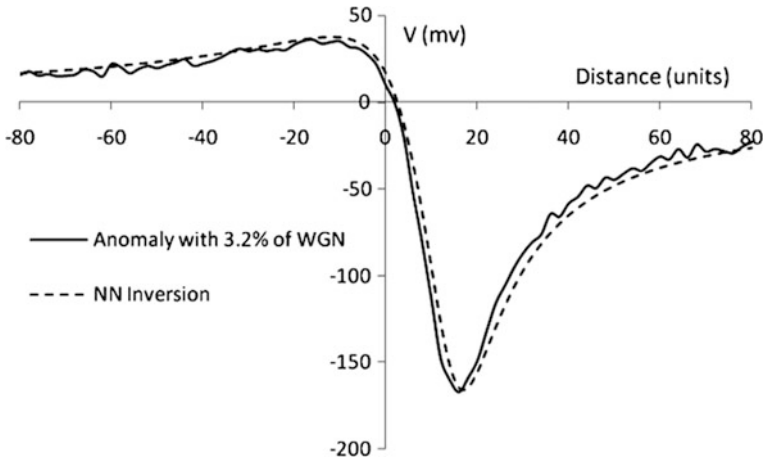


Fig. 2.41 Synthetic SP anomaly profile over a sheet-like body with 3.2% of random WGN and its NN inversion response (El-Kaliouby and Al-Garni 2009)

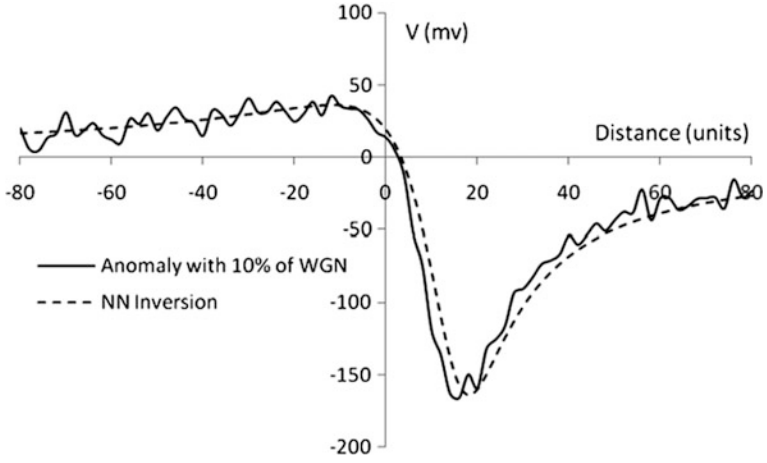


Fig. 2.42 Synthetic SP anomaly profile over a sheet-like body with 10% of random WGN and its NN inversion response (El-Kaliouby and Al-Garni 2009)

and then these are combined into an overall RMS error and if this is within acceptable limits the misfit between model and data is calculated but if not, then the range of parameters is automatically refined until and acceptable error is obtained. Once the MNN is well trained it will be able to invert any field information that fall inside the range of training (El-Kaliouby and Al-Garni 2009).

The results for the synthetic data version are illustrated in Fig. 2.38. In the next stage the designed neural network is tested on noisy synthetic data. I El-Kaliouby

Table 2.16 Example of synthetic data parameters (El-Kaliouby and Al-Garni 2009)

Parameters	h(m)	a(m)	α(degrees)	k(mV)	x_o(m)
Assumed values	10	5	40	100	10
NN inversion values without noise	10.23	5.42	38.65	96.67	10.70
NN inversion with 3.2% of WGN	10.32	5.54	37.89	96.35	10.71
NN inversion with 10% of WGN	10.85	5.49	38.66	97.3	11.93

Table 2.17 Parameters used to train the MNN for Rakha mines, Singbhum copper belt, Bihar, India (El-Kaliouby and Al-Garni 2009)

Parameter	Range	Number of points in the range
Depth	$10 \leq h \leq 40$ m	7
Half-width	$10 \leq a \leq 30$ m	7
Inclination	$20° \leq \alpha \leq 50°$	7
Polarization	$90 \leq K \leq 180$ mV	7
Amplitude	$-20 \leq x_0 \leq 40$ m	7

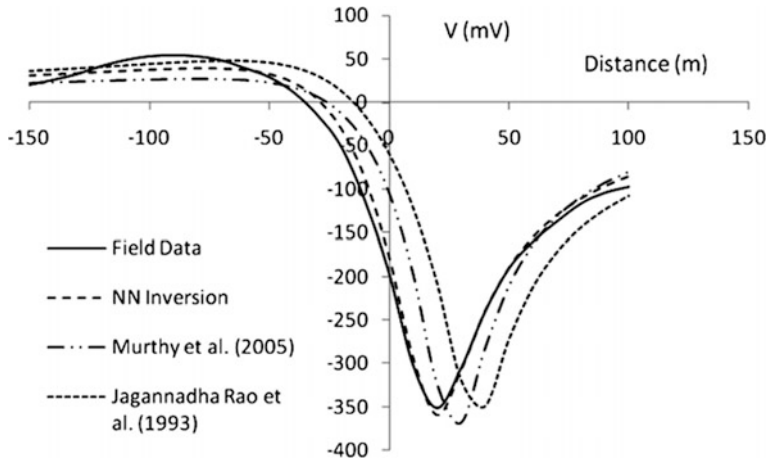


Fig. 2.43 Self-potential anomaly profile over a sulfide mineralization zone in the Surda area of Rakha mines (after Murthy and Haricharan 1984) and its neural network inversion compared with Murthy et al. (2005) and Jagannadha et al. (1993) (El-Kaliouby and Al-Garni 2009)

and Al-Garni (2009) tested the MNN by adding 3.2% (3 dB) and 10% (20 dB) of white Gaussian noise (WGN) to the SP anomaly (Figs. 2.41 and 2.42).

Adding White Gaussian Noise is a simple procedure in MATLAB using the command 'awgn', which adds white Gaussian noise to a signal.

Table 2.18 Comparison of NN interpreted SP parameters of Kalava anomaly with two other interpretations (El-Kaliouby and Al-Garni 2009)

Parameters	h(m)	a(m)	α(degrees)	k(mV)	x_o(m)
Jagannadha et al. (1993)	7.59	3.75	80	–	0.4
Murthy et al. (2005)	9.38	3.96	80.76	–	−0.4
NN inversion	7.2	3.15	78.72	68.29	−0.9

Table 2.19 The parameters ranges used to train the network for the Kalava fault zone Cuddapah basin, India

Parameter	Range	Number of points in the range
Depth	$5 \leq h \leq 15$ m	10
Half-width	$1 \leq a \leq 7$ m	10
Inclination	$80° \leq \alpha \leq 120°$	5
Polarization	$50 \leq K \leq 100$ mV	5
Origin location	$-5 \leq x_0 \leq 5$ m	5

$$Y = awgn(x, SNR) adds\ white\ Gaussian\ noise\ to\ the\ vector\ signal\ x.$$

where the scalar SNR specifies the signal-to-noise ratio per sample, in dB.

For example to add 3.2% noise to SP data when the noise power is 30 dB we use the 'awgn (SP, 30)' command and to add 10% of white Gaussian noise when the noise power is 20 dB the command in MATLAB is 'awgn (SP, 20)'.

El-Kaliouby and Al-Garni (2009) observed that the MNN inversion result for up to 10% random Gaussian white noise was satisfactory (Table 2.16).

Finally, they used two field cases to determine whether the MNN model worked for real data. The first field was a very significant SP anomaly (Murthy et al. 2005) observed from the Surda zone of the Rakha mines, Singbhum copper belt, Bihar, India. For this SP anomaly they used 16807 training models to cover the ranges of the parameters. The parameters they used to train the MNN are listed in Table 2.17.

The MNN inversion response that El-Kaliouby and Al-Garni (2009) used was compared with the Murthy et al. (2005) results and in Fig. 2.43 with the inversion in Table 2.18.

The next field example that El-Kaliouby and Al-Garni (2009) used was the SP anomaly profile across a zone of mineralization in the Kalava fault zone, Cuddapah basin, India (Rao et al. 1982). The SP anomaly was sampled at 41 stations over a 40 m distance with a 1 m interval. They used 12500 training models and the parameter ranges that they used to train the network are listed in Table 2.19.

El-Kaliouby and Al-Garni (2009) compared the MNN results with the inversion of Rao et al. (1982) and with the observed data (Fig. 2.44) and the inversion parameters are shown in Table 2.20.

They concluded that the MNN inversion fitted the measured SP data better than the Rao et al. (1982) results convincingly demonstrating the successful application

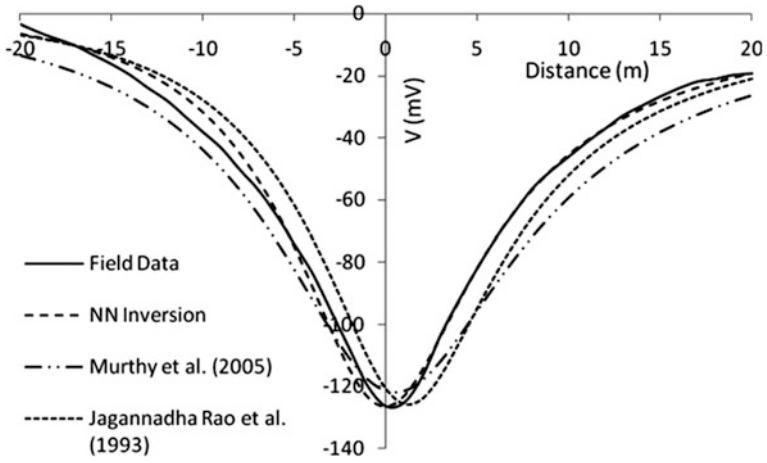


Fig. 2.44 Self-potential anomaly profile over a sulfide body in the Kalava fault zone, Cuddapah Basin, India (after Rao et al. (1982), and its neural network inversion compared with Murthy et al. (2005) and Jagannadha et al. (1993) (El-Kaliouby and Al-Garni 2009)

Table 2.20 Comparison of NN Kalava fault zone Cuddapah basin interpreted SP parameters of with five other interpretations (El-Kaliouby and Al-Garni 2009)

Parameters	h(m)	a(m)	α(degrees)	k(mV)	x_o(m)
Paul (1965)	21	40.20	20.01	–	–
Rao et al. (1970)	30.48	34.87	10.01	–	–
Jagannadha et al. (1993)	29.88	29.40	45	–	15.00
Sundararajan et al. (1998)	27.65	32.35	13.20	–	–
Murthy et al. (2005)	26.52	19.81	57.63	–	15.84
NN inversion	27.78	19.51	50.96	130.86	5.86

of MNN to synthetic data (free-noise, noisy) and field data and confirming the capability and increased accuracy of MNN models to invert SP anomaly data.

2.4 Application of ANN for Sea Level Prediction

Imani et al. (2014) used various types of neural networks, (MLP, GRNN and Radial Basis neural systems) using satellite altimetry to determine their relevance for forecasting Caspian Sea levels.

5 years of Topex/Posedion (T/P) and Jason-1 (J/1) altimetry data were used by Imani et al. (2014), covering 1993–2008, to train the neural networks along profiles which corresponded to available tide-gauge data. The results of different neural

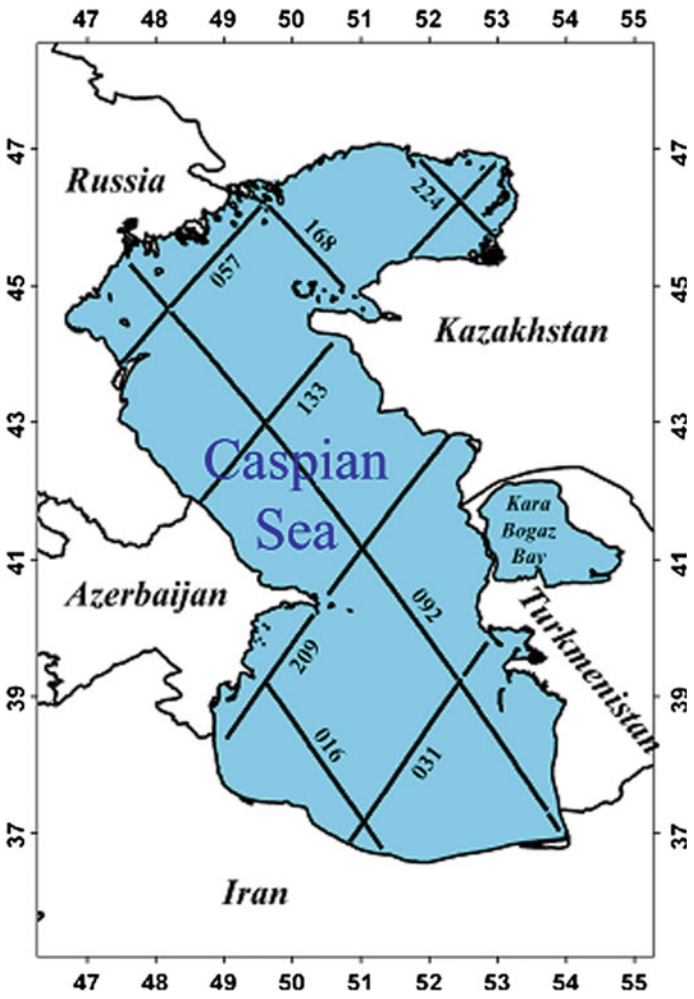


Fig. 2.45 T/P and J-1 ground tracks over the Caspian sea (Imani et al. 2014)

Table 2.21 Correlation coefficients of sea level anomalies between pass 092 and other passes (Imani et al. 2014)

Ground track	Pass 016	Pass 031	Pass 057	Pass 133	Pass 209
Correlation	0.91	0.81	0.77	0.88	0.83
P value[a]	<0.001	<0.001	<0.001	<0.001	<0.001

[a]*P* values <0.05 are considered statistically significant

networks used for sea-level prediction, were compared to the conventional autoregressive moving average (ARMA) technique results.

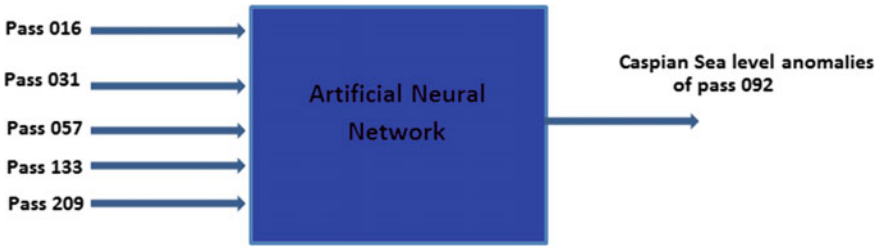


Fig. 2.46 Input/output structure of neural network to predict sea level anomalies

Table 2.22 RMSE and R for training and testing data sets using different MLP networks (Imani et al. 2014)

Model	Training				Testing	
	No. of neurons in hidden layer	No. of epoch	RMSE (m)	*R*	RMSE (m)	*R*
MLP (5, 4, 1)	4	10	0.065	0.90	0.086	0.85
MLP (5, 6, 1)	6	50	0.045	0.92	0.061	0.90
MLP (5, 8, 1)	8	36	0.084	0.86	0.107	0.80
MLP (5, 10, 1)	10	100	0.106	0.82	0.124	0.78
MLP (5, 6–3, 1)	6–3	15	0.074	0.88	0.085	0.85
MLP (5, 16–6, 1)	l2–6	60	0.057	0.91	0.063	0.90
MLP (5, 8–4, 1)	8–4	110	0.097	0.83	0.111	0.79
MLP (5, 16–8, l)	16–8	42	0.039	0.93	0.054	0.91
MLP (5, 16–8, l)	16–8	42	0.042	0.93	0.060	0.91

The satellites have repeat pass times of c 9 days and from the period April 1993 to January 2008 selected passes 092, 031, 016, 209, 133 and 057 were used for this analysis while passes 168 and 224 were rejected for atmospheric and hydrological reasons (Fig. 2.45).

To select suitable inputs for the neural network, Imani et al. (2014) calculated the correlation coefficients (P value) between pass 092 and the other passes and rejected those where the P values were less than 0.05 (Table 2.21).

In the Imani et al. (2014) study, pass 092 was selected as the output layer because it was a better indicator of Caspian Sea level anomalies possesses more data and is relatively free from strong winds and ice. As can be seen from Table 1.24, Imani et al. (2014) decide to use passes 031, 016, 209, 133, and 057 as they correlated best with pass 092 as the output layer (Fig. 2.46).

After the standard geophysical instrumental and media corrections were applied, sea level anomalies (h_{SLA}) were calculated with respect to the geoid heights from the EGM 2008 geopotential model (Pavlis et al. 2008). Imani et al. (2014) then calculated the time-averaged, along-track mean sea level from Eq. 1.59.

$$h_{SLA(t)} = \frac{l}{n}\sum_{1}^{n} h_{SLA}(t,n) \tag{2.18}$$

where t is the index of cycles and n is the number of altimetry along-track measurements.

For ANN processing Imani et al. (2014), divided the sea-level anomalies into two parts: with 1993–2004 as the training data set and 2005–2008 as the test data set. For the MLP architecture optimization they used both MLP with one and two hidden layers and with different number of neurons (Table 2.22).

MLP (x, y, z) means a 3 layer MPL with x neurons in the output layer. y neurons in the hidden layer and z neurons in the output layer. As shown in Table 2.25, the RMSE is reduced for two hidden-layers compared to one hidden layer. Finally they proved that the MLP (5, 16–8, 1) structure with two hidden layers having 5 neurons

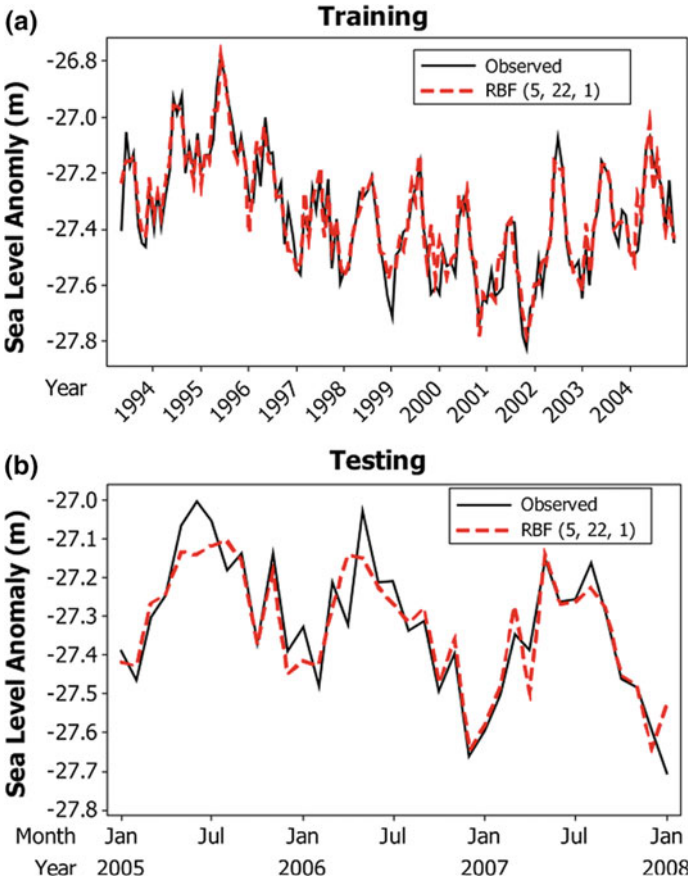


Fig. 2.47 Observed and estimated sea level anomalies derived by the optimal RBF model during **a** the training and **b** the testing periods (Imani et al. 2014)

Table 2.23 RMSE and R for training and testing data sets using different RBF and GRNN networks (Imani et al. 2014)

Model	Training				Testing	
	No. of neurons in hidden layer	Speed parameter	RMSE (m)	*R*	RMSE (m)	*R*
RBF (5, 11, 1)	11	0.50	0.087	0.85	0.095	0.82
RB F (5, 8, l)	8	0.55	0.055	0.91	0.063	0.90
RB F (5, 22, 1)	22	0.45	0.030	0.95	0.042	0.92
RB F (5, 33, 1)	33	0.65	0.065	0.90	0.083	0.85
GRNN (5, 0.05, 1)		0.05	0.087	0.85	0.102	0.80
GRNN (5, 0.5, 1)		0.50	0.069	0.89	0.091	0.83
GRNN (5, 0.3, 1)		0.30	0.047	0.92	0.059	0.90
GRNN (5, 0.1, 1)		0.10	0.098	0.83	0.127	0.78

and with 42 epochs has the least RMSE and the best correlation and this was selected as the appropriate model.

As mentioned previously, even though the same network design was used, different outputs (forecast values) are obtained after each simulation because different initial random weights are applied at the commencement of each training run. Imani et al. (2014), conducted simulations several times with the same network structure until the best performance was obtained in order to avoid this problem. As mentioned previously MLP (5, 16–8, 1) gave the optimal results of 0.039 m with a R of 0.93 for the training period and an RMSE of 0.054 m and R of 0.91 for the testing interval indicating that the MLP model designed by Imani et al. (2014) works very well for sea-level forecasting.

Imani et al. (2014) also examined different numbers of hidden layer neurons and spread constants for the RBF and GRNN network models to find the optional structures of RFB and GRNN. They found that the optional RFB model has a minimum RMSE of 0.042 and R of 0.92 for the testing period with a spread constant of 0.45 and 22 neurons in the hidden layer and that the estimations obtained from the RFB (5, 22, 1) network agree well with the observed sea level data (Fig. 2.47).

Imani et al. (2014) also examined GRNN models with different structures and found the GRNN (5, 0.3, 1) design with a parameter spread of 0.3 performed best giving an RMSE of 0.059 m with R of 0.090 during the testing time frame shown in Table 2.23 and Fig. 2.48.

Imani et al. (2014) compared the results of ANNs with the results of best ARMA conventional methods collected among different types of ARMA models (Table 2.24) and showed that artificial intelligence neural network approaches can reduce the RMSE error by c 50% as compared to presently used ARMA methods with R also improving by 15% and that the RFB method was the best Neural Network model saving significant computation time with similar performance.

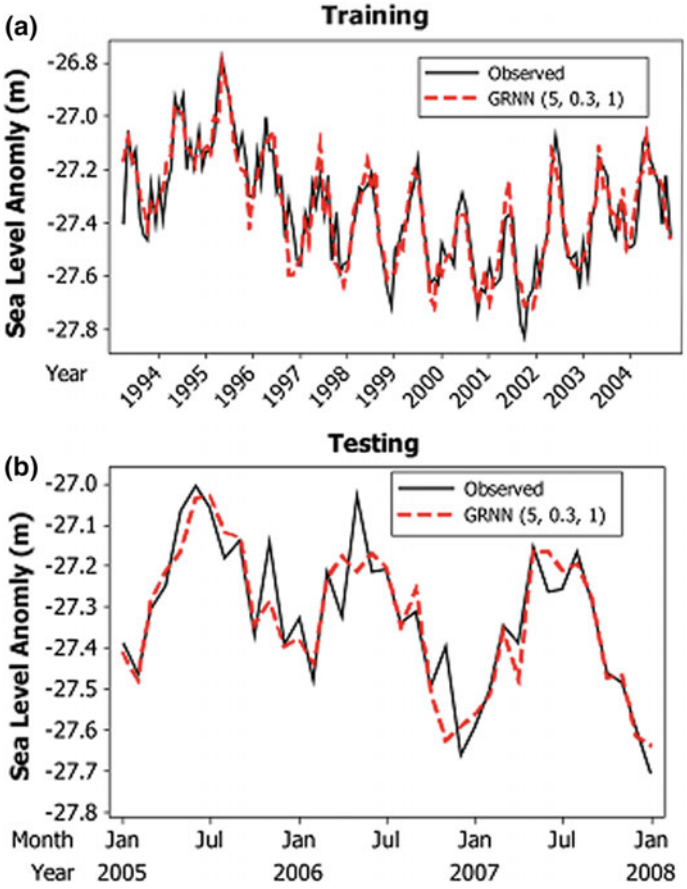


Fig. 2.48 Observed and estimated sea level anomalies derived by the optimal GRNN model during the **a** training and **b** testing periods (Imani et al. 2014)

Table 2.24 Testing statistics of estimates using ARMA models (Imani et al. 2014)

Type of model	Statistics of ARMA models	
	RMSE (m)	*R*
AR (1)	0.138	0.76
AR (2)	0.137	0.76
AR (3)	0.126	0.78
ARMA (1, 1)	0.127	0.78
ARMA (1, 2)	0.123	0.78
ARMA (2, 1)	0.125	0.78
ARMA (3, 1)	0.121	0.78
ARMA (1, 3)	0.125	0.78
ARMA (3, 2)	0.121	0.78
ARMA (2, 3)	0.121	0.78
ARMA (3, 3)	0.119	0.79
ARMA (2, 2)	0.121	0.78

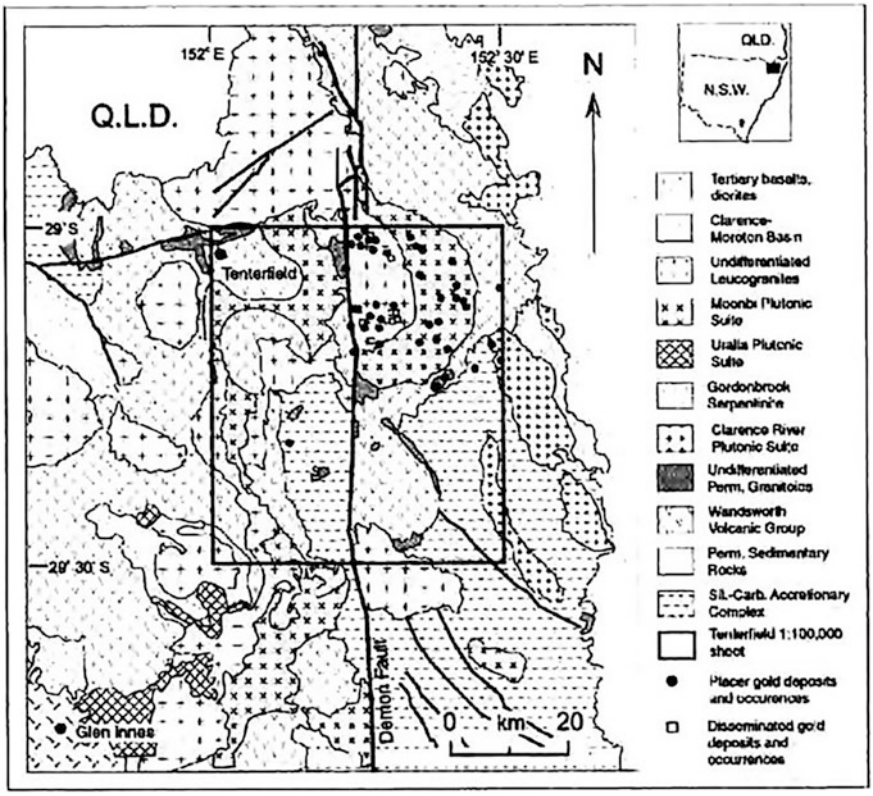


Fig. 2.49 Location and simplified geology of the Tenterfield 1:100,000 sheet area. Adapted from an unpublished regional digital map supplied by the Geological Survey of New South Wales (Brown et al. 2003)

2.5 Application of Neural Network for Mineral Prospectivity Mapping

Brown et al. (2003) used a multilayer perceptron neural system to combine exploration and GIS data in order to estimate mineral prospectivity for Gold in the Tenterfield region of NSW, Australia (Fig. 2.49).

The steps that Brown et al. used for neural network modeling are depicted in Fig. 2.50.

The 41 individual rock units were reduced to a much simpler set as many contained no or very few gold deposits. Values of magnetic and radiometric grey-scale images were subdivided into eight classes.

The fault network was digitized with cell values indicating distance to the nearest fault and assigned as an input to the neural network (Fig. 2.51).

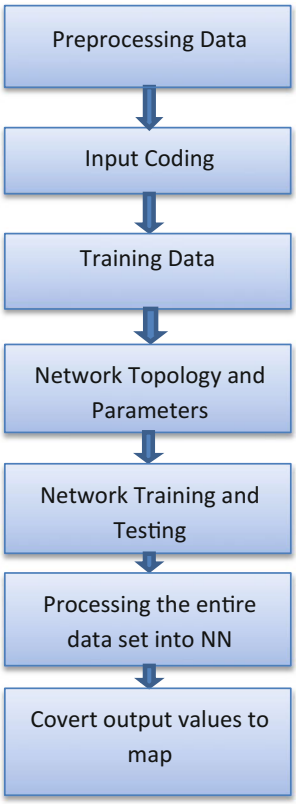


Fig. 2.50 Flowchart for mineral prospectivity mapping using a neural network

The training data should be linearly scaled based on the maximum and minimum value because for a feed forward network with asymptotic logistic activation function, the values of output activation are ordinarily restricted to between 0.1 and 0.9 (Masters 1993) and so the training data were scaled to the range [0.1, 0.9].

At the next stage a single input value was derived based on the separate radiometric magnetic and fault distance layers.

A one-of-n coding scheme was applied to the solid geology layers, so that each of the 12 rock types in the simplified layer was assigned a separate input layer unit in the neural network input layer.

The MLP that Brown et al. (2003) used in their study is shown in Fig. 2.52.

Brown et al. (2003) divided the data into: training, test and validation sets with the TSS: the total sum of squares error as the test for convergence. They used approximately equal numbers of deposit and no-deposit cells in the training of the network with the rationale that if real ratio of null to very rare deposits (1:1000) was used for training, the system would not be able to learn to identify them (Masters 1993; Zaknich 1999). The optimum topology of for the MLP was an 18-2-1

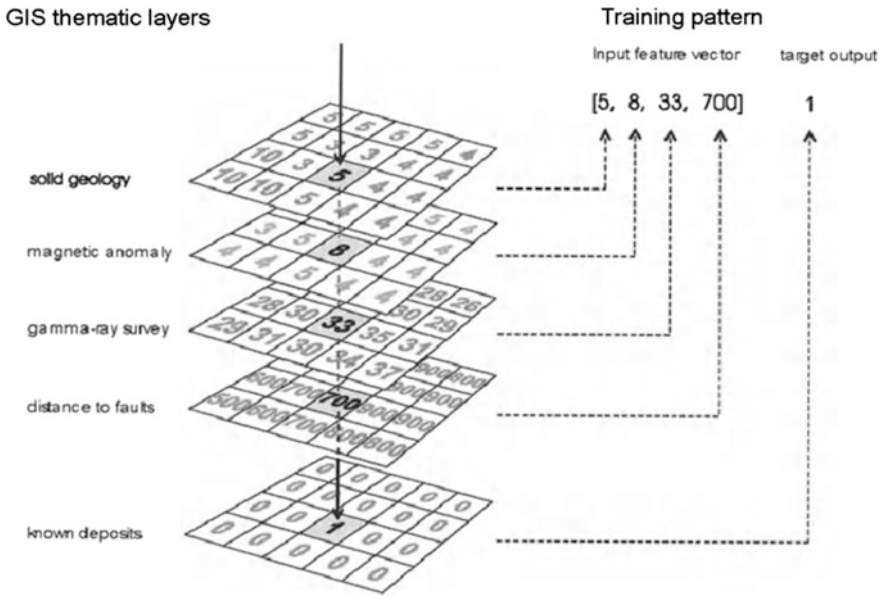


Fig. 2.51 Relationship between GIS thematic layers and feature vectors used as input to the neural network. At each location on the map grid, the cell values for each thematic layer are combined to form an input feature vector. Patterns used to train the network consist of an input feature vector paired together with the desired output, represented by the value of the binary layer showing the locations of known deposits (Brown et al. 2003)

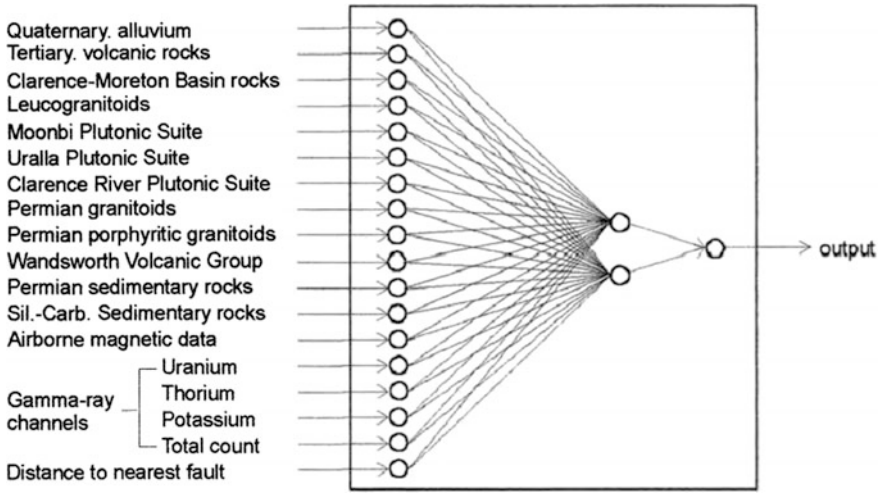


Fig. 2.52 Plan of the MLP neural network Brown et al. (2003) used. The network has an 18-2-1 topology (where the numbers refer to input, hidden and output units, respectively). Each map layer in the GIS database was assigned an input unit except the geology layer, for which 1-of-n encoding was used (Brown et al. 2003)

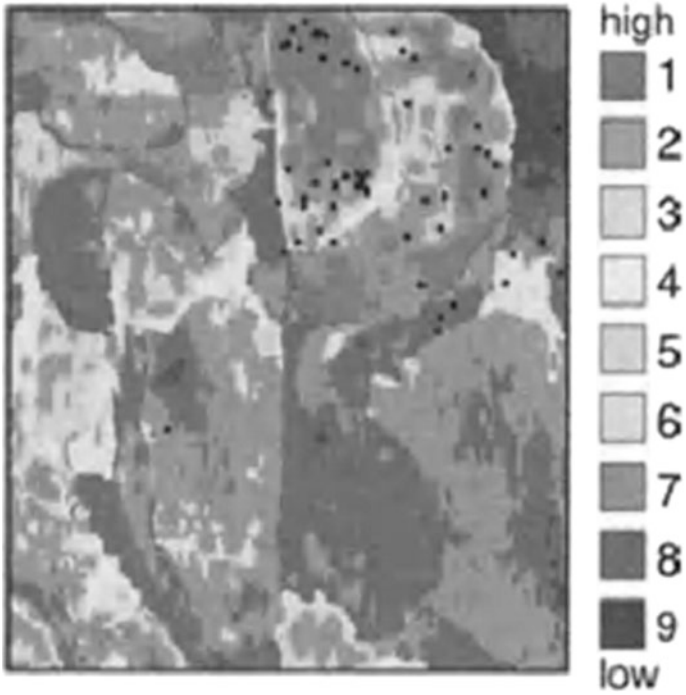


Fig. 2.53 Prospectivity maps for the Tenterfield (I: 100,000) area using neural network (Brown et al. 2003)

Table 2.25 Statistical measures of map quality for prospectivity maps produced using different methods (Brown et al. 2003)

Quality static	Method	
	Weights-of-evidence	Neural network
Chi-Square	66	83
Spearman's ρ	0.80	0.93
Kendall's τ	0.67	0.83

network. The initial weights were assigned randomly in the range of [−5, +5] and a series of networks were trained using ten different random sets of initial weights and for each case the SSE error was calculated for both the training and test data sets. The training error decreased during training. The test error first decreased to a minimum but then began increase and so training was stopped at the point at which the test SSE error reached the minimum observed over for all ten weights. The network with the best classification performance for the validation data set was selected as the optimum network and used to process the entire Tenterfield grid (Brown et al. 2003).

The outputs of the optimized neural network were used to produce the nine-class prospectivity map, shown in Fig. 2.53.

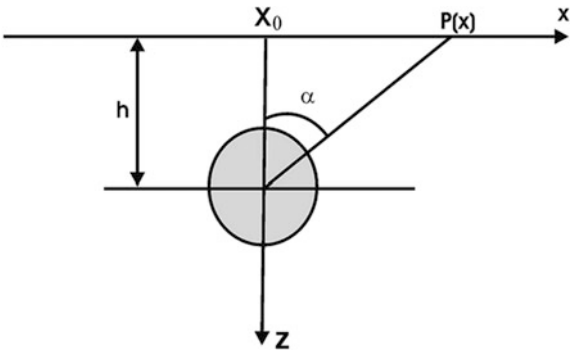


Fig. 2.54 Sphere model for SP method (Kaftan et al. 2014)

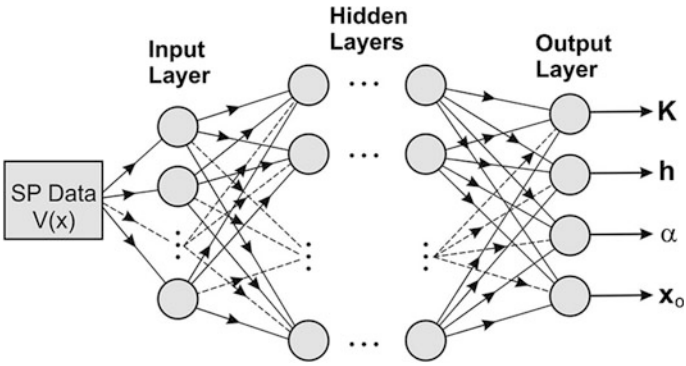


Fig. 2.55 MLPNN structure for SP application (Kaftan et al. 2014)

The statistical measurements of map quality for prospectivity map produced by neural network were compared to that of the weight-of-evidence method (Table 2.25).

The general similarity between the neural network and the empirically-derived weights-of-evidence maps suggested that the neural network result accounts reasonably well for the spatial relationships between known Gold deposits and the parameters of the GIS database. The main difference between the prospectivity maps is that the neural network map contains more tightly defined areas of high prospectivity because the network responds in a highly non-linear way so that a high favorability value is only assigned to areas where there is a combination of the critical favorable parameters (Brown et al. 2003).

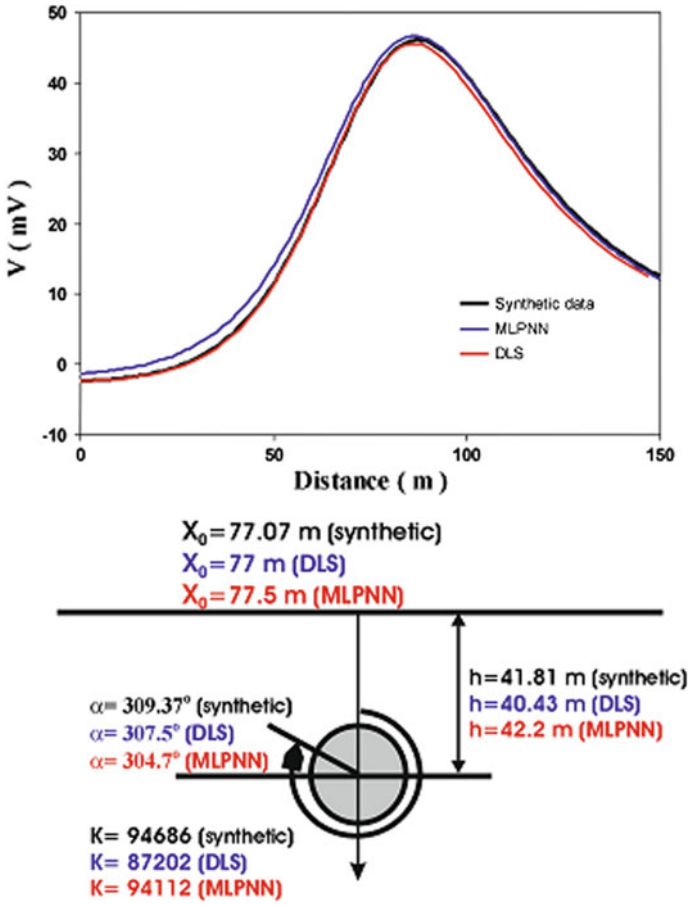


Fig. 2.56 Synthetic SP anomaly profile over a spherical body and its MLPNN and DLS inversion response (Kaftan et al. 2014)

2.6 Application of NN for SP Inversion Using MLP

Kaftan et al. (2014) used MPL neural networks to find the inverse solution for a buried and polarized sphere-shaped body, as shown in Fig. 2.54.

The outputs are the geometrical parameters of the sphere-like body: the polarization angle (α), the depth to the center of the sphere (h), the electrical dipole moment (k) and the distance from the origin (x_0) while the inputs are the SP values along the selected principal profile (Fig. 2.55).

In preparing the training data set the SP anomaly at any point P(x) over a spherical model can be calculated by the formula given by (Bhattacharya and Roy 1981):

Table 2.26 Training parameters of MLPNN

Number of training set	5000
Normalization data range	[−1, 1]
Number of hidden layers	2
Learning rate	0.1
Number of neurons in the first hidden layer	2
Number of neurons in the second hidden layer	5
Activation function hyperbolic tangent (tanh)	Sing/maid
Number of epoch to reach optimum	150
MSE at optimum epoch	0.00306092

Table 2.27 Range of sphere model parameters for training MLPNN

Parameter	Range
k	$10000 - 110000$
α	$260^\circ - 340^\circ$
x_0	$50 - 100\,\text{m}$
h	$7 - 75\,\text{m}$

$$V(x) = k\left(\frac{(x - x_0)\cos\alpha - h\,\sin(\alpha)}{\left((x - x_0)^2 + h^2\right)^{3/2}}\right) \tag{2.19}$$

where k is the electrical dipole moment, $k = I\rho/Z\pi$ is the depth to the center of sphere, x_0 is the zero distance from the origin and α is the angle between the polarization axis direction and vertical axis.

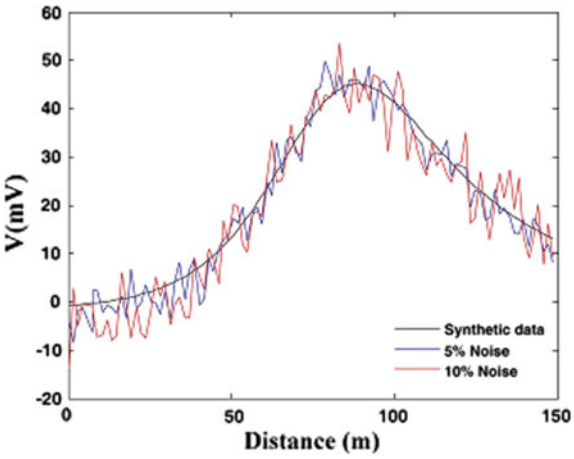


Fig. 2.57 Synthetic SP anomaly profile over a sphere-like body with and without noise (Kaftan et al. 2014)

Table 2.28 MLPNN inversion results for a synthetic SP anomaly profile over a spherical body with 5 and 10% of random Gaussian noise added (Kaftan et al. 2014)

	K	α	x_0	h
Model without noise	94,686	309.37	77.07	41.81
S/N ratio: 5%	94,095	301.46	78.25	41.95
S/N ratio: 10%	94,651	302.85	78.12	39.68

Table 2.29 DLS inversion results of a synthetic SP anomaly profile over a spherical body with 5% and 10% of random Gaussian noise added (Kaftan et al. 2014)

	K	α	x_0	h
Model without noise	94,686	309.37	77.07	41.81
S/N ratio: 5%	88,758	307.57	77.5	40.7
S/N ratio: 10%	85,386	306.48	77.5	41.04

Kaftan et al. (2014) used a two-hidden layer MLP neural network. The model was first tested on noise-free synthetic data, which was assumed to have the following parameters:

$$k = 94.686\ h = 41.81\ m, \alpha = 309.37^{\circ}, x_0 = 77.07\ m,$$

Which gives the response shown in Fig. 2.56.

The parameters/ranges that were used for training are listed in Tables 2.26 and 2.27. The MLPNN results were compared to the DLS inversion method (Damped Least Square) and showed a very good agreement (Fig. 2.56).

The parameter value range should be selected based on the shape of the anomaly curves and should not be too narrow. The important points to be considered are that deep SP sources generate broad anomaly curves over a large distance while shallow sources generate sharp anomaly curves over a short distance. Also the consideration of the monopolar or bipolar character of the anomaly can improve the determination of the parameter range for the polarization angle.

In the next stage, Kaften et al. (2014) examined the network performance under noisy conditions by adding 5, 10% of Gaussian noise to previously generated, noise-free SP data in order to estimate the k, α, x_0 and h for the synthetic model (Fig. 2.57).

The results of MLPNN for the noisy test data showed that the estimated model parameters do not vary significantly, and so the MLPNN solution is not particularly affected by noise level (Tables 2.28 and 2.29).

In order to evaluate the MLPNN performance for real data, Kaftan et al. (2014) used Self Potential data collected from Izmir-Urla-Demircili Village (Fig. 2.58).

The Urla S-P anomaly data was collected on 7×7 profiles oriented N–S and E–W with the survey details of Table 2.30.

They applied MLPNN to the data along 150 m long profiles $A-A'$ and $B-B'$ cross-sections taken from the contour in N–S and E–W directions (Fig. 2.59).

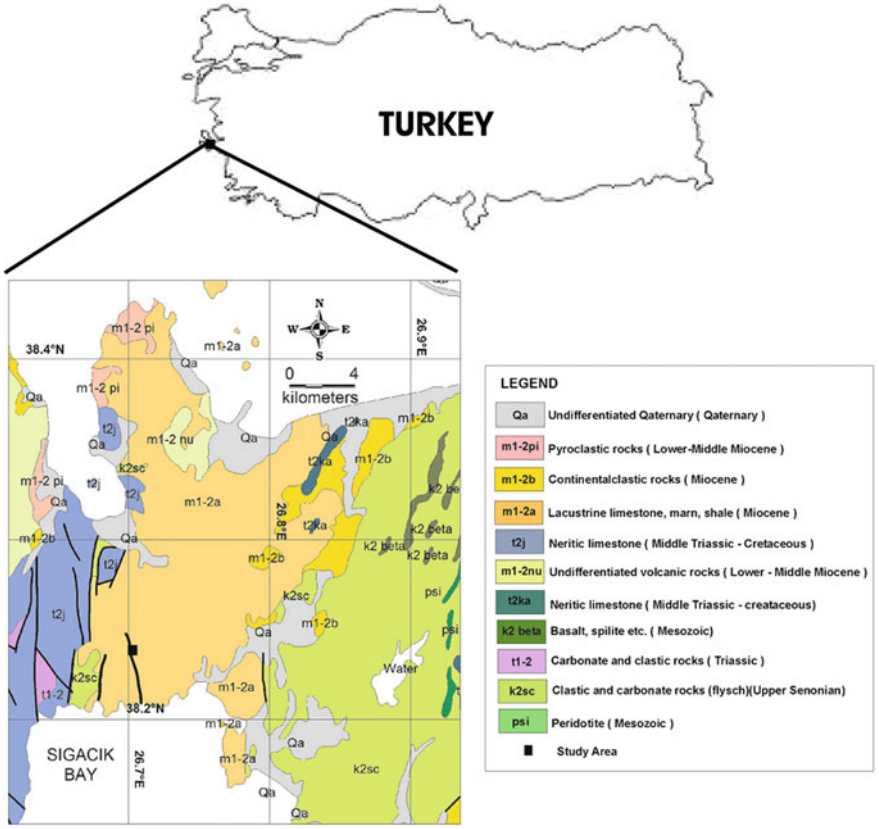


Fig. 2.58 The geological map of the study area after Kaftan et al. 2014

Table 2.30 SP surveying specifics

Measurement technique	Gradient
Each profile length	300 m
Distances between profiles	50 m
Electrode spacing	1.5 m

The MLPNN they used for real data had the same architecture and specifications and parameter ranges as the MLPNN which they used for the synthetic data. The MSE of MLPNN was 0.032719 after 300 epochs for cross section $A-A'$ and 0.0033268 after 250 epochs for cross section $B-B'$. To assess the accuracy of MLPNN the results were compared to that of DLS inversion results and agreed well with the least squares method (Table 2.31 and Fig. 2.60).

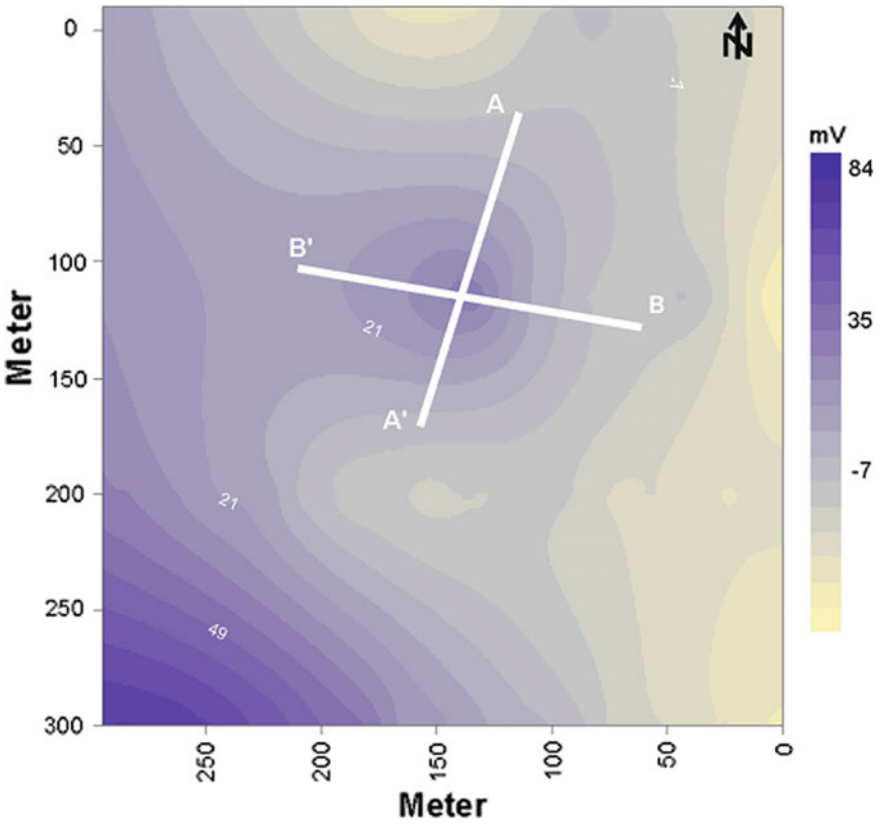


Fig. 2.59 SP contour map of the, A–A′ and B–B′ cross-sections across the Urla anomaly (Kaftan et al. 2014)

Table 2.31 The inversion results for MLPNN and DLS (Kaftan et al. 2014)

	K	α	x_0	h
A−A′ (MLPNN)	95,460	302	76.1	47
A−A′ (DLS)	89,454	301	75	46
B−B′ (MLPNN)	94,937	300	78	46
B−B′ (DLS)	105,079	296.3	80	49.8

2.7 Determination of Facies from Well Logs Using Modular Neural Networks

Interpretation of rock facies from well logs requires the recognition of changes which occur across layer boundaries as well as identifying facies from their characteristic log responses.

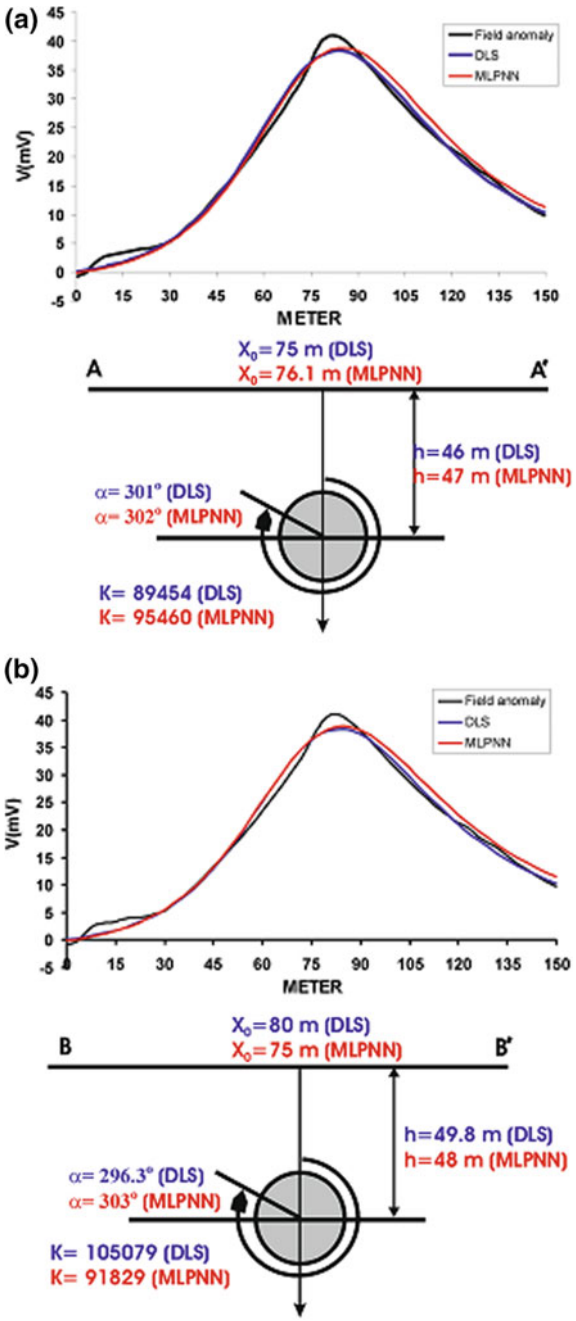


Fig. 2.60 **a** A–A′ cross-section SP anomaly over the Urla–Demircili village data set, and its MLPNN and DLS inversion solutions. **b** B–B′ cross-section SP anomaly over the Urla–Demircili village data set, and its MLPNN and DLS inversion solutions (Kaftan et al. 2014)

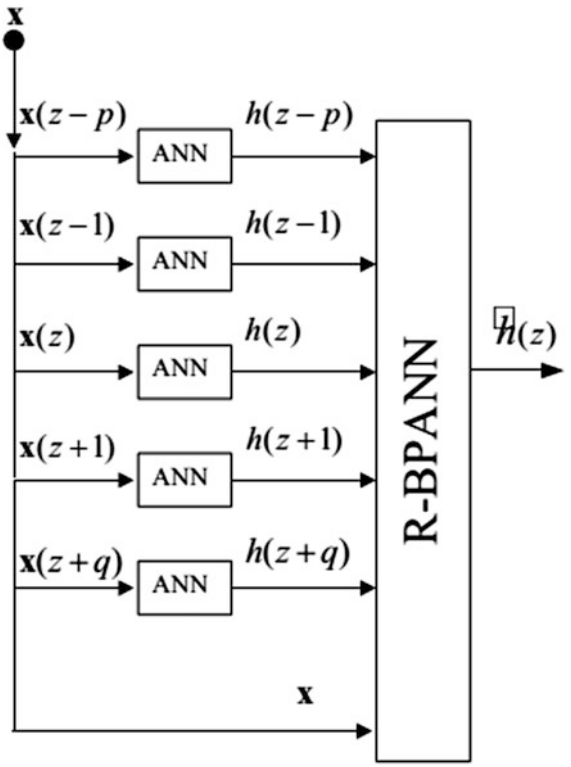


Fig. 2.61 Architecture of the modular neural network for lithofacies prediction using a two-step prediction with initial classification by ANN for each depth point, following by a recurrent R-BPANN that uses input from adjacent depth points. Both networks are 3-layer MLPs with single output (Bhatt and Helle 2002)

Bhatt and Helle (2002) used this property to determine facies from well logs. This property was exploited using a recurrent P-ANN derived from time-series analysis (Fig. 2.61).

Bhatt and Helle (2002) (Fig. 2.63) combined back-propagation neural networks in ensembles and modular systems where the multi-class classification problem of facies identification was reduced to a number of two-class problems (Fig. 2.62).

The idea was to reduce the multi-classification problem to a number of two-class classification tasks to maintain the simple architecture of the MLP with a minimum of hidden units. The multi-class classification problem was accomplished by constructing a modular neural network (MNN) using simple MLP as the building block with individual components of the MNN, or group of MLPs assigned to predicting a given log facies, enabling the MNN to solve the multi-classification problem by voting.

The log data used to train the network were those commonly available from wire line core logging:

Density, sonic, gamma, resistivity and neutron porosity

The synthetic data were modeled based on three distinct log facies and random noise at the level of 10% was added to each log as a tenth of the typical variation

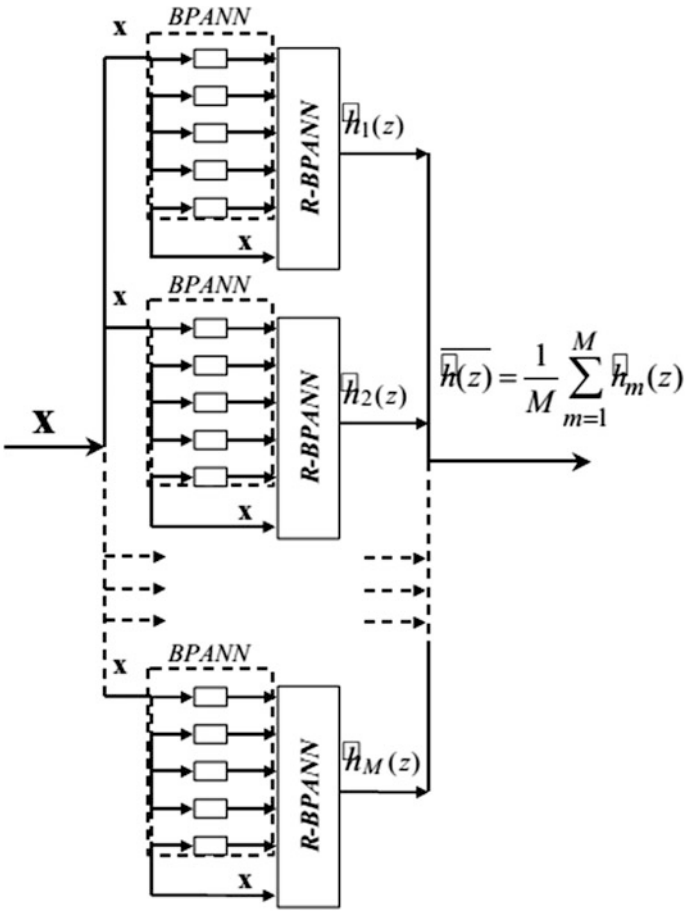


Fig. 2.62 Architecture of the CM (Cooperative Modular) for redundant combination of M modular networks as shown in Fig. 2.61 (Bhatt and Helle 2002)

range observed for a Brent Group North Sea reservoir at Jurassic level. Figure 2.64 shows the distribution of log values for the 3-facies model with 10% noise.

The training procedure for the component network in the modular system was optimized based on synthetic log network responses adapted from realistic models (Fig. 2.65). The building blocks used were simple three-layer back propagation ANNs.

They selected 150 samples at regular intervals from 3000 samples, at a nominal sampling distance of 10 cm for the training pattern, for the various tests. The network performance was tested using all 3000 samples. The resulting performance in terms of miss-hits for the 3 CMs, one for each facies, is illustrated in Figs. 2.66 and 2.67. Bhatt and Helle (2002) compared the results with the individual

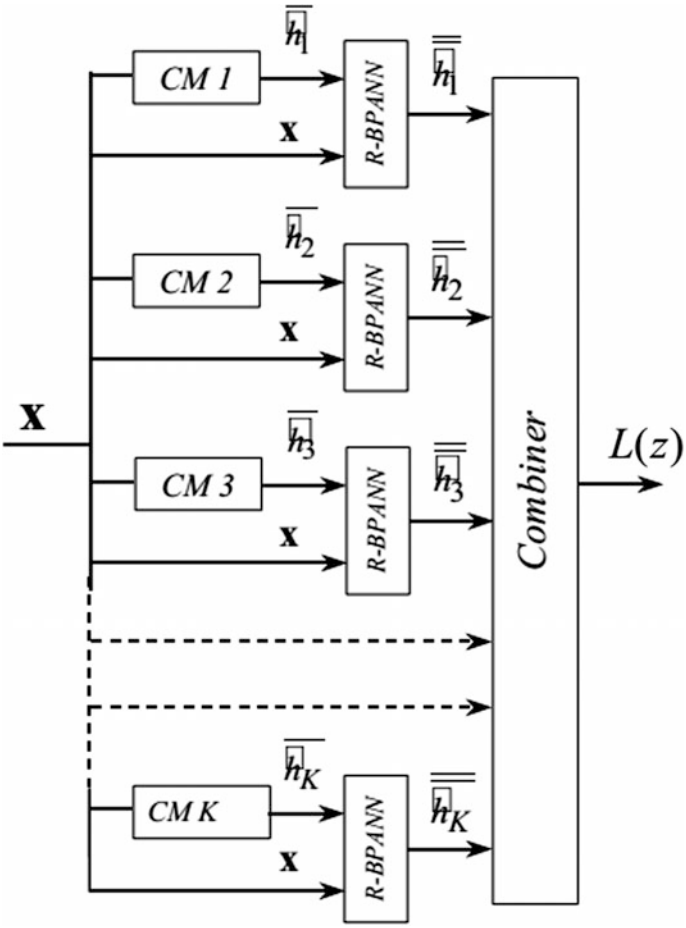


Fig. 2.63 Architecture of the modular neural network for predicting log-facies where each log-facies is designated to an expert CM-k as shown in Fig. 2.62. The combiner is a voter that votes the most probable log-facies using soft-max or prior model data; e.g. local stratigraphy, to eliminate remaining ambiguities (Bhatt and Helle 2002)

components of the stacked outputs and the miss-classification was less than 1% of 3000 samples which is a significant reduction.

Bhatt and Helle (2002) also used the Ness formation (Ejsberg field in the North Sea) which correlates with the stratigraphic interval between the Tarbet and Etvine-Rannoch formations of the Brent group. The Ness formation is divided into four main lithofacies:

(1) Channel sands (2) Crevasse splays (3) Lake Deposits and (4) Coals, which each have significant detectable differences in the log response. The main log feature of each facies is summarized in Table 2.32.

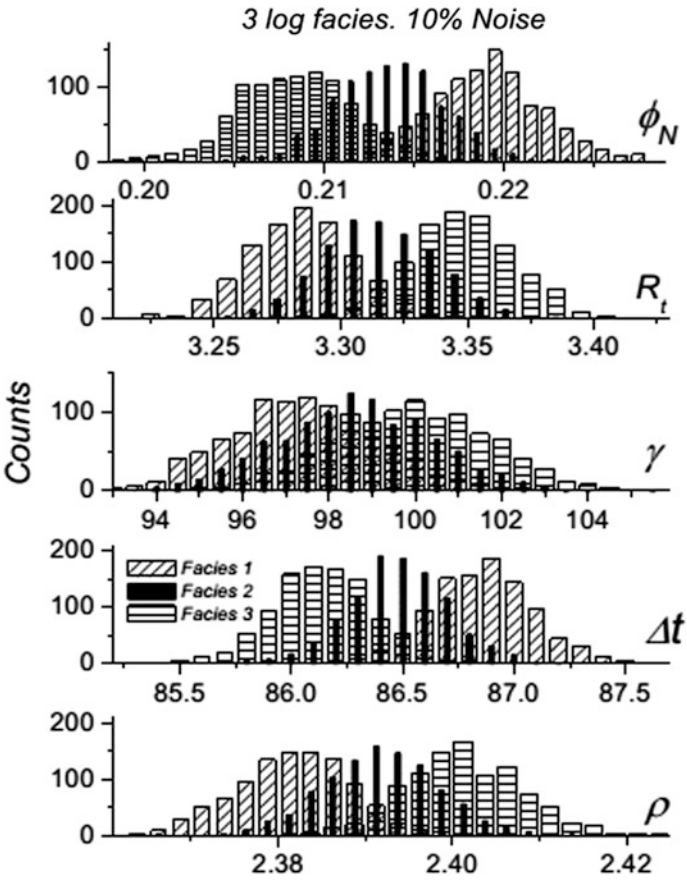


Fig. 2.64 Distribution of log values for the 3-facies model with 10% noise (Bhatt and Helle 2002)

The zonation was compared based on the cores and neural networks for four wells Q+2, Q-1, Q-2 and Q-21, and the classification performance shown in Table 2.33.

Training patterns were principally extracted from well Q-12, while the other three wells were kept anonymous to test the network. The mishit rates are shown in Table 2.34 with respect to facies, for this research and other methods.

Comparison of log facies from the Bhatt and Helle (2002) study (left) and the lithofacies shows very good adaption (Fig. 2.68). The average hit-rate is well above 90% in wells unknown to the network which shows the high performance of the MNN.

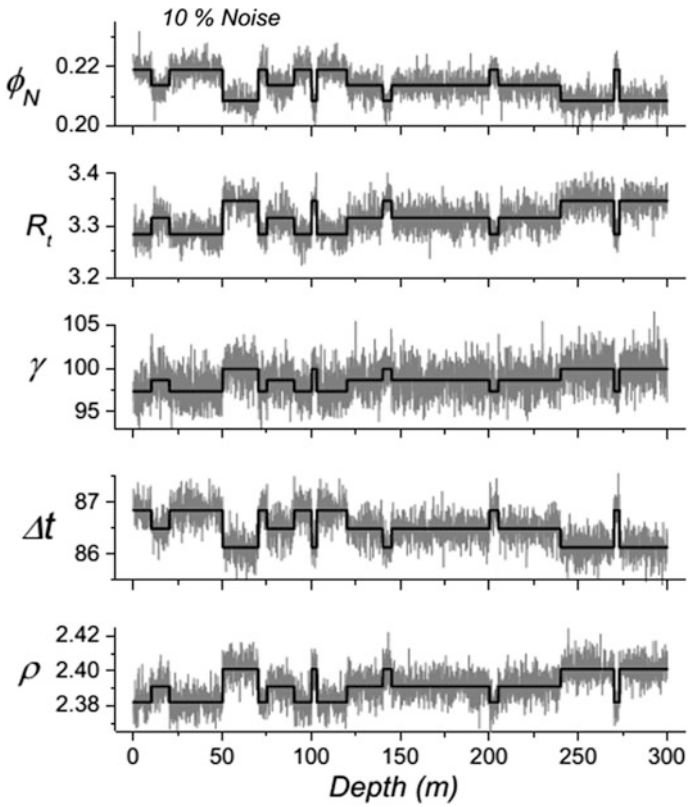


Fig. 2.65 Synthetic logs for a multi-layer (17 layers) model composed from three different log facies. Layer thickness varies from 3 to 55 m (Bhatt and Helle 2002)

2.8 Estimation of Surface Settlement Due to Tunneling

Tunneling has always posed dangers in shallow urban areas and soft soils, where ignorance of the presence of the hazard can have very unpleasant and costly repercussions. The problem of surface settlement and its effects on surface structures are the main dangers, which can and should be controlled by geomechanical modeling to prevent damage to surface structures. The surface settlement due to tunneling depends on various factors such as the drilling system, drilling parameters, the geometrical shape of the tunnel, geological conditions and geotechnical properties. Empirical and semi-empirical formulas are available for the prediction of surface settlement and different analytical and numerical methods have been used to predict this kind of settlement. Hajian et al. (2014) applied artificial neural networks and finite element methods to the estimation of surface settlement, in the context of the geology of the studied route of the Mashhad Subway Line 2, in Iran. Using the database related to tunneling in Mashhad Subway Line 2 project, a series of inputs

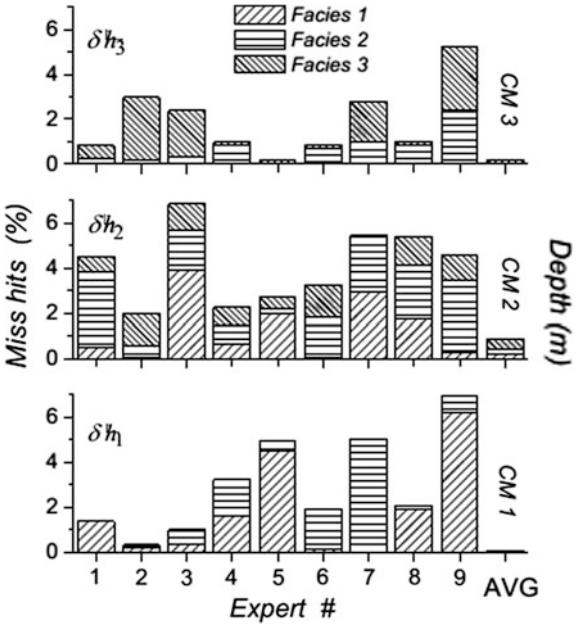


Fig. 2.66 Distribution of miss-hits for the three committees CM1, CM2 and CM3 and their component networks designated for predicting the individual log-facies (Bhatt and Helle 2002)

which are relevant to surface settlement and the outputs of mean settlement measured by instrumentation, different models of artificial neural networks were designed using MATLAB software which were then trained and tested. From both of the models, the one with the optimal performance in testing and showing the best correlation between the actual and the predicted settlement is chosen as the optimal model. The errors of the network were shown in the models, and the correlation between actual settlements and the predicted ones is studied. Moreover, some cross-sections of the route were modeled using the finite element method using the 2-dimensional Plaxis software, and their results are introduced. Finally, the amounts of settlements estimated by neural network and finite element were compared and tested against some actual measured cross-sections of the route.

2.8.1 Introduction

In general, the drilling of tunnels and other underground structures leads to the removal of soil and rock masses leading to considerable changes in stresses around them. Surface settlement is one of the consequences of the disturbance of the soil and shallow bedrock and is especially important in urban areas and particularly when the tunnel passes under residential areas. Consequently, surface settlement

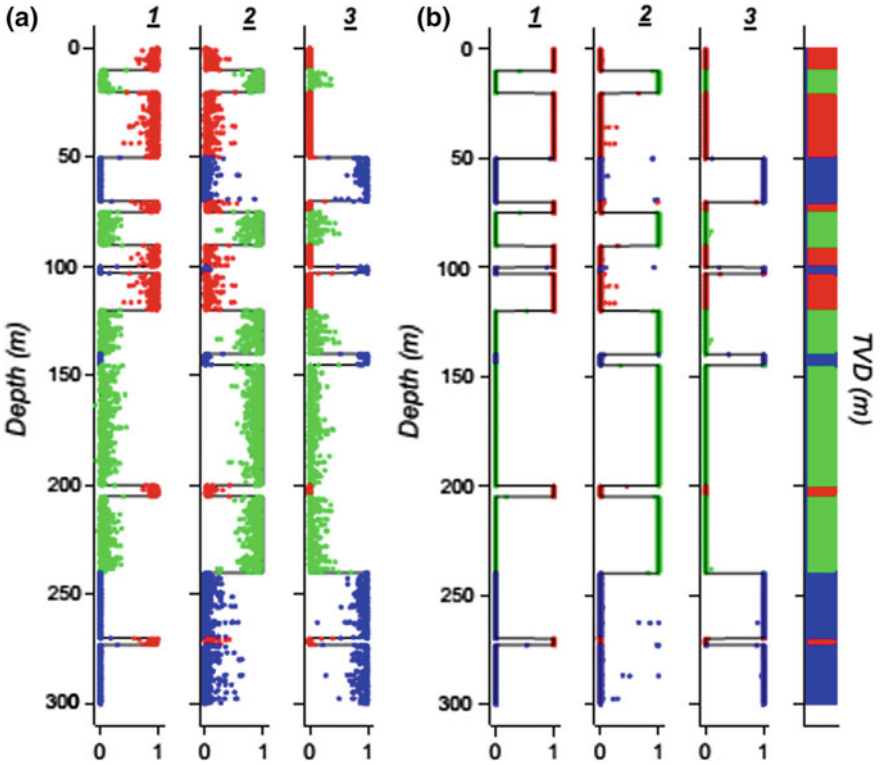


Fig. 2.67 Output from the Committees (**a**) and the sequential application of the R-BPANN. **b** The predictions for the 17-layer model shown in Fig. 2.63 (Bhatt and Helle 2002)

Table 2.32 Main log feature of each facies

Face No	Name	Material	Log features
1	Channel sands	– Relatively clean – Poorly to well-sorted sand	– Low values of Gamma, density, velocity – Average permeability: 1700 mD – Porosity $\propto$ 26% Resistivity and neutron porosity variable, depending on the type of pore fluid
2	Crevasses	– Fine-grained sand inter bedded with clay-rich material	– Log responses more variable than channel sands dependent on the clay content & pore fluid
3	Lake	Jurassic mud rocks	– High gamma – High density – High velocity – Low resistivity
4	Coal	–	– Low density – Low sonic velocity – Distinct peak at high Values of the neutron porosity

Table 2.33 The classification performance of zonation (Bhatt and Helle 2002)

	Q-12	Q-l	Q-2	Q-21	Avg
Channel	3.6	1.3	9.1	12.3	6.6
Crevasse	1.1	4.3	4.0	2.5	3.0
Lake	11.3	16.0	16.5	11.3	13.6
Coal	2.8	3.0	2.7	6.7	3.8
Average	4.7	6.2	8.1	10.0	

Table 2.34 Miss-hits with respect to wells and facies (Bhatt and Helle 2002)

Technique	Researcher(s)	Results	
		Facies	Mishit rate (%)
Discriminant factor analysis	Busch et al. (1987)	7	25
Discriminant factor analysis	Jian et al. (1994)	8	23
Recurrent neural network	Yang et al. (1996)	6	17
Conventional clustering analysis	Yang et al. (1996)	6	40
Hybrid neural network	Chang et al. (2000)	4	13
Modular neural networks	Bhatt and Helle (2002)	4	10

should be estimated before the commencement of construction to prevent damage arising from tunneling affecting surface structures and sub-surface structures (Mair and Taylor 1997), Several different empirical, analytical and numerical methods have been used to date to predict the surface settlement due to tunneling. One of the methods for the estimation of surface settlement uses artificial neural networks (ANN). Although, ANNs are not directly comparable with natural nervous systems, they possess some attractive features, which make them distinctive in certain applications like pattern distinction, robotic functions, and wherever linear or non-linear learning is required. ANNs, due to their flexibility and high capability for learning, can take information from past experience and improve their behavior during learning (Hajian et al. 2011a, b); consequently, this technique can be used to predict surface settlement in regard to the geometric properties of tunnels and geotechnical properties of the surrounding area.

With the development of new technology in the fields of both software and hardware, researchers began to use numerical methods. The application of finite element methods (FEM) as one of the preeminent numerical methods began to be used in geotechnical engineering in 1966 (Zinkiewicz and Taylor 2000). These days, powerful software based on numerical methods is available, which can be used in tunnel modeling and the assessment of surface settlement due to tunneling. FEM using Plaxis software in the field of numerical methods is a new technique for estimating surface settlement due to tunneling. The high accuracy in the calculations is the main advantage of this method if the input data are precise.

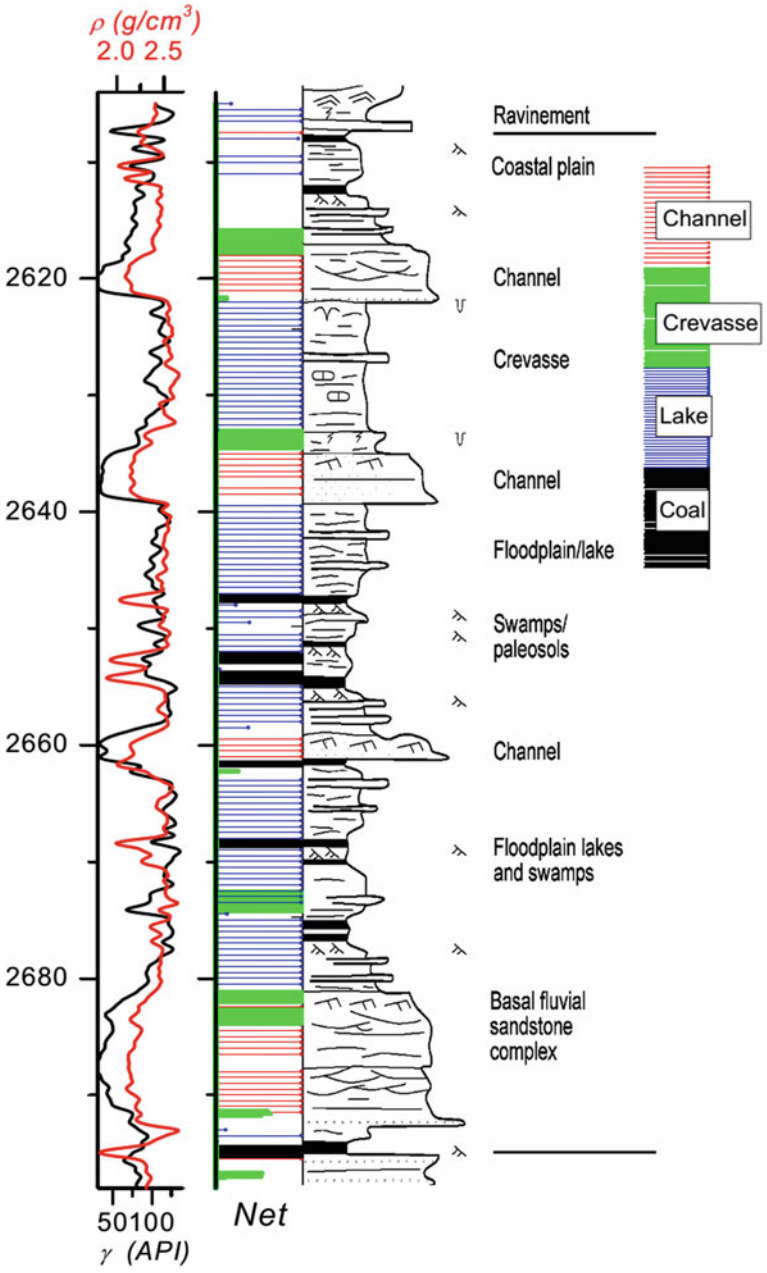


Fig. 2.68 Detailed comparison of log facies from Bhatt and Helle (2002) study (left) and lithofacies (right) after Ryseth et al. (1998) and Bhatt and Helle (2002)

2.8.2 The Finite Element Method in Plaxis Software

Plaxis is a computer program based on the finite element method of Zinkiewicz and was designed for 2-D and 3-D geotechnical analysis of the deformation and stability of soil structures, groundwater and heat flow, in geo-engineering applications such as excavation, foundations, embankments and tunnels. The 2-dimensional edition of this software is used for two-dimensional analysis of plane strain. The plane strain model is applicable to structures with large aspect rations, in which their cross-section, geometry and loading are fixed, and supposing that the length displacements are zero.

2.8.3 The Available Elements for Modeling

In the 2-dimensional edition, 6-node or 15-node plane elements can be used for modeling materials like soil, rock, concrete, etc. The 6-node elements provide the possibility of second-order interpolation for displacement, and the stiffness matrix can be generated by numerical integration using three stress points, while, fourth-order interpolation is used for 15-node elements, and the integration calculated using twelve points of stress.

2.8.4 Soil and Rock Behavior Models

In the Plaxis software, different advanced models are considered for soil and rock behavior modeling. These models and their applications are briefly explained below.

(A) Linear Elastic: This is the simplest model in this software, which is defined by using two parameters: Young's modulus and Poisson's ratio. While the linear elastic model can be used to model the behavior of stiff structures located in soil, like steel or concrete linings; it cannot be considered suitable for soil itself.
(B) Mohr-Coulomb: This is a nonlinear model but strong and simple, which can introduce an appropriate behavior for soil and rock. Elastic-absolute plastic behavior can be simulated based on five parameters including Young's modulus, Poisson's ratio, cohesion, friction angle and dilation angle. The dilation angle is used for modeling the soil volume increase (e.g., dense sand).
(C) Jointed rock model: This model was developed based on the Mohr-Coulomb model, in which the effects of stratifications, joints and anisotropies can be observed. In this model, the environment is elastic, which can include elastic modulus and Poisson's ratio in orthogonal directions. Plastic sliding can only occur along pre-defined cracks.

(D) Soft soil creep model: This model is formulated based on visco-plastic behavior, and includes soil diachronous changes such as creep and secondary compression; therefore, it can be effectively used for long-term loading simulations such as the soil under foundations, but is not applicable to unloading activities like tunneling.

(E) Hardening soil model: This model which is more advanced in comparison with Mohr-Coulomb model is considered as a hyperbolic model of elasto-plastic type, which is formulated based on frictional hardening. In elastic-absolute plastic models, the yield surface of the hardening model is not fixed in the space of principal stresses, and can deform due to plastic strain. This model's parameters are similar to those of Mohr-Coulomb model. The only difference is that in this model, the soil hardness is considered more accurately in three conditions including oedometer loading, triaxial loading and unloading (reloading) under the reference confining stress. The hardening model can be used for simulating the behavior of sand, gravel and pre-consolidated clay; in particular this model can be effectively used for activities like tunneling which include unloading.

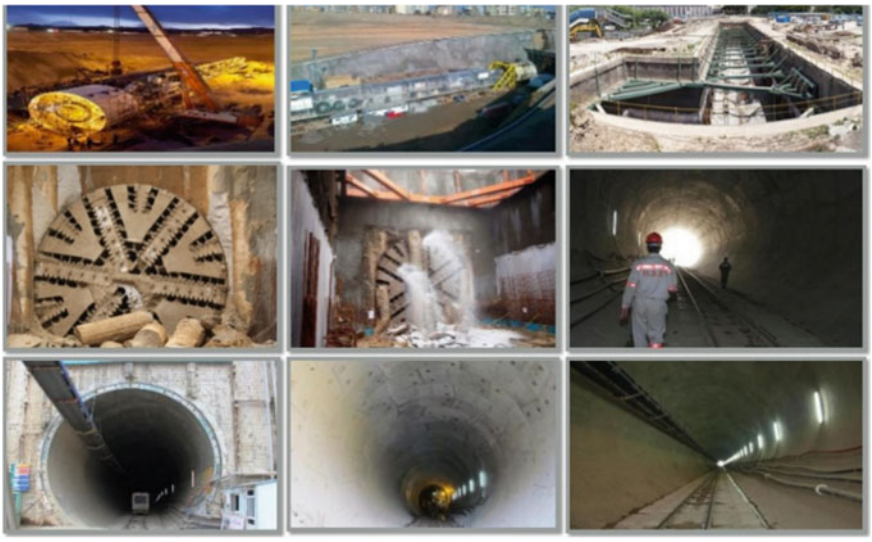


Fig. 2.69 Photos of the Line2 Mashhad metro construction from early to final stages. *Source* www.slideshare.net/khakestary1363/muroc-mashhad-urban-railway-operation-company

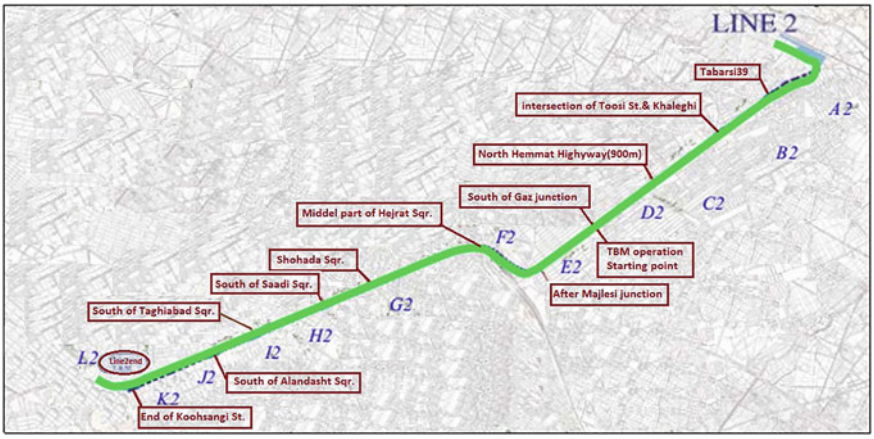


Fig. 2.70 Map of the Line2 route and stations (www.slideshare.net/khakestary1363/muroc-mashhad-urban-railway-operation-company)

2.8.5 *The Studied Route of the Mashhad Subway Line 2 Project*

The basic design studies for Mashhad metro line2 was accomplished by a joint effort between the project's main local consultant "Pajoohesh" and the French consultant "Systra". Twelve groups of qualified engineering consultants assisted the main consultant in preparation of basic studies such as topographical and geotechnical studies, civil design, traffic studies,

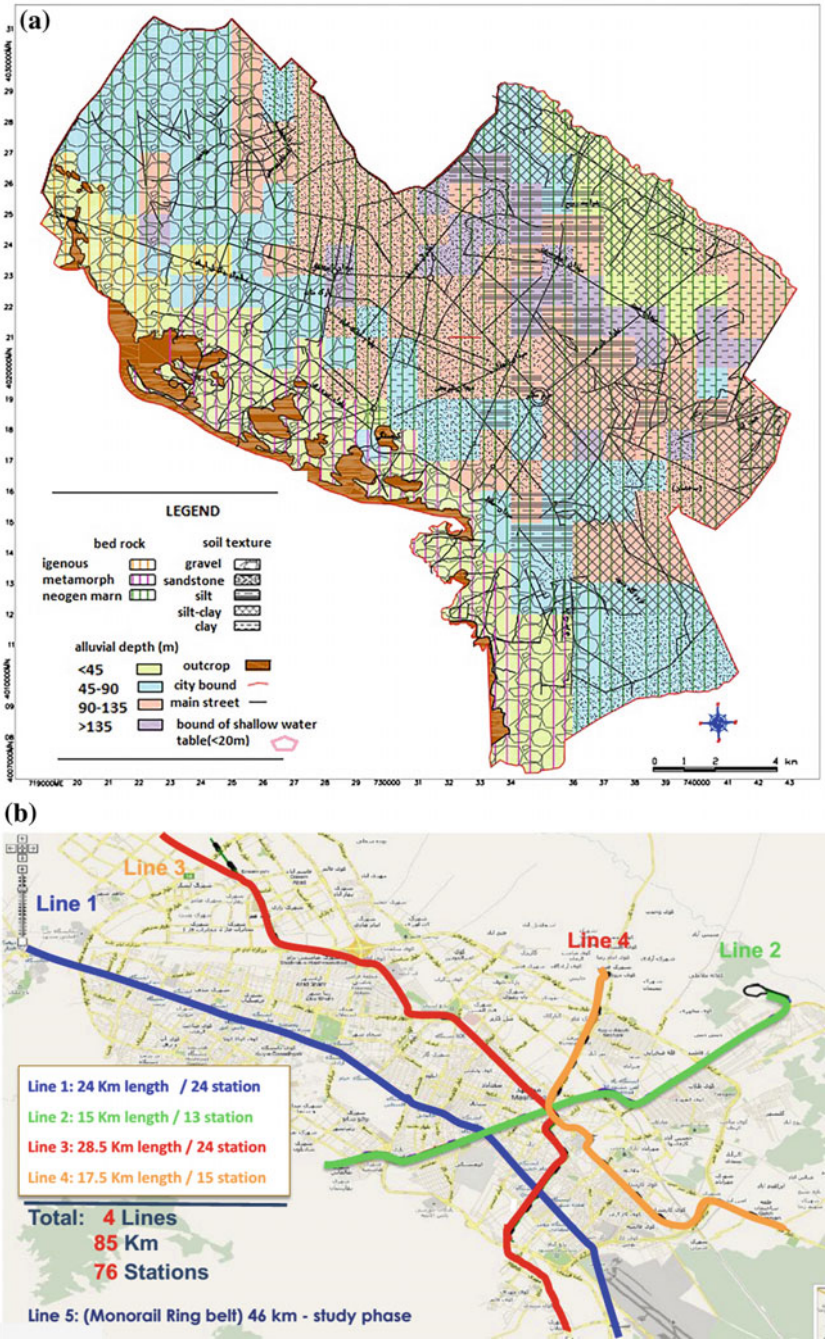
Urban utility tracing, track design, rolling stock functional specification, bio-eco and value engineering. The preliminary design of line2 was carried out and a call for international tender (E.P.C) was launched and a local General Contractor (GE) was appointed. A grouped photo of the Line2 Mashhad metro construction from early to final stages is illustrated in Fig. 2.69.

The geographical route of Mashhad Subway Line 2 project (Fig. 2.70) runs from northeast to southwest of Mashhad city from Tabarsi Blvd. to Shazan Blvd. (Fig. 2.70).

The studied route is 4.3 km long, a part of line 2. One side of this route begins at the North shaft and passes the stations A2, B2 and C2, and then ends at D2 (Fig. 2.70).

In addition, part of the south shaft leading to station L2 was carried out. Therefore, the database used in this study was gathered between chainages −365 and +3400 as well as between the chainages +13080 and +13614 of Mashhad Subway Line 2.

The Mashhad Plain lies between the Binalud and Hezarmasjed mountains, running from northwest to southwest. Mashhad city is located on young alluvial deposits deposited by the main branches of the Kashafrud River, which has its


(a)
LEGEND
bed rock
igenous
metamorph
neogen marn
soil texture
gravel
sandstone
silt
silt-clay
clay
alluvial depth (m)
<45
45-90
90-135
>135
outcrop
city bound
main street
bound of shallow water table(<20m)
0 1 2 4 km
(b)
Line 3
Line 1
Line 4
Line 2
Line 1: 24 Km length / 24 station
Line 2: 15 Km length / 13 station
Line 3: 28.5 Km length / 24 station
Line 4: 17.5 Km length / 15 station
Total: 4 Lines
85 Km
76 Stations
Line 5: (Monorail Ring belt) 46 km - study phase

◄**Fig. 2.71** **a** Geotechnical Zonation map of Mashhad **b** The route of lines across Mashhad city (to match with the geotechnical zonation). Source of a: http://www.preventionweb.net/files/…/30411_2infoaboutearthquakegemmashhad.pdf and Source of b: www.slideshare.net/khakestary1363/muroc-mashhad-urban-railway-operation-company

source in the northern hillsides of the Binalud Mountains and the southern hillsides of the Hezarmasjed Mountains. Mashhad's surface soil texture varies from gravel to clay. The surface soil of the northeastern border tends to be clay, which changes to silt-clay and silt. Gravel channels occur in the western and southern parts. The sandy soil lies between gravel and silt-clay soils. The geotechnical Zonation map of Mashhad city is shown in Fig. 2.71.

Along the studied route, the soil of the North Shaft as far as station D2 is mostly comprised of two layers including clay and sand or clayey sand. Moreover, the soil texture of south shaft as far as station L2 comprises two layers, which are made from clay, sand and gravel. The level of the groundwater along the tunneling route only changes under the tunnel crown.

2.8.6 Characteristics of the Tunnel

The tunnel for the Mashhad Subway Line 2 was dug by an Earth Pressure Balance (EPB)-type Tunnel Boring Machine (TBM) There are two TBMs, one of which drills the north shaft and the other the south shaft. The external diameter of the tunnel is 9.1 m. For the studied route, the depth of tunnel varies between 13.15 and 21.65 m.

In an EPB type of TBM, the concrete segments are emplaced at the end of the shield to be installed in the wall of the tunnel. The diameter of the shield is greater than that of concrete rings; therefore, some space is created between the concrete lining and the drilled soil, which is called trailing empty space. This trailing empty space tends to be filled with the surrounding soil when the shield moves forward. In fact, the soil around the steel shield collapses; consequently, the convergence of the soil into the inside tunnel is one of the significant reasons for surface settlement. Grouting operations are performed to prevent this settlement; thus, the grouting pressure plays an important role in controlling surface settlement.

Along the studied route, grouting has been performed with pressures of up to 3 bars. The grouting operations were performed from rings #12 to #2267 in the northern route as well as from rings #24 to #380 in the southern route.

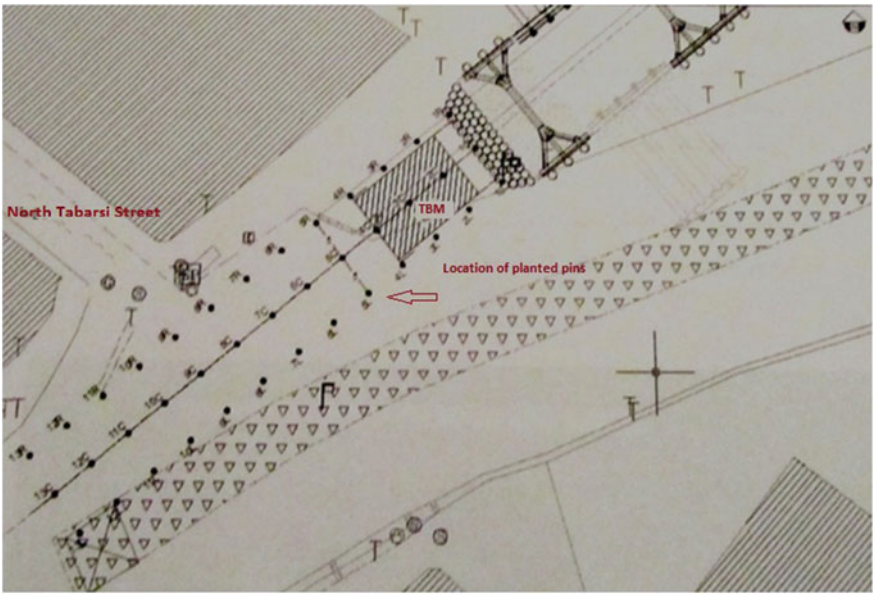


Fig. 2.72 The location of planted pins in the rout of Mashhad Subway Line 2 Project

Table 2.35 Description of the inputs and output of the network (Hajian et al. 2014)

Type of datum	Description of datum
Input #1	Depth of tunnel (m)
Input #2	Dry density of the soil of first layer (kN/m^3)
Input #3	Cohesion of the soil of first layer (kN/cm^2)
Input #4	Frictional angle of the soil of first layer (Degree)
Input #5	Modulus of elasticity of the soil of first layer (kg/cm^2)
Input #6	Dry density of the soil of second layer (kN/m^3)
Input #7	Cohesion of the soil of second layer (kN/cm^2)
Input #8	Frictional angle of the soil of second layer (Degree)
Input #9	Modulus of elasticity of the soil of second layer (kg/cm^2)
Input #10	Grouting pressure (Bar)
Input #11	Level of groundwater (*1* for above the tunnel, *0.5* for inside the tunnel, *0* for under the tunnel)
Output	Surface settlement (mm)

2.8.7 The Surface Settlement Measurement Operations

To control surface settlement in underground projects, earth monitoring is carried out in order to evaluate the accuracy of predictions and preventive measures are put in place to decrease the possibility of dangers and financial losses. The simultaneous control of drilling processes as well as the ground surface displacements is also of great importance, especially in urban areas. Mashhad Urban Railway Company's geotechnical consultant Arthe Civil & Structure BV performed the measurements and calculations of the surface settlement for the route of the Mashhad Subway Line 2 project.

The surface settlement measurement operations are based on planting and installing the markers, reading them using appropriate surveying instruments, recording, primary processing and presentation of the obtained data. To measure the surface settlement in the Mashhad Subway Line 2 project, marker pins are used. These pins are planted at between 5 and 10 m spacing along the route on both left and right hand side as shown on Fig. 2.66. In this study, the final measured amounts of surface settlement due to the height changes of central pins in each cross-section are used (Fig. 2.72).

2.8.8 Surface Settlement Prediction Using ANN

The ANN used in this study is designed using the MATLAB software. The existing database includes 346 data values related to cross-sections of the studied route. The training, testing and validation data related to each model are selected randomly. The number and description of the inputs and outputs (targets) of network are presented in Table 2.35.

The tunnel of Mashhad Subway Line 2 has the same diameter and the same kind of drilling, and so variations in these parameters are not applied to the input of the network. For the cross-sections, which have one soil layer, the inputs are specified as two equal layers. Where cross-sections have four soil layers, and where the geotechnical parameter of two soil layers are close to each other, these are considered as one equivalent layer using the average of the parameters and then the inputs into the network modeling are specified as two equivalent layers.

With regard to transfer functions, it is better to assign initial values between 0 and 1 to the input and target data; and these are then normalized using Eq. 1.61, to become the suitable input data for processing:

$$x_N = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{2.20}$$

The feed-forward back-propagation ANN was used, with Levenberg-Marquardt used as the training algorithm. The mean squared error (MSE) was used as the

Table 2.36 The results obtained from the training and testing of some models of ANN (Hajian et al. 2014)

Number of layers	Type of Transfer function	Number of neurons	Correlation of testing data	Correlation of all data	Training MSE	Validation MSE	Testing MSE
2	LOGSIG	15	0.88	0.85	1.630e-05	6.139e-06	6.608e-05
	TANSIG	1					
2	TANSIG	20	0.03	0.56	6.735e-05	3.541e-05	3.019e-05
	PURELIN	1					
3	PURELIN	20	0.50	0.88	1.177e-05	6.881e-06	6.483e-05
	TANSIG	18					
	PURELIN	1					
3	LOGSIG	10	0.65	0.46	1.516e-05	1.013e-05	3.486e-04
	PURELIN	14					
	TANSIG	1					
4	PURELIN	20	0.98	0.86	3.194e-05	4.349e-06	6.086e-06
	LOGSIG	18					
	LOGSIG	13					
	TANSIG	1					
4	PURELIN	20	0.35	0.57	6.727e-05	1.644e-05	2.242e-05
	TANSIG	20					
	LOGSIG	18					
	PURELIN	1					

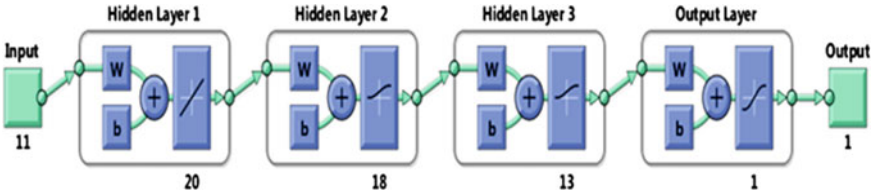


Fig. 2.73 The structure of the optimal model of ANN (Hajian et al. 2014)

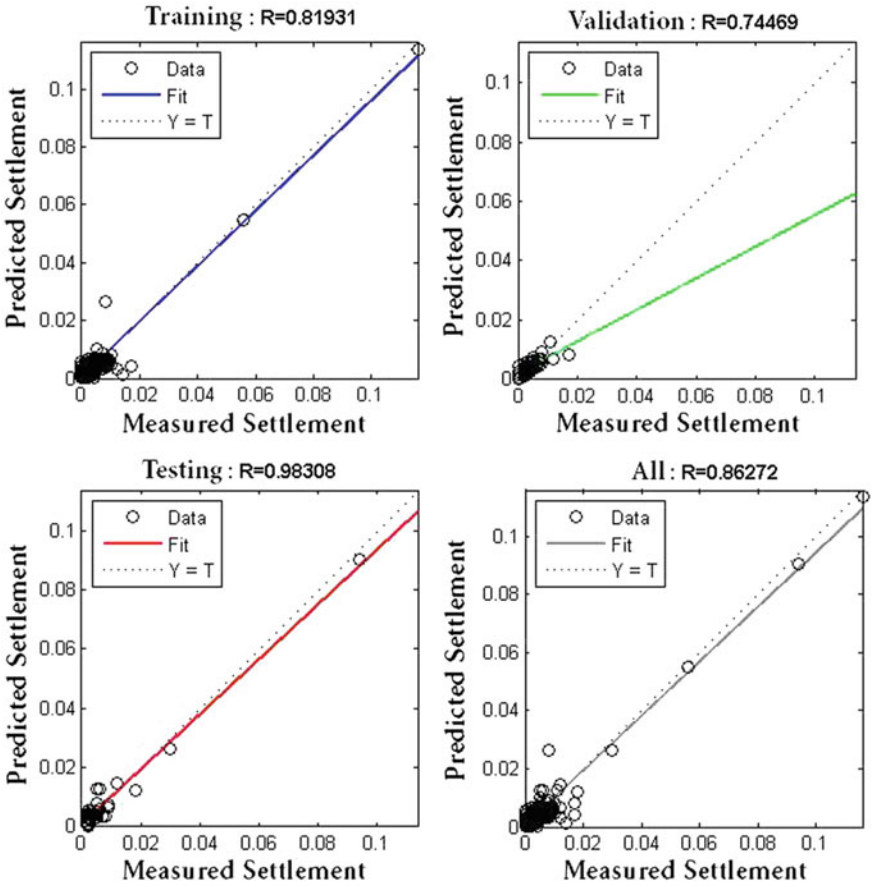


Fig. 2.74 The actual amounts of settlement versus the amounts of settlement predicted using ANN (Hajian et al. 2014)

Table 2.37 Geotechnical properties of soil related to the chainage 1+050 (Hajian et al. 2014)

Description of layer	Thickness of layer	Dry density, γ_d	Cohesion, C'	Frictional angle, ϕ'	Modulus of elasticity, E
	(m)	(kN/m^3)	(kg/cm^2)	(Degree)	(kg/cm^2)
CL-ML	25	16	0.25	25	200

Table 2.38 Geotechnical properties of soil related to the chainage 2+090 (Hajian et al. 2014)

Description of layer	Thickness of layer	Dry density, γ_d	Cohesion, C'	Frictional angle, ϕ'	Modulus of elasticity, E
	(m)	(kN/m^3)	(kg/cm^2)	(Degree)	(kg/cm^2)
CL-ML	21	16.7	0.3	25	300
SM	4	16.2	0	30	500

Table 2.39 Geotechnical properties of soil related to the chainage 13+160 (Hajian et al. 2014)

Description of layer	Thickness of Layer	Dry density, γ_d	Cohesion, C'	Frictional angle, ϕ'	Modulus of elasticity, E
	(m)	(kN/m^3)	(kg/cm^2)	(Degree)	(kg/cm^2)
SC-SM	13	19	0	35	800
CL-ML	16	17	0.2	24	300
GC-GM	3	19	0.11	40	1000
CL-ML	8	17.5	0.25	20	300

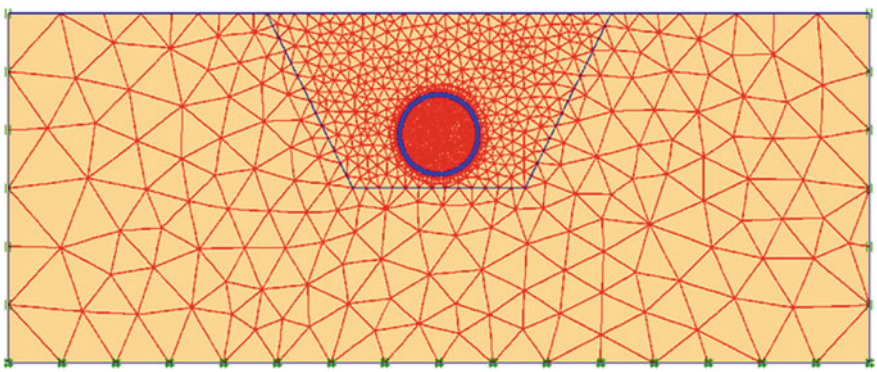


Fig. 2.75 The meshed model of cross-section located at the chainage 1+050 (Hajian et al. 2014)

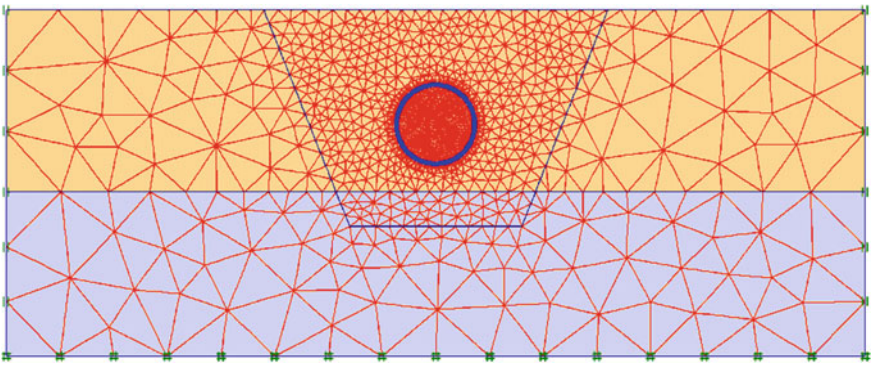


Fig. 2.76 The meshed model of cross-section located at the chainage 2+090 (Hajian et al. 2014)

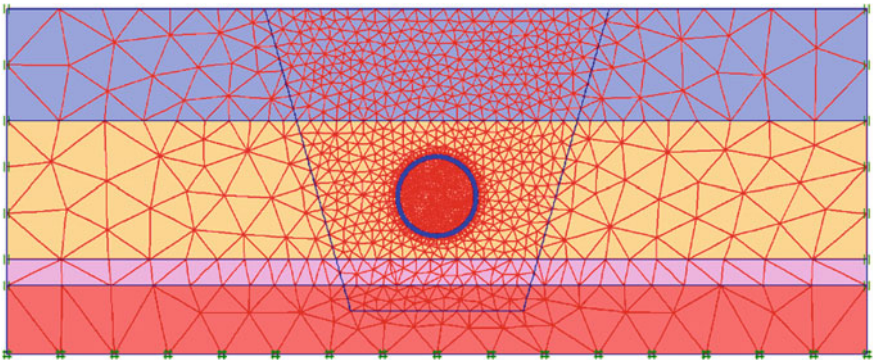


Fig. 2.77 The meshed model of cross-section located at the chainage 13+160 (Hajian et al. 2014)

performance function. 96 models of ANN including one with up to three hidden layers and one output layer were designed, trained and tested. These networks with different types of transfer functions in each layer with differing numbers of neurons were modeled. The correlations and MSE values for some samples of the models are presented in Table 2.36.

After 24 models had been analyzed, it was observed that using the Logsig transfer function for the output layer led to very low correlations between the actual data and the predicted values and so this function was not used as the transfer function for the output layer in the next models. The results obtained from training and testing of the network, the four-layer model including Tansig, Logsig, Logsig and Purelin transfer functions containing 20, 18, 13 and 1 neurons in each layer respectively, showed better results in the testing of network than the other models. The correlation coefficient obtained from this model between actual and predicted amounts of surface settlement is 0.86. The schematic of the structure of this

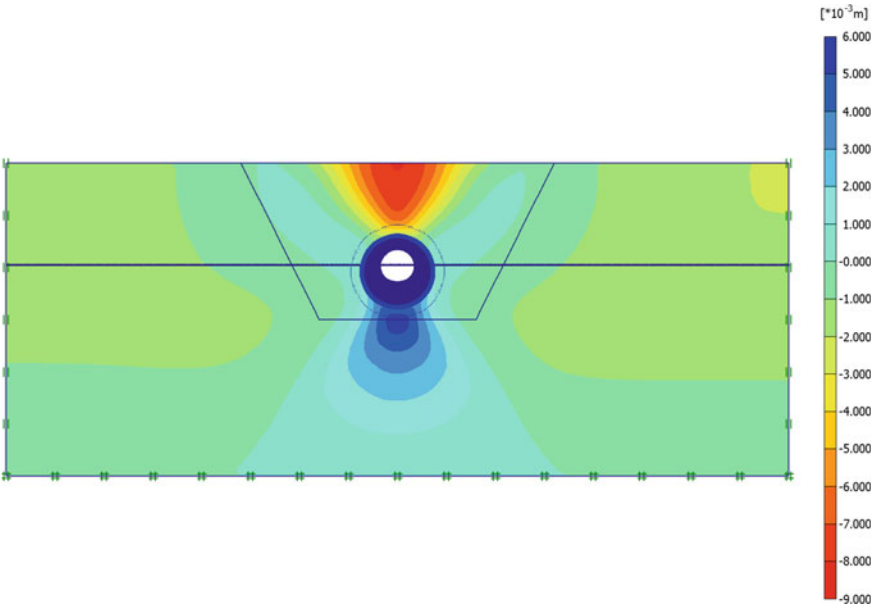


Fig. 2.78 Vertical displacements related to the cross-section located at chainage 1+050 (Hajian et al. 2014)

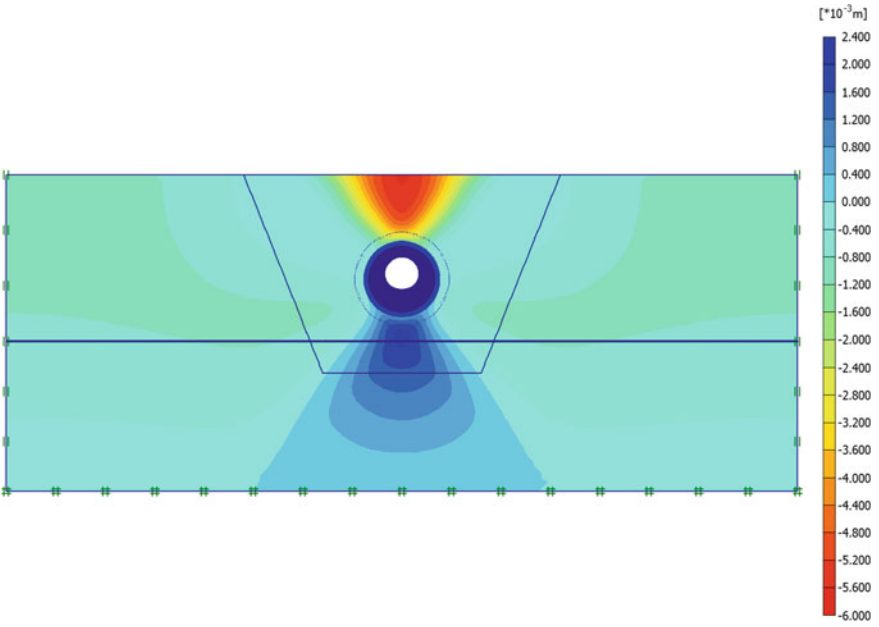


Fig. 2.79 Vertical displacements related to the cross-section located at chainage 2+09 (Hajian et al. 2014)

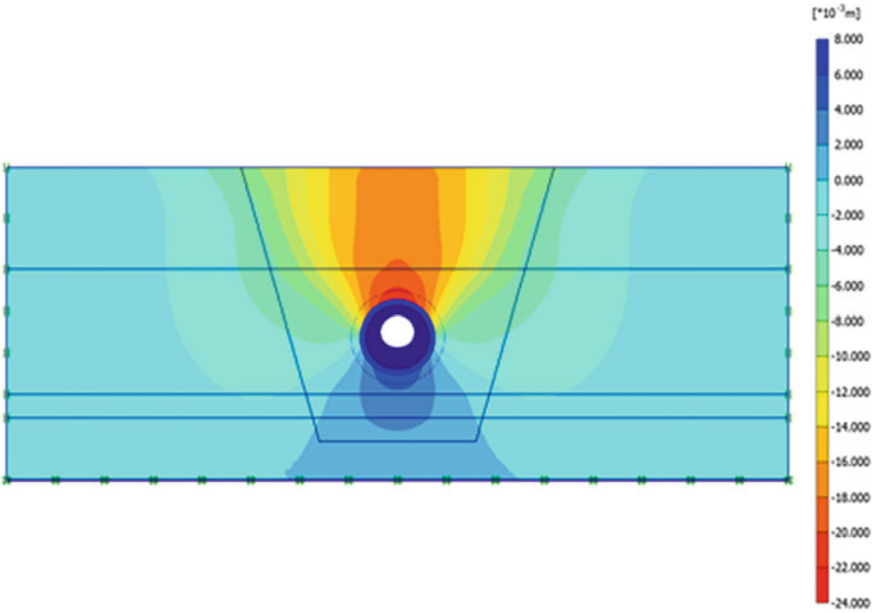


Fig. 2.80 Vertical displacements related to the cross-section located at chainage 13+160 (Hajian et al. 2014)

network is shown in Fig. 2.73. Figure 2.74 illustrates the correlation coefficient existing between the actual data and the predicted.

2.8.9 Surface Settlement Calculation Using FEM

Finite Element Modeling using the 2-dimensional version of Plaxis software was used to calculate the surface settlement which occurred in three cross-sections located at chainages 1+050 (between stations A2 and B2), 2+090 (between stations B2 and C2) and 13+160 (between the south shaft and the station L2).

Table 2.40 The actual amounts of surface settlement and the ones estimated using ANN and FEM (Hajian et al. 2014)

Chainage	Settlement measured by instrumentation, *mm*	Settlement predicted using ANN, *mm*	Settlement calculated using FEM, *mm*
1+050	3.00	3.40	8.05
2+090	5.00	4.60	5.64
13+160	0.00	0.40	0.017

As has been already stated, the tunnel of the Mashhad Subway Line 2 has a diameter of 9.1 m. The geotechnical properties of the soil at chainages 1+050, 2 +090 and 13+160 are presented in Tables 2.37, 2.38 and 2.39 respectively. At these chainages, the depth of the tunnel is 13.85, 13.15 and 21.65 m respectively.

For 2-D modeling in the Plaxis software, the plane strain model was used with 15-node elements;, the materials of soil for all layers were set as hardening soil models. The lining of tunnel is 35 cm thick. An elastic plate model with a modulus of elasticity of 35 million kg/cm^2 is used for the materials of the lining.

The model meshing was first done for the whole model, and then refined three times for particularly critical areas. Figures 2.75, 2.76 and 2.77 show the meshed model of cross-sections located at the specific chainages.

It should be mentioned that, as the middle and upper parts of the model include areas closest to the desired point set for determination of settlement, they are distinguished from the other areas of the model so that finer mesh can be generated in these areas.

After the mesh generation, at the stage of the initial conditions, the level of groundwater is set at the level beneath the tunnel invert. The calculations comprise four phases include tunneling for five days, 1% contraction for half a day, grouting for one day, lining for half a day and consolidation for three days. The desired point for determining of settlement in software is set at the precise location of the planted marker pin on ground surface.

Figures 2.78, 2.79 and 2.80 show the vertical displacements at the mentioned cross-sections. The vertical displacement considered refers to the surface settlement above the central axis of the tunnel.

2.8.10 Results

The surface settlement estimated using ANN and FEM as well as the actual amounts of surface settlement at the three chainages are presented in Table 2.40. The amounts of surface settlement estimated using all mentioned methods are close to the actual ones. The small difference between the actual amounts of settlement and those of the estimates is due to error related to ANN or the calculation errors in the Plaxis software.

2.8.11 Conclusions

Measurement of surface settlement while tunneling is essential to mitigate hazard and/or financial losses. Powerful tools are available for the estimation of this surface settlement. Applying ANNs, which use the numerical data to predict the results, is a powerful tool in this field. While FEM can be an exact numerical method for

surface settlement calculation, ANNs have the potential to predict the results very accurately and rapidly and provide a more powerful tool compared to classical FEM methods. By checking the observed data with the results of ANN, it was deduced that the correlation coefficient between the actual amounts of settlement and the predicted ones is 0.86 for ANN. These analyses show the higher correlation resulting from ANN in comparison with FEM.

Table 2.41 Review of some TBM performance prediction models

Prediction value	Reference	Rock mass factors	Machine factors
Penetration rate (m/h)	Graham (1976)	Uniaxial compressive strength	Cutter force
Penetration rate (m/h)	Farmer and Glossop (1980)	Tensile strength	Cutter force
Penetration rate (m/h)	Büchi (1984)	Compressive and tensile strength	Cutter spacing, cutter tip width, cutter radius, cutter force, TBM diameter, RPM
Penetration rate (m/h)	Hughes (1986)	Uniaxial compressive strength	Cutter force, Fn
Penetration rate (m/h)	CSM model	Uniaxial compressive strength	Cutter spacing, cutter tip width, cutter
Penetration rate (mm/rev)	Gehring(1995)	Uniaxial compressive strength	Cutter force, Fn
Penetration rate (m/h)	NTH Bruland (1998)	Uniaxial compressive strength, drilling rate index (DRI), number of	Cutter force, RPM, cutter spacing
Penetration rate (m/h)	QTBM Barton (2000)	RQD0, Jn, Jr, Ja, Jw, SRF, rock mass strength, cutter life index (CLI), quartz	Cutter force
Penetration rate (m/h)	RME Bieniawski et al. (2006)	Uniaxial compressive strength, abrasivity, rockmass jointing at the	Total cutter head thrust, RPM
Bore ability IndexBI (kN/mm/rev)	Gong and Zhao (2009)	Compressive strength, volumetric joint count, brittleness index, angle between main discontinuities and tunnel axis	Cutter force
Field penetration index: FPI (kN/mm/rev)	Hassanpour et al. (2011)	Uniaxial compressive strength and RQD	Cutter force, RPM

2.9 Comparison of Neural Networks for Predicting the Penetration Rate of Different Models for Tunnel Boring Machines (TBM)

Drilling TBMs are machines which, by applying a driving force and rotating the cutter head with cutter discs, dig a tunnel with uniform and specified circular shape. In spite of their huge investment requirements, their high speed and high quality performance have enabled them to compete with traditional methods of drilling. In addition to their high advance rates and excellent safety performance, they decrease the extent of damaged zones and the other difficulties associated with drilling and blasting. The analysis of TBM performance is the basis for estimating the expenditure and timetable for each tunnel project. One of the methods used for the prediction of the performance of these machines is the estimation of their penetration rate. Different researchers have studied the effect of the properties of intact rock and rock mass on the penetration rate performance of TBM of these machines. The inputs of the models are the uniaxial compression strength, the point load strength index, the penetration force of each disc and the number of cutter head revolutions per minute; and the output is the penetration rate of the machine.

2.9.1 Literature Review of the Prediction of the Penetration Rate of TBM

Tunnels are often excavated by Tunnel Boring Machines TBMs in order to complete the construction of a tunnel in a reasonable timescale, Therefore, the prediction of TBM performance is a primary requirement for planning purposes and the selection of the optimal construction method, Many performance prediction models have been developed, during the last 30 years ranging from single factor models (Graham 1976; Farmer and Glossop 1980; Hughes 1986) to multiple factor models (Büchi 1984; Rostami and Ozdemir 1993; Gehring 1995; Bruland 1998; Barton 2000; Bieniawski et al. 2006; Gong and Zhao 2009; Hassanpour et al. 2011), and are summarized in Table 2.41.

The most widely used models at present include the Colorado School of Mines (CSM) model (Rostami and Ozdemir 1993; Rostami 1997) and the Norwegian Institute of Technology (NTH) model (Bruland 1998). Each of these has advantages and disadvantages. The original CSM model did not consider the influence of joints on TBM penetration rate, while the modified CSM model only considers jointing as a factor reducing rock mass strength. The NTH model requires special laboratory tests, (not commonly available in a standard rock mechanics laboratory). TBMs can generally achieve very good advance rates (up to 150 m/day and 2000 m/month; Barton 2000) but are very sensitive to adverse geological conditions, such as spalling, rock bursting, rock squeezing and high water inflows (Mobarra and Hajian 2013).

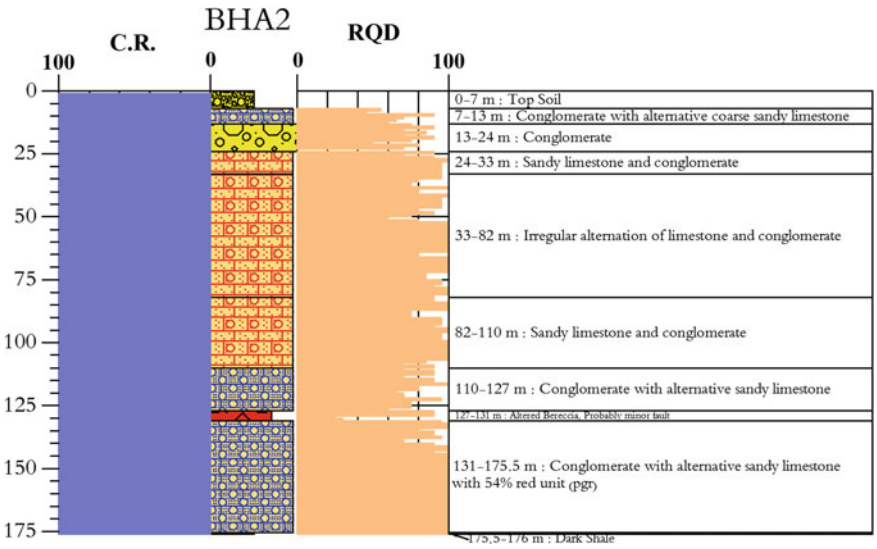


Fig. 2.81 Log data from borehole BHA2 (Mobarra et al. 2013)

2.9.2 Case Study of the Golab Tunnel

The main purpose of the Golab tunnel project was the transfer of water from the Tanzimi dam river to Kashan, along with supplying the Karoun desert with water from the Zayande-Rud River. The area of the project is located to the west of Isfahan within a distance of 100 km and in the north of Chaharmahal-o-Bakhtiari Province. The nearest towns to this project are Tiran and Chadegan. The highest mountain range of this region is the Dadan mountain range with a height of 3890 m and the lowest point of this region lies in the Zayande-Rud valley with a height of 1900 m.

2.9.2.1 Geotechnical Investigation of the Tunnel Route

At the request of the project supervisor, nine geotechnical boreholes were drilled to a total length of 3105 m along the ten kilometer route of the Golab tunnel as well as two other boreholes located at the water entrance to a length of 100 m by the Azemooneh Steel Company. The drilling was carried out with rotary and continuous core drilling, and then borehole and laboratory tests were done by the foregoing company. The purpose of this drilling was to obtain more geotechnical information at the planned depth of the tunnel from boreholes. Figure 2.81 shows a sample of borehole log data (BHA2).

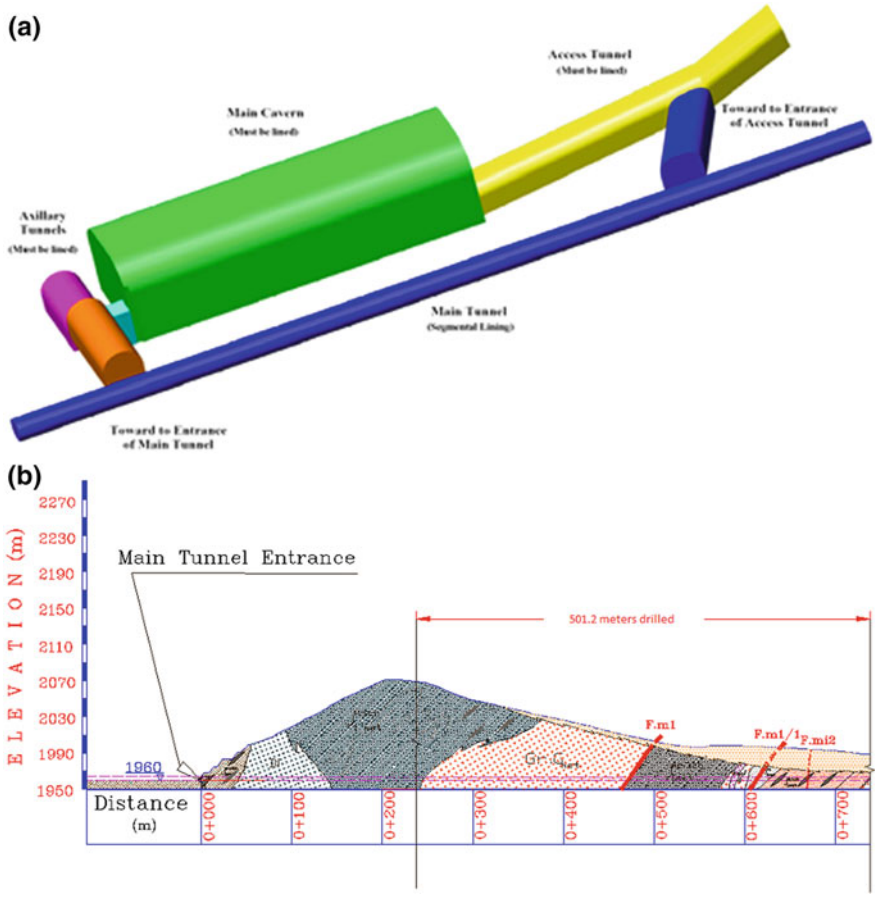


Fig. 2.82 **a** Generalized diagram of tunnels and gouge in the general case **b** A part of the Golab tunnel rout, (Mobarra et al. 2013)

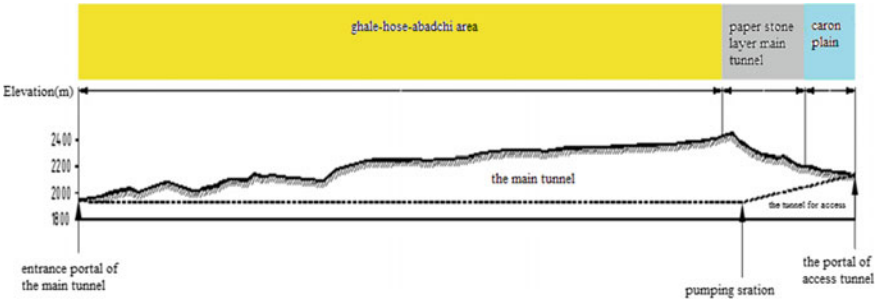


Fig. 2.83 Three morphological regimes of the regions near the Golab tunnel (Mobarra et al. 2013)

Fig. 2.84 TBM with dual shield model TB 453 E/TS

2.9.2.2 Introduction to the Project

Generalized diagram of tunnels and gouge in the general case is depicted in Fig. 2.82a. The water transfer project of Golab includes the following engineering operations:

1. Digging a 9.2 km long tunnel whose slope and inner diameter are 1 in 1000 and 3.5 m respectively.
2. Excavating an underground gouge at the junction of the main tunnel and making the required spaces and their accessibility.
3. Constructing a tunnel 1.5 km long with a slope of 13.5% and laying pipe with 1400 mm diameter inside the tunnel for accessibility
4. Constructing a water outlet structure including water intake structures and a canal as long as 900 m with sediment catcher facilities.

2.9.3 Geomorphology

Isfahan province is located in the center of Iran's plateau and includes different mountainous and plain areas that cover the eastern outskirts of the Zagros Mountains, mountainous valleys and part of the low lands of the east and south–east of Iran .In the west of Isfahan province lie the Zagros Mountains stretching

from north–west to south–east, extending from the western mountains of Golpayegan to the hills of the Dena in Semirom. These hills are sedimentary, but sometimes metamorphic formations can be found. The most important mountains in the western area include Daran, with particular peaks of Dalankouh, Bazmekouh and Tamandarkouh with the highest Shahankouh, with an altitude of 4040 m.

2.9.3.1 The Morphology of the Area

There are three differing geomorphological terrains with differing geological and tectonic characteristics, which are shown in Fig. 2.83.

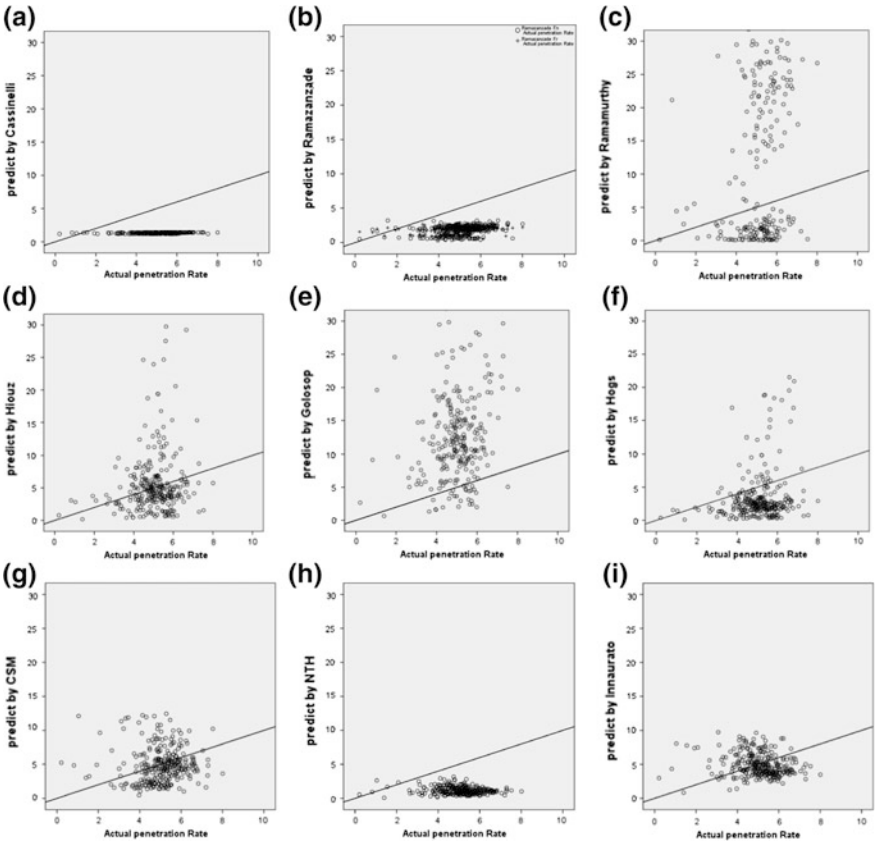


Fig. 2.85 Comparison of actual penetration rates in m/h and the penetration rates obtained from different predictive equations (A-Cassinily, B-Ramazanzade, C-Ramamurthy, D-Hious, E-Glossop, F-Hogs, G-CSM, H-NTH, K-Innaurate)

2.9.4 The TBM Machine Used for the Golab Project

Because of the differing geological conditions and the increasing demand for tunnel making, different kinds of whole-section drilling machines are manufactured all around the world; which can be categorized into different classes, one of which is a double-shield drilling machine. A TBM machine was used to dig the tunnel to transfer water from Golab area. This drilling machine was equipped with two different models of shield known as TB 453 E/ TS. This machine is shown in Fig. 2.84.

2.9.5 Data Collection

In tunnel drilling projects carried out using mechanized systems, the performance of the machine is analyzed via three kinds of information, which are gathered, recorded, estimated and archived:

1. The machine-running parameters (torque, pushing power, consumption capacity, rpm, etc.)
2. Data pertaining to the performance of machine (starting and ending hours of drilling cycle, the delay of the machine, etc.)
3. Sundry combined parameters

Collecting these data in mechanized drilling projects comprises one of the most significant processes in the study of TBM performance. In the Golab tunnel, the drilling parameters were also recorded manually in drilling logs and then their averages were used as daily parameters in drilling reports.

2.9.6 A Static Model for Predicting the Penetration Rate

Many researchers and scientists have presented formulae for the estimation of penetration rate for drilling machines all over the world. In this section we have calculated the penetration rate predicted by the equations presented by different scientists, and compared them with the actual penetration rates of drilling machines used in the Golab tunnel drilling. The correlation coefficients between the actual data and the models have been compared to choose the optimal procedure. Comparisons between these different static models, to the database gathered from the procedures of the Golab tunnel drilling, are presented in Fig. 2.85. Consequently, to determine which of the above models has the best agreement between predicted penetration rates and the actual data; we calculated Pearson's Rank correlation coefficient for different models, with the results (Table 2.40). Table 2.42 shows that Rostami's CSM model gave the best agreement with the

Table 2.42 Pearson's correlation coefficient of the penetration rates predicted by different models and measured penetration rates (Mobarra et al. 2013)

Method	Graham	Golosop	Hogs	Cassinelli	Hiouz	Innaurato	Alber	Palmstrom	Barton	Ramazanzade	Ramazanzade	Ramamurthy	PR
Graham	1.00	0.57	0.99	0.41	0.99	0.45	−0.12	0.10	0.78	0.39	0.10	−0.07	0.19
Golosop	0.57	1.00	0.57	0.50	0.53	0.26	−0.21	−0.02	0.55	0.34	0.27	−0.10	0.13
Hogs	0.99	0.57	1.00	0.39	1.00	0.36	−0.17	0.02	0.80	0.36	0.16	−0.14	0.22
Cassinelli	0.41	0.50	0.39	1.00	0.39	−0.01	−0.56	−0.07	0.50	0.82	0.11	0.34	0.25
Hiouz	0.99	0.53	1.00	0.39	1.00	0.36	−0.17	0.02	0.72	0.32	0.08	−0.10	0.15
Innaurato	0.45	0.26	0.36	−0.01	0.36	1.00	0.63	0.78	0.29	0.10	−0.27	0.25	−0.16
Alber	−0.12	−0.21	−0.17	−0.56	−0.17	0.63	1.00	0.82	−0.12	−0.25	−0.18	0.10	−0.29
Palmstrom	0.10	−0.02	0.02	−0.07	0.02	0.78	0.82	1.00	0.12	0.26	−0.23	0.51	−0.15
Barton	0.78	0.55	0.80	0.50	0.72	0.29	−0.11	0.12	1.00	0.57	0.21	0.02	0.23
RamazanzadeFn	0.39	0.34	0.36	0.82	0.32	0.10	−0.25	0.26	0.57	1.00	0.18	0.48	0.27
RamazanzadeFr	0.10	0.27	0.16	0.11	0.08	−0.27	−0.18	−0.23	0.21	0.18	1.00	−0.45	0.36
Ramamurthy	−0.07	−0.10	−0.14	0.34	−0.10	0.25	0.10	0.51	0.02	0.48	−0.45	1.00	−0.08
PR	0.19	0.13	0.22	0.25	0.15	−0.16	−0.29	−0.15	0.23	0.27	0.36	−0.08	1.00

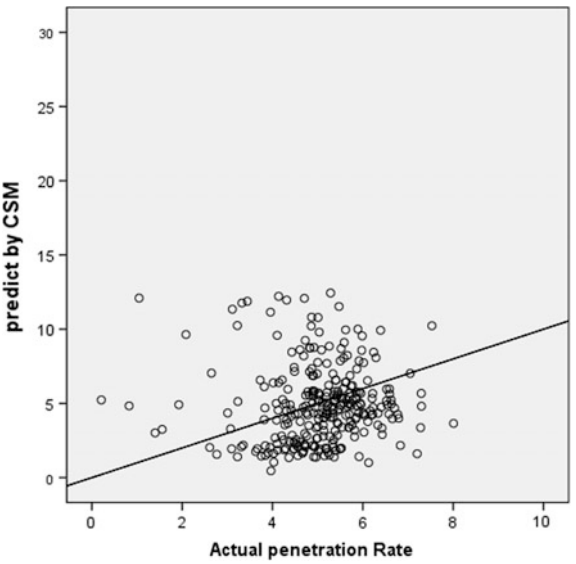


Fig. 2.86 The predicted values vs. actual values of the penetration rate (Mobarra et al. 2013)

measured data and in addition, its penetration rate was closest to actually obtained penetration rates (Figs. 2.85 and 2.86).

2.9.7 Input Parameters

The determination of all the relevant parameters that influence the prediction of penetration rate is difficult but not all of the parameters are independent and some of them are strongly correlated and it is not necessary to use all the variables as input parameters. The parameters which have an important effect on TBM performance and affecting Rate of Penetration (ROP) which were selected for this study were: Uniaxial Compressive Strength (UCS), point load strength index ($I_{S(50)}$), RPM and normal force (f_n). Each parameter Although machine characteristics (e.g., thrust or torque) are very important for overall performance it was assumed here that these characteristics remain unchanged and all possible effects were due to the geotechnical and machine conditions. Data were obtained from the Golab water conveyance tunnel project.

The main purpose of the Golab tunnel project is the transfer of water from the Tanzimi Dam River to Kashan, along with supplying the Karoun desert with water from the Zayande-Rud River. The highest mountain range of this region is the Dadan mountain range with a height of 3890 m and the lowest point of this region belongs to the Zayande-Rud valley with a height of 1900 m above sea level.

Table 2.43 Comparison of some of the models error indexes (Mobarra et al. 2013)

R	Model	T-F	MSE (Train)	MSE (validation)	MSE (Test)	R (Test)	R (All)
1	4-4-1	s-p	$5e^{-5}$	$1.22e^{-5}$	$4.98e^{-3}$	0.111	0.955
2	4-7-1	s-p	$9.81e^{-17}$	$4.615e^{-5}$	0.1	0.131	0.705
3	4-8-1	s-p	$10e^{-4}$	0.0118	$10e^{-4}$	0.147	0.637
4	4-10-1	S-p	$6.7e^{-4}$	0.0146	$9.5e^{-4}$	0.513	0.630
5	4-15-1	s-p	$7.9e^{-11}$	0.00971	0.00971	0.289	0.68
6	4-12-10-1	s-s-p	$10e^{-15}$	0.00197	0.00197	0.84	0.905
7	4-7-4-1	s-s-p	$1.894e^{-5}$	$1.894e^{-5}$	0.00165	0.227	0.954
8	4-13-4-1	s-s-p	$8.451e^{-6}$	0.00338	$1.116e^{-6}$	0.923	0.913
9	4-14-4-1	s-s-p	$1.188e^{-19}$	0.00557	0.00557	0.402	0.723
10	4-15-4-1	s-s-p	0.00770	0.00973	0.00475	0.38	0.45
11	4-8-5-1	s-s-p	0.00017	0.01241	0.00020	0.316	0.530
12	4-13-5-1	s-s-p	0.00046	0.00141	$2.37e^{-5}$	0.297	0.938
13	4-7-7-1	s-s-p	$6.34e^{-21}$	0.00294	$1.71e^{-5}$	0.301	0.911
14	4-15-10-1	s-s-p	0.00044	0.01913	0.00083	0.428	0.313
15	4-17-1	s-p	$9.6e^{-4}$	0.00002	0.00034	0.465	0.809
16	4-6-5-1	s-s-p	0.000197	0.000345	0.000345	0.86	0.90
17	4-12-9-5-1	s-s-s-p	$2.95e^{-9}$	$2.12e^{-5}$	0.00221	0.69	0.94
18	4-12-9	s-s-p	$1.78e^{-13}$	$2.37e^{-5}$	0.00222	0.36	0.94
19	4-4-7-2-1	s-s-s-p	$7.12e^{-11}$	$2.11e^{-5}$	0.02483	−0.08	0.009
20	4-9-3-1	s-s-p	$6.92e^{-6}$	$1.57e^{-5}$	0.00204	0.65	0.94

2.9.7.1 Input Dataset

A critical stage in the ANN technique is data collection. Before training and implementation was carried out, the data were randomly divided into training, validation and test subsets. 289 datasets were collected in the present study and from these, 65% of the data were chosen for training, 15% for validation and the remaining 20% were allotted to the final test phase. The training set was used to generate the model and the validation set was used to check the general applicability capability of the model.

2.9.8 ANN Topology

An appropriate architecture was obtained from feed-forward back propagation. A three-layer network was chosen with logarithmic sigmoid transfer function neurons in the hidden layer, and a purely linear transfer function neuron corresponding to Rate Of Penetration in the output layer. Since there is no exact method of determining the appropriate number of hidden layers, and the number of neurons

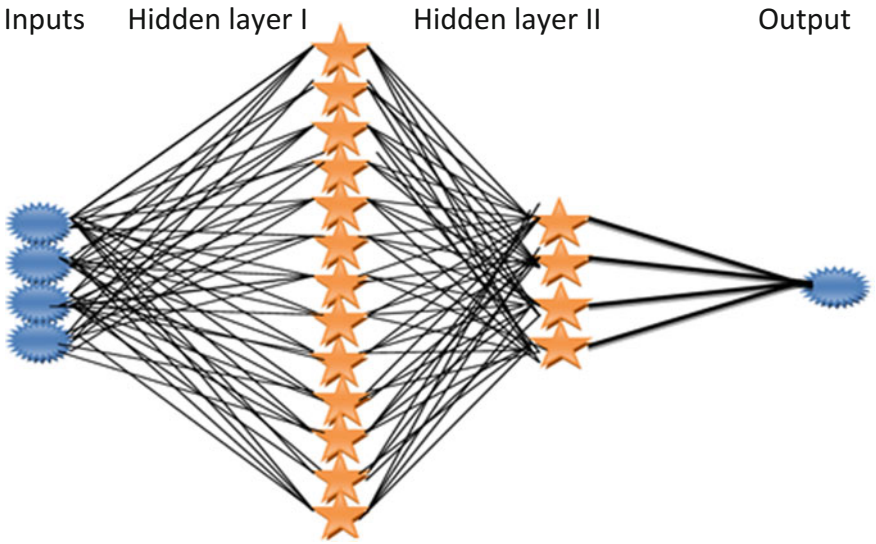


Fig. 2.87 Suggested A.N.N for predicting ROP (Mobarra et al. 2013)

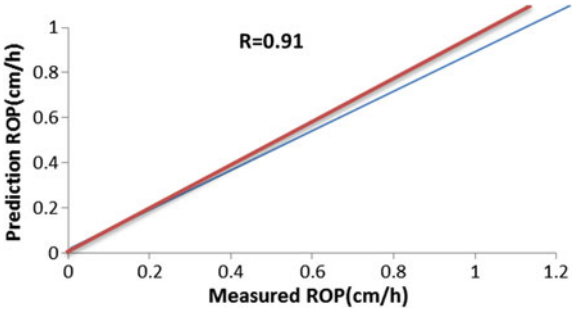


Fig. 2.88 Correlation between the measured and predicted values of ROP (Mobarra et al. 2013)

in each hidden layer, a trial and error procedure is typically used to identify the best network for a particular problem. Several network topologies were examined and the Levenberg-Marquardt algorithm chosen for training the ANNs because it is the fastest method for training moderate-sized feed-forward neural networks.

The resulting target network should produce a minimum error for the training pattern and give a generalized solution that performs well with the testing pattern.

2.9.9 Testing and Validation of the ANN Model

Testing and validation of the ANN model was done with new datasets which were not previously used while training the network. The results here demonstrate the performance of the networks. The Mean Squared Error (MSE) and coefficient of correlation between the predicted and measured values were taken as the performance measures. The MSE was calculated as:

$$\mathrm{MSE} = \frac{1}{Q}\sum_{Q}(d-0)^2 \tag{2.21}$$

where d, o and Q represent the predicted output, the measured output and the number of input-output data pairs, respectively. The prediction was based on the input datasets shown above. The quality of the results obtained for some models is shown in Table 2.43. The correlation coefficient and mean squared errors for the different models are presented there.

The errors suggest that the network with a 4-13-4-1 architecture is optimal. This network is shown in Fig. 2.87.

Figure 2.88 shows a graph comparing measured and predicted data for the preferred ANN model. It appears that the ROP model has predicted values close to the measured; the correlation coefficient between measured and predicted ROP is very high.

2.10 Application of Neural Network Cascade Correlation Algorithm for Picking Seismic First-Breaks

It is possible to perform first break picking using neural networks through training them using the samples of first arrivals to obtain a classification rule by which we can make a distinction between first breaks and non-first breaks.

Song et al. (2011a, b) developed a first-break auto-picking method based on cascade-correlation neural networks. First-break picking was performed earlier using multi-layer perceptron neural networks with the Back propagation (BP) algorithm, as a pattern recognition problem. The motivation to use the cascade correlation (CC) algorithm was some of the limitations in the BP algorithm such as:

- low convergence speed
- non-trivial applications and proneness to getting trapped in local minima on the error surface
- Fixed BP network's topology which can't expand the size of the network to incorporate extra training data into the internal knowledge

The advantages of using the CC learning architecture compared with BP are:

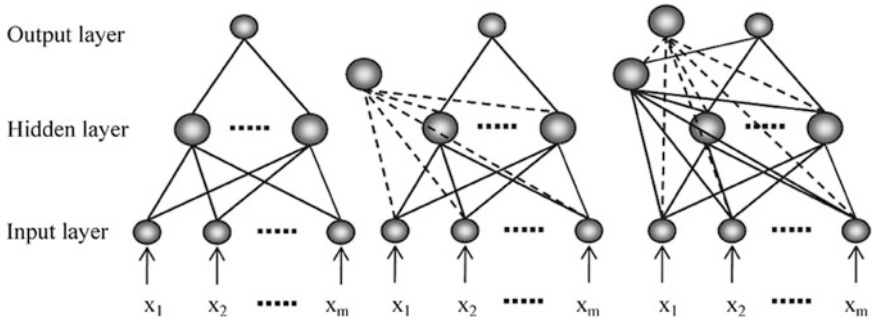


Fig. 2.89 The improved CC learning architecture, initial BP network (left) and after adding two hidden units (right) (redrawn after Song et al. 2011a, b)

- CC learning architecture is a dynamic network which can determine its own size and topology.
- It uses gradient-based weight optimization without the complexity of back-propagation.
- Compared with the BP method, CC learning architecture learns very quickly.
- CC retains the structure it has built even when the training set changes (Fahlman and Lebiere 1990).

The original CC algorithm begins with a minimum network, which has no hidden layers, consequently the initial stage of training is very time consuming. Yang et al. (2006) developed the improved CC algorithm which differs from the original CC algorithm, in that the improved algorithm begins with appropriate network architecture. Song et al. (2011a, b) adopted the idea to improve the CC algorithm for picking first breaks. Furthermore, in order to guarantee the network's high generalization ability, they added a regularization item to the covariance (or correlation) when training candidate units in the manner of Wu and Nakayama (1997). It is necessary to mention that this method not only effectively avoids ill-weight growth and improves generalization (Wu and Nakayama 1997), but also accelerates the convergence of the neural network.

To avoid increasing the errors in first break picking using neural networks it is very important to recognize the error sources and prevent or attenuate them. Errors in classifying first breaks arise from two sources (Hart 1996):

- The neural network input data may contain inherent statistical ambiguities
- The neural network approximation of the statistical classification rule may not be accurate

To classify the first breaks and non-first breaks with a neural network, appropriate attributes must be selected as inputs to the network. The attributes should have two main properties:

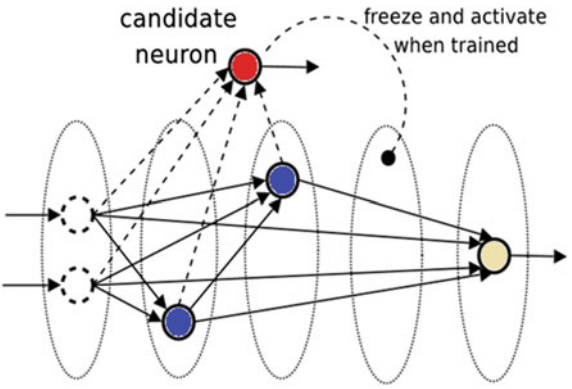


Fig. 2.90 The schematic of cascade correlation algorithm for candidate neuron and freezing before training

- The selected attributes should have high stability and reliability in order to eliminate statistical ambiguities of input data.
- The combination of the selected attributes must be able to make clear distinction between first breaks and non-first breaks.

2.10.1 The Improvement of CC Algorithm

The CC algorithm was proposed by Fahlman (1989) and Fahlman and Lebiere in (1990) and provides a way to automatically adjust the neural network architecture. The CC algorithm begins with minimal network architecture without hidden units, then automatically trains and adds new hidden units one by one until the performance of the network is satisfactory.

Four main steps are performed in the CC algorithm:

Step1: Begin with no hidden-layer neuron
Step2: Add hidden neurons one at a time
Step3: After adding hidden neuron H_i, optimize the weights from "upstream" neurons to H_I to maximize the effect of H_i on the outputs
Step4: Optimize the output weights of H_i to minimize training error.

The theory of the improved CC algorithm is the same as the original CC algorithm mentioned above, but starts from an appropriate BP network (see Fig. 2.83). The initial BP network has only three layers, and the number of hidden units is calculated from Eq. 2.63 (Fig. 2.89):

$$N = 2^{m-1}/(m+1) \tag{2.63}$$

The training process of the improved algorithm includes three main stages:

Stage 1: Initial network training
Stage 2: candidate unit training

Stage 3: output-layer unit training.

The detailed algorithm to perform the mentioned three main stages is as below:

- Step 1: Initial all connection weights with a random value between −1 and +1.
- Step 2: Use the Quickprop to adjust the connection weights*.
  * the Quickprop algorithm acts essentially like the delta rule, except that it converges much faster (Fahlman 1989).
- Step 3: Is there any significant error reduction after a certain number of training cycles? If No: continue until get to this condition. If Yes: Calculate the performance of the network
- Step 4: Is providing the desired performance? If yes then stop training; if no, begin the candidate units training and freeze the input weights of hidden units (Fig. 2.90).
- Step 5: Candidate training stage: train a pool of candidate units with a different set of random initial weights.
- Stage 6: Add the candidate whose correlation score is the best to the network as a new hidden unit.

Fig. 2.91 The situation map of Yenan

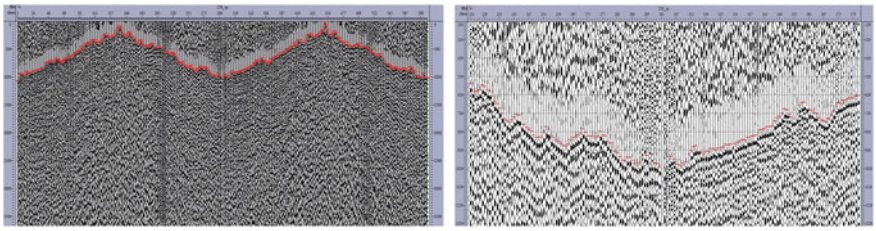


Fig. 2.92 FBP with neural network used in the Yenan loess tableland, on the right is an enlarged part of the figure (Song et al. 2011a, b)

The candidate unit's input connects all of the network's input units and all pre-existing hidden units, but its output is not yet connected to the active network and will be connected in the output training stage (see Fig. 2.90).

In order to avoid weight-ill growth when training candidate units, Fahlman (1989) proposed to add Gauss regularization to the correlation:

$$C = \sum_{o} \sum_{p} (V_p - \bar{V})(E_{p,o} - \overline{E_o}) - \frac{\beta}{2} \sum w_i^2 \tag{2.22}$$

where β is a regularization coefficient and w_i is the incoming weight of the candidate unit.

Step 7: Is the correlation C increasing? If yes go to step 6, if no then apply the best one to the active network as a new hidden unit, and freeze its incoming weights. Then delete other candidate units, and begin output-layer units training step (8).

Step 8: output-layer units training:

- Retrain all the weights to output layer units using the Quickprop algorithm, including the new hidden units.
- If no significant error reduction has occurred after a certain number of training cycles, stop training and begin a new round of candidate unit training.

It's very important to note that throughout the whole training process, the candidate units training stage and output layer training stage repeat alternately, until an overall stopping criterion is satisfied.

2.10.2 Attribute Extraction for Neural Network Training

Selection of attributes is one of the important aspects which affects the accuracy of first break picking. Song et al. (2011a, b) proposed five attributes which were used in their method:

- Power ratio
- Maximum amplitude of the peak

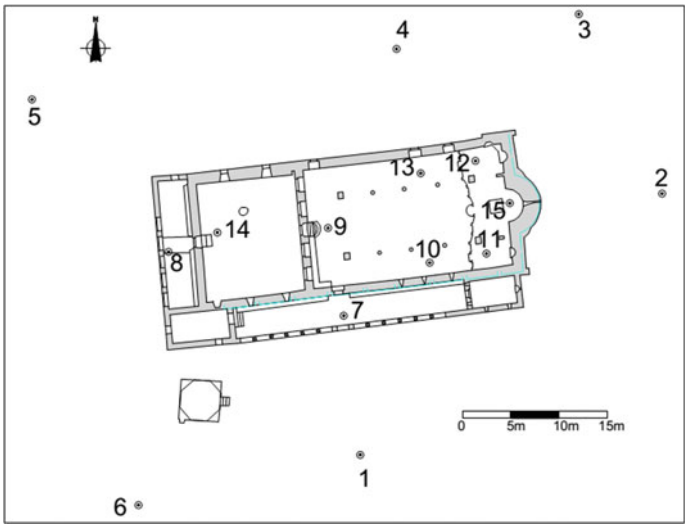


Fig. 2.93 The geodetic network used for church vertical displacement (Pantazis and Alevizakou 2013)

- Frequency
- Curve length ratio
- Adjacent seismic channel correlation.

Curve length is defined as the line integral of the seismic wave over a time window. Its value will change dramatically when the first break arrives. Correlation of adjacent seismic channels is recommended for detection of the first arrival of the event according to their similarity. The inputs to network for training are these five attributes and the output is '1' for first break and '0' for non-first breaks.

Song et al. (2011a, b) applied the CC neural network method in Western China to field test data. The seismic data for testing were from the Loess tableland of Yenan (Fig. 2.91), which has extremely complex near-surface conditions, and the seismic signal to noise ratio is also low for first-break picking. The area is also covered by sand dunes and gravels.

As shown in Fig. 2.92, the Earth's surface is undulating and the results obtained by First-Break Picking are not ideal when using traditional methods. But the

method discussed above achieves approximately 99% accuracy after being trained with a training data set of 400 traces. It only installs 8 new hidden units (initial net has 4 hidden units) and takes 56 s on average to finish the training, but the original CC algorithm would have installed 17 or more, so this is a significant improvement over the original.

Table 2.44 The vertical displacements (ΔH_i) of the 15 control points of the network—Data (Pantazis and Alevizakou 2013)

Code	Coordinates			2009			2010			2011			2012		
				June–July	July–Aug	Aug–Sept	June–July	July–Aug	Aug–Sept	June–July	July–Aug	Aug–Sept	June–July	July–Aug	Aug–Sept
	X (m)	Y (m)	H (m)	ΔH_i (mm)	ΔH_i (mm)	ΔH_i (mm)	ΔH_i (mm)	ΔH_i (mm)	ΔH_i (mm)	ΔH_i (mm)	ΔH_i (mm)	ΔH_i (mm)	ΔH_i (mm)	ΔH_i (mm)	ΔH_i (mm)
1	100.000	100.000	10.000	1.8	−2.9	10.4	2.0	−3.1	10.6	1.4	−2.5	10.0	1.9	−3.0	10.5
2	131.357	127.094	9.339	0.0	0.0	0.0	0.2	−0.2	0.2	−0.4	−0.4	−0.4	0.1	0.1	0.1
3	122.704	145.690	10.180	−1.6	−7.0	6.7	−1.8	−7.2	6.9	−1.2	−6.6	6.3	−1.7	−7.1	6.8
4	103.775	142.079	11.511	1.5	−1.5	−4.9	1.7	−1.7	−5.1	1.1	−1.1	−4.5	1.6	−1.6	−5.0
5	65.910	136.834	15.538	−1.8	0.1	−3.7	−2.0	0.3	−3.9	−1.4	−0.3	−3.3	−1.9	0.2	−3.8
6	76.974	94.788	11.922	−2.2	−3.2	−4.9	−2.4	−1.4	−5.1	−1.8	−2.8	−4.5	−2.1	−1.3	−5.0
7	98.281	114.420	10.201	−0.3	5.0	−9.3	−0.5	5.2	−9.5	0.1	4.6	−8.9	−0.4	5.1	−9.4
8	80.091	121.045	11.812	−1.9	1.6	−2.6	−2.1	1.8	−2.8	−1.5	1.2	−2.2	−2.0	1.7	−2.7
9	96.648	123.524	9.972	−3.1	2.2	−2.7	−3.3	2.4	−2.9	−2.7	1.8	−2.3	−3.2	2.3	−2.8
10	107.197	119.925	9.915	−3.4	2.4	−3.2	−3.6	2.6	−3.4	−3.0	2.0	−2.8	−3.5	2.5	−3.3
11	113.077	120.872	10.071	−2.3	1.0	−4.0	−2.5	1.2	−4.2	−1.9	0.6	−3.6	−2.4	1.1	−4.1
12	111.948	130.470	10.041	−5.2	0.1	−1.6	−5.4	0.3	−1.8	−4.8	−0.3	−1.2	−5.3	0.2	−1.7
13	106.270	129.193	9.915	−2.5	1.6	−3.4	−2.7	1.8	−3.6	−2.1	1.2	−3.0	−2.6	1.7	−3.5
14	85.150	123.043	11.164	0.9	−0.1	−4.5	1.1	−0.3	−4.7	0.5	0.3	−4.1	1.0	−0.2	−4.6
15	111.521	126.089	10.020	−1.9	0.9	−4.0	−2.1	1.1	−4.2	−1.5	0.5	−3.6	−2.0	1.0	−4.1

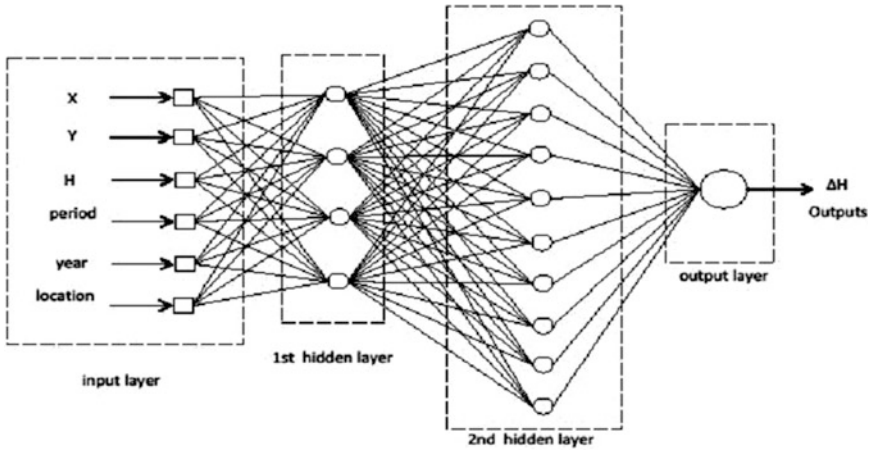


Fig. 2.94 Architecture of the ANN for predicting vertical displacements (Pantazis and Alevizakou 2013)

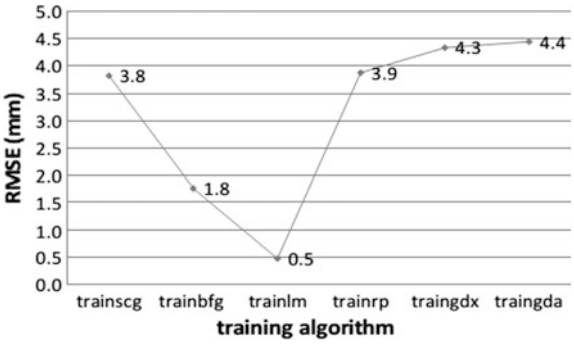


Fig. 2.95 RMSE variation for different training algorithms (Pantazis and Alevizakou 2013)

2.11 Application of Neural Networks to Engineering Geodesy: Predicting the Vertical Displacement of Structures

Pantazis and Alevizakou (2013) used the geodetic method for monitoring a church structure to predict its future position through training an ANN. The trained ANN was used to predict vertical displacement, with the ultimate aim to prevent it failing. The geodetic network they used had 15 central points (Fig. 2.93) and the results of twelve series of geodetic measurements and adjustments to this network were used.

Six of the central points were sited around the church and nine more station points were installed inside the church. The measurements were carried out on this

Table 2.45 Actual and desired outputs from the ANN's (Pantazis and Alevizakou 2013)

i	Actual output (y_i) (mm)	Desired output (d_i) (mm)	i	Actual output (y_i) (mm)	Desired output (d_i) (mm)	i	Actual output (y_i) (mm)	Desired output (d_i) (mm)	i	Actual output (y_i) (mm)	Desired output (d_i) (mm)
1	−4.4	−2.3	10	5.4	5.1	19	−1.6	−1.6	28	−4.0	−4.1
2	−3.3	−3.5	11	10.3	10.5	20	−4.7	−5	29	0.2	0.2
3	−1.6	−1.9	12	−9.2	−9.4	21	0.3	0.2	30	−1.5	−1.7
4	−9.3	−8.9	13	0.0	0.1	22	−3.6	−3.8	31	1.6	1.7
5	−2.5	−2.4	14	−2.9	−2.7	23	−3.2	−3.3	32	−3.9	−3.5
6	−4.6	−5.3	15	−7.2	−7.1	24	−4.9	−5	33	−0.3	−0.2
7	−4.2	−3	16	6.9	6.8	25	2.9	2.5	34	−4.1	−4.6
8	1.6	1.7	17	2.2	2.3	26	−3.5	−3.3	35	0.8	1
9	−0.5	0.1	18	−2.8	−2.8	27	1.5	1.1	36	−4.1	−4.1

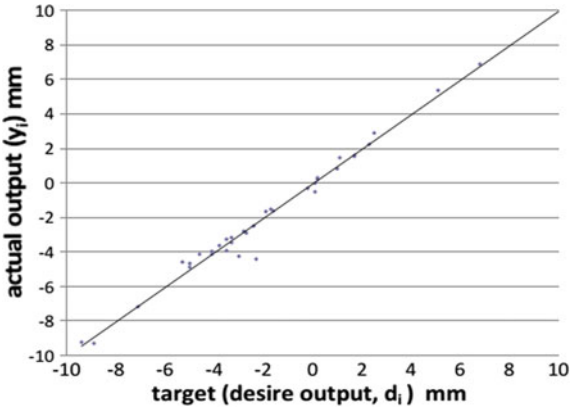


Fig. 2.96 Correlation of the outputs and targets of the final ANN (Pantazis and Alevizakou 2013)

network at regular time intervals. The height differences between the points were measured using leveling in order to determine the height of each point with an uncertainty of ±0.2 mm to ±1 mm. The geodetic network was adjusted through the least squares method. The vertical displacements (ΔH_i) of all the 15 points of the network were calculated for a time period of June to July (1st period), July–August (2nd period) August–September(3rd period) of the years 2009, 2010, 2011, 2012 (Table 2.44).

Pantazis and Alevizakou (2013) used 180 (12*3*5) data items to develop the ANN because 12 vertical displacement values were available for each of the points (three periods per year). They divided the data into 3-subsets: 60% (108 points) as a training set, 20% as a validation set (36 points) and 20% as a test set (36 points).

The inputs of the network were the six parameters below:

- The coordinates X, Y, H of each point
- The period during which the displacement was observed
- The year in which displacement took place
- The location of the point, i.e. inside or outside of the church.

The output was the resulting vertical displacement (ΔH) of each point (in mm).

The optimized structure for their multi-layer perceptron had a 6 × 4 × 10 × 1 architecture which means the network consists of 6 inputs, two hidden layers with 4 and 10 hidden neurons respectively and one output, illustrated in Fig. 2.94.

The mean sequence error (MSE), root mean square error (RMSE) and correlation coefficient (R) of the test set were used to evaluate and select the best ANN. Also to select the best training algorithm amongst **trainscg**[1]**, trainbfg**[2]**, trainlm**[3]**, trainrp**[4]**,**

[1]Scaled conjugate gradient back propagation.

[2]FGS quasi-newton.

[3]Levenberg-Marquardt back propagation.

[4]Resilient back propagation.

traingdx[5] and **traingda**[6] the RMSE variation for different training algorithms were calculated and plotted, and based on this **trainlm** was selected as the best training algorithm (Fig. 2.95). To clarify these subroutines it's necessary to mention that **trainlm** is a network training function that updates weight and bias values according to Levenberg-Marquardt optimization and **traingdx** is a network training function that updates weight and bias values according to gradient descent momentum and an adaptive learning rate backpropagation while **traingda** is Gradient descent with adaptive learning backpropagation. An adaptive learning rate requires some changes in the training procedure used by **traingd**. First, the initial network output and error are calculated. At each epoch new weights and biases are calculated using the current learning rate. New outputs and errors are then calculated. In appendix 2 of this chapter a brief list of the training functions in NN toolbox of Matlab is presented, as we are limited to the volume of this chapter more description of the details of the procedures we refer the reader to NN toolbox tutorial in Matlab help.

The $6 \times 4 \times 10 \times 1$ ANN with the "trainlm" algorithm was used to estimate the vertical displacement of the points of the network and the results that they achieved using this ANN are shown in Table 2.45.

The results of Pantazis and Alevizakou (2013) showed that ANNs can be used to successfully predict vertical displacement using measurements obtained from geodetic displacement control points (Fig. 2.96).

Additionally a linear regression was preformed between the ANNS actual output and the target to determine their degree of correspondence and the correlation coefficient (R) between the actual and desired output was found to be 0.993 which implies a very high correlation.

2.12 Attenuation of Random Seismic Noise Using Neural Networks and Wavelet Package Analysis

Here, we present a method for random background noise suppression to enhance the resolution of seismic data, using automatic training of a Feed Forward Back Propagation (FFBP) Artificial Neural Network (ANN) in a multi scale domain obtained from Wavelet Packet Analysis (WPA). We calculate approximate input seismic sections which are used to train the neural network to model coherent events through an automatic algorithm. After coherent events are modeled, the remaining data are assumed to be background random noise and this is applied to both synthetic and real seismic data. The results are then compared with the Adaptive Wiener Filter (AWF) for synthetic shot gathers and real common-midpoint gathers, and also with band-pass filtering on real common-offset-gathers indicating substantially higher

[5]Adaptive learning rate back propagation.

[6]Gradient descent with adaptive learning back propagation

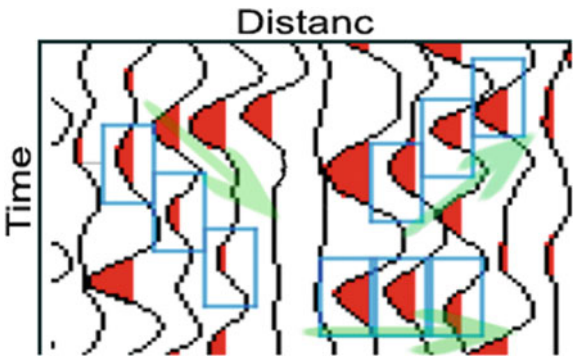


Fig. 2.97 The three routes (Blue Boxes) considered for tracing events in a three by three neighborhood (Kimiaefar et al. 2016)

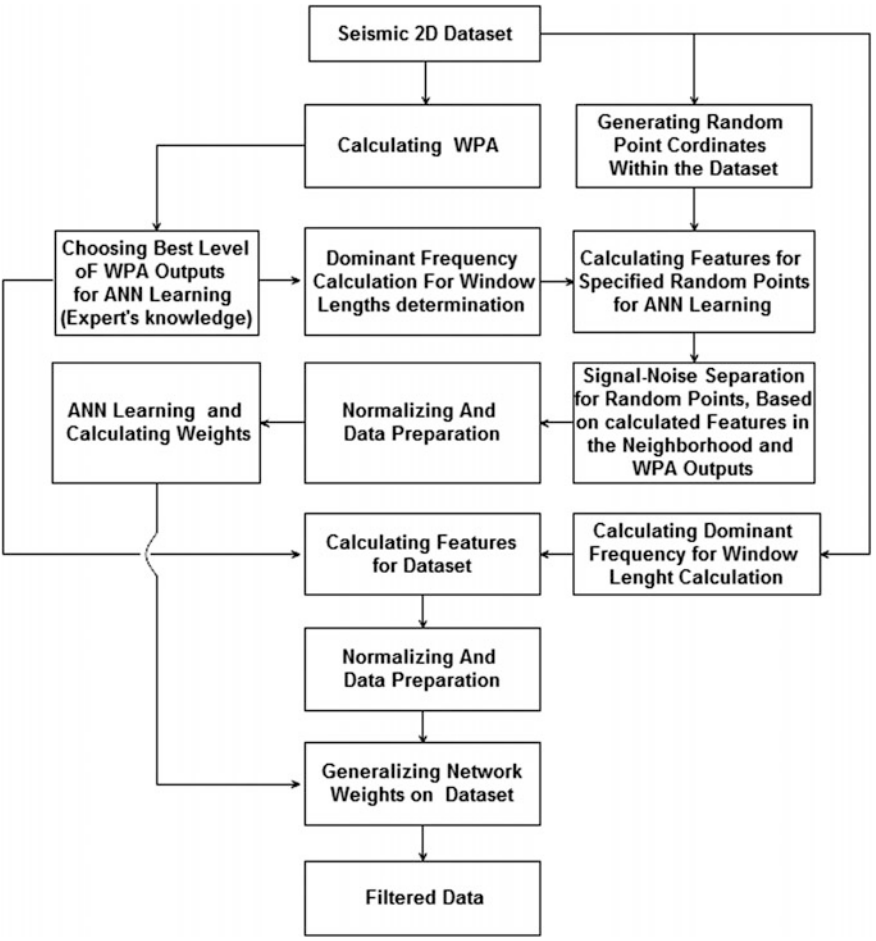


Fig. 2.98 Method used for attenuating random noise by ANN (Kimiaefar et al. 2016)

efficiency of this proposed method in attenuating random noise and enhancing seismic signals. The Wiener filter used here is proposed by Lim in 1990. First, the local mean, μ, and local variance, σ^2, around each point, $a(n_1, n_2)$, are calculated in an N_1 by N_2 local neighborhood, η, as (Lim 1990):

$$\begin{aligned} \mu &= \frac{1}{N_2 N_2} \sum_{n_1, n_2 \in \eta} a(n_1, n_2) \\ \sigma^2 &= \frac{1}{N_2 N_2} \sum_{n_1, n_2 \in \eta} a^2(n_1, n_2) - \mu^2 \end{aligned} \tag{2.23}$$

Since the noise variance (σ^2 in Eq. 2.23) is not usually available, the average of all local estimated variances is used and with this estimate, the AWF will be calculated from:

$$W(n_1, n_2) = \mu + \frac{\sigma^2 - \nu^2}{\sigma^2}(a(n_1, n_2) - \mu) \tag{2.24}$$

Considering low noise level case (i.e. noise variance will be small), AWF is nearly equivalent to the input value. And for a noise free input, the Eq. 2.24 become as (Abe and Shimamura 2012):

$$W(n_1, n_2) = \mu + \frac{\sigma^2}{\sigma^2}(a(n_1, n_2) - \mu) = a(n_1, n_2) \tag{2.25}$$

Equation 2.25 confirms that, for noise free points, output of AWF is not dependent to window size.

2.12.1 Methodology

Earth's filtering properties affect the energy of seismic waves, for long travel times and at far offsets. Therefore, the signal energy may be comparable to background

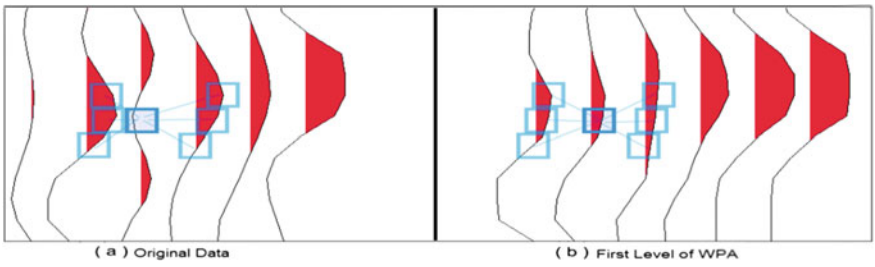


Fig. 2.99 Qualitative illustration of noise-signal separation in the proposed method (Kimiaefar et al. 2016)

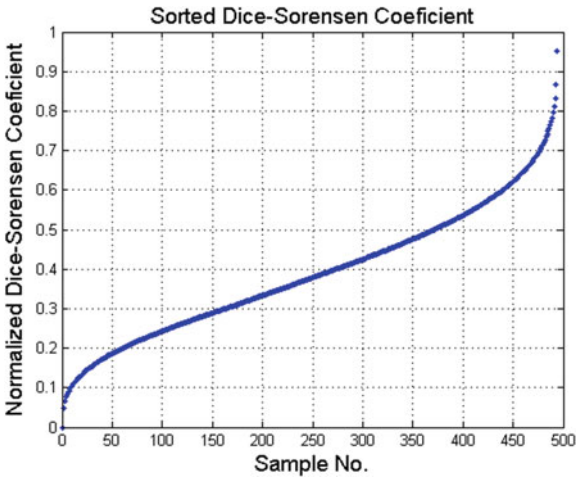


Fig. 2.100 Normalized and sorted DSSC values for 500 random points (Kimiaefar et al. 2016)

random noise. Therefore, separating noise from seismic events through mathematical modeling is difficult due to the lack of a clear phase relation between the coherent events on adjacent traces and is the biggest challenge in the application of ANN for attenuating random noise. ANN training is based on defining and

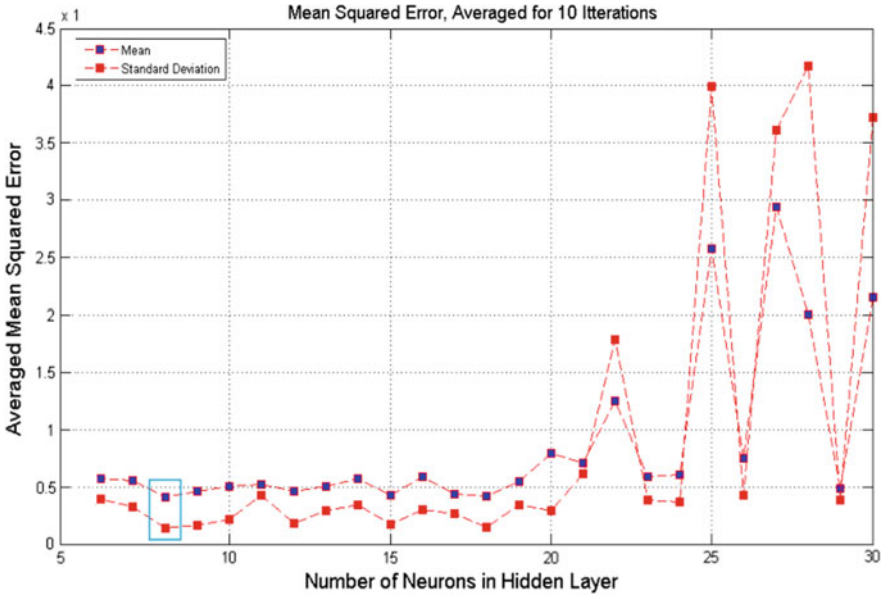


Fig. 2.101 Plot of averaged MSE with ANN performance over 10 iterations for 6–30 Neurons in hidden layer (Kimiaefar et al. 2016)

calculating features and solving problems using prior knowledge. However, fitting a local model appears almost impossible in the situation described here.

The similarity of coherent seismic events between adjacent traces could be the key solution for distinguishing coherent from non-coherent components. As an example, if the median, mean, extremum, or center of gravity of a window with specified length is calculated in the neighborhood of each sample, similar values might indicate that the sample belongs to the coherent event space, however, as Fig. 2.97 show, a seismic event could have three possible correlations. It is clear that at this stage, the expert's knowledge of appropriate features and the relationships between them is essential for training an ANN and that, this tracing could be done in either time or time-frequency domains. Figure 2.98 presents the general workflow for this method.

In order to utilize expert knowledge, the following steps are taken. The first few levels of WPA are calculated and one of the approximations chosen by the expert for ANN. Next, the dominant frequency is calculated using the approximation data and the Gabor transform, (the adaptive peak-to-trough intervals could be used as an alternative solution). Feature extraction is now possible using the extremum of each window around a central point and its 3×3 neighbors for the three specified routes and for each route, the standard deviation of the feature values is calculated and that the one with minimum standard deviation is selected. By using the Dice-Sorensen Similarity Coefficient, DSSC, (Roux and Rouanet 2006) we can calculate and compare the similarity between the two standard deviations and this normalized value considered as a signal-noise separation indicator for each point (coherent events).

The ANN input data is all features calculated for a point around a central point. For a 'noise indicated point', the output is a weighted combination of all WPA approximations from level one to the selected level of approximation and for a 'signal indicated point', the output is the amplitude of the point in the original signal. A diagram of this procedure is presented in Fig. 2.99, a central point and its neighbors in the original section are compared to their first level of WPA. The feature values (e.g. extremum in any of the blue windows in Fig. 2.99) in the original section are different for the three mentioned routes so that the standard deviation is large, but in the WPA approximation image, the DSSC will produce a number close to one.

Table 2.46 Network specifications used for filtering random noise

Type of ANN	Feed forward back propagation
Number of neurons in hidden layer	8 Neurons
Activation function	Sigmoid
Training function	Levenberg-Marquardt
Performance function	Mean squared error
Training points	500 points (randomly chosen)

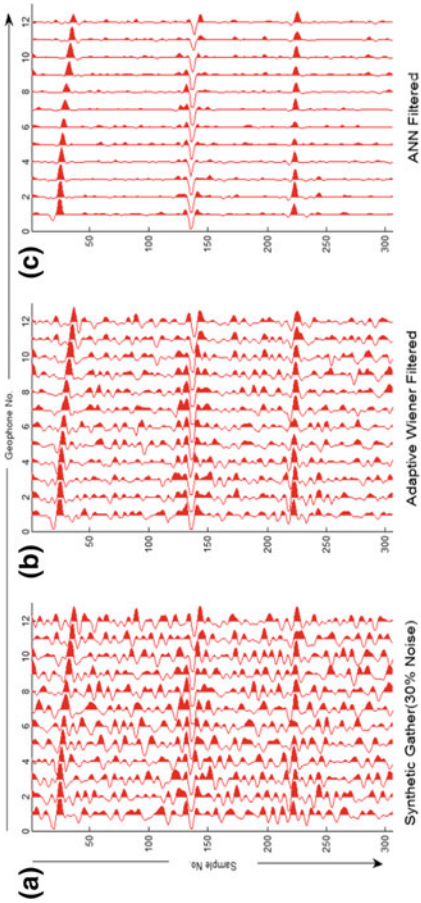


Fig. 2.102 **a** Synthetic common offset sorted gather with 30% additive Gaussian noise, **b** result of AWF, **c** result of ANN filtering (Kimiaefar et al. 2016)

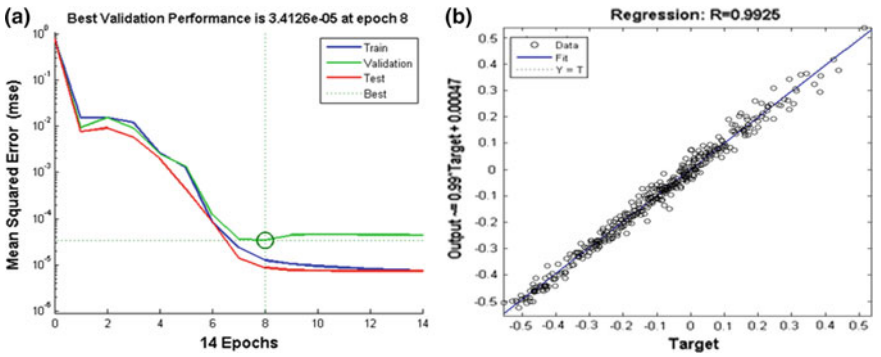


Fig. 2.103 **a** Mean squared error as indicator of network's performance, **b** linear

The normalized DSSC values for 500 randomly chosen points are sorted from zero to one (Fig. 2.100). Small values (e.g. smaller than 0.2 or the first 50 points) correspond to the noise while higher probabilities and higher values (e.g. greater than 0.6 or the last 50 points) represent points correspond to the signal.

Based on the DSSC value and this logic, input data are chosen. The first and last 50 points are considered as ANN training points and then the architecture of the ANN must be determined. Since we are using FFBP, we must decide the optimal number of neurons (or nodes) in the hidden layer. Weak model discrimination may result from too small a number of neurons, while a large number of neurons over-trains the system. The optimum number of nodes was determined using a modification of the method of Hajian et al. (2012). Because random weights are allocated during the training procedure of an ANN, the performance may differ even for the same inputs, outputs, and network architecture. Hajian et al. (2012) used the Mean Squared Error (MSE) between output and target as a performance indicator for the trained network, and averaged these performance values over 10 iterations for each number of nodes in the hidden layer. In addition to calculating averages, they used the standard deviation of values for each neuron quantity to determine the optimum value and the results are plotted in Fig. 2.101 suggesting it is optimal to use eight nodes in the hidden layer. We note that the training procedure becomes unstable with 21 or more nodes, as the MSE and standard deviation values never achieve a steady state.

After ANN training, the procedure is applied to the whole dataset and finally, the filtered section is the output of applying the generalized network weights to all data.

2.12.2 Experimental Philosophy

MLP neural networks with more than one hidden layer are generally implemented when using one hidden layer has already failed. But in this research, as one hidden

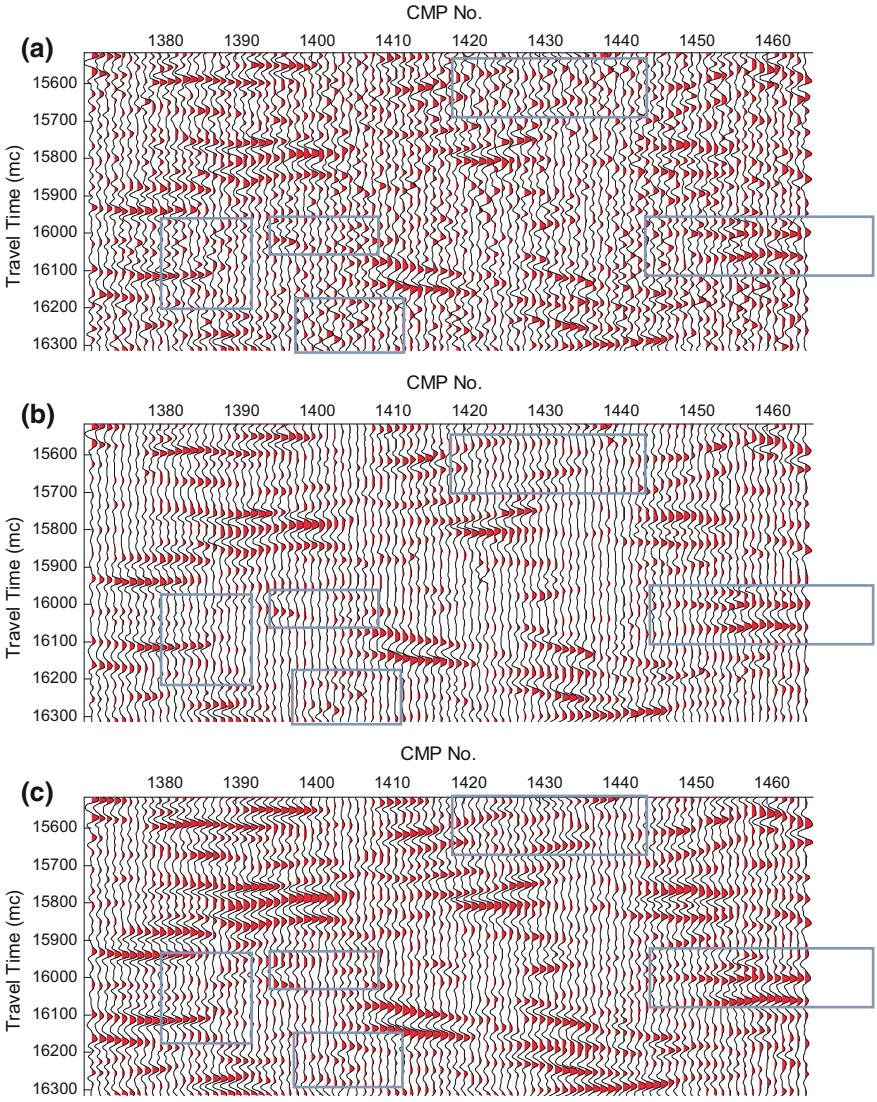


Fig. 2.104 **a** Real stacked data recorded in Australia, **b** result of AWF and **c** result of ANN filtering (Kimiaefar et al. 2016)

layered MLP network, produces the desired performance, we do not need to use an MLP with more than one hidden layer. A network with the specifications mentioned in Table 2.46 was applied to a synthetic common-offset-sorted gather 4 ms sampling interval, 25 m geophone spacing and 35 Hz. central frequency and Ricker wavelet as source impulse) with 30% random Gaussian noise added (with zero mean and variance equal to one).

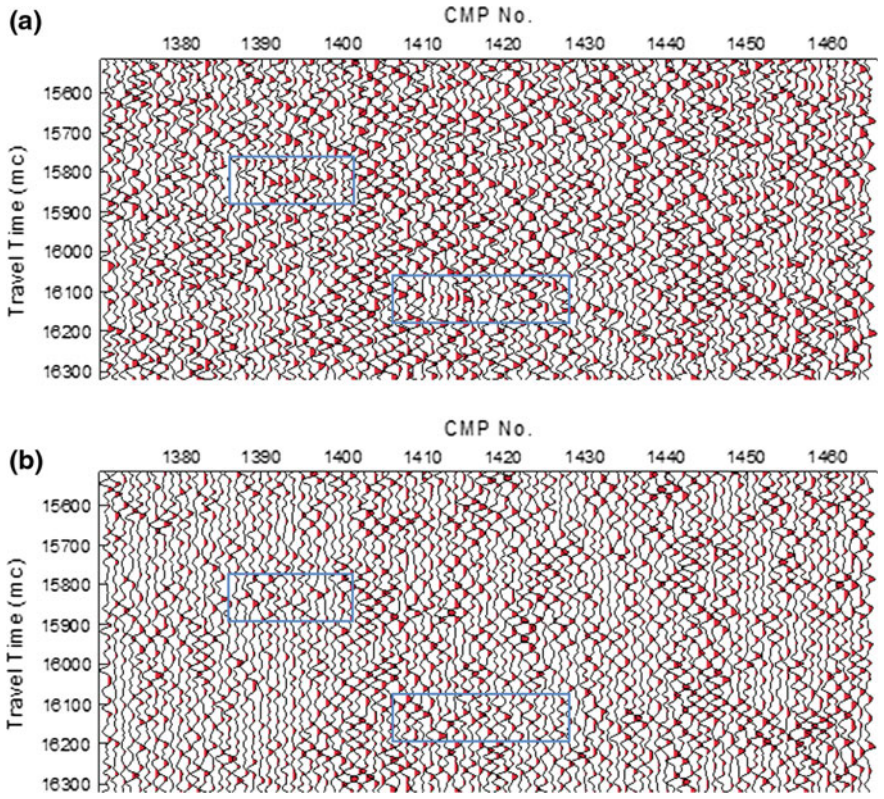


Fig. 2.105 The difference between the original data and denoising result based on **a** AWF and **b** ANN filtering (Kimiaefar et al. 2016)

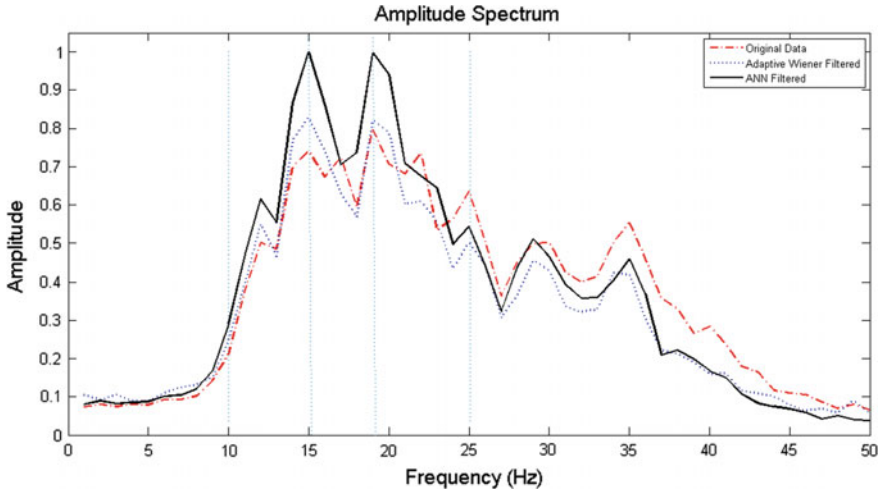


Fig. 2.106 Amplitude spectra of the sections illustrated in Fig. 2.104 (Kimiaefar et al. 2016)

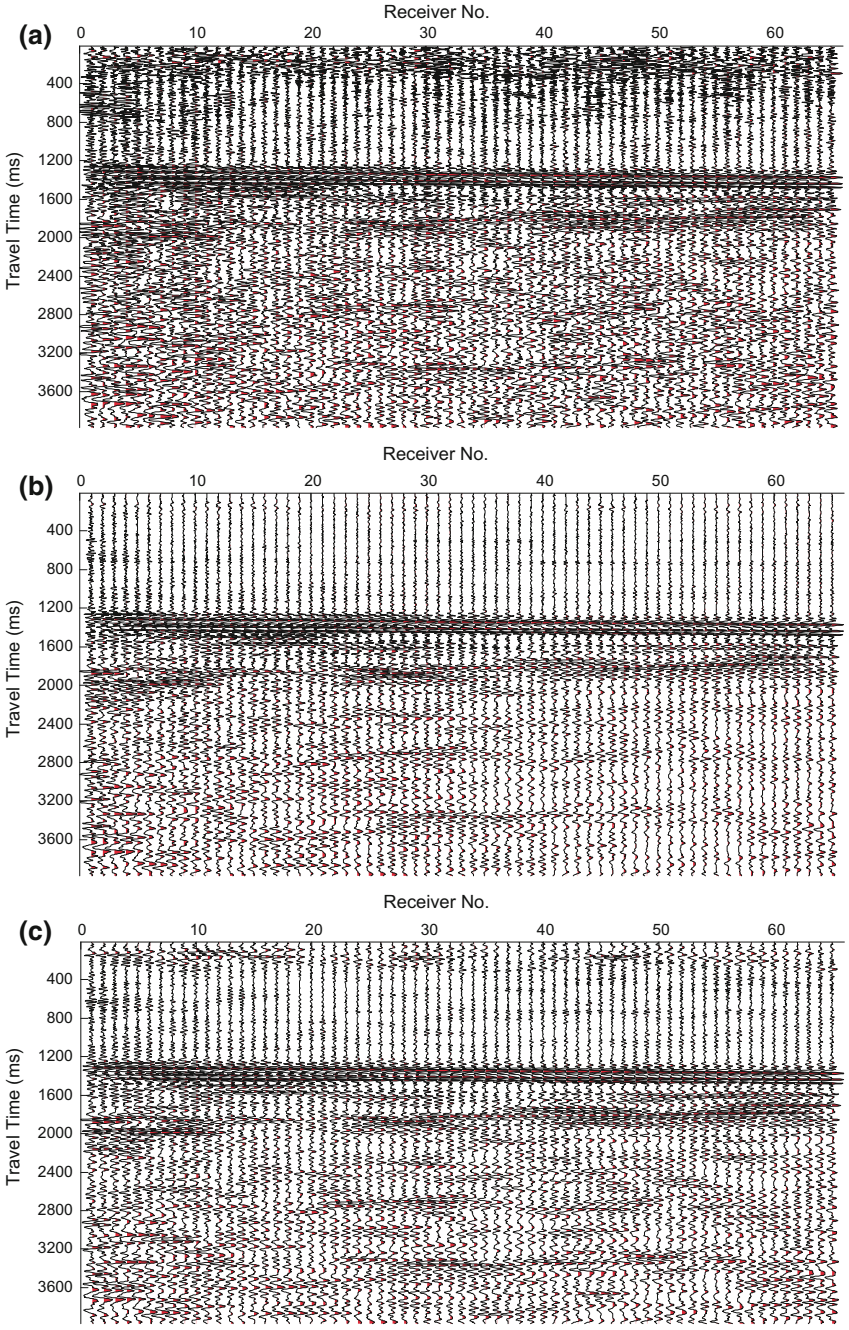


Fig. 2.107 **a** Common offset sorted gather recorded in Iran, **b** result of band-pass filtering **c** result of ANN filtering (Kimiaefar et al. 2016)

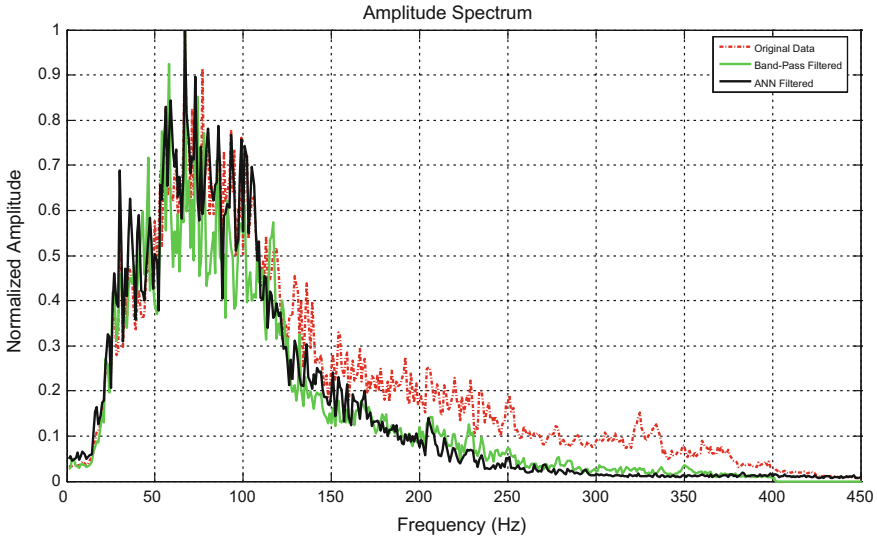


Fig. 2.108 Amplitude spectra of sections illustrated in Fig. 2.107 (Kimiaefar et al. 2016)

In order to implement WPA with this synthetic data as input, the images of approximations and details were calculated at three successive levels using a level four Daubechies wavelet. The approximation image at the second level was selected for the training of the network and subsequently 500 random points of the input data were selected. Network training was carried out based on the three parameters of:

- Mean,
- Median,
- Extremum of a window with a variable length of data for a 3 × 3 neighborhood for each of these points.

The noisy input synthetic data, the adaptive wiener filtered (AWF) signal, and the data filtered by our (ANN-filtering) are shown in Fig. 2.102 and it is clear that the ANN-filtered is much less noisy than the AWF filter.

Two real seismic sections were used to assess the performance of the proposed method. The first dataset was chosen from a relatively deep seismic acquisition (20 s) recorded in Victoria, Australia (Jones 2010), which has a 4 ms sampling interval, 20 m CMP spacing and subsurface fold of 75, The training procedure was implemented using the specifications listed in Table 2.46. Figure 2.103 illustrates the network's training performance and linear regression calculated between targets relative to outputs. Both parts of Fig. 2.103 confirm the acceptable training processes. The difference between the original data and the filtered data are illustrated

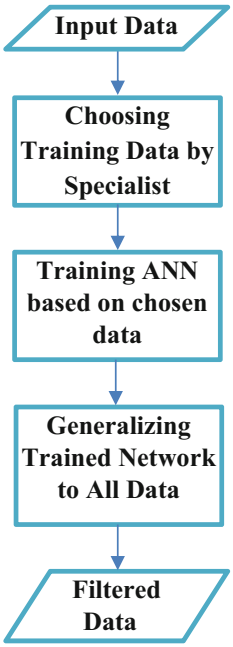


Fig. 2.109 General flowchart of random noise attenuating by ANN

in Fig. 2.105 to confirm these results. While there is almost no useful information in the difference section obtained by ANN filtering method, there seem to be two additional coherent seismic events on the difference section as compared to the AWF method.

A broadened amplitude spectrum and improved resolution is a definite advantage for a processing method (Stanton and Sacchi 2014). The amplitude spectra of the seismic sections in Fig. 2.104 are shown in Fig. 2.106. Across the range from 10 to 30 Hz, the frequency components of the ANN-filtered section are higher than the two other sections because >30 Hz. which are probably related to random events (Liu et al. 2009) the amplitude of the ANN-filtered section are generally less than the others for most frequencies. The comparison of the results in Figs. 2.106, 2.107 and 2.108 indicate improved attenuation of random noise by the proposed method. The related Matlab codes to attenuate the noise of seismic data are explained in appendix 1 of this chapter.

The second real dataset recorded in an oil field in south-western Iran is a portion of a common-offset-sorted gather with 4 ms time sampling and total length of 4 s. Figure 2.107a shows a considerable amount of random seismic noise is contaminating the signal. One of the most widely used methods in processing seismic data is band-pass filtering which is widely used for common offset sorted gathers, and is suitable for comparison with the proposed method. The comparison of ANN filtering and band-pass filtering (Fig. 2.107a, b) shows that, the section processed by

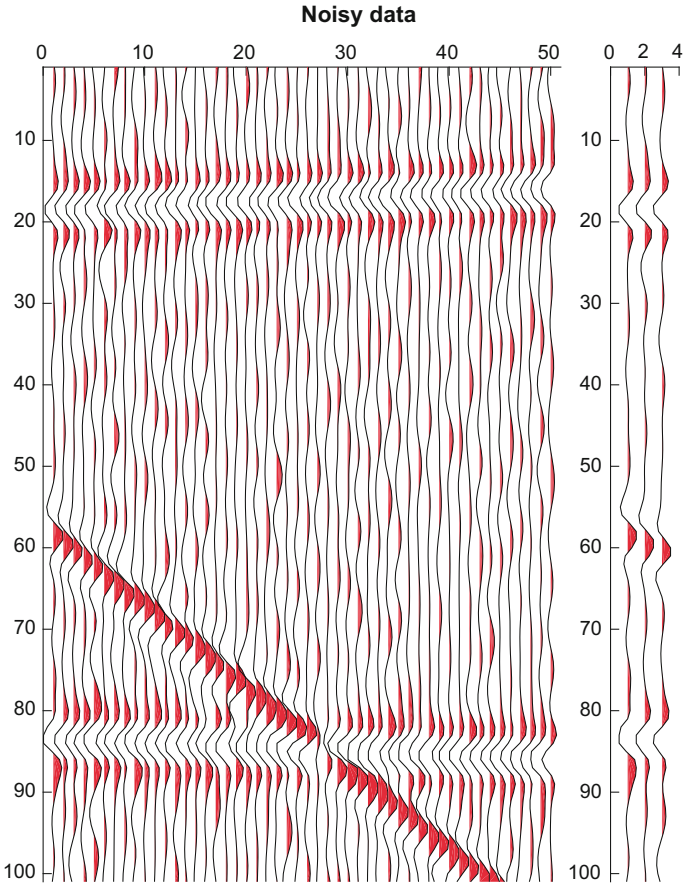


Fig. 2.110 A synthetic noisy CMP gather and the first 3 traces selected for training ANN

the trained neural network has successfully eliminated the random noise, while preserving the true reflection data.

Figure 2.108 plots the amplitude spectra of the seismic sections in Fig. 2.107. Most spectral >120 Hz are attenuated with frequency band-pass filtering (green line) but ANN filtering (black line) is substantially better. Below 120 Hz, ANN filtering has higher amplitude than the output of band-pass filtering, for almost all frequencies.

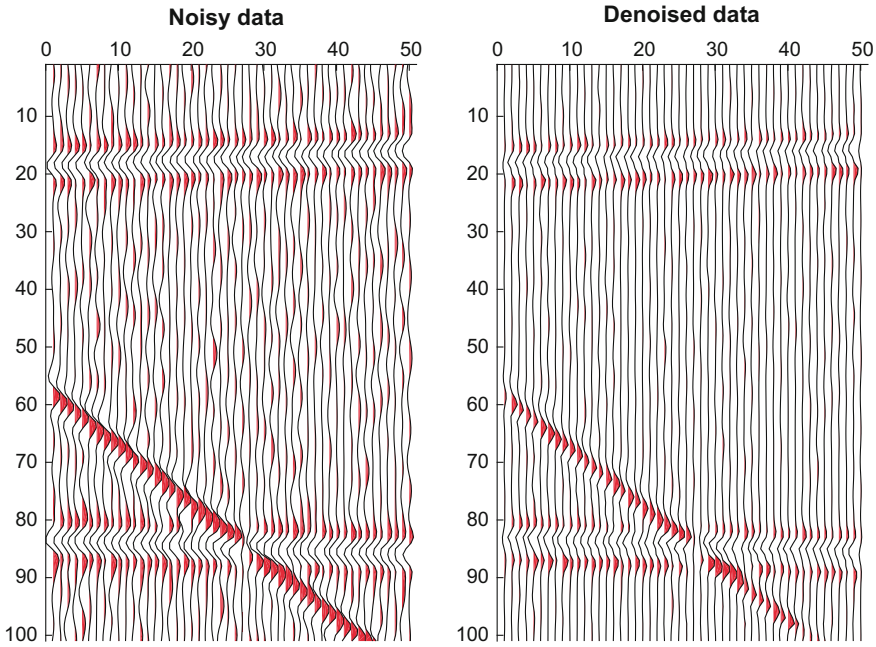


Fig. 2.111 Noisy data (right) and denoised version (left) using feed forward back propagation ANN

2.12.3 Conclusion

ANN filtering seems to be an advantageous alternative method for denoising seismic data. This research reveals that instead of utilizing the original section, approximating the input section with coarser scales of the WPA is a novel approach to neural network semi-automatic training and will noticeably improve the resolution of both synthetic and real seismic datasets. A comparison of AWF and band-pass filtering methods, demonstrates that the proposed method is more efficient in eliminating background random noise in real and synthetic CMP and common offset sorted gathers.

Appendix 1 of Chapter Two

The procedure of teaching algorithms for MLP networks consist of 4 steps: defining network architecture, defining parameters related to network and learning algorithm (iterations, maximum epoch number and etc., training the network and finally validating and simulating (generalizing) the trained network. This general process is illustrated in Fig. 2.109.

In the example described here, a synthetic noisy seismic CMP gather is chosen and the ability of ANN in random noise attenuation is shown in a simple manner.

As random noise components are supposed not to be predictable in adjacent traces, the logic used for attenuating random noise is based on attenuating whatever is not confirmed to be coherence data.

By comparing the elements of a 7 by 3 neighborhood data and by user's decision, the network will be trained with some examples (14 training pairs are used here). In Fig. 2.110 the noisy data and the selected adjacent traces from trace No. 1 to No. 3 is illustrated. These three traces are selected for extracting training pairs. The expert knowledge is required here in order to make the decision for choosing best points for marking noise contaminated data as well as pure coherence data related points. The decision could take place as a number (0 for noisy and 1 for noise free points). It should be mentioned that as an advanced algorithm, partly noise contaminated data also should be present in the trained network but in this appendix, the simplicity of the algorithm was the priority. Selected pairs are chosen from trace No. 2 as the training points are supposed to be check in a 7 by 3 neighborhood.

After training the network, simulation process will be held. In doing so, the input will be the 7 by 3 neighborhood amplitude (as training input data) for each sample. Now, the weight for each sample could show whether the sample belongs to noise or coherence data space (as training output data).

The results of the Generalizing trained network will be a matrix that contains weights for all data. Denoised version of input data will be achieved by production of weights into noisy data. The results are shown in Fig. 2.111.

```
% Loading, Plotting and selecting (for ANN training) some points from synthetic noisy CMP gather
load('noisy_data.mat');
% Calculating size of input-data
[m,n] = size(noisy_data);
% Choosing train-data matrix
train_matrix = noisy_data(:,1:3);
% Plotting input-data and train-data
figure;
subplot(1,7,1:6);
subplot(1,7,1:6);
plotseis(noisy_data); title('Noisy data');
subplot(1,7,7,'align');
plotseis(train_matrix); title('Selected for training ANN');
 % Choosing desired points from train-data matrix
% Selected trace
trace = 2;
% Selected times
times = [18, 21, 17, 83, 39, 38, 48, 67, 60, 83, 15, 44, 35, 71];
% Determined weights (noise = 0, signal = 1)
targets  = [1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0];
% Preparing ANN training input and target data
% (The neighborhood data for each time in "times" as input and the weights determined in "targets" as network target)
for i = 1:size(times,2)
   in(i,1:7)    =   [train_matrix(times(1,i)-3:times(1,i)+3,trace-1)];
   in(i,8:14)  =   [train_matrix(times(1,i)-3:times(1,i)+3,trace)];
   in(i,15:21)=   [train_matrix(times(1,i)-3:times(1,i)+3,trace+1)];
   out(i) =  targets(i);
end
% Determining iteration for algorithm
iteration = 6;
% Training Feed Forward Back Propagation ANN and Generalizing to all data
net_out = zeros(m,n, iteration);
ANN_out = zeros(m,n);
```

```
for k = 1: iteration
%  Defining Feed Forward network with variable hidden layer size in each iteration
   net = feedforwardnet(4+(k-1)*4);
%  Training network using input and target data
   [net, trace] = train(net, in', out);
%  Generalizing trained network to all data
   for I = 4:m-3
      for j = 2:n-1
         net_in(1,1:7) = noisy_data(i-3:i+3,j-1);
         net_in(1,8:14) = noisy_data(i-3:i+3,j);
         net_in(1,15:21) = noisy_data(i-3:i+3,j+1);
         net_out(i,j,k) = sim(net,net_in');
      end
   end
   ANN_out = ANN_out+net_out(:,:,k);
end
% Normalizing simulated weights between 0 and 1 and then using them for denoising input data
denoised = nor_mat((1/iteration)*ANN_out,0).*noisy_data;
% Plotting results
figure;
subplot(1,2,1);
plotseis(noisy_data); title('Noisy data');
subplot(1,2,2);
plotseis(denoised);   title('Denoised data');
```

Appendix 2 of Chapter Two

Brief list of Training functions in Matlab's NN Toolbox is presented in Table 2.47.

Table 2.47 List of Training functions in NN toolbox of Matlab

Function name	Algorithm
trainb	Batch training with weight and bias learning rules
trainbfg	BFGS quasi-Newton backpropagation
trainbr	Bayesian regularization
trainc	Cyclical order incremental training w/learning functions
traincgb	Powell-Beale conjugate gradient backpropagation
traincgf	Fletcher-Powell conjugate gradient backpropagation
traincgp	Polak-Ribiere conjugate gradient backpropagation
traingd	Gradient descent backpropagation
traingdm	Gradient descent with momentum backpropagation
traingda	Gradient descent with adaptive lr backpropagation
traingdx	Gradient descent w/momentum and adaptive lr backpropagation
trainlm	Levenberg-Marquardt backpropagation
trainoss	One step secant backpropagation
trainr	Random order incremental training w/learning functions
trainrp	Resilient backpropagation (Rprop)
trains	Sequential order incremental training w/learning functions
trainscg	Scaled conjugate gradient backpropagation

Source www.cs.ucr.edu/∼vladimir/cs171/nn_summary.pdf

References

Abe C., Shimamura T., 2012, Iterative Edge-Preserving Adaptive Wiener Filter for Image Denoising, International 253 Journal of Computer and Electrical Engineering, 4(4), 503–506.

Abdelrahman E.M., Bayoumi A.I., and El-Araby H.M., 1991, A least-squares minimization approach toinvert gravity data, Geophysics 56, 1, 115–118.

Abdelrahman E.M., Bayoumi A.I., Abdelhady Y.E., Gobashy M.M., and ElAraby, 1989, Gravity interpretation using correlation factors between successive least-squares residual anomalies, Geophysics 54, 12, 1614–1621, https://doi.org/10.1190/1.1442629.

Abdelrahman E.M., and El-Araby T.M., 1993, A least-squares minimization approach to depth determination from moving average residual gravity anomalies, Geophysics 58, 12, 1779–1784, https://doi.org/10.1190/1.1443392.

Abdelrahman E.M., El-Araby T.M., El-Araby H.M. and Abo-Ezz E.R., 2001, A new method for shape and depth determinations from gravity data, Geophysics, 66: 1774–178.

Aghajani H., Moradzadeh A. and Zeng H., 2009, Estimation of depth to salt domes from normalized full gradient of gravity anomaly and examples from the USA and Denmark, Journal of Earth Science, 20: 1012. https://doi.org/10.1007/s12583-009-0088-y.

Albora A.M., Uçan O.N. and Aydogan D., 2007, Tectonic modeling of Konya-Beysehir Region (Turkey)using cellular neural networks, Annals of geophysics, VOL. 50, N. 5,603–614, https://doi.org/10.4401/ag-3060.

Albora, A.M., Ucan O.N., Özmen A. and Ozkan T., 2001, Separation of Bouguer anomaly map using cellular neural network, Journal of Applied Geophysics, 46, 129–142, https://doi.org/10.1016/s0926-9851(01)00033-7.

Al-Garni M.A., 2009, Interpretation of some magnetic bodies using neural networks inversion, Arabian Journal of Geosciences, 2, 175–184.

Barráez D., Garcia-Salicetti S., Dorizzi B., Padrón M., Ramos E., 2002, Modular neural networks for seismic tomography, 2002 IEEE, Object recognition supported by user interaction for service robots, in: https://ucv.academia.edu/DanielBarraez.

Barton N., 2000, TBM Tunneling in Jointed and Faulted Rock, Balkema, Brookfield.

Beiki M., and Pedersen L.B., 2010, Eigenvector analysis of gravity gradient tensor to locate geologic bodies, Geophysics 75, 6, I37–I49, DOI:10.1190/ 1.3484098.

Bhatt A. and Helle H.B., 2002, Determination of facies from well logs using modular neural networks. Petroleum Geoscience, 8(3), 217–229(Expanded abstract presented in EAGE 64th conference and technical exhibitions-Florence, Italy).

Bhattacharya B.B. and Roy N., 1981, A note on the use of a nomogram for self-potential anomalies, Geophysical Prospecting, 29(1), 102–107, https://doi.org/10.1111/j.1365-2478.1981.tb01013.x.

Bieniawski von P., Celada Tamames Z.T., Galera Fernandez B., J.M., Alvarez Hernandez, M., 2006, Rock masse excavability indicator: new way to selecting the optimum tunnel construction method, Tunnell. Underground Space Technol. 21 (3–4), 237.

Bowin C., E. Scheer, and W. Smith, 1986, Depth estimates from ratios of gravity, geoid, and gravity gradient anomalies, Geophysics 51, 1, 123–136, https://doi.org/10.1190/1.1442025.

Brown W. M., DA Yid, Grovest and Tamas d. Gedeon, 2003, An artificial neural network method for mineral prospectivity mapping: a comparison with fuzzy logic and bayesian probability metiiod, in :Geophysical applications of artificial neural networks and fuzzy logic, Sandham and Leggett,179-198. Springer Science + Business Media Dordrecht ISBN 978-90-481-6476-9 ISBN 978-94-017-0271-3 (eBook) https://doi.org/10.1007/978-94-017-0271-3.

Bruland A., 1998, Hard rock tunnel boring, Ph.D. Thesis, Norwegian University of Science and Technology, Trondheim.

Büchi E., 1984, Einfluss Geologischer Parameter auf die Vortriebsleistung einer Tunnelbohrmaschine, PhD Thesis, University of Bern.

Busch J.M., Fortney W.G. and Berry L.N., 1987, Determination of lithology from well logs by statistical analysis. SPE Formation Evaluation 2, 412–418.

Butler D.K., 1984, Microgravimetric and gravity gradient techniques for detection of subsurface cavities, Geophysics 49(7), 1084–1096, https://doi.org/10.1190/1.1441723.

Chang H.C., Kopaska-Merkel D.C. Chen H.C. and Durrans S. R., 2000, Lithofacies identification using multiple adaptive resonance theory neural networks and group decision expert system. Computers & Geoscience 26, 591–601. https://doi.org/10.1190/1.1442946.

Chamoli, A., Srivastava R.P., and Dimri V.P., 2006, Source depth characterization of potential field data of Bay of Bengal by continuous wavelet transform, Indian J. Marine Sci. 35(3), 195–204.

Douglas I. Hart, 1996, Improving the reliability of first-break picking with neural networks, Annual Meeting, Society of Exploration Geophysicists, 1662–1665.

Del Negro C., Greco F., Napoli R. and Nunnari G., 2008, Denoising gravity and geomagnetic signals from Etna volcano (Italy) using multivariate methods, Nonlinear Processes in Geophysics,15, 735–749, ww.nonlin-processes-geophys.net/15/735/2008.

Efron, B., and Tibshirani R.J., 1993, An Introduction to the Bootstrap, Chapman & Hall, London.

Elawadi, E., Salem A., and Ushijima K., 2001, Detection of cavities and tunnels from gravity data using a neural network, Exploration Geophysics, 32(4), 204–208, https://doi.org/10.1071/eg01204.

El-Kaliouby H.M. and Al-Garni M.A., 2009, Inversion of self-potential anomalies caused by 2D inclined sheets using neural networks, Journal of geophysics and engineering, 6, 29–34, https://doi.org/10.1088/1742-2132/6/1/003.

Eshaghzadeh A. and Kalantari. R.S., 2015, Anticlinal Structure Modeling with Feed Forward Neural Networks for Residual Gravity Anomaly Profile, 8th Congress of the Balkan Geophysical Society 5–8 October 2015, Chania, Greece.

Eshaghzadeh. A., 2011. Optimization of gravity anomalies due to folded structures of an oil field with parabolic density contrast. M.Sc. Thesis, Tehran University, Iran (in Persian, Unpublished).

Eshaghzadeh A. and Hajian A., 2018, 2D inverse modeling of residual gravity anomalies from Simple geometric shapes using Modular Feed-forward Neural Network, Annals of geophysics, 61(1), SE115(1-5), https://doi.org/10.4401/ag-7540

Fahlman S.E, 1989, Fast learning variations onback—propagation: An empirical study, In Proceedings of the 1988 Connectionist Models Summer School, 38–51.

Fahlman S.E and Lebiere C., 1990, The cascade-correlation learning architecture, Advances in Neural Information Processing Systems, 2, 524–532.

Farmer I.W. and Glossop N.H., 1980, "Mechanics of disc cutter penetration". Tunnels .Tunnell. Int. 12 (6), 22–25

Gehring K., 1995, Leistungs -und Verschleissprognosen in maschinellen Tunnelbau, Felsbau 13 (6).

Ghaffari M.R. and Mohammadzadeh A., 2015, Modeling of crustal velocity using Artificial Neural Networks (case study: Iran Geodynamic GPS Network), Iranian Journal of Geophysics, 9(3), 91–103 (Abstract in English, Original: in Persian).

Gong Q.M. and Zhao, J., 2009,Development of a rock mass chara cteristics model for TBM penetration rate prediction, Int J Rock Mech Min Sci. 46(1), 8–18.

Graham P.C., 1976, Rock exploration for machine manufacturer, In: Bieniawski, Z.T. (Ed.), Exploration for Rock Engineering. Balkema, Johannesburg, 173–180.

Gravesen, P., 1990, Geological map of Denmark 1:50.000, Kortobladet 116I Thirsted, Geological basis datakort, Geological Survey of Denmark, Map series no.13.

Grêt A.A., E.E.Klingelé, Kahle H.G., 2000, Application of artificial neural networks for gravity interpretation in two dimensions: a test study, Bollettino Digeofisica teorica Edapplicata, 41(1), 1–20.

Gupta O.P., 1983, A least-squares approach to depth determination from gravity data, Geophysics 48(3), 357–360, https://doi.org/10.1190/1.1441473.

Hajian A., Zomorrodian H., Styles P., 2012, Simultaneous Estimation of Shape Factor and Depth of Subsurface Cavities from Residual Gravity Anomalies using Feed-Forward Back-Propagation Neural Networks, Acta Geophysica, 58(4), 317–33, https://doi.org/10.2478/s11600-012-0049-1.

Hajian A., Shirazi M, 2015, "Depth estimation of salt domes using gravity data through General Regression Neural Networks, case study: Mors Salt dome Denmark", Journal of the Earth and Space Physics, 41(3), 425–438.

Hajian A., Rezazadeh Anbarani M., Mobarra Y., 2014, Comparison of neural networks, fuzzy - neural networks and finite element methods for the estimation of surface settlement due to tunneling in the Mashhad subway line 2, 2nd International Congress of Recent Advances in Engineering – Islamic Azad University – Khomeinishahr Branch, ICRAE2014.

Hajian A., Styles P., Zomorrodian H., 2011a, Depth Estimation of Cavities from Microgravity Data through Multi Adaptive Neuro Fuzzy Interference Systems, 17th European Meeting of Environmental and Engineering Geophysics, Leicester, UK, 2011.

Hajian A., 2009, Detection of Hazardous subsidence by microgravity method through Artificial Neural Networks, 6th Annual Meeting of the Asia Oceania Geosciences Society, AOGS, Singapore.

Hajian A., 2004, Depth Estimation of Gravity Anomalies by Neural Network, M.Sc. Thesis, Tehran University, Iran (in Persian, unpublished).

Hajian A., 2008, Depth estimation of gravity anomalies by Hopfield Network. In: Proc. 5th Annual Meeting, AOGS: Asia Oceania Geosciences Society, Busan, Korea, Ext. Abstr. SE92-D3-PM2-P001.

Hajian A., Zomorrodian H., Lucas C., and Amini N., 2007, Detection of Subsurface Sinkholes Made by Earthquakes Through Artificial Neural Networks from Microgravity Data, International Earthquake Symposium, Kocaeli University, Turkey, October 22–24.

Hajian A., Ardestani V.E. and Lucas C., 2011b, Depth estimation of gravity anomalies using Hopfield Neural Networks, Journal of the Earth and Space Physics(Scopus), 37, 1–9.

Hassanpour J., Rostami J. and Zhao, J., 2011, A new hard rock TBM performance prediction model for project planning. Tunnell. Underground Space Technol. 26, 595–603.

Hesham E.L.K., 2009, Inversion of self-potential anomalies caused by 2D-inclined sheets using, Journal of geophysics and engineering 6, 29–34. https://doi.org/10.1088/1742-2132/6/1/003.
http://www.preventionweb.net/files/…/30411_2infoaboutearthquakegemmashhad.pdf
https://en.wikipedia.org/wiki/Hopfield_network#/media/File:Energy_landscape.png.
Hughes, H.M., 1986, The relative cut ability of coal measures rock". Min. Sci. Technol., 3, 95–109.
Imani M., You RJ. and Kuo C.Y., 2014, Caspian Sea level prediction using satellite altimetry by artificial neural networks, International Journal of Environmental Science and Technology, 4 (11), 1035–1042, https://doi.org/10.1007/s13762-013-0287-z.
Jagannadha R. S., Rao R.P. and Radhakrishna M. I V, 1993, Automatic inversion of self-potential anomalies of sheet-like bodies, Computer Geoscience. 1961–73.
Jian F.X., Chork C.Y., Taggert I.J., McKay D.M. and Bartlett R.M., 1994, A genetic approach to the prediction of petrophysical properties. Journal of Petroleum Geology, 17, 71–88.
Jones LEA, 2010, L194 Ararat Seismic Survey, VIC, 2009. Stack and migrated SEG-Y seismic data and images for line 09GA-AR1. Geoscience Australia, Victoria.
Jorgensen F., Sandersen B. E. P., Auken E., Lykke-Andersen H. and Sorensen K., 2005, Contributions to the geological mapping of Mors, Denmark – A study based on a large –scale TEM survey, Bulletin of the Geological Society of Denmark, 52, 53–75.
Kaftan I., Sındırgı P. and Akdemir Ö., 2014, Inversion of Self Potential Anomalies with Multilayer Perceptron Neural Networks. Pure and Applied Geophysics 171:8, 1939–1949.
Kimiaefar R., Siahkoohi H.R., Hajian A., and Kalhor A., 2016, Seismic random noise attenuation using artificial neural network and wavelet packet analysis, Arabian Journal of Geosciences, 9 (234),1–11.
Langer H., Nunnari G., and Occhipinti L., 1996, Estimation of seismic waveform governing parameters with neural networks, J. Geophys. Res. 101, B9, 20109–20118, https://doi.org/10.1029/96jb00948.
Li X., 2006, Understanding 3D analytic signal amplitude, Geophysics 71(2), 13–16, https://doi.org/10.1190/1.2184367.
Lim J. S., 1990, Two-Dimensional Signal and Image Processing, Englewood Cliffs, NJ, Prentice Hall.
Lines L.R., and Treitel S., 1984, A review of least-squares inversion and its application to geophysical problems, Geophysical Prospecting, 32(2), 159–186., https://doi.org/10.1111/j.1365-2478.1984.tb00726.x.
Liu Y., Liu C. and Yang D., 2009, A 1D time-varying median filter for seismic random, spike-like noise elimination. Geophysics 74(1), 17–24.
Mair R.J. and Taylor R.N., 1997, Bored Tunneling in the Urban Environment, 14th International Conference on Soil Mechanics and Foundation Engineering, Hamburg, Rotterdam, 2353–2358
Masters T., 1993, Practical neural network recipes in C++: Academic Press, Inc.
McCormack M.D., Zaucha D.E., and Dushek D.W., 1993, First-break refraction event picking and seismic data trace editing using neural networks, Geophysics 58, 1, 67–78, https://doi.org/10.1190/1.1443352.
Mobarra Y. and Hajian A., 2013, Application of Artificial Neural Networks to the Prediction of TBM Penetration Rate in TBM-driven Golab Water Transfer Tunnel, International Conference on Civil Engineering Architecture & Urban Sustainable Development 27 & 28 November, Tabriz , Iran
Mobarra Y., Hajian A., Rezazadeh Anbarani M., 2013, Application of Artificial Neural Networks to the Prediction of TBM Penetration Rate in TBM-driven Golab Water Transfer Tunnel, International Conference on Civil Engineering Architecture, Tabriz, Iran, 120–126.
Moghtased-Azar K., and Zaletnyik P., 2009, crustal velocity field modeling with neural network and polynomials, SIDERIS, M.G. (Ed.), observing our changing Earth, International Association of Geodesy symposia, 133, 809–816.
Mohan N.L., Anandababu L., and Rao S.V.S., 1986, Gravity interpretation using the Mellin transform, Geophysics 51, 1, 114–122, https://doi.org/10.1190/1.1442024.
Murat M.E., and Rudman A.J., 1992, Automated first arrival picking: A neural network approach, Geophysical Prospecting, 40(6), 587–604, https://doi.org/10.1111/j.1365-2478.1992.tb00543.x.

Murthy B.V.S., and Haricharan P., 1984, Self-potential anomaly over double line of poles—interpretation through log curves Proc. Indian Acad. Sci. Earth Planet Sci. 93437–45

Murthy B.V.S., and Haricharan P., 1985, Nomograms for the complete interpretation of spontaneous potential profiles over sheet like and cylindrical 2D structures. Geophysics 50(11), 27–35.

Murthy I V R, Sudhakar K.S. and Rao P.R., 2005, A new method of interpreting self-potential anomalies of two-dimensional inclined sheets Computer Geoscience, 31661–5.

Nettleton L.L., 1976, Gravity and Magnetic in Oil Prospecting, McGraw-Hill, New York.

Oruç B., 2010, Depth estimation of simple causative sources from gravity gradient tensor invariants and vertical component, Pure Appl. Geophys. 167(10), 1259–1272, https://doi.org/10.1007/s00024-009-0021-4.

Osman O., Albora A.M. and Ucan O.N., 2007, Forward modeling with forced Neural Network for Gravity Anomaly Profile, Mathematical Geology 39, 593–605. https://doi.org/10.1007/s11004-007-9114-8.

Paul M.K., 1965, Direct interpretation of self-potential anomalies caused by inclined sheets of infinite extension Geophysics 30(4), 18–23.

Pavlis N., Kenyon S., Factor J., and Holmes S., 2008, Earth gravitational model. In SEG technical pro-gram expanded abstracts 27, 761–763.

Pantazis G. and Alevizakou E., 2013, The Use of Artificial Neural Networks in Predicting Vertical Displacements of Structures, International Journal of Applied Science and Technology, 3(5),1–8.

Pedersen G.K., and Shurlyk F., 1983, The fur formation, a late Paleocene ash-bearing diatomite from northern Denmark, Bulletin of the Geological Society of Denmark, 32, 43-65.

Poulton M.M., Sternberg B.K., and Glass C.E., 1992, Location of subsurface targets in geophysical data using neural networks, Geophysics 57(12), 1534–1544, https://doi.org/10.1190/1.1443221.

Wu Q., Nakayama K., 1997, Avoiding weight-ill growth: cascade correlation algorithm with local regularization. Neural Networks, 1954–1959.

Rao B.S.R., Murthy I.V.R., and Reddy S.J., 1970, Interpretation of self-potential anomalies of some simple geometrical bodies Pure Appl. Geophys. 7860–77.

Rao, B.S.R., Murty, I.V.R., 1978. Gravity and Magnetic Methods of Prospecting. Arnold-Heinemann Publishers, New Delhi, India, 390.

Rao R. M., Babu H. and Sivakumar Sinha G., 1982, A Fourier transform for the interpretation of self-potential anomalies due to two-dimensional inclined sheet of finite depth, Pure Applied Geophysics, 120365–74.

Reid A.B., Allsop J.M., Granser H., Millett A.J., and I.W. Somerton, 1990, Magnetic interpretation in three dimensions using Euler deconvolution, Geophysics 55(1), 80–91, https://doi.org/10.1190/1.1442774.

Rostami J., Ozdemir L., 1993, A new model for performance prediction of hard rock TBM. In: Proceedings of the Rapid Excavation and Tunnelling Conference. Chapter 50, Boston, MA, USA, 793–809.

Rostami, J., 1997, Development of a force estimation model for rock fragmentation with disc cutters through theoretical modeling and physical measurement of crushed zone pressure, Ph. D. thesis, Colorado School of Mines, Golden, Colorado, USA.

Roth G., and Tarantola A., 1994, Neural networks and inversion of seismic data, J. Geophys. Res. 99, B4, 6753–6768, https://doi.org/10.1029/93jb01563.

Roux BL. And Rouanet H., 2006, Geometric data analysis: from correspondence analysis to structured data analysis. Springer Science &Business Media, Dordrecht, 192–197.

Ryseth A., Fjellbirkeland H., Osmundsen I.K., Skålnes Å., and Zachariassen E., 1998, High-resolution stratigraphy and seismic attribute mapping of a fluvial reservoir: Middle Jurassic Ness Formation, Oseberg Field. AAPGBulletin, 82, 1627–1651.

Salem A., Elawadi E., and Ushijima K., 2003a, Short note: Depth determination from residual gravity anomaly data using a simple formula, Comput. Geosci. 29(6), 801–804, https://doi.org/10.1016/s0098-3004(03)00106-7.

Salem A., Elawadi E., Abdelaziz A., and Ushijima K., 2003b, Imaging subsurface cavities from microgravity data using Hopfield neural network. Proceedings of the 7th International Symposium on Recent Advances in Kyoto, RAEG2003, Expanded Abstracts, 199–205.

Sarzeaud O.M., LeQuentrec-Lalancette F. and Rouxel D., 2009, Optimal Interpolation of Gravity Maps Using a Modified Neural Network, Math Geosci, 41, 379–395, https://doi.org/10.1007/s11004-009-9214-8.

Sharma B., and Geldart L.P., 1968, Analysis of gravity anomalies of two dimensional faults using Fourier transforms, Geophysical Prospecting 16(1), 77–93, https://doi.org/10.1111/j.1365-2478.1968.tb01961.x.

Song J. G., Zeng W. H., Xu Y. and Xu W. X., 2011, The Improvement of Neural Network Cascade-correlation Algorithm and its Application in Picking Seismic First Break, 73rd EAGE Conference and Exhibition incorporating SPE EUROPEC 2011, DOI: https://doi.org/10.3997/2214-4609.20149418.

Song J.G., Zeng W.H., Xu Y., Xu W.X., 2011, The Improvement of Neural Network Cascade correlation Algorithm and its Application in Picking Seismic, 2011, First Break 73rd EAGE Conference & Exhibition incorporating SPE EUROPEC, Vienna, Austria.

Stanton A. and Sacchi M.D., 2014, Least squares migration of converted wave seismic data. CSEG Recorder 39(10), 48–52.

Styles P. and Hajian A., 2012, Generalized Regression Neural Networks for Cavities Depth Estimation using Microgravity Data, Case Study: Kalgorlie Gold, Near Surface Geoscience, 18th European Meeting of Environmental and Engineering Geophysics, Paris, France.

Styles, P., R. McGrath, E. Thomas, and Cassidy N.J., 2005, The use of microgravity for cavity characterization in karstic terrains, Quart. J. Eng. Geol. Hydrogeol. 38(2), 155–169, https://doi.org/10.1144/1470-9236/04-035.

Sundararajan N., Srinivasa Rao P., and Sunitha V., 1998, An analytical method to interpret self-potential anomalies caused by 2D inclined sheets. Geophysics 63(15), 51–5.

Thompson D.T., 1982, EULDPH: A new technique for making computer-assisted depth estimates from magnetic data, Geophysics 47(1), 31–37, https://doi.org/10.1190/1.1441278.

Tierra A. and de Freitas S., 2005, Artificial Neural Network: A Powerful Tool for Predicting Gravity Anomaly from Sparse Data. In: Jekeli C., Bastos L., Fernandes J. (eds) Gravity, Geoid and Space Missions. International Association of Geodesy Symposia, 29. Springer, Berlin, Heidelberg.

Weiss, S.M., and Kulikowski C.A., 1991, Computer Systems That Learn, Morgan Kaufmann, San Mateo.

Wu, Q., and Nakayama, K., 1997, Avoiding WeightIllgrowth: Cascade Correlation Algorithm with Local Regularization. Neural Networks, 1954-1959.

www.cs.ucr.edu/∼vladimir/cs171/nn_summary.pdf

Yang H., Chen H.C. and Fang J.H., 1996, Determination of carbonate lithofacies using hierarchical neural networks. Intelligent Engineering Systems through Artificial Neural Networks 6, 481–486.

Yang, H. Z., Wang, W. N., & Ding, F., 2006, Two Structure Optimization Algorithm for Neural Networks. Information and Control, 35(6), 700-705.

Yilmaz M., 2013, Artificial Neural Networks pruning approach for geodetic velocity field determination Boletim de Ciências Geodésicas, 19(4), 558–573, http://dx.doi.org/10.1590/S1982-21702013000400003.

Zaknich A., 1999, Artificial Neural Networks: a research dominated engineering discipline: Centre for Intelligent Information Processing Systems (CIIPS), Department of Electrical and Electronic Engineering, Univ. of Western Australia, (unpublished).

Zhang Y., and Paulson K.V., 1997, Magneto telluric inversion using regularized Hopfield neural networks, Geophysical Prospecting, 45(5), 725–743, DOI:10.1046/ j.1365-2478.1997.660299.x.

Zhongjin Y. and Zhongke Shi, 2004, Fast Approach for Cascade-Correlation Learning, Mathematics in Practice and Theory, 34(9), 114–118.

Zinkiewicz O.C .and Taylor R.L., 2000, The finite Elements, Fifth Edition, Volume1: The Bais, McGraw-Hill. ISBN: 0750650494.

Part II
Fuzzy Logic

Chapter 3
Fuzzy Logic

3.1 Introduction

Fuzzy logic is an extension to Boolean logic and was developed by Zadeh in 1965. While classical logic only allows values of 1 for 'true' and '0' for 'false' as the value of a variable, fuzzy logic permits any value in the interval [0, 1] (PourMohammadBagher et al. 2009). This attempts to more closely mimic the way in which humans think and communicate which is often vague and uncertain way, partly because of limited information and partly due to the way the human brain works (Nelles 2001).

This fuzzy logic approach can be useful for help engineers and researchers in dealing with uncertainty and in the use of imprecise information in complex situations (Nikravesh and Aminzadeh 2003). The application of fuzzy logic for solving complex problems with associated uncertainty has grown steadily and fuzzy logic now plays an important role in many engineering disciplines (Aminzadeh and Chatterjee 1984/1985; Aminzadeh 1989; Aminzadeh and Jamshidi 1995; Adams et al.1999a, b; Hajian et al. 2012).

The fuzzy approach, based on fuzzy logic, has many advantages some of which are listed below (Mathworks 2009):

- Conceptually easy to understand.
- The Mathematical concepts are very simple and provide an intuitive approach without added complexity.
- Flexibility to add more functionality during the operation without starting afresh.
- Tolerance of imprecise data.
- Fuzzy reasoning builds the inherent uncertainty in natural processes implicitly rather than as an 'add-on'.
- Ability to model non-linear functions of arbitrary complexity.
- Fuzzy systems can match any set of input-output data using adaptive neuro-fuzzy inference systems (ANFIS).

A. Hajian and P. Styles, *Application of Soft Computing and Intelligent Methods in Geophysics*, Springer Geophysics,
https://doi.org/10.1007/978-3-319-66532-0_3

- Expert knowledge can be built in using the experience of interpreters who already understand the system combining expert partial knowledge with learning from examples
- Fuzzy logic is based on linguistic labels and implements computing with words.

3.2 Motivation for Using Fuzzy Logic in Geophysics

The value of Fuzzy Logic to Geophysics can be described from two main viewpoints:

3.2.1 First Viewpoint

One of the inherent properties of Geophysical data is their uncertainly and non-uniqueness when they are interpreted for geological applications (Jongmans and Garambis 2007).

Geophysical data is naturally vague because of the presence of noise, which occurs in the measured signals. Whether environmental, human and/or instrumental noise, or indeed often all three simultaneously the geophysical data is uncertain and there always is under real conditions a level of "vagueness" or "uncertainty". Geophysical data intrinsically has a "fuzzy" nature and so fuzzy mathematics and generally fuzzy thinking about this data either in the processing or interpretation stages (or usually both) can help us to be much closer to real conditions. Klir and Floger (1988) introduce some useful terms to introduce fuzzy set theory:

- "Uncertainty" is the degree of conformity between the natures of data when compared to reality. This 'nature' is a qualitative description attached to the quantitative value and can be related to a concept that, for instance, geomorphologists and geologists have identified from field observations.
- "Inaccuracy" means the quantitative imperfection of any data value and can be considered as the difference between a measurement and its prediction estimated from a probabilistic model; and so effectively imperfection is the standard deviation of the probability law.
- "Ambiguity" can be due to the above-mentioned imperfections and naturally leads to the possibility of different interpretations such that the resolution of the conflict between two solutions can only be reduced by adding external constraints.
- Uncertainty in geophysical data is therefore inevitable and interpretation of parameters (such as seismic data, wire line logs, geological and lithological data, etc.) makes fuzzy set theory a very useful tool (Nikravesh et al. 2004). Common sources of geophysical noise are depicted in Fig. 3.1 and different types of sources of uncertainty in geophysical data are shown in Fig. 3.2.

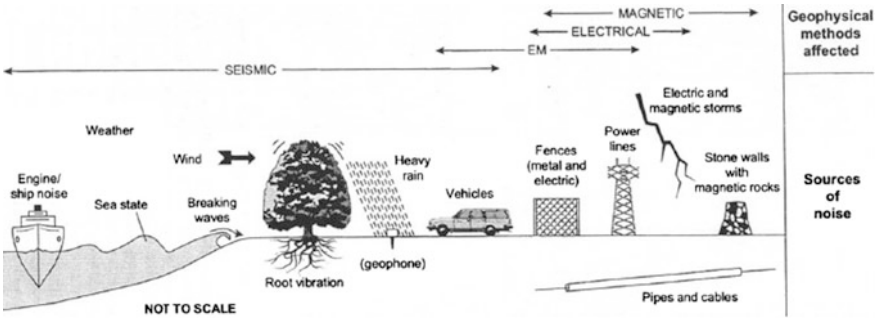


Fig. 3.1 Common sources of geophysical noise. http://www.slideshare.net/oncel/geophysics-overview2, slide17

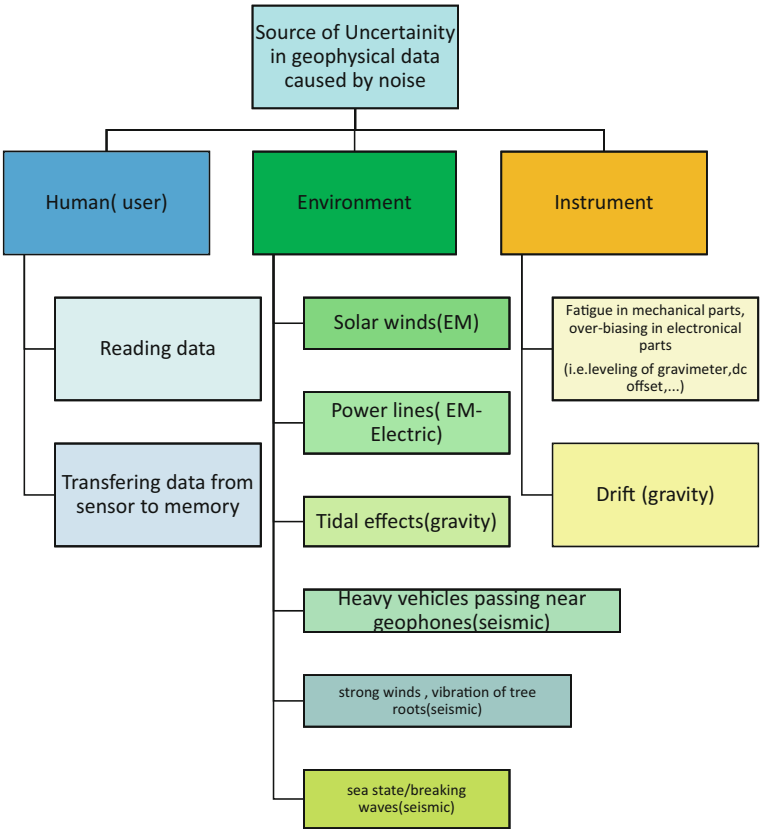


Fig. 3.2 Several sources of uncertainty in geophysical data

Because we apply simple models or non-precise models in the forward modeling of the geophysical causative bodies there is always a difference between the synthetic data produced based on assumed models and the real data.

For example in cavity detection using micro-gravity data in the forward modeling of cavities the common simple initial models are sphere, vertical cylinder and horizontal cylinder while the actual shape of the natural cavities are not 'Completely' like the assumed mentioned simple geometrical models but may be '*to some extent*' close to or very close to these shapes (Fig. 3.3a–c). Most cavities in karstic areas are connected in a some fashion to each other in what might be termed "banana holes" which are neither spherical nor cylindrical (Fig. 3.4a–c), and similarly while the 2D cross-section of the cavity is not completely a circle it might be near a circle; therefore the set of sections with the shape 'near circle' is a fuzzy set. This fuzzy set has members with shapes that are close to a complete circle with a degree or percentage of variation from perfection (see Fig. 3.5).

Banana Holes are oval chambers, generally less than 12 m diameter and 4 m deep which form at the top-surface of a fresh-water lens where vadose waters (occurring above the water table), meet the phreatic flows within the lens. They are tabular in aspect with a depth/width ratio of less than one. The median floor area is estimated to be 28 m^2 and the minimum 4 m^3 (Wilson et al. 1995). Banana holes are relatively small, relatively shallow caves having a subcircular shape in plan view. They are phreatic dissolution features formed at the top of an ancient groundwater lens (at the water table), away from the margins of a carbonate island (such as the Bahamas). Like flank margin caves, banana holes originally had no entrance connected to the surface. With erosional denudation of limestone bedrock surfaces, banana hole ceilings collapse, resulting in cave entrances. From a fuzzy point of view they can be considered to be members of the set of spherical cavities with a certain degree of membership while simultaneously belonging to the set of cylindrical cavities with another different degree of membership make in the 'Fuzzy' approach excellent tool for modeling these subsurface objects.

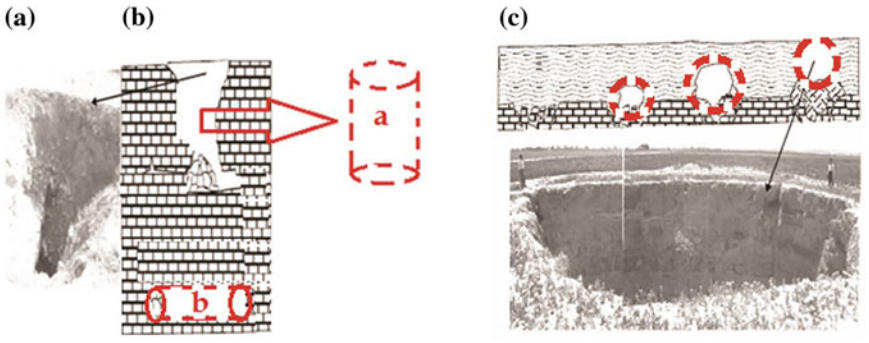


Fig. 3.3 Real cavities simulated with simple models: **a** vertical cylinder **b** horizontal cylinder **c** sphere (Hajian et al. 2012)

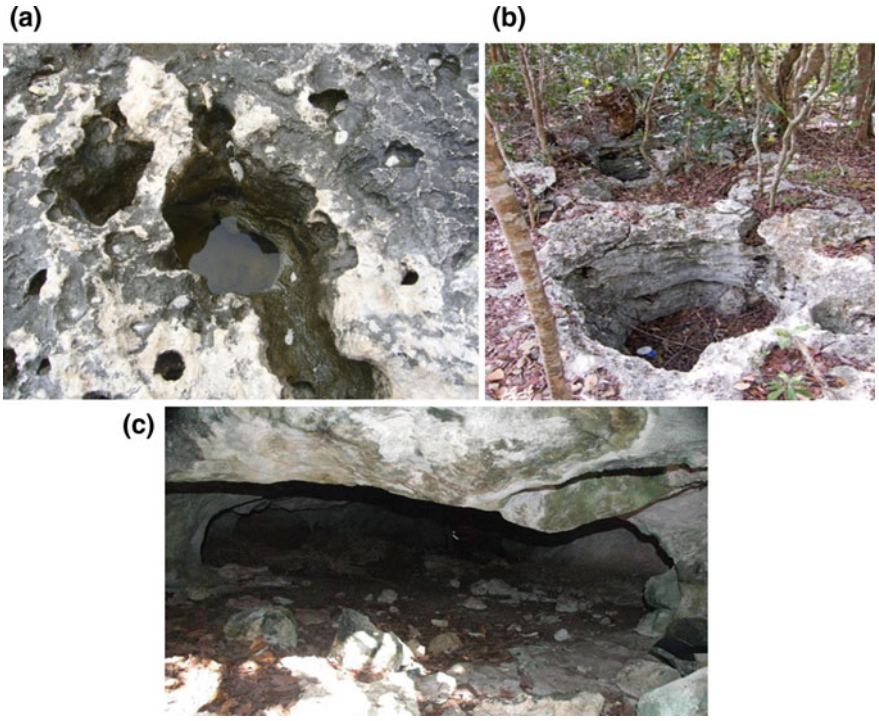


Fig. 3.4 Samples of Banana holes **a** near sea, filled with salt water **b** far distant from the shoreline **c** entrance of a cave. http://www.jsjgeology.net/San-Salvador,Bahamas-caves-karst.htm

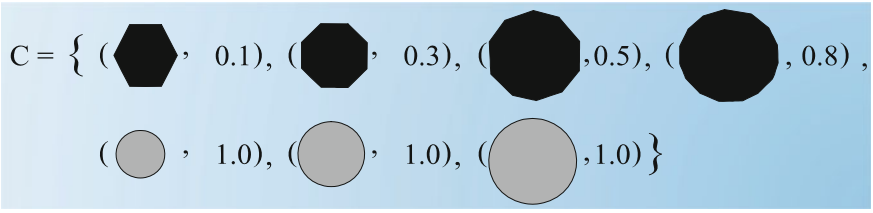


Fig. 3.5 An example of a fuzzy set of "approximately a circle"

As another example, when we describe the propagation of seismic waves in geological media, wave equation modeling is performed through the solution of partial differential equations (PDE) with deterministic coefficients, with boundary conditions, which are assumed to be very clear and crisp, and as both the deterministic coefficients and the crisp boundaries are idealized assumptions this leads to unrealistic inversion solutions. Using fuzzy coefficients for the solution of PDEs is in fact much more realistic and easy to parameterize (Table 3.1). An easy to

Table 3.1 Comparison of conventional PDEs and fuzzy PDEs in geophysics

Equation	Conventional PDE	Fuzzy PDEs
Laplace's equation in potential theory	$\nabla^2 \mathrm{v} = 0$ $\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 v}{\partial z^2} = 0$	$\nabla^2 \tilde{v} = 0$ (or $\nabla^2 \mathrm{V} = \tilde{0}$) $\frac{\partial^2 \tilde{v}}{\partial x^2} + \frac{\partial^2 \tilde{v}}{\partial y^2} + \frac{\partial^2 \tilde{v}}{\partial z^2} = 0$
Poisson's equation	$\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 v}{\partial z^2} = 4\mathrm{h}\pi\rho$	$\frac{\partial^2 \tilde{v}}{\partial x^2} + \frac{\partial^2 \tilde{v}}{\partial y^2} + \frac{\partial^2 \tilde{v}}{\partial z^2} \approx 4\mathrm{h}\,\pi\,\rho$
Seismic travelling wave	$\alpha^2 \underline{\nabla}^2 \phi - \frac{\partial^2 \phi}{\partial t} = -\frac{1}{\rho} F_P$	$\tilde{\alpha}^2 \nabla^2 \tilde{\varphi} - \frac{\partial^2 \tilde{\varphi}}{\partial a^2} = -\frac{1}{\rho} \tilde{F}_p$
Euler's equation of homogeneity	$(x - x_0)\frac{\partial V}{\partial x} + (y - y_0) + \frac{\partial V}{\partial y} + (z - z_0)\frac{\partial V}{\partial z} = -nV$	$(x - x_o)\frac{\partial \tilde{v}}{\partial x} + (y - y_o)\frac{\partial \tilde{v}}{\partial y} + (z - z_0)\frac{\partial \tilde{v}}{\partial z} = -n\tilde{v}$

understand example is when modeling seismic waves traveling in subsurface layers of the earth the boundaries are often assumed to be straight inclined surfaces (Fig. 3.7a) but in fact the boundary between a geological layer and another neighboring is more likely to be a curved or a 'fuzzy' line. In other words the boundary between adjacent strata is much more 'fuzzy' than crisp. It is obvious that Fig. 3.7b is much closer to the true natural conditions of geological structures.

All real-world geophysical data are intrinsically vague or fuzzy in nature and so during geophysical data processing and interpretation the fuzzy logic approach is very likely to be fruitful. Also in attempting to predict the onset or occurrence of geophysical events like earthquakes, landslides, volcano eruptions, in addition to the available methods in time series prediction i.e. regression, multi-variate analysis, on-linear extrapolation, neural networks etc., fuzzy approaches are very likely to be helpful and informative. An overview of various applications of fuzzy logic in geophysics is shown in Fig. 3.6.

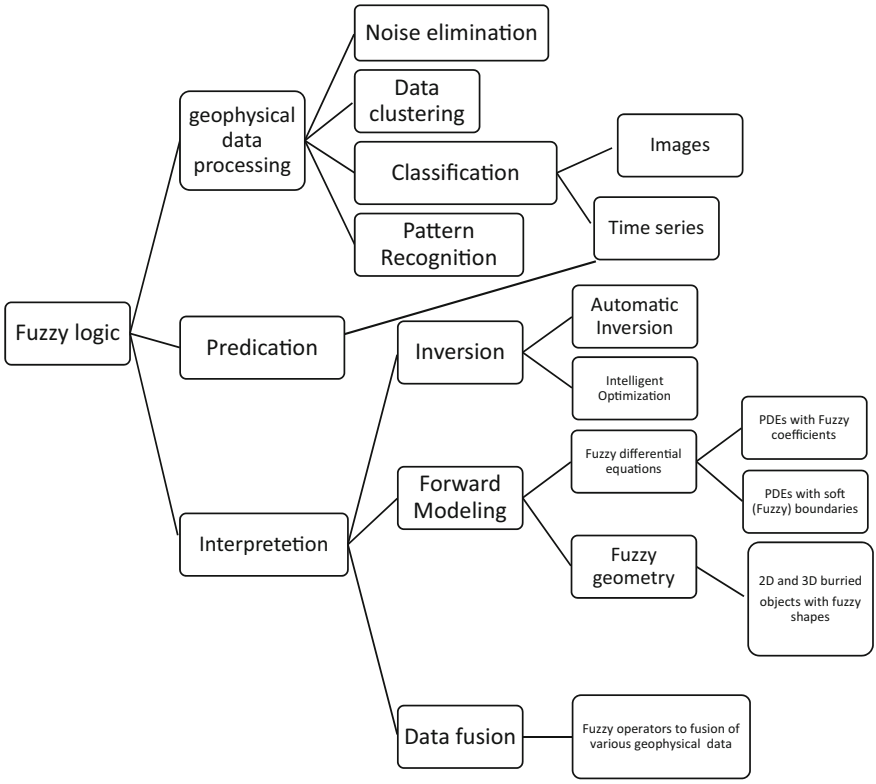


Fig. 3.6 Applications of fuzzy logic in various disciplines in geophysics

3.2.2 The Second Viewpoint

In most real cases the problem of finding the model, which fits best to the observed data, is problematic because of the intrinsic non-uniqueness of geophysical data. One way to overcome the challenge posed by non-uniqueness is to measure different complementary types of geophysical data over an area or the region and/or use which a priori information such as other geological studies of the area, well-core data or other similar additional constraints. The combination of geophysical data often leads to more realistic and precise solutions than when a single geophysical parameter is used alone. There are also some classical methods of geophysical data fusion such as GIS-based models which use different geophysical and/or geological data as layers of the GIS-Model and then integrate these layers to interpret the exploration data (Fig. 3.7).

A new way to implement the fusion of geophysical data and/or geophysical and geological data is by using fuzzy aggregator operations can merge of data of disparate different types or even natures. Furthermore in some cases there is a need to use both quantitative and qualitative geophysical, geological data simultaneously and especially in this case fuzzy operators can help us to gain more precise results which approach real conditions more closely. One of the problems faced when using different types of geophysical data for interpretation of the exploration area is that if we want to aggregate data is deciding how much weight should be applied for each type of data and/or how to interpret the situation when a specific type of data leads to an interpretation which conflicts with the interpretation of other types of data. One way is by using the OWA (Ordered Weighted Average)[1] which is a general well-known fuzzy aggregator.

In this chapter fuzzy logic is described as succinctly as possible with the focus on the concepts and their potential applications in geophysical problems and then some geophysical applications of fuzzy methods are explained in the following Chap. 4 Fuzzy logic attempts to describe through mathematics how human reasoning processes uncertain or imprecise information to make a decision or to draw a conclusion; this may prove to be a useful 'bio-algorithm' as an analogue to carrying out geophysical approaches. Three main concepts of fuzzy logic are described in this chapter (Fig. 3.8).

The first concept of fuzzy logic that will be more explained in this chapter is the fuzzy set which is the fundamental underpinning of this approach (Ghasem-Aghae

[1]In fuzzy logic the ordered weighted averaging (OWA) operators introduced by Yager (1988) define a class of 'mean type' aggregation operators. Mean operators such as max, arithmetic average, median and min, are members of this class and are commonly used in computational intelligence because they can model linguistically expressed aggregation instructions.

Formally, an **OWA** operator of dimension n is a mapping F: R_n R that has an associated collection of weights W = [w_1 w_2 …w_n] lying in the unit interval and summing to one and with $F(a_1, a_2, \ldots, a_n) = \sum_{j=1}^{n} w_j b_j$

Where b_j is the *j*th largest of the a_j. If we choose different weights we implement different aggregation operators.

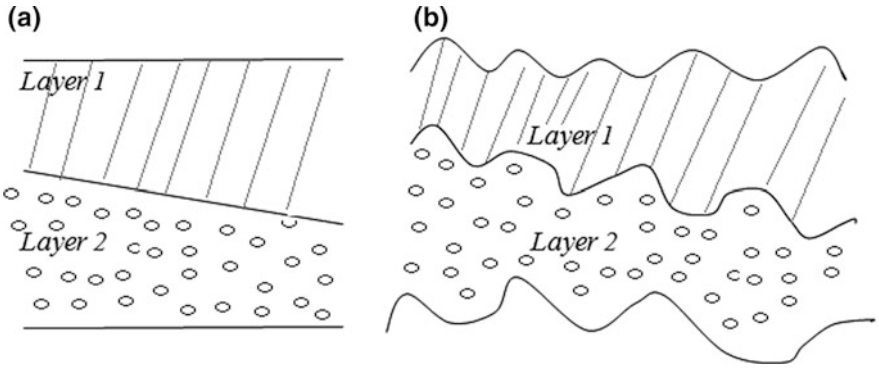


Fig. 3.7 **a** Conventional geometrical model used for waves traveling in layers. **b** Fuzzy geometrical model, a model with 'fuzzy' or 'soft' boundaries rather than crisp boundaries

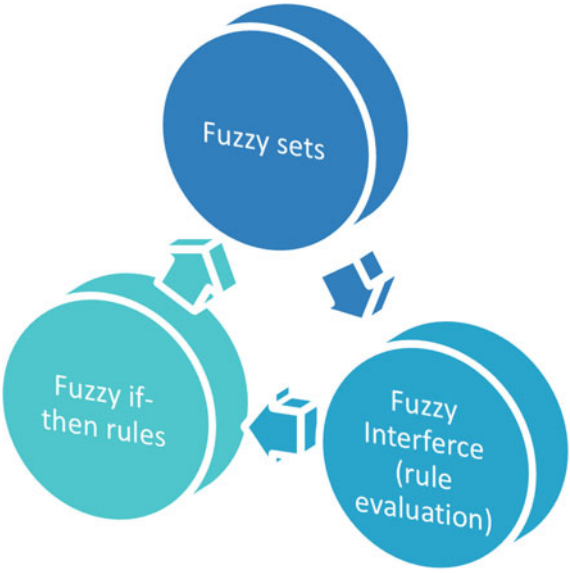


Fig. 3.8 Three main concepts of fuzzy logic

et al. 2005). Examples of fuzzy sets and linguistic variables are presented in the later sections. The second basic concept of fuzzy logic is the "***fuzzy if-then rule***". Fuzzy if-then rules use logic operators such as 'and' or 'or' to ***fuzzily*** link linguistic variables together to give a conclusion.

Examples of ***if-then fuzzy*** rules are given later. The final basic concept of fuzzy logic is the ***fuzzy interference system: FIS*** or rule evaluation. The two main methods of rule evaluation are known as Mamdani interference (Mamdani and Assilian 1975) and Sugeno interference (Takagi and Sugeno 1985). The output of Mamdani interference is a fuzzy set while the output of Sugeno interference is a constant value or a polynomial function.

3.2.3 *Geophysical Data Fusion Based on Fuzzy Logic Rules*

Often, different types of geophysical data are combined to enhance the accuracy of the interpretation, especially in those cases where there are no physical or empirical relations between the different types of measured data and there are no significant useful correlations. In these cases the expert interpreter plays a vital role in obtaining the most accurate interpretation through the combining of geophysical data of different type.

The expert must define rules, derived from his/her own experience as the basis of the combination of geophysical parameters to achieve the optimized and more accurate interpretation. Generally in the classic approach, the interpretation of geophysical data is carried out this way. Recently, more objective new methodologies based on fuzzy logic have been tested as replacements for expert interpretation in carrying out such operations by formulating the expert's rules as mathematical functions, which can be used to combine geophysical data into an improved interpretation (Grandjean et al. 2006).

Figure 3.9 shows how a 2D tomogram of P-wave velocity (Vp) is combined with a likelihood function given by the inversion process to produce a possibility cross-section indicating the areas where the medium may be fissured or fractured (Grandjean et al. 2006).

The motivation for using different types of geophysical data set and combining them (or geophysical data fusion) is to overcome the ambiguity in an ill-posed inversion when a single dataset is used and it can generate two or more solutions with the same degree of confidence.

Here, we think we should mention that there is a difference between the **probability** and the **possibility** which is often confused in the literature. A great mistake is in thinking that the degree of fuzziness of the data is identical to its probability but this not at all true because the probability is used if an event hasn't yet happened (and the value of probability shows how likely it is that the event will happen!) but the possibility is used after the occurrence of an event.

The "fuzzy degree" or the "fuzzy membership" indicates how much an 'occurred' event is similar to an 'expected' event. On the other hand, the degree of fuzzy membership of an element indicates the possibility that it belongs to the defined set (or subset). The essence of using the fuzzy degree is to quantify the degree of uncertainty which is always present in geophysical data.

3.3 Fuzzy Sets

Zadeh (1965) introduced the concept of fuzzy sets and possibility theory. These fuzzy sets pose a mathematical model which can quantify the 'vagueness', fuzziness or uncertainty of a data set. While the word 'fuzzy' implies the 'uncertainty' and 'inaccuracy' of the data it can also be used as a linguistic variable expressing vagueness, i.e. 'good', 'tall', 'bad', 'light', etc.

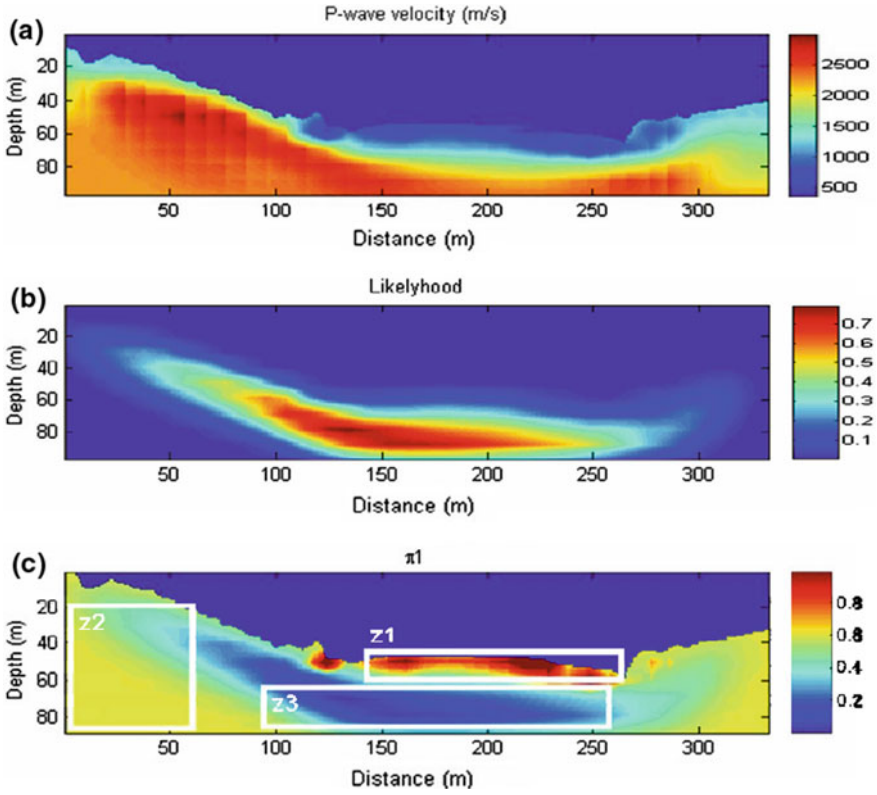


Fig. 3.9 Example of fuzzy logic rule for combining P-wave velocity (Vp) and its likelihood function (LVp) into a possibility map (p1) that the rocks are fissured (Grandjean et al. 2006)

In this part of the chapter the basic concepts of fuzzy set theory are explained. First, the fuzzy set is defined and described with some examples. Then the operations on the fuzzy set, fuzzy-cuts and some other concepts corresponding to ordinary sets are investigated in the context of technical geophysical reports and/or interpretations where we use vague variables: 'The depth of subsurface cavity is low', 'The hydrocarbon accumulated value is high', 'The rock electrical resistivity is very low', 'The clay content is to some extent high'.

3.3.1 The Concept of a Fuzzy Set

In our daily conversation we use a lot of 'vague' words such as John is 'tall', 'The car's speed is high', and 'The weather is very hot' which are not easily defined using classical sets. Fuzzy sets however can define these linguistic variables, i.e. the 'tall' persons set could be defined using classical sets but through fuzzy sets they are characterized in a manner closer to the real world.

A classical set is defined as:

$$\{X : P(x)\} \tag{3.1}$$

where P(x) is the property of the elements 'X' belonging to the set. When P(x) is well-defined the set is precisely defined. For example the set: "natural numbers greater than ten" is represented by: $\{x \in N : x > 10\}$, but if p(x) is not well-defined the related set or set elements cannot be defined.

It is obvious that when 'fuzzy' or 'vague' or inaccurate descriptions are involved a well-defined p(x) does not exist, for example when p(x) is "the natural numbers very much greater than 10", the 'very much greater' property is a fuzzy concept which can't be presented via a well-defined p(x). In these cases we can't designate the set elements definitely and explicitly.

As another example, consider the set of "very tall persons" the variable 'very tall' is not definitive. Generally the linguistic variables that we have used in our dialogues have a kind of vagueness and complexity. As another example we refer to An et al. (1991): a subset of gravity data, defined as B where: "Bouger gravity anomalies are greater than 31.52 mGal". We could define this as a subset with a single member, "31.52" mGal. But it is impractical to create an infinite number of subsets to represent uncertain information. A more logical step is to include a family of observations whose values are close to this particular measurement. The elements in this subset can have varying degrees of uncertainty.

The best way to assign the linguistic variables to a set is using the 'fuzzy' method.

Example 3.1 In the Table 3.2 the name and tallness of the people is illustrated in cm, determine the tall person amongst the people belonging to this set.

'Tallness' as a property is in contradiction with the well-defined set but if we stretch a point the set could be presented via a classical set (pseudo-set).

To distinguish this set we consider the concept of a deliberation index, for example the person whose stature is greater than 180 cm is tall. According to this feature the set of tall persons is defined as:

T = { Abtin, John, Peter, David, Andy}
X = Abtin, John, Peter, David, Andy
X = Alireza, Paul

$$X_T(x) = \{ \begin{matrix} 1 \\ 0 \end{matrix}$$

Where X_T (x) is the membership of X with a value of zero or 1. "1" means that X belongs to set 'T' and '0' means X doesn't belong to set 'T' (Fig. 3.10).

Table 3.2 Tallness value for a set of men

Name	Abtin	John	Peter	Alireza	David	Paul	Andy
Tallness (cm)	205	195	180	179	212	165	185

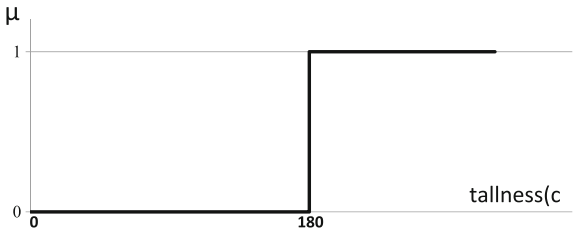


Fig. 3.10 Crisp set of 'Tallness'

As can be seen from this example Peter and John both belong to the 'tall' set (T) but the stature of Peter is 180 cm while John is 195 cm.

The difference between Peter's tallness and the defined deliberation index is 1 cm but for John's stature this difference is 15 cm. So the 'tallness' of John is greater than the 'tallness' of Peter. Classical sets can't show this difference because they have a 'crisp' boundary.

Also there is a problem here that one can consider another deliberation index for even taller people for example greater than 175 cm.

In the mentioned example 'Alireza' doesn't belong to the set of tall by a margin of only 1 cm, whereas 'Peter' does belong to the set of tall persons even though his stature is only 1 cm greater than 180 cm. This tolerance in the boundary of classical sets is very common and is unsuitable for the classification and/or clustering of data or the adequate characterization of the properties of a set of data.

Another disadvantage of using a defined deliberation index is the occurrence of different result for using a different index for different populations. For example, for a basketball team member the concept of 'tallness' is completely different than for the general population. In this case for example the deliberation index is defined as 195 cm where the member with stature greater than 195 cm is tall. The property of 'Tallness' consists of a vague and imprecise natural concept which cannot be represented using classical set theory but is easily possible via 'fuzzy' sets. The 'tallness' property is scalable or quantifiable if we define the degree of 'tallness' (Table 3.3).

The decimal number beside each person indicates the degree of tallness, i.e. Abtin is 80% tall and Peter is 30% tall. On the other hand 'Abtin' belongs to 'T'

Table 3.3 Tallness degree for a set of people (Soleimani and Hajian 2017)

Name	Abtin	John	Peter	Alireza	David	Paul	Andy
Degree of tallness (degree of membership to 'tall' set)	0.8	0.7	0.3	0.2	0.9	0	0.4

$\tilde{T}$ = {(Abtin, 0.81), (John, 0.7), (Peter, 0.3), (Alireza, 0.2), (David, 0.9), (Paul, 0), (Andy, 0.9)}

with a degree of 0.8 and 'Peter' belongs to 'T' with a degree of 0.3. In set 'T' the degree of membership for each element lies between '0' and '1'. Zero degree means that element doesn't belong to the set and 1 means that it completely belongs to set 'T'. In 'vague' concepts different people have their own different conclusion and/or interpretations about the degree of membership of an element to the defined fuzzy set but this difference has no contradiction in the context of fuzzy sets because with reference to local, temporal, personal evaluation the vague concept can have different meanings.

For example someone 30 years old among people more than 50 years age is considered as 'young' but this person among people with the age of 18–25 is not thought to be young or when a policeman says "The speed is high" on a Motorway it means the speed is more than 120 km/h but if it is about a local urban street it corresponds to a to the speed of more than 70 km/h. Also the degree of 'speed' depends on the time of the occurrence, i.e. day or at night.

3.3.2 Definition of a Fuzzy Set

In classical sets one way to represent a set is to illustrate it with its membership function (MF). In this method the MF is defined as below:

$$\chi_A : U \rightarrow \{0, 1\}$$

$$\chi_A(x) = \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases} \tag{3.2}$$

Here the membership degree is only 0 or 1 to extend this definition for gradual belonging in fuzzy sets the membership degree is given a value between 0, 1 and is defined as:

$$\tilde{A} = \{(x, \mu_A(x)) : \quad \mu_A(x) : X \rightarrow [0, 1]\} \tag{3.3}$$

where X is a collection of objects and μA (x) is called the membership function or degree of compatibility of x in A and μA (x) maps X to the membership space. The range of μA (x) is normally defined in the range [0, 1], where "0" expresses non-membership and "1" full membership.

So from a fuzzy logic point of view, elements of a fuzzy set $\tilde{A}$ are assigned a degree of belonging to it and are not confined to only 1 or zero. On the other hand an element may have a partial membership ranging from non-membership "0" to full membership "1". In general, the greater the value of the function with which the element x belongs to the fuzzy set $\tilde{A}$, the greater the evidence that the object x belongs to the category described by the set A.

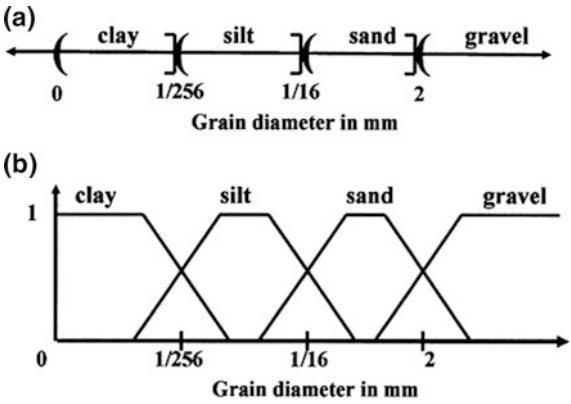


Fig. 3.11 Comparison of crisp-set **a** versus fuzzy-set **b** representation of the geologic variable "grain size" (Demicco and Klir 2004)

Example 3.2 We start here with an interesting example from the book "Fuzzy logic in geology" by Demicco and Klir (2004).

An example is of the classification of rock grains based on their diameter into: clay, silt, sand and gravel, which are terms, used to describe the 'size' of sedimentary particles (Fig. 3.11).

These terms are most commonly used for exact sets, so that a grain can only belong to one-grain size; therefore in this traditional view, a spherical grain of diameter 1.999 mm is sand whereas a grain with diameter of 3.001 is gravel. However, an alternative representation of the crisp set "sand" is:

$$\text{"Sand"} = \{\text{Particle}|0.0625 \text{ diameter } 2 \text{ mm}\}$$

Here we assign a valve of '1' to grain diameters that are members of the set "sand" and '0' to grain diameters that are not sand. One possible representation of the sedimentary site terms clay, silt, sand and gravel as a fuzzy set is shown in Fig. 3.11b (Demicco and Klir 2004).

In the fuzzy representation the range of membership is any value between [0, 1]. The hypothetical 1.999 and 3.001 mm diameter grains can be members of both sets: sand and gravel to a degree of about 0.5. Here the simple trapezoids are the membership functions (MF's).

Example 3.3 The second example is using a fuzzy approach to classify pyroclastic (volcanic ash deposits) deposits by a factor. This example was introduced by Cagnoli (1998).

In sorting terms, pyroclastic deposits are (Cas and Wright 1988):

(1) Very well sorted: $\delta\phi$ = [0 1]
(2) Well sorted: $\delta\phi$ = [1 2]
(3) Poorly sorted: $\delta\phi$ = [2 4]
(4) Very poorly sorted: $\delta\phi > 4$

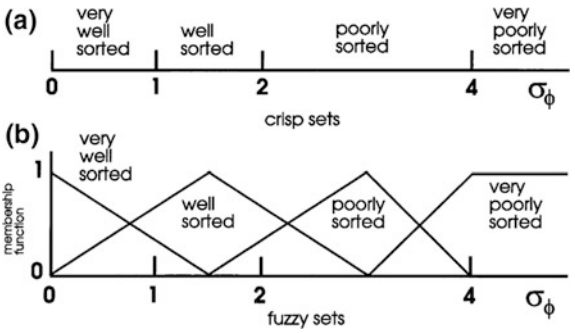


Fig. 3.12 **a** Sorting of a pyroclastic deposit with crisp sets (Cas and Wright 1988). **b** Sorting using fuzzy sets. The shapes and sizes of the membership functions are arbitrary and used only as an example to show the philosophy of fuzzy logic (Cagnoli 1998)

These crisp sets whose elements are either 100% yes or 100% no, with no intermediate possibilities which at first glance seems logical produces some strange situations. For instance, is a deposit with $\delta\phi = 2$ well sorted or poorly sorted? Is a rock with $\delta\phi = 1.1$ as well sorted as one with $\delta\phi = 1.5$? A more useful is possible using fuzzy sets with simple triangular membership functions (Fig. 3.12). A sample with $\delta\phi = 2$ would have intermediate values between [0 1] and belong to both the sets: "well sorted" and "poorly sorted"; similarly a sample with $\delta\phi = 1.1$ close to the boundary of the set and a sample with $\delta\phi = 1.5$ located in the center of the set would have different values of the MF.

This simple example shows how fuzzy sets can improve the description of the natural world by representing vagueness in a simple way. A sample with $\delta\phi = 2$ can be simultaneously well-sorted and poorly-sorted with different degrees without posing a contradiction in fuzzy logic.

Presentation of fuzzy sets

We could define a fuzzy set as:

$$\tilde{A} = \{(x, \mu_A(x)\}; \text{where x is the universal set}$$

Example 3.4 Let x = {1,2,3,4,5,6,7,8,9} be the universal set, describing the fuzzy set $\tilde{A}$: the numbers close to 5.

$\tilde{A}$ = {(1, 0.2) (2, 0.4) (3, 0.60 (4, 0.8) (5, 1) (6, 0.8) (7, 0.6) (8, 0.4) (9, −2)}

Example 3.5 Solve the example 3.4 for the case when $X = \mathbb{R}$ (x = real numbers).

Here the reference of real numbers is continuous. $2 \in \mathbb{R}$ but what then is the first real number after 2? So to determine the real number closest to 5 it impossible to assign a membership degree for each of the elements. In this case we choose to use a membership function which is continuous. One can assign:

$$\mu_A(x) = \begin{cases} \frac{x-3}{2} & 3 < x \leq 5 \\ \frac{7-x}{2} & 5 < x < 7 \\ 0 & \text{o.w.} \end{cases} \tag{3.4}$$

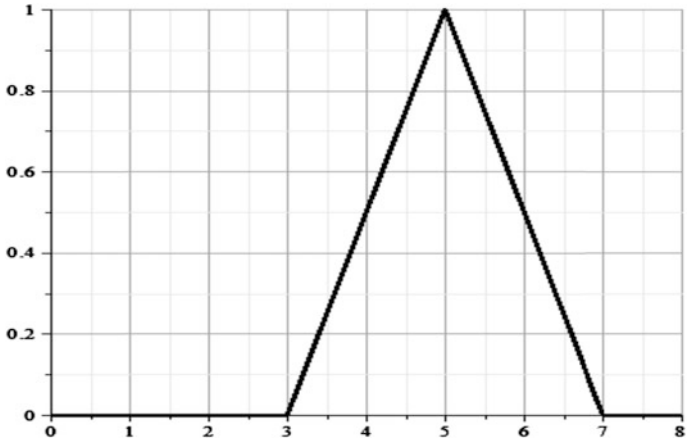


Fig. 3.13 The fuzzy set real number close to 5 with linear membership

This fuzzy (set real number close to 5) with linear membership is shown in Fig. 3.13.

Here to calculate the membership degree of "4" it is enough to put this number in the functional equation:

$$\mu_A(4) = \frac{4-3}{2} = 0.5$$

The assigned membership function is linear but it could easily enough be assigned as nonlinear: i.e. which is depicted in Fig. 3.14.

$$\mu_A(x) = \frac{1}{1+(x-5)^2} \tag{3.5}$$

Example 3.6 For a furnace heater temperature control we need to define a set of: low, medium and high heats. If the universal set is X = [50, 450], then determine the mentioned set with classical and fuzzy sets.

(a) Set via classical sets (Fig. 3.15):
(Low) l = [50, 100]
(Medium) M = [100, 250]
(High) H = [250, 450]
(b) Via fuzzy sets (Fig. 3.16):

$$\mu_M(x) = \begin{cases} \frac{x-90}{60} & 90 < x < 150 \\ 1 & 150 \le x \le 200 \\ \frac{260-x}{60} & 200 < x < 260 \end{cases} \tag{3.6}$$

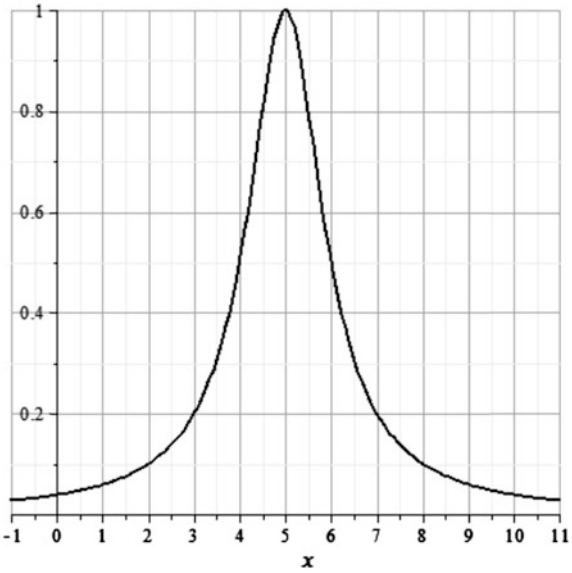


Fig. 3.14 Fuzzy set of real number close to 5 with nonlinear membership

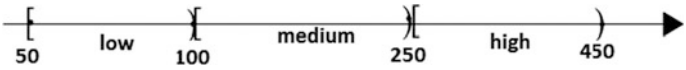


Fig. 3.15 Classical set presentation for temperature of a furnace

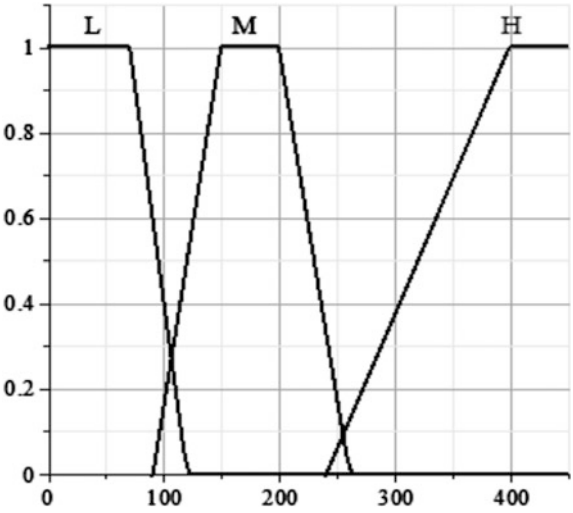


Fig. 3.16 The fuzzy memberships for low, medium and high temperature of a heater

$$\mu_H(x) = \begin{cases} 1 & x \geq 400 \\ \frac{x-240}{160} & 240 < x < 400 \end{cases}$$

$$\mu_L(x) = \begin{cases} 1 & x \leq 70 \\ \frac{120-x}{50} & 70 < x \leq 120 \end{cases}$$

Here, as an example, the temperature 100 °C belongs to the set 'low temp' with a degree of 0.4 and belongs to 'medium temp' with a degree of 1/6, its membership degree belongs more to the medium set than to the low set.

It is necessary to mention that for a fuzzy system there is usually a need to have a couple of fuzzy set definitions operational at any time with their processing done simultaneously.

3.3.3 *Different Types of Fuzzy Sets According to Their Membership Functions*

3.3.3.1 π-Shaped Fuzzy Sets

In this type of fuzzy set, the membership function starts from zero (or close to zero) and increases to a maximum (with a value less or equal to 1) and after this point it starts to decrease to zero (or close to zero).

The "trapezoid" (Fig. 3.17) and "triangle" (Fig. 3.18) and "bell" membership functions belong to this group of π-shaped fuzzy sets (Fig. 3.19). The π-shaped membership function is used to model fuzzy concepts such as: 'close to', 'approximately', 'medium', 'around'.

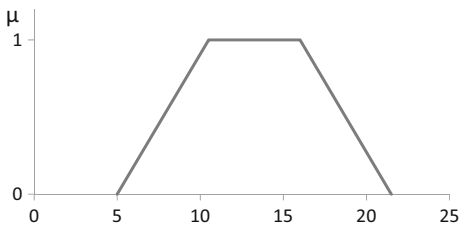


Fig. 3.17 Trapezoidal membership function

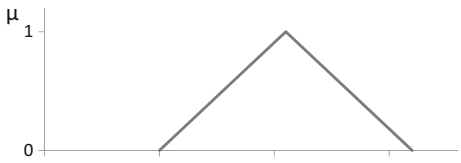


Fig. 3.18 Triangular membership function

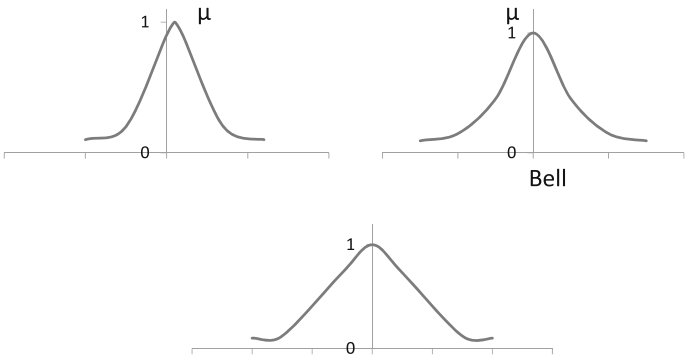


Fig. 3.19 A group of "π" shaped membership functions

3.3.3.2 Matlab Code for Non-conventional Pi-Shaped MF

PIMF(X, PARAMS) returns a matrix that is the Pi-shaped membership function evaluated at X. PARAMS = [A B C D] is a 4-element vector that determines the break points of this membership function.

The parameters 'A' and 'D' specify the "feet" of the curve, while 'B' and 'C' specify its "shoulders". This membership function is the product of SMF and ZMF:

PIMF(X, PARAMS) = SMF(X, PARAMS (1:2)).*ZMF(X, PARAMS (3:4))

Note that this Pi-MF could be asymmetric because it has four parameters. This is different from the conventional Pi = −MF which only uses two parameters. For example

```
x = (0:0.1:10)′;
y1 = pimf(x, [1 4 9 10]);
y2 = pimf(x, [2 5 8 9]);
y3 = pimf(x, [3 6 7 8]);
y4 = pimf(x, [4 7 6 7]);
y5 = pimf(x, [5 8 5 6])
plot(x, [y1 y2 y3 y4 y5]);
```

The result of the above commands is shown in Fig. 3.20.

Example 3.7 Let the universal set of the grades of an examination with full score of 20: X = [0 20] then the sets of 'medium grades' and the grades about 10 can be presented as shown in Fig. 3.21.

A general triangle membership function shown in Fig. 3.22 is easily defined by the equation below:

$$\mu(x) = \begin{cases} \frac{x-a}{m-a} & a < x \le m \\ \frac{b-x}{b-m} & m < x < b \end{cases} \tag{3.7}$$

where "α = m – a" is the left band width and "β = b − m" is the right band width.

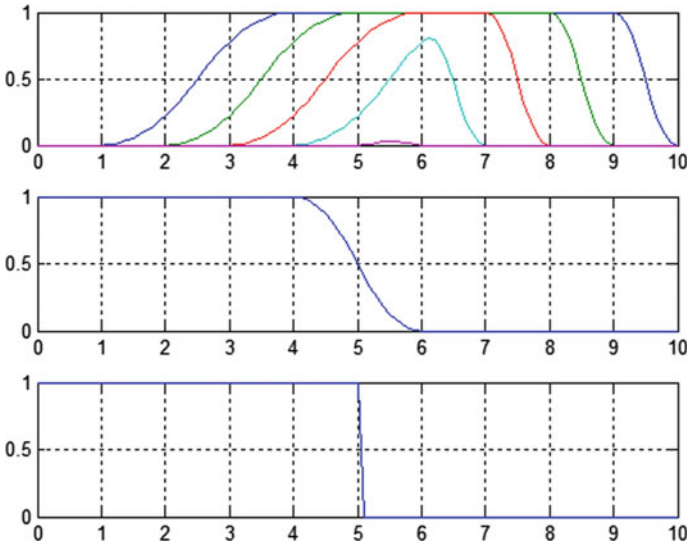


Fig. 3.20 Illustration of different Pi-shaped MFs using the 'pimf' command in MATLAB (Mathworks 2009)

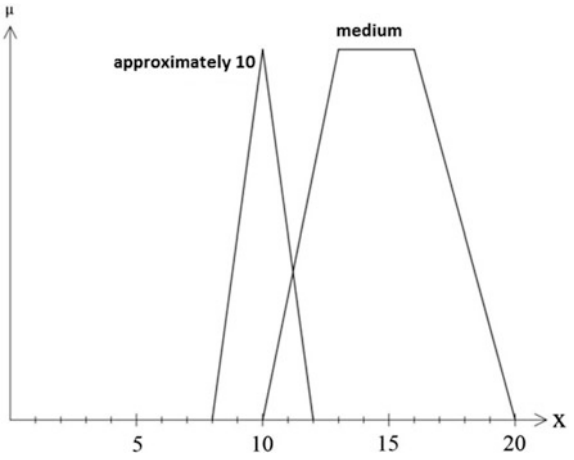


Fig. 3.21 Fuzzy set of medium grades of an examination

3.3.3.3 Matlab Code for a Triangular Membership Function (TRIMF)

TRIMF(X, PARAMS) returns a matrix which is the triangular membership function evaluated at X. PARAMS = [A B C] is a 3-element vector that determines the break points of this membership function. Usually we require $A \leq B \leq C$. Note that this MF always has a height of unity. If we require a triangular MF with a height less than unity, we would use TRAPMF (Mathworks 2009).

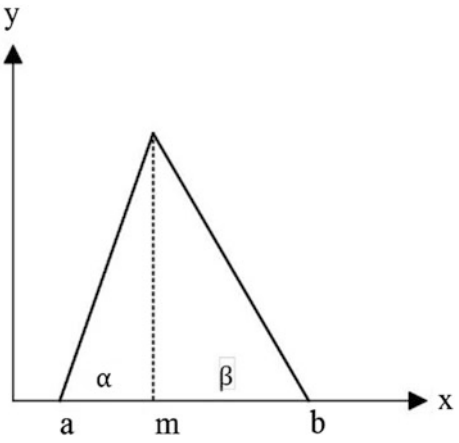


Fig. 3.22 Triangular membership function and its geometrical parameters

For example:

```
x = (0:0.2:10)′;
y1 = trimf(x, [3 4 5]);
y2 = trimf(x, [2 4 7]);
y3 = trimf(x, [1 4 9]);
subplot(211), plot(x, [y1 y2 y3]);
y1 = trimf(x, [2 3 5]);
y2 = trimf(x, [3 4 7]);
y3 = trimf(x, [4 5 9]);
subplot(212), plot(x, [y1 y2 y3]);
```

The results are illustrated in Fig. 3.23.

The triangle membership function is mostly used when the maximum membership occurs only in one point but when it happens in a domain the trapezoid membership function (Fig. 3.24) is used. The general format of a trapezoid membership function is given by the equations:

$$\mu(x) = \begin{cases} \dfrac{x-a}{m-a} & a < x \le m \\ 1 & m < x \le n \\ \dfrac{b-x}{b-m} & n < x \le b \end{cases} \tag{3.8}$$

3.3.3.4 Matlab Command for a Trapezoidal Membership Function

TRAPMF(X, PARAMS) returns a matrix which is the trapezoidal membership function evaluated at X. PARAMS = [A B C D] is a 4-element vector that determines the break points of this membership function.

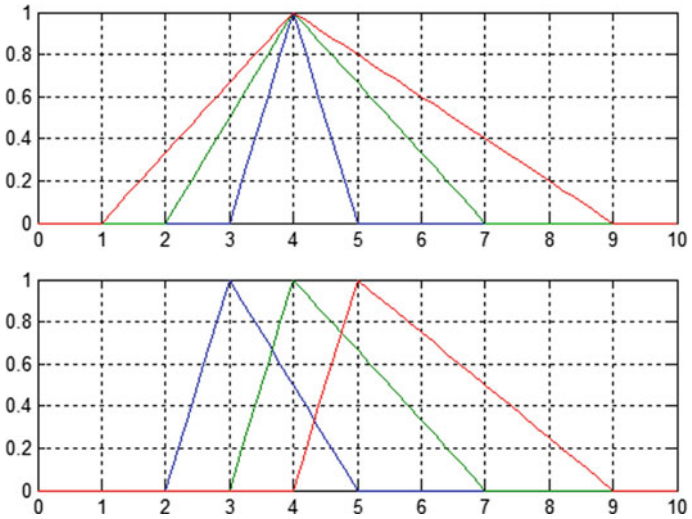


Fig. 3.23 Illustration of different triangular MFs using the 'trimf' command in MATLAB (Mathworks 2009)

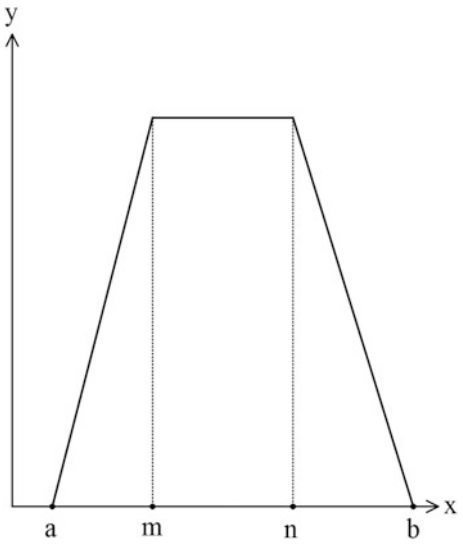


Fig. 3.24 Trapezoidal membership function and its geometrical parameters

We require that $A \leq B$ and $C \leq D$. If $B \geq C$, this membership function becomes a triangular membership function which could have a height less than unity (See the example below from Matlab Help).

For example:

```
x = (0:0.1:10)′;
y1 = trapmf(x, [2 3 7 9]);
y2 = trapmf(x, [3 4 6 8]);
y3 = trapmf(x, [4 5 5 7]);
y4 = trapmf(x, [5 6 4 6]);
plot(x, [y1 y2 y3 y4]) grid
```

The results are shown in Fig. 3.25.

When the π-shaped membership function is non-linear it will commonly be bell-shaped or Gaussian and is defined by a second order fractional or exponential function as:

Bell shaped:

$$\mu(x) = \frac{1}{1 + (x - a)^2} \tag{3.9}$$

Or Gaussian:

$$\mu(x) = \mathrm{e}^{-\frac{(x-a)^2}{k}} \tag{3.10}$$

The latter is a Gaussian Membership Function with

$$\mathrm{k} = 2\sigma^2, c = a \tag{3.11}$$

where c is the mean and σ is the standard deviation.

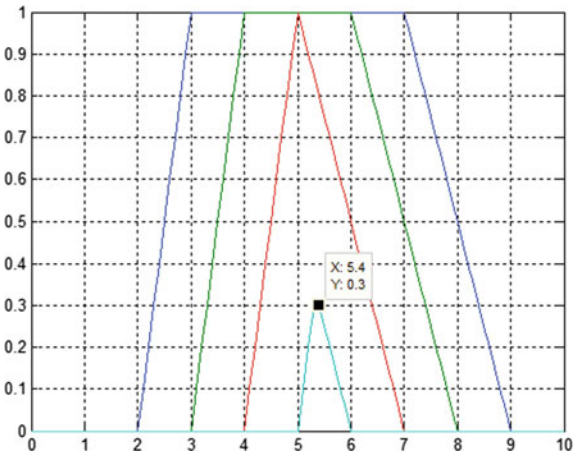


Fig. 3.25 Illustration of different trapezoidal MFs using the 'trapmff' command in MATLAB. *Source* Matlab Fuzzy Toolbox Tutorial Help, Mathworks (2009)

3.3.3.5 Matlab Code for Gaussian Curve Membership Function (GAUSSMF)

GAUSSMF(X, PARAMS) returns a matrix which is the Gaussian membership function evaluated at X. PARAMS is a 3-element vector which determines the shape and position of this membership function. Specifically, the formula for this membership function is:

$$\text{GAUSSMF}(\text{X}, [\text{SIGMA}, \text{C}]) = \frac{e^{-(x-c)^2}}{2\sigma^2} \tag{3.12}$$

For example (results are illustrated in Fig. 3.26) (Mathworks 2009):

```
x = (0:0.1:10)';
y1 = gaussmf(x, [0.5 5]);
y2 = gaussmf(x, [1 5]);
y3 = gaussmf(x, [2 5]);
y4 = gaussmf(x, [3 5]);
subplot(211); plot(x, [y1 y2 y3 y4]);grid;
y1 = gaussmf(x, [1 2]);
y2 = gaussmf(x, [1 4]);
y3 = gaussmf(x, [1 6]);
y4 = gaussmf(x, [1 8]);
subplot(212); plot(x, [y1 y2 y3 y4]);grid;
```

Figure 3.26 shows the results of the above Matlab code.

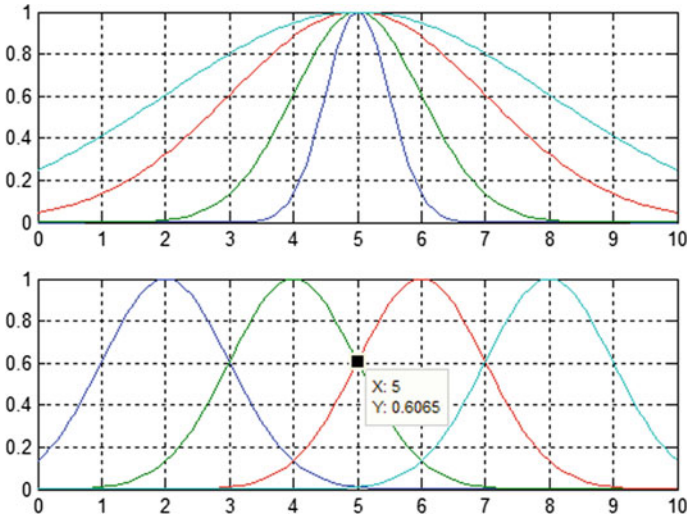


Fig. 3.26 Ilustration of different Gaussian MFs using the 'gaussmf' command in MATLAB (Mathworks 2009)

3.3.3.6 Two-Sided Gaussian Membership Function (Gauss2mf)

Another example for the group of pi-shaped MFs is the two-sided-Gaussian MF.

The Matlab command is gauss2mf with the format as below:

y = gauss2mf(x, params)
y = gauss2mf(x, [sig_1 c_1 sig_2 c_2])

As mentioned in the last section, the Gaussian function depends on two parameters sig and c as given by:

$$\text{gaussmf}(\text{X}, [\text{SIGMA}, \text{C}]) = \frac{\text{e}^{-(x-c)^2}}{2\sigma^2} \tag{3.13}$$

The function gauss2mf combines of two functions, one specified by sig1 and c1, determines the shape of the left hand curve and The second function determines the shape of the right hand curve. When $c_1 < c_2$, the gauss2mf function reaches a maximum value of 1 but otherwise, the maximum value is less than one. The parameters are listed in the order: [sig1, c1, sig2, c2] (Mathworks 2009).

Examples:

```
x = (0:0.1:10)′;

y1 = gauss2mf(x, [2 4 1 8]);
y2 = gauss2mf(x, [2 5 1 7]);
y3 = gauss2mf(x, [2 6 1 6]);
y4 = gauss2mf(x, [2 7 1 5]);
y5 = gauss2mf(x, [2 8 1 4]);
Plot(x, [y1 y2 y3 y4 y5]);
Grid
```

Results are shown in Fig. 3.27.

3.3.3.7 Generalized Bell Curve Membership Function and Its Matlab Code

Another example for the group of Pi-shaped MFs is a generalized bell-shaped membership function. GBELLMF(X, PARAMS) returns a matrix which is the generalized bell membership function evaluated at X. 'PARAMS' is a 3-element vector that determines the shape and position of this membership function.

Specifically, the formula for this membership function is:

$$\text{gbellmf}(\text{x}, [\text{a}, \text{b}, \text{c}]) = \frac{1}{1 + \left(\frac{x-c}{a}\right)^{2b}} \tag{3.14}$$

This membership function is a modification of the Cauchy probability distribution function. For example (for results see Fig. 3.28) (Mathworks 2009):

Fig. 3.27 Various two sided Gaussian MFs with different parametric values using 'Gauss2mf' command in MATLAB (Mathworks 2009)

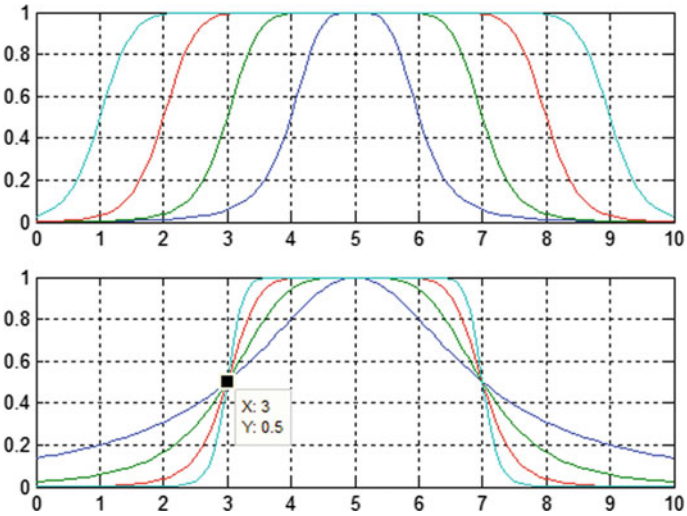


Fig. 3.28 Generalized bell-shaped MFs for different values of parameters a, b and c (Mathworks 2009)

```
x = (0:0.1:10)′;
y1 = gbellmf(x, [1 2 5]);
y2 = gbellmf(x, [2 4 5]);
y3 = gbellmf(x, [3 6 5]);
y4 = gbellmf(x, [4 8 5]);
subplot(211);
plot(x, [y1 y2 y3 y4]);
grid;
y1 = gbellmf(x, [2 1 5]);
y2 = gbellmf(x, [2 2 5]);
y3 = gbellmf(x, [2 4 5]); y4 = gbellmf(x, [2 8 5]);
subplot(212); plot(x, [y1 y2 y3 y4]);grid;
```

3.3.3.8 Z-Shaped Fuzzy Sets

In this type of fuzzy sets the membership value is maximum at first at a point or a domain and after that it decreases to zero (or close to zero). This kind of membership function is used to model fuzzy concepts such as: low, down, short, bad, etc. The shape of the membership function in this state is pseudo-triangular or pseudo-trapezoid (Fig. 3.29).

Example 3.8 The fuzzy set of 'bad' grades is a Z-shaped fuzzy set (Fig. 3.30).

3.3.3.9 Matlab Code for Z-Shaped Membership Function

ZMF(X, PARAMS) returns a matrix which is the Z-shaped membership function evaluated at X. PARAMS = $[X_1\ X_0]$ is a 3-element vector that determines the break points of this membership function. When $X_1 < X_0$, ZMF is a smooth transition

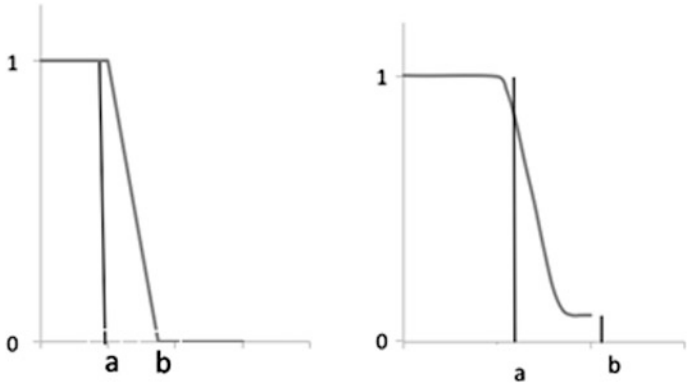


Fig. 3.29 Z shaped membership function

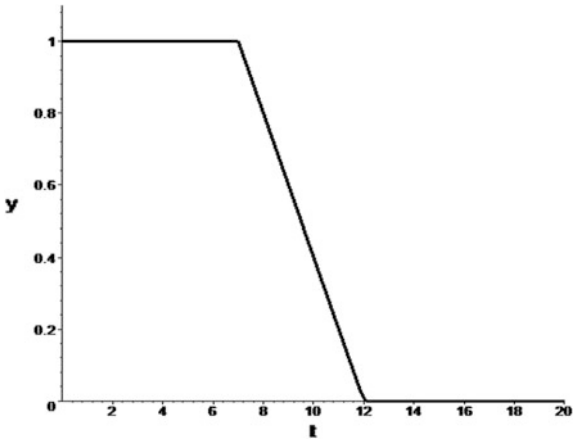


Fig. 3.30 An example for Z-shaped fuzzy set

from 1 (at X_1) to 0 (at X_0). When $X_1 \geq X_0$, ZMF becomes a reverse step function which jumps from 1 to 0 at $(X_0 + X_1)/3$.

For example (Mathworks 2009):

```
x = 0:0.1:10;
subplot(311);
plot(x, zmf(x, [2 8]));
subplot(312);
plot(x, zmf(x, [4 6]));
subplot(313);
plot(x, zmf(x, [6 4]));
set(gcf, 'name', 'zmf', 'numbertitle', 'off');
```

The results are illustrated in Fig. 3.31.

3.3.3.10 S Shaped Fuzzy Sets

In this type of fuzzy sets the membership value starts from zero (or close to zero) and increases to get to its maximum value at a point or a domain (Fig. 3.32). This type of membership function is used for modeling of fuzzy concepts like: high, very, good, tall, etc.

Example 3.9 The set of 'good' grades is an S-shaped fuzzy set:

$$\mu(x) = \begin{cases} 1 & 18 < x \leq 20 \\ \frac{x-16}{2} & 16 < x \leq 18 \end{cases} \tag{3.15}$$

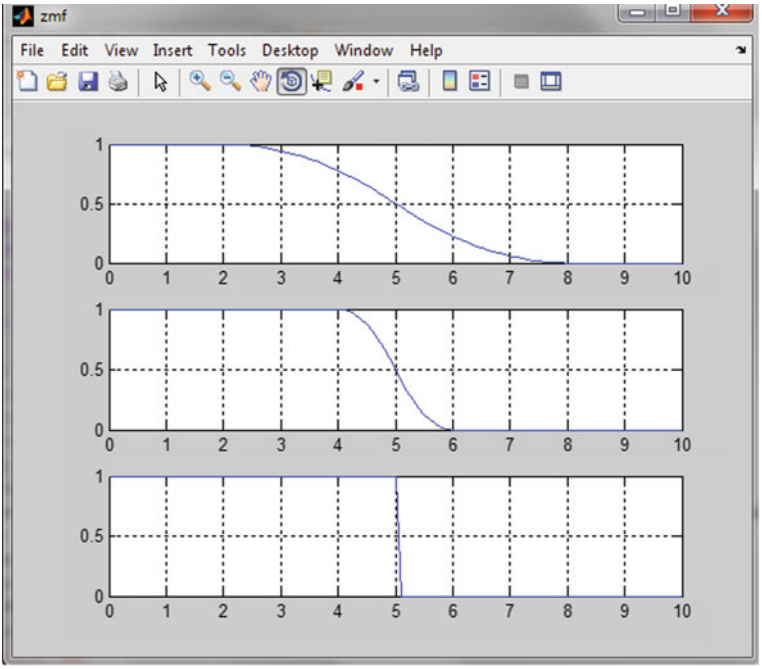
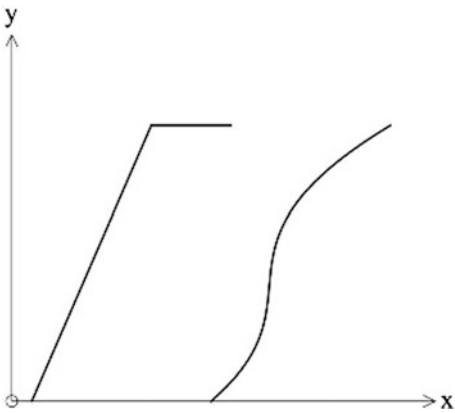


Fig. 3.31 Illustration of different z-shaped MFs using the 'zmf' command in MATLAB (Mathworks 2009)

Fig. 3.32 S shaped MFs



3.3.3.11 Matlab Code for S-Shaped Curve Membership Function (SMF)

SMF(X, PARAMS) returns a matrix which is the S-shaped membership function evaluated at X. PARAMS = $[X_0\ X_1]$ is a 3-element vector that determines the break points of this membership function (Fig. 3.33).

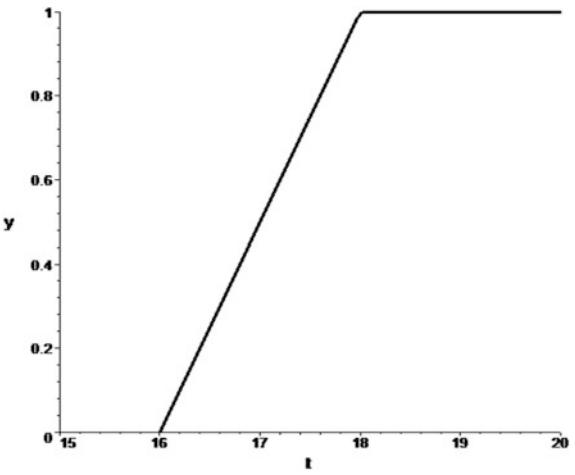


Fig. 3.33 The fuzzy set of 'good' grades is an S-shaped MF

When $X_0 < X_1$, SMF is a smooth transition from 0 (at X_0) to 1 (at X_1). When $X_0 \geq X_1$, SMF becomes a step function which jumps from 0 to 1 at $(X_0 + X_1)/3$.

For example (Mathworks 2009):

```
x = 0:0.1:10;
subplot(311); plot(x, smf(x, [2 8]));grid;
subplot(312); plot(x, smf(x, [4 6]));grid;
subplot(313); plot(x, smf(x, [6 4]));grid;
```

The results are shown in Fig. 3.34.

3.3.3.12 V Shaped Fuzzy Set

In this type of fuzzy set the membership function starts at its maximum value and decreases to zero (or close to zero) value, and then increases regaining its maximum value (Fig. 3.35). This kind of membership functions is used to model the fuzzy concepts like: 'not good not bad', 'not low not high', etc.

Example 3.10 The set of 'not good-not bad' grades is a V shaped fuzzy set.

$$\mu_v(x) = \begin{cases} \dfrac{10-x}{10} & 0 < x \leq 10 \\ \dfrac{x-10}{10} & 10 < x \leq 20 \end{cases} \tag{3.16}$$

A brief list of various membership functions with their typical shape and their MATLAB commands is shown in Fig. 3.36.

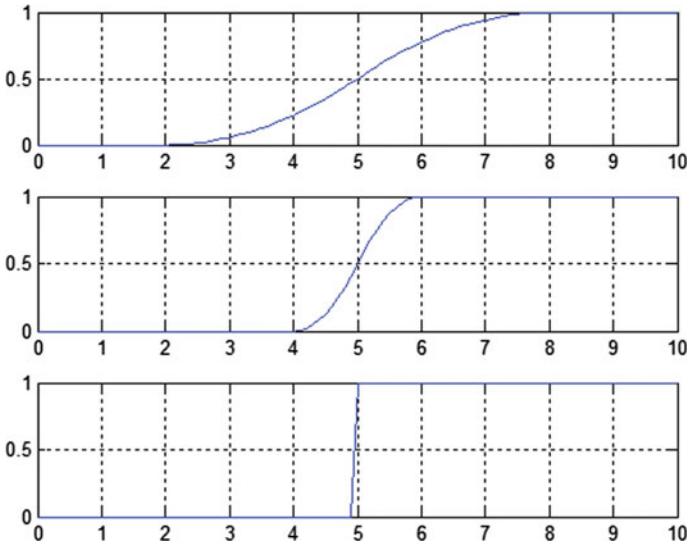


Fig. 3.34 S-shaped MFs for different parameters (x_o and x_1) using the 'smf' command in MATLAB

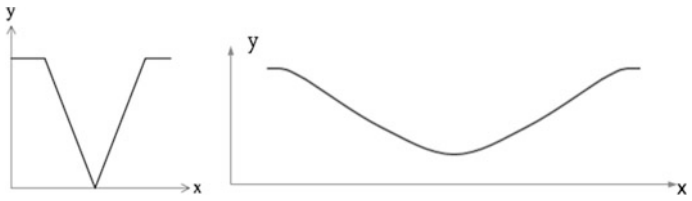


Fig. 3.35 V shaped fuzzy sets

3.3.4 Connecting Classical Set Theory to Fuzzy Set Theory

There are two main ways to connect classical sets to fuzzy sets:

(1) Using α-cut[2]
(2) Using the extension principle

[2]A: Given a fuzzy set A defined on a particular number α in the unit interval [0 1], the α-cut of A denoted by α_A is a crisp set containing all element of X whose membership in A are greater than or equal to α:

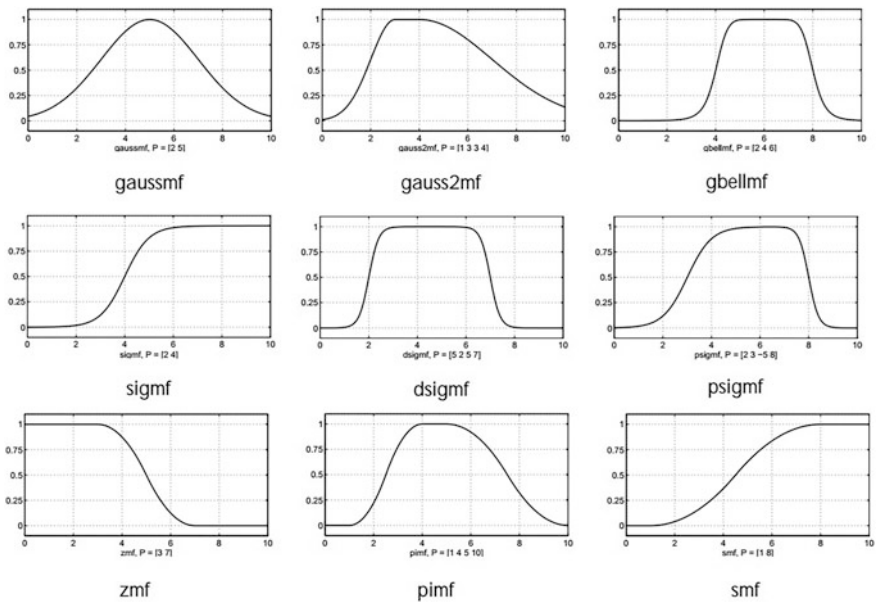


Fig. 3.36 Illustration of various MFs in Matlab (the abbreviations under each figure is its related Matlab command)

3.3.4.1 α-cut

The α-cut representation of fuzzy sets maps the properties of crisp sets, from classical set theory, into their fuzzy counterparts by requiring that the classical property be satisfied by all α-cuts of the fuzzy set concerned.

Properties extended from classical set theory into fuzzy set theory are called 'cut-worthy'. For instance, if the convexity of a fuzzy set requires that all α-cuts of a fuzzy convex set are classically convex, this concept of fuzzy convexity is cut-worthy.

Other important concepts are that of a fuzzy partition, fuzzy equivalence fuzzy compatibility, and various fuzzy orderings that are cut-worthy (Demicco and Klir 2004) but many properties of fuzzy sets are not cut-worthy and cannot be derived from classical set theory; in these cases the extension principle (following) is very useful.

3.3.4.2 Extension Principle

If we have a function f: X → Y where X and Y are crisp sets, we can say that the function is 'fuzzified' when it is extended to act on fuzzy sets defined on X and Y.

The 'fuzzified' function (F) maps $\mathcal{F}(X)$ into F(Y):

$$F : \mathcal{F}(X) \rightarrow F(Y) \tag{3.17}$$

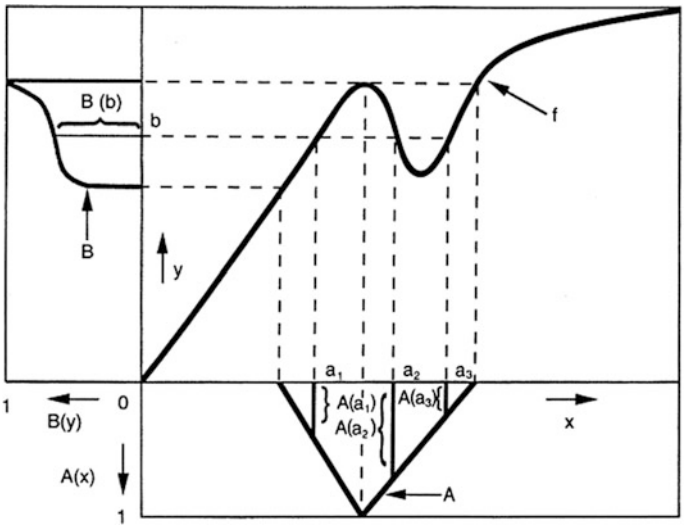


Fig. 3.37 Illustration of the extension principle (Demicco and Klir 2004)

Where $\mathcal{F}(X)$ and F(Y) denote the fuzzy power set (the set of all fuzzy subsets) of X and Y, a fuzzified version of, function must conform to F within the extended domain $\mathcal{F}(X)$ and F(y); this condition is guaranteed using the extension principle.

$$\alpha_A = \{\alpha | A(x) \geq \alpha\}$$

Therefore, B = F (A) is determined for any given fuzzy set $A \in F(x)$ through the formula:

$$B(y) = \max A(x) \quad x|y = f(x) \quad \text{for all } y \in Y. \tag{3.18}$$

When the maximum in equation (Eq. 3.18) does not exist we replace it with the supremum,[3] the inverse function:

$$F^{-1} : \mathcal{F}(Y) \rightarrow F(X) \tag{3.19}$$

of F is defined, according to the extension principle for any given $B \in F(Y)$ by the formula:

$$[F^{-1}(B)](x) = B(y) \tag{3.20}$$

The extension principle is illustrated in Fig. 3.37.

[3]The **supremum** of a subset *S* of a partially ordered set T is the least element in T that is greater than or equal to all elements of *S*, if such an element exists. Consequently, the supremum is also referred to as the *least upper bound* (https://en.wikipedia.org/wiki/Infimum_and_supremum).

3.4 Operations on Fuzzy Sets

In this section we introduce some standard operations on fuzzy sets such as intersection, complementation averaging, etc.

3.4.1 Standard Union

Assume that $\tilde{A}$ and $\tilde{S}$ are fuzzy sets of the universal set X then the union of A, B is:

$$(A \cup B)(x) = \max\{A(x), B(x)\} \tag{3.21}$$

Example 3.11 See Fig. 3.38 as an example of standard union.
Other common fuzzy union's relisted in Table 3.4.

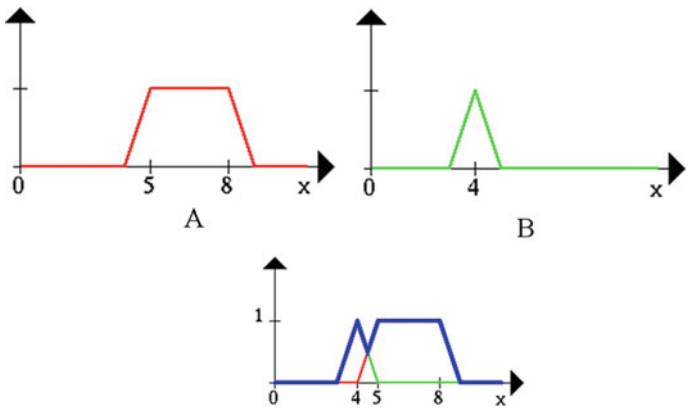


Fig. 3.38 Standard union of fuzzy sets $\tilde{A}$, $\tilde{B}$

Table 3.4 List of some non-standard fuzzy union operators

Name union operator	Formula
Algebra sum	U(a, b) = a + b − ab
Bounded sum	U(a, b) = min(1, a + b)
Drastic union	$U_{max} = \begin{cases} a \; when \; b = 0 \\ b \; when \; a = 0 \\ 1 \qquad o.w. \end{cases}$
Hamacher union	$U(a,b) = \frac{a+b-(2-\gamma)ab}{1-(1-\gamma)ab}$
Prad&Dubios union	$U\alpha(a,b) = \frac{a+b-a-b-min\{a,b,1-\alpha\}}{\max\{1-a,1-b,\alpha\}}$

3.4.2 Standard Intersection

$$(\tilde{A} \cap \tilde{B})(x) = \min\{A(x), B(x)\} \tag{3.22}$$

Example 3.12 See Fig. 3.39 as an example of standard intersection.

Other common fuzzy intersections are listed in Table 3.5 with their names and related formula 3.5 and it is obvious that when $w \to \infty$ the Yager operation will be a standard intersection and when $\gamma = 0$, the Hamacher operation will be the standard intersection.

3.4.3 Standard Complement

$$\left(\tilde{A}'\right)(x) = 1 - A(x) \tag{3.23}$$

Example 3.13 See Fig. 3.40 as an example of standard complementary.

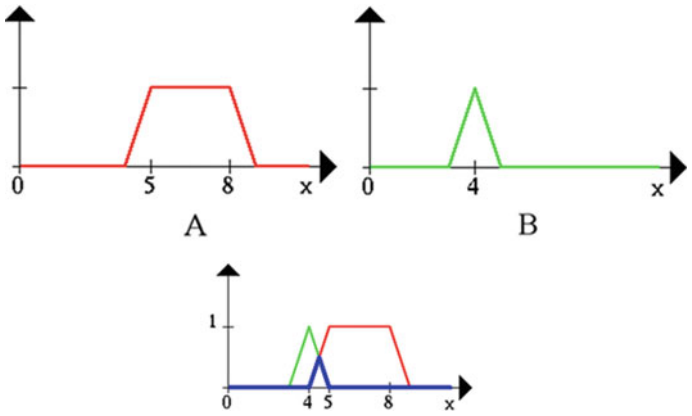


Fig. 3.39 Standard intersection of $\tilde{A}$, $\tilde{B}$

Table 3.5 List of some of non-standard fuzzy intersections

Name of intersection	Formula
Algebraic product	I(a, b) = a.b
Bounded difference	I(a, b)= max(0, A + b − 1)
Drastic intersection	$I_{min}(a,b) = \begin{cases} a \text{ when } b = 1 \\ b \text{ when } a = 1 \\ 0 \quad \text{otherwise} \end{cases}$
Yager intersection	i_w (a, b) = 1 − min{1, {$(1-a)^w$+$(1-b)^w$]$^{1/w}$} w ≥ 0
Hamacher intersection	$i_\gamma(a,b) = \frac{ab}{\gamma+(1-\gamma)(a+b-ab)} \gamma \geq 0$
Pradd and Dubios intersection	$i_\alpha(a,b) = \frac{ab}{\max\{\alpha,ab\}} \alpha \in [0,1]$

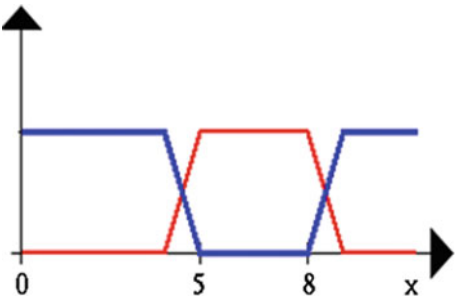


Fig. 3.40 Standard complementary of a fuzzy set Ã (red: Ã, blue: Ã′)

The classes of fuzzy intersections, i_w, and, fuzzy union, u_w, over the full range of these operations are defined for all a, b ∈ [0, 1] by the formula:

$$i_w(a,b) = 1 - \min\left\{1, [(1-a)^w + (1-b)^w]^{1/w}\right\} \tag{3.24}$$

$$u_w(a,b) = \min\left\{1, (a^w + b^w)^{1/w}\right\} \tag{3.25}$$

where w is a parameter whose range is $(0, \infty)$. These operations are often referred as the 'Yager' classes of intersection and union. When $w \to \infty$ the standard fuzzy operations are obtained. Other classes of fuzzy intersections and unions have been described and information on how new classes of fuzzy intersection and unions can be generated is given in Klir and Yuan (1995).

Example 3.14 Assume Ã, B̃ are fuzzy sets over {1, 2, 3, 4, 5} defined as below:

Ã = {(1, 0.3), (2, 0.7), (3, 1), (4, 0.7), (5, 0.3)}
B̃ = {(1, 0), (2, 0.2), (3, 0.5), (4, 0.8), (5, 1)}

Calculate their intersection and union.
Solve:

$$\tilde{A} \cup \tilde{B} = \{(1,0.3), (2,0.7), (3,1), (4,0.8), (5,1)\}$$

$$\tilde{A} \cap \tilde{B} = \{(1,0), (2,0.2), (3,0.5), (4,0.7), (5,3)\}$$

Example 3.15 Let Ã, B̃ fuzzy sets ((real numbers close to zero)) and ((real number close to 2)) respectively and with membership functions:

$$A(x) = \frac{1}{1+x^2}, \; B(x) = \frac{1}{1+(x-2)^2} \tag{3.26}$$

Find the intersection and union of Ã, B̃.

Solve: First we find the intersection point of the two functions:

$$A(x) = B(x) \rightarrow \frac{1}{1+x^2} = \frac{1}{1+(x-2)^2} \rightarrow x = 1 \tag{3.27}$$

Then we minimize and maximize for the points located before and after it to get the intersection and union (of $\tilde{A}$, $\tilde{B}$) respectively.

$$(A \cup B)(x) = \begin{cases} \max\left\{\frac{1}{1+x^2}, \frac{1}{1+(x-2)^2}\right\} x \leq 1 \\ max\left\{\frac{1}{1+x^2}, \frac{1}{1+(x-2)^2}\right\} x > 1 \end{cases} \tag{3.28}$$

So:

$$(A \cap B)(x) = \begin{cases} \frac{1}{1+x^2} x \leq 1 \\ \frac{1}{1+(x-2)^2} x > 1 \end{cases} \tag{3.29}$$

Also:

$$(A \cap B)(x) = \begin{cases} \min\left\{\frac{1}{1+X^2}, \frac{1}{1+(X-2)^2}\right\} & x \leq 1 \\ \min\left\{\frac{1}{1+X^2}, \frac{1}{1+(X-2)^2}\right\} & x > 1 \end{cases} \tag{3.30}$$

So:

$$(A \cap B)(x) = \begin{cases} \frac{1}{1+(x-2)^2} x \leq 1 \\ \frac{1}{1+x^2} x > 1 \end{cases} \tag{3.31}$$

Example 3.16 Assume that $\tilde{A}$ is the set of fuzzy numbers greater than 1 and $\tilde{B}$ is the fuzzy set of numbers very much greater than 1 with the membership functions below (Fig. 3.41).

$$A(x) = \frac{1}{1+(x-1)^{-1}}, B(x) = \frac{1}{1+10(x-1)^{-1}} \tag{3.32}$$

(a) Imply that $\tilde{B} \subseteq \tilde{A}$
(b) Calculate A′

Solve:

(a) It is obviously that for all $x \in \mathbb{R} : B(x) \leq A(x)$. Thus $\tilde{B}$ is a subset of A.
(b) $A'(x) = 1 - \frac{1}{1+(x-1)^{-1}} = \frac{1}{x}$

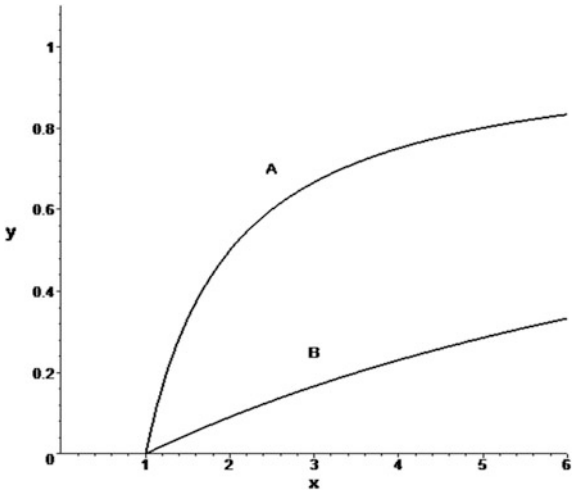


Fig. 3.41 Fuzzy numbers greater than one (A) and Very greater than one (B) (Soleimani and Hajian 2017)

3.4.4 Applications of the Intersection of Fuzzy Set

An important application of the intersection of fuzzy sets is in fuzzy decision making and fuzzy optimization. Where $\tilde{G}$ is the target set with the limitation property of $\tilde{C}$ then $\tilde{D} = \tilde{G} \cap \tilde{C}$ is called the fuzzy decision space and the optimized selection is the point of space in which it takes the maximum membership degree.

Example 3.17 A gravity interpreter is looking for a profile with negative residual anomaly on a gravity map of a power plant site to find large subsurface cavities. On the map there are 5 profiles with negative anomalies (A to E) close to the cooling towers, which are more at risk of subsidence.

$$\text{Negative Anomaly} : \tilde{\text{N}}\text{A} = \frac{0.1}{\text{A}} + \frac{0.3}{\text{B}} + \frac{0.6}{\text{C}} + \frac{0.8}{\text{D}} + \frac{1}{\text{E}}$$

$$\text{Close to Cooling Tower} : \text{CC}\widetilde{\text{T}} = \frac{0.8}{\text{A}} + \frac{0.2}{\text{B}} + \frac{0.4}{\text{C}} + \frac{0.7}{\text{D}} + \frac{0}{\text{E}}$$

On the other hand if the interpreter needs to find the profile which has two main properties: "negative anomaly" and "close to cooling tower" $\text{CC}\widetilde{\text{T}}$ then:

$$\text{N}\widetilde{\text{A}} \cap \text{CC}\widetilde{\text{T}} = \frac{0.1}{\text{A}} + \frac{0.2}{\text{B}} + \frac{0.4}{\text{C}} + \frac{0.7}{\text{D}} + \frac{0}{\text{E}}$$

The most important region that satisfies with both mentioned properties is region D.

3.4.5 *Fuzzy Averaging Operations*

The fuzzy intersections and fuzzy union operators cannot cover all aggregating operations and represent only aggregating operations that are associative, therefore another type of aggregation operations namely, averaging operation are defined.

In various ways they average membership functions of two or more fuzzy sets defined on the same universal set. Averaging operations run the gamut between the standard fuzzy intersection and standard fuzzy union. One class of averaging operations spanning the entire range between min and max operations is the generalized mean, which is defined below (Demicco and Klir 2004):

$$\mathrm{h_p}(\mathrm{a_1}, \mathrm{a_2}, \ldots, \mathrm{a_n}) = \left(\frac{a_1^p + a_2^p + \ldots + a_n^p}{n}\right)^{1/\mathrm{p}} \tag{3.33}$$

Where P is a parameter whose range is the set of all real numbers expect (0).

For P = 0 the function $\mathrm{h_p}$ is defined by the limit mentioned in Table 3.6.

The special conditions of P values with the equivalent means operator are listed in Table 3.6 and it seems sensible to consider the arithmetic mean as the standard averaging operation. Generalized means are symmetric averaging operations, but if symmetry is not necessary, we use the weighted generalized means $w_{h\mathrm{p}}$, defined as:

$$\mathrm{W_{hp}}(\mathrm{a_1}, \mathrm{a_2}, \ldots, \mathrm{a_n}, \mathrm{w_1}, \mathrm{w_2}, \ldots, \mathrm{w_n}) = \left(\sum_{i=1}^{n} w_i a_i^p\right)^{1/\mathrm{p}} \tag{3.34}$$

where $\mathrm{w_i}(\mathrm{i} \in \mathrm{n})$ are non-negative real number called weights for which

$$\sum_{i=1}^{n} w_i a_i^p = 1 \tag{3.35}$$

The weights are expressing the relative importance of the sets to be aggregated. More sophisticated classes of function which cover more than the three basic types of aggregation operations (min, max, mean) by Klir and Yuan (1995).

Table 3.6 Fuzzy averaging operation for special values of '*P*'

P value	Formula	Equivalent
$P = 0$	$\lim_{p\to 0} h_p(a_1, a_2, \ldots, a_n) = (a_1 a_2, \ldots, a_n)^{1/\mathrm{n}}$	Geometric mean
$P \to -\infty$	$\lim_{p\to -\infty} h_p(a_1, a_2, \ldots, a_n) = \min(a_1 a_2, \ldots, a_n)^{1/\mathrm{n}}$	Standard intersection
$P = 1$	$\mathrm{h_1}(a_1, a_2, \ldots, a_n) = \frac{a_1, a_2, \ldots, a_n}{n}$	Arithmetic mean
$P = -1$	$\mathrm{h_{-1}}(a_1, a_2, \ldots, a_n) = \frac{n}{\frac{1}{a_1} + \frac{1}{a_2} + \ldots + \frac{1}{a_n}}$	Harmonic mean

3.4.6 Matlab Codes for Fuzzy Operations

As an example here we have prepared a Matlab code for union, intersection and complement; this program takes two fuzzy sets A and B and gives any of the mentioned operations based on user selection.

```
% Enter two matrices
A=input ('Enter the first matrix')
B=input ('Enter the second matrix')
Option=input ('enter the option')
% Option 1 union
% Option 2 intersection
% Option 3 complement
If (options==1)
U=max (A/B);
fprintf('unions is ');
Print f (u)
end
If (option==2)
I=min (A/B),'
fprintf ('Intersection is');
Printf(I)
End
If (option==3)
Op=input ('Enter whether to find complement for first set or second set'),'
If(op==1)
[min]=size (u);
C=ones (m)-u;
Else
C=ones (m)-v;
end
printf('complement is')
Printf(c)
end
```

3.4.7 Other Operations on Fuzzy Sets

3.4.7.1 Non-Joint Union

$$A \oplus B = (A \cap B') \cup (A' \cap B) \tag{3.36}$$

$$(a \oplus b)(x) = \max\{\min(a, 1-b), \min(1-a, b)\} \tag{3.37}$$

Example 3.18 Let: $\tilde{A}$ = {(a, 0.2), (b, 0.7), (c, 1), (d, 0)}, $\tilde{B}$ = {(a, 0.5), (b, 0.3), (c, 1), (d, 0.1)}

Calculate $\tilde{A} \oplus \tilde{B}$.

Solve:

$$\tilde{A} \oplus \tilde{B} = \{(a, 0.5), (b, 0.7), (c, 0), (d, 0.1)\} \tag{3.38}$$

3.4.7.2 Non-cross Sum

$$\mu({}_a\Delta_b) = |\mu_a - \mu_b| \tag{3.39}$$

Example 3.19 If $\tilde{A}$ and $\tilde{B}$ are as the same as example 3.18 then:

$$\tilde{A}\Delta\tilde{B} = \{(a, 0.3), (b, 0.4), (c, 0), (d, 0.1)\}$$

3.4.7.3 Difference in Fuzzy Sets

In classical sets A − B = A ∩ B′, in fuzzy sets the standard difference of two sets $\tilde{A}$, $\tilde{B}$, belongs to universal set. X is: $\mu_{a-b} = \min(\mu_a, 1 - \mu_b)$.

Example 3.20 Let: $\tilde{A}$ = {(a, 0.2), (b, 0.7), (c, 1), (d, 0)}, $\tilde{B}$= {(a, 0.5), (b, 0.3), (c, 1), (d, 0.1)}

Then $\tilde{A} - \tilde{B}$ = {(a, 0.2), (b, 0.7), (c, 0.1), (d, 0)}

3.4.7.4 Bounded Difference

If $\tilde{A}$, $\tilde{B}$ are fuzzy sets belong to universal X then

$$(\tilde{A} \ominus \tilde{B})(x) = \max(0, a - b) \tag{3.40}$$

As an example for the $\tilde{A}$, $\tilde{B}$ sets defined in the last example,

$$(\tilde{A} \ominus \tilde{B})(x) = \{(a, 0), (b, 0.4), (c, 0), (d, 0)\}$$

3.4.7.5 Distance in Fuzzy Sets

The concept of distance is sometimes shown as subtraction but in the measurement math these two concepts are different (Fig. 3.42).

The distance between fuzzy sets has been widely studied and many definitions exist, differing in the type of information conveyed by their formal properties and the underlying mathematical approaches (Bloch 1999; Zwick et al. 1987). Most deal with the point-wise comparison of the membership functions but other distances introduce metrics into the considered space. In the first type of distance, complexity is linear in the cardinality of the space, so it is easy to compute. Typically, two fuzzy sets to be compared represent the same structure or a structure and a model and this is closely related to the notion of fuzzy similarity (if s is a similarity measure between fuzzy sets, then $1 - s$ is a distance, see e.g. (Bouchon-Meunier et al. 1986) for a review of fuzzy comparison measures. For instance, applications in model-based or case-based pattern recognition can make use of such distances. Definitions combining spatial distance and fuzzy membership comparison allow a more general analysis of structures in the considered space, where topological and spatial arrangement of the structures is important. This is enabled by the fact that these distances combine membership values at different points in the space and so take into account their proximity or distance in this measurement space. Here we introduce some fuzzy distance measures.

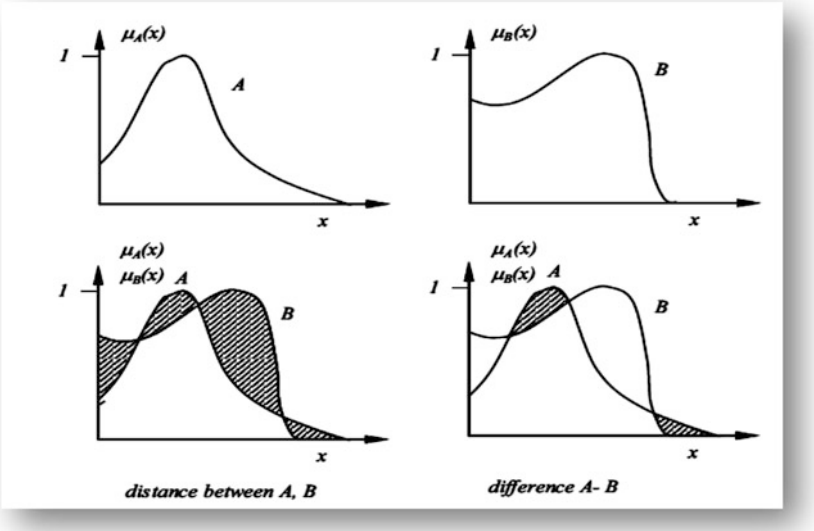


Fig. 3.42 Graphical comparison of 'Distance' and 'Difference' between two fuzzy sets

Hamming Distance

If $\tilde{A}$ and $\tilde{B}$ are fuzzy sets belong to universal set X then the Hamming distance of them is:

$$d\left(\tilde{A}, \tilde{B}\right) = \sum_{x \in X} |A(x) - B(x)| \tag{3.41}$$

Example 3.21 If: $\tilde{A}$ = {(1, 0.4, (2, 0.8), (3, 1), (4, 0)}, $\tilde{B}$= {(1, 0.4), (2, 0.3), (3, 0), (4, 0)}then

$$d(\tilde{A}, \tilde{B}) = 0 + 0.5 + 1 + 0 = 1.5$$

The Hamming distance has the properties below:

D (A, B) ≥ 0
d (A, B) = d(B, A)
d (A, C) $\leq$ d(A, B)+d(B, C)
d(A, A) = 0

If |X| = n then the Hamming distance is defined as:

$$\delta(\tilde{A}, \tilde{B}) = \frac{1}{n} d(\tilde{A}, \tilde{B}) \tag{3.42}$$

The non-cross sum is a kind of distance namely "symmetric distance":

$$\tilde{A} \Delta \tilde{B} = |a - b| \tag{3.43}$$

Euclidean Distance

When $\tilde{A}$, $\tilde{B}$ are fuzzy sets belonging to universal set X then the Euclidean distance is defined as:

$$e(\tilde{A}, \tilde{B}) = \sqrt{\sum_{x \in X} (A(x) - B(x))^2} \tag{3.44}$$

Example 3.22

$$\tilde{A} = \{(1, 0.4, (2, 0.8), (3, 1), (4, 0)\}$$

$$\tilde{B} = \{(1, 0.4, (2, 0.3), (3, 0), (4, 0)\}$$

$$e\left(\tilde{A}, \tilde{B}\right) = \sqrt{(0 + \left(0.5\right)^2 + 1^2 + 0\right)} = \sqrt{1.25 \approx 1.12}$$

The relative Euclidean distance is defined as:

$$\varepsilon(\tilde{A}, \tilde{B}) = \frac{e(\tilde{A}, \tilde{B})}{\sqrt{n}} \tag{3.45}$$

In general, however, the distance between fuzzy sets $\tilde{A}$, $\tilde{B}$ is defined by the Minovsky distance.

Minovsky Distance

When $\tilde{A}$, $\tilde{B}$ are fuzzy sets of universal set X, the Minovsky's distance is defined as:

$$d_w(\tilde{A}, \tilde{B}) = [\sum_{x \in x} |A(x) - Bx|^w]^{1/w}, w \in [1, \infty) \tag{3.46}$$

If w = 1 then d_1 ($\tilde{A}$, $\tilde{B}$) is Hamming distance and if w = 2 $d_2(\tilde{A}, \tilde{B})$ is the Euclidean distance.

Cartesian Product of Fuzzy Sets

Power of a fuzzy set $\tilde{A}$: Assume $\tilde{A}$ is a fuzzy set over universal set X, and then the m-power of this fuzzy set is defined as:

$$A^m(x) = \left(A_{(x)}\right)^m \tag{3.47}$$

On the other hand for the m-power of a fuzzy set the memberships of each element of the set is raised to power m. Power of fuzzy sets is a useful concept for linguistic variables.

3.4.8 Cartesian Product

Assume $\tilde{A}_1$, $\tilde{A}_2$, …,$\tilde{A}_n$ are fuzzy sets over X, X_2, …, X_n then the Cartesian product of them is defined as:

$$(A_1 * XA_2 * \ldots, A_n)(x, x_2, \ldots, x_n) = \min(A_1(x), A_2(x), \ldots, A_n(x_n) \tag{3.48}$$

Example 3.23 Assume $\tilde{A}$, $\tilde{B}$ are as:

$\tilde{A}$ = {(1, 0.3), (2, 0.7), (3, 1)}, $\tilde{B}$ = {(3, 0.7), (4, 1), (5, 0.7)}

Then:

$\tilde{A}$*$\tilde{B}$ = {((1, 3), 0.3), ((1, 4), 0.3), ((1, 5), 0.3), ((2, 3), 0.7), ((2, 4), 0.7), ((2, 5), 0.7), ((3, 3), 0.7), ((3, 4), 1), ((3, 5), 0.7)} (Table 3.7)

Table 3.7 Cartesian product of Ã and B (Soleimani and Hajian 2017)

A B	3	4	5
1	0.3	0.3	0.3
2	0.7	0.7	0.7
3	0.7	1	0.7

3.5 Fuzzy Relationships

One of the fundamental concepts is ***relationship*** with many applications in science and engineering. Relationships are used in approximated (or non-precise) reasoning, pattern recognition, control, etc. Fuzzy relationship is an extension of classical relationship. In classical relations, we only have two conditions:

First condition: there is a relationship between an element with another, and

Second condition: there is no relationship between an element with another

There are two crisp conditions; '0' if not included or '1' if the relationship is included but in fuzzy relationships the degree of relationship between two elements is included which means that how much of a relationship an element has to another has a value between zero and one. From the fuzzy point of view the degree of relationship is a fractional number between 0 and 1.

The classical relationship between X and Y is defined in a two dimensional space; but fuzzy relationship is defined in a multidimensional space, two elements (X, Y) are two of its dimensions and one other dimension is the membership degree of the relation between these two elements .i.e. X R Y in classical set shows x has relation R with Y, but X $\tilde{R}$ Y from fuzzy relation viewpoint means X how much has relation R with Y and this may be from zero to one. In this part the fuzzy relationships and operations over them are examined.

3.5.1 Definition of Fuzzy Relationship

3.5.1.1 Definition of a Classical Relationship

A classical relationship from classical set A to classical set B is a subset of Cartesian A*B and is presented as:

$$X_R : A^*B \rightarrow [0, 1]]$$

$$X_R(x, y) = \begin{cases} 1 & (x, y) \in R \\ 0 & (x, y) \notin R \end{cases} \tag{3.49}$$

where R means the relationship.

Example 3.24 Assume A = {1, 2, 3}, B = {a, b, c}. The relationship R is:

$R = \{(1,a),(1,c),(2,b),(3,b),(3,c)$

This relationship representation via matrix or index function is an below:

$$R = \begin{matrix} \\ 1 \\ 2 \\ 3 \end{matrix} \begin{matrix} \begin{matrix} a & b & c \end{matrix} \\ \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} \end{matrix}$$

As can be seen in the matrix, the availability of a relationship between each of two elements is shown with value 1 and otherwise is 0, but in fuzzy relations this value is represented by a number between zero and one.

According to the fact that relationship is a kind of set the representation of index function and the operation are the same as for sets.

3.5.1.2 Definition of a Fuzzy Relationship

A fuzzy relationship $\tilde{R}$ is defined from set x to set Y as below:

$$\tilde{R} = \{((x,y), \mu_R(x,y) | \mu_R(x,y) : X * Y \rightarrow [0,1]\} \tag{3.50}$$

where μ_R is the membership function of element (x, y) and means the degree of relationship between element x and y.

Example 3.25 Assume X = {a, b} Y = {c, d, e} which are the set of cities and the relationship R is the quality of the main connecting road between each of two cities. Then this fuzzy relationship could be as below (Table 3.8):

This means for example that the quality of the main connection road between city 'a' and c is 90% while the quality of the main connection road between city b and d is 60%.

Example 3.26 Assume x = {1, 2, 3}, y = {3, 4, 5} the relationship "very much smaller" i.e. is as shown in Table 3.9.

Table 3.8 Matrix presentation of Fuzzy relationship in example 3.25 (Soleimani and Hajian 2017)

$\tilde{R}$	C	D	e
A	0.9	0.8	0.7
B	0.8	0.6	0.4

Table 3.9 Fuzzy relationship "very much smaller" for "x" very much smaller than y

$\tilde{R}$	3	4	5
1	0.6	0.8	1
2	0.4	0.6	0.8
3	0	0.4	0.6

Or in Matrix format:

$$R = \begin{matrix} \\ 1 \\ 2 \\ 3 \end{matrix} \begin{matrix} \begin{matrix} 3 & 4 & 5 \end{matrix} \\ \begin{bmatrix} 0.6 & 0.8 & 1 \\ 0.4 & 0.6 & 0.8 \\ 0 & 0.4 & 0.6 \end{bmatrix} \end{matrix}$$

$\tilde{R}$ (1, 4) = 0.8 means that the degree of very much smaller than 4 is 0.8 for the element 1 or the number 1 is 80% very much smaller than 4.

Example 3.27 If $x = y = \mathbb{R}$ then the fuzzy relationship $\tilde{R}$: very much smaller can be defined as:

$$\mu_R(x, y) = \begin{cases} \frac{1}{1+(y-x)^{-2}} & x < y \\ 0 & x \geq y \end{cases} \tag{3.51}$$

Example 3.28 If $x = y = \mathbb{R}$ then the relationship "approximately equal" can be defined as: $\mu_R(x, y) = e^{-(x-y)^2}$.

3.5.1.3 General Representation of Fuzzy Relationship

Generally a fuzzy relationship over $X_1 * X_2 * \ldots * X_n$ is defined as:

$$\tilde{R} = \{((x_1, x_2, \ldots, x_n), \mu_R(x_1, x_2, \ldots, x_n)) | \mu_R : x_1, x_2, \ldots, x_n \rightarrow [0, 1]\} \tag{3.52}$$

Hereafter to present the membership function of a fuzzy relationship

We use $R(x_1, x_2, \ldots, x_n)$ instead of $\mu_R(x_1, x_2, \ldots, x_n)$ and when $x_1 = x_2 = \ldots$, $x_n = x$ then $\tilde{R}$ is defined as a fuzzy relationship over X.

Example 3.29 Assume $x = y = z = \mathbb{R}$ the $R(x, y, z) = e^{-k\left(x^2+y^2+z^2-r^2\right)}$ is defining the geometrical shape near to a sphere or the fuzzy relationship of the points with the same approximate distance of "r" from the general X point.

3.5.2 Domain and Range of Fuzzy Relationship

Assume $\tilde{R}$ is a fuzzy relationship over x*y then its domain and range are defined as:

$$\text{Dom}(R(x, y)) = \max_y R(x, y) \tag{3.53}$$

$$R_a(x, y) = \max_x R(x, y) \tag{3.54}$$

Table 3.10 Fuzzy relationship $\tilde{R}$ in example 3.30 (Soleimani and Hajian 2017)

$\tilde{R}$	10	20	30
1	1	0.5	0.2
2	0	1	0.3
3	0.4	0.7	1

Example 3.30 Assume x = {1, 2, 3}, y = {10, 20, 30} and fuzzy relationship $\tilde{R}$ on x, y is (Table 3.10):

Then:

$$\mathrm{dom}(\tilde{R}) = \{(1,1),(2,1),(3,0.7)\}$$
$$\mathrm{Ra}(\tilde{R}) = \{(10,1),(20,1),(30,0.3)\}$$

3.5.3 *Operations on Fuzzy Relationships*

As we mentioned, before, a relationship is a kind of set so an operation on fuzzy relationships is the same as an operation on fuzzy sets.

3.5.3.1 Standard Union of Fuzzy Relationships

Assume $\tilde{R}$, $\tilde{S}$ the fuzzy relations over X*Y then the standard union of two fuzzy relations is defined as:

$$(R \cup S)_{(X,y)} = \max\{R(x,y), S(x,y)\} \tag{3.55}$$

3.5.3.2 Standard Intersection of Fuzzy Relationships

Assume $\tilde{R}$, $\tilde{S}$ are fuzzy relationships over X*Y then the standard intersection of $\tilde{R}$, $\tilde{S}$ is defined as:

$$(\tilde{R} \cap \tilde{S})_{(X,y)} = \min\{R(x,y), S(x,y)\} \tag{3.56}$$

3.5.3.3 Standard completion Is Defined as

$$R'(x,y) = 1 - R(x,y) \tag{3.57}$$

Example 3.31 Let $\tilde{R}$, $\tilde{S}$ be the fuzzy relationship over X*Y with the memberships shown in Tables 3.11 and 3.12 respectively. The standard union $\tilde{R} \cup \tilde{S}$, intersection $\tilde{R} \cap \tilde{S}$ and completion $\tilde{R}'$, are calculated and represented in Tables 3.13, 3.14, 3.15, respectively.

Table 3.11 $\tilde{R}$ fuzzy relationship in example 3.31

$\tilde{R}$	2	3	4
1	0.3	0.6	0.9
2	0	0.3	0.6
3	0	0	0.3

Table 3.12 $\tilde{S}$ fuzzy relationship in example 3.31

$\tilde{S}$	2	3	4
1	0.2	0	0
2	1	0.2	0
3	0.2	1	0.2

Table 3.13 Standard union of $\tilde{R}$, $\tilde{S}$ fuzzy relations

$\tilde{R} \cup \tilde{S}$	2	3	4
1	0.3	0.6	0.9
2	1	0.3	0.6
3	0.2	1	0.3

Table 3.14 Standard intersection of $\tilde{R}$, $\tilde{S}$ fuzzy relations

$\tilde{R} \cap \tilde{S}$	2	3	4
1	0.2	0	0
2	0	0.2	0
3	0	0	0.2

Table 3.15 Standard completion of $\tilde{R}$

$\tilde{R}'$	2	3	4
1	0.7	0.4	0.1
2	1	0.7	0.4
3	1	1	0.7

3.5.4 Projection of Fuzzy Relationship and Cylindrical Extension

3.5.4.1 Projection of Fuzzy Relations

Assume $\tilde{R}$ is a fuzzy relationship over X*Y then the projection of this relationship over each of X and Y sets is defined as (see Fig. 3.43):

$$Proj[R, X] = \{(x, \max(x, y))\} \tag{3.58}$$

$$Proj[R, Y] = \{(y, \max(x, y))\} \tag{3.59}$$

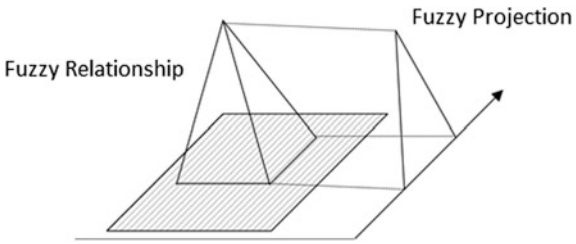


Fig. 3.43 Illustration of projection of fuzzy relation (Soleimani and Hajian 2017)

3.5.4.2 Cylindrical Extension

Assume $\tilde{A}$ is a fuzzy set over x. Then the cylindrical extension of $\tilde{A}$ over x, y is a fuzzy relationship defined as:

$$C(A) = \begin{Bmatrix} & y_1 & y_2 & y_m \\ x_1 & A(x_1) & A(x_1)\ldots & A(x_1) \\ x_2 & A(x_2) & A(x_2)\ldots & A(x_2) \\ . & & & \\ . & & & \\ . & & & \\ . & & & \\ x_n & A(x_n) & A(x_n)\ldots & A(x_n) \end{Bmatrix} \tag{3.60}$$

And if $\tilde{B}$ is a fuzzy set over y then cylindrical extension from $\tilde{B}$ over X*Y is a fuzzy relationship defined as:

$$C(B) = \begin{Bmatrix} & y_1 & y_2 & y_m \\ x_1 & B(y_1) & B(y_1)\ldots & B(y_1) \\ x_2 & B(y_2) & B(y_2)\ldots & B(y_2) \\ . & & & \\ . & & & \\ . & & & \\ . & & & \\ x_n & B(y_n) & B(y_n)\ldots & B(y_n) \end{Bmatrix} \tag{3.61}$$

The projection in continuous conditions is defined like the discrete condition but the Σ operator is changed to the integration operator. A 3D illustration of the cylindrical extension of a typical fuzzy set is shown in Fig. 3.44.

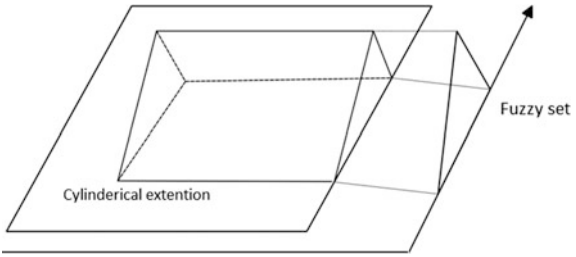


Fig. 3.44 3D illustration of cylindrical extension of a fuzzy set (Soleimani and Hajian 2017)

Example 3.32 Assume x = {x_1, x_2, x_3}, y = {y_1, y_2, y_3} and the fuzzy relationship $\tilde{R}$ is defined as:

$$\tilde{R} = \begin{matrix} x_1 \\ x_2 \\ x_3 \end{matrix} \begin{matrix} y_1 & y_2 & y_3 \end{matrix} \begin{bmatrix} 1 & 0.8 & 0.6 \\ 0.9 & 0.4 & 0.5 \\ 0 & 0.3 & 0.3 \end{bmatrix}$$

Then:

$$\text{Proj}[R, X] = \{(x_1, 1), (x_2, 2), (x_3, 0.3)\}$$

$$\text{Proj}[R, Y] = \{(y_1, 1), (y_2, 0.8), (y_3, 0.6)\}$$

And if $\tilde{A}$ = {(x_1, 1), (x_2, 0.9), (x_3, 0.3)}
The cylindrical extension of $\tilde{A}$ over x is as below:

$$C(A) = \begin{matrix} x_1 \\ x_2 \\ x_3 \end{matrix} \begin{matrix} y_1 & y_2 & y_3 \end{matrix} \begin{bmatrix} 1 & 1 & 1 \\ 0.9 & 0.9 & 0.9 \\ 0.3 & 0.3 & 0.3 \end{bmatrix}$$

Example 3.33 The Fuzzy set A with Gaussian membership A(x) is given (Fig. 3.45), its cylindrical extension over y axis is depicted in Fig. 3.46 (http://slideplayer.com/slide/5336605/).

3.5.5 Composition of Fuzzy Relations

Assume $\tilde{R}$ is a fuzzy relationship over X × Y with a membership function R(x, y), $\tilde{S}$ is a relationship over Y × Z with membership function S (y, z) then the composition of these fuzzy relations is defined as:

$$\left(\tilde{R}O\tilde{S}\right)_{(x,y)} \max_y \{T[r(x, y), S(y, z)]\} \tag{3.62}$$

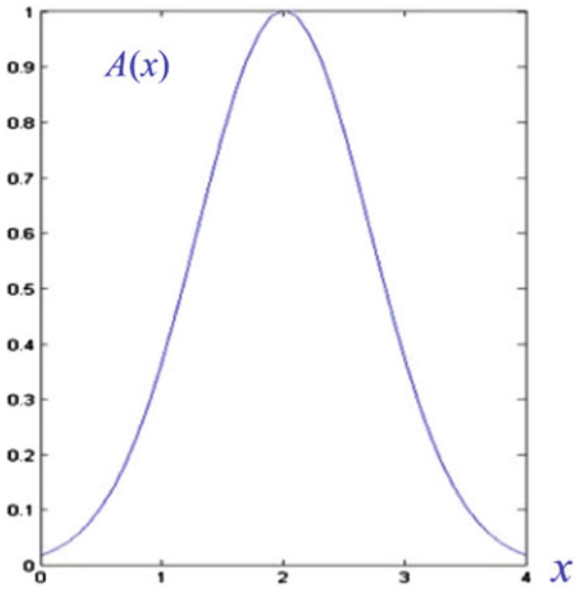


Fig. 3.45 Set A with Gaussian membership function

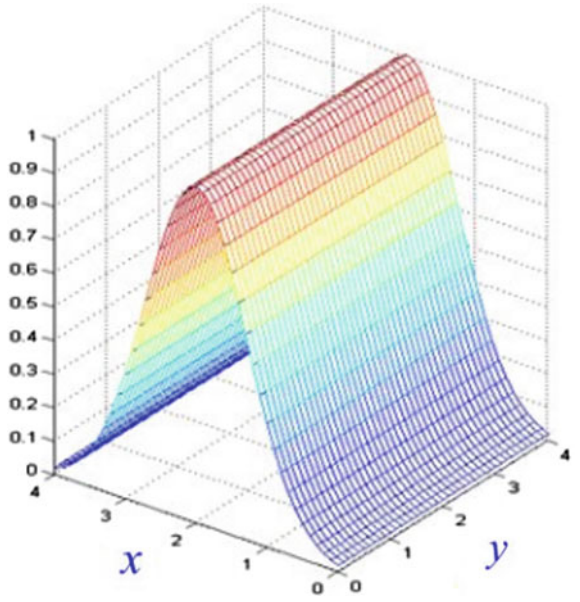


Fig. 3.46 Cylindrical projection of set A in Fig. 3.45

Where T is a triangular norm and mostly the T-norm: 'min' is used:

$$\left(\tilde{\mathrm{R}}\mathrm{O}\tilde{\mathrm{S}}\right)_{(\mathrm{x,y})} = \min_{y}\{min[r(x,y), S(y,z)]\} \tag{3.63}$$

Example 3.34 Assume $X = \{x_1, x_2, x_3\}$, $Y = \{y_1, y_2, y_3\}$, $Z = \{z_1, z_2, z_3, z_4\}$, $\tilde{R}$ is a fuzzy relationship over x, y (Table 3.16) and $\tilde{S}$ is a fuzzy relationship over Y and Z (Table 3.17). Then calculate $\tilde{R}O\tilde{S}$.

The $\tilde{R}O\tilde{S}$ is a fuzzy relationship over x, z as shown in Table 3.18.

As an example the way of calculating $\tilde{R}O\tilde{S}(x_1, z_1)$ is described, here. First min (0.7, 0, 0.1), min (1, 0.8), min (0.9, 0.4) are calculated and among these membership degrees the maximum value is calculated:

$$\left(\tilde{\mathrm{R}}\mathrm{O}\tilde{\mathrm{S}}\right)_{(\mathrm{x1,z1})} = \max\{\min(0.7, 0.1), \min(1, 0.8), \min(0.9, 0.4)\}$$

In this example if x, y, z are sets of the hydrophones linked to a network and $\tilde{R}$ is the degree of linkage security between hydrophones set X to hydrophones set Y and $\tilde{S}$ is the degree of linkage security between hydrophones set X to Z then $\tilde{R}O\tilde{S}$ determines the degree of linkage security between hydrophones set x to z where the hydrophones in set x are linked to hydrophones set z indirectly through hydrophone set Y (Fig. 3.47).

There are three routes for linkage between hydrophone X_1 to Z_1:

$$\text{Path 1} : x_1 \overset{0.7}{\rightarrow} y_1 \overset{0.1}{\rightarrow} z_1 \overset{min}{\rightarrow} 0.1$$

$$\text{Path 2} : x_1 \overset{1}{\rightarrow} y_2 \overset{0.8}{\rightarrow} z_1 \overset{min}{\rightarrow} 0.8$$

Table 3.16 Matrix representation of fuzzy relationship $\tilde{R}$

$\tilde{R}$	y_1	y_2	y_3
x_1	0.7	1	0.9
x_2	0.9	0.5	0.6
x_3	0.1	0.4	1

Table 3.17 Matrix representation of fuzzy relation $\tilde{S}$

$\tilde{S}$	z_1	z_2	z_3	z_4
y_1	0.1	0.3	0.9	0.7
y_2	0.8	0.7	0.2	0.6
y_3	0.4	0.3	0.5	1

Table 3.18 Matrix representation of fuzzy relationship $\tilde{R}O\tilde{S}$

$\tilde{R}O\tilde{S}$	z_1	z_2	z_3	z_4
x_1	0.8	0.7	0.7	0.7
x_2	0.5	0.5	0.9	0.7
x_3	0.4	0.4	0.5	1

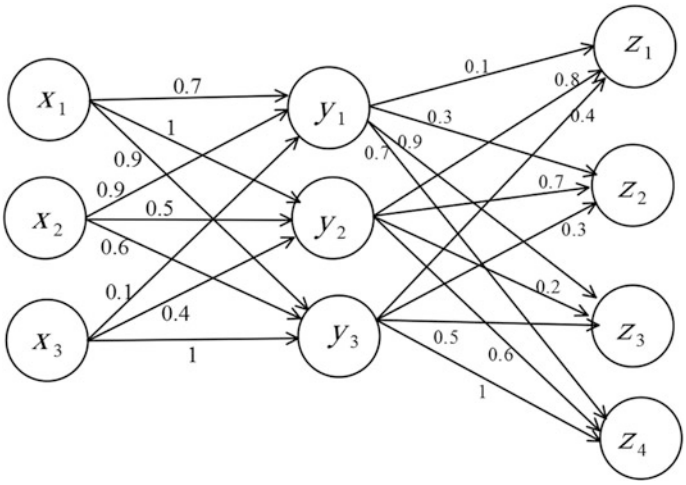


Fig. 3.47 The connection graphical presentation for $\tilde{R}$, $\tilde{S}$ (Soleimani and Hajian 2017)

$$\text{Path } 3 : x_1 \overset{0.9}{\rightarrow} y_3 \overset{0.4}{\rightarrow} z_1 \overset{min}{\rightarrow} 0.4$$

This means that path 2 is the best path to link hydrophone x_1 to z_1 (from x_1 to y_2 to z_1) with the membership degree 0.8.

Example 3.35 Assume x = {1, 2, 3}, y = {2, 3, 4}, $\tilde{R}$ is a fuzzy relationship for similar elements over x, y, and $\tilde{A}$ is the fuzzy set of 'small elements' over x then what is $\tilde{A}O\tilde{R}$ = ?

$\tilde{A}$ = {(1, 0.9), (2, 0.3), (3, 0.1)} (Table 3.19)

$$\tilde{A}O\tilde{R} = [0.9 \quad 0.3 \quad 0.1]O\begin{bmatrix} 0.7 & 0.4 & 0.1 \\ 1 & 0.7 & 0.4 \\ 0.7 & 1 & 0.7 \end{bmatrix} = [0.9 \quad 0.4 \quad 0.3]$$

$$\Rightarrow \tilde{B} = \tilde{A}O\tilde{R} = \{(2, 0.9), (3, 0.4), (4, 0.3)\}$$

Fuzzy set $\tilde{B}$ is the set of small elements over universal set y and $\tilde{R}$ it is a demonstrative for elements x similar to y. When set $\tilde{A}$ is combined with relationship $\tilde{R}$ it determine small elements among x members thus the combination of $\tilde{A}$ and $\tilde{R}$ will define small elements among members of set y.

Table 3.19 Fuzzy relationship in example 3.35 in its Matrix format

$\tilde{R}$	2	3	4
1	0.7	0.4	0.1
2	0.1	0.7	0.4
3	0.7	1	0.7

The calculation of a combination of two relationships is similar to the product of matrices but with a difference that the add operator '+' is replaced by max and the '.' product is replaced by min.

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} o \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} a.e + b.g & a.f + b.h \\ c.e + d.g & c.f + d.h \end{bmatrix} = \begin{bmatrix} (a \wedge e) \vee (b \wedge g) & (a \wedge f) \vee (b \wedge h) \\ (c \wedge e) \vee (d \wedge g) & (c \wedge f) \vee (d \wedge h) \end{bmatrix} \tag{3.64}$$

The above method is namely the 'max-min' method. As we mentioned in definition of combination of relations the general condition of T-norm is max. Another combination method which has many applications is max-product method.

Example 3.36 Combine the relations $\tilde{R}$, $\tilde{S}$ via max-product

$$\tilde{R} = \begin{bmatrix} 0.1 & 0.7 & 0.9 \\ 1 & 0.8 & 0.3 \end{bmatrix} \quad \tilde{S} = \begin{bmatrix} 0 & 1 \\ 0.4 & 0.5 \\ 0.8 & 0.7 \end{bmatrix}$$

$$\tilde{R} o \tilde{S} = \begin{bmatrix} 0.72 & 0.63 \\ 0.32 & 1 \end{bmatrix}$$

One of the important applications of fuzzy relations and their combination is the interpretation of fuzzy if-than rules which will be investigated in future sections.

3.5.6 Matlab Coding for Fuzzy Relations

To calculate fuzzy operations and combination of fuzzy relations Matlab software is very useful. For example the program for combination of fuzzy relations is described during the next example.

Example 3.37 Calculation of the combination of the last example relations through max-product with a Matlab program.

Solve: We have presented here a program for general condition to combine two fuzzy relations via max-product method.

```
R=input('enter the first relationship (matrix)')
S = input ('enter the second relationship (matrix)')
% size of matrix
[min] = size(R)
[p,q] = size(S)
```

```
If (n =  = p)
fori = 1:m
for j = 1:q
c = R(i,:);
d = S(:,j);
              [f,g] = size(c);
              [h,k] = size(d);
fot  l = 1;g
e(1,l) = c(1,l)*d(l,1)
end
T(i,j) = max(e)i
end
end
display(t)
```

After running the above codes in Matlab:
Enter the first relationship (matrix) [0.1,0.7,0.9;1,0.8,0.3]

$$\mathrm{R} = \begin{matrix} 0.1000 & 0.1000 & 0.9000 \\ 1.0000 & 0.8000 & 0.3000 \end{matrix}$$

Enter the second relationship (matrix) [0,1;0.4,0.5,0.8,0.1]

$$\mathrm{S} = \begin{matrix} 0.0000 & 1.0000 \\ 0.4000 & 0.5000 \\ 0.8000 & 0.1000 \end{matrix}$$

$$\mathrm{t} = \begin{matrix} 0.7200 & 0.6300 \\ 0.3200 & 1.000 \end{matrix}$$

3.5.7 Properties of Fuzzy Relations

As in classical set theory a relationship can be reflexive, symmetric, anti-symmetric, associativity.

Definition, Assume $\tilde{R}$ is a fuzzy relationship over X with membership function $\overline{R}(x, y)$ then:

1. $\tilde{R}$ is a reflexive relationship if:

$$\forall x \in: \tilde{R}(x, x) = 1 \tag{3.65}$$

2. $\tilde{R}$ is a symmetric relationship if:

$$\forall (y, x) \in X \times Y : \tilde{R}(y, x) = \tilde{R}(y, x) \tag{3.66}$$

3. $\tilde{R}$ is on oppressive relationship if:

$$\forall (x, y) \in X \times Y, (y, z) \in Y \times Z : \tilde{R}(x, z) \geq \max[\min_{y} R(x, y), R(y, z)] \tag{3.67}$$

Or on the other hand: $\tilde{R}O\tilde{R} \leq \tilde{R}$

4. $\tilde{R}$ is an anti-symmetric relationship if:

$$\forall (x, y) \in X \times Y : \tilde{R}(x, y) \neq \tilde{R}(y, x) \text{or} \, \tilde{R}(x, y) = \tilde{R}(y, x) = 0 \tag{3.68}$$

3.5.7.1 Equivalence Fuzzy Relations

$\tilde{R}$ is an equality fuzzy relationship when it has all properties: reflectivity, symmetry and transitive.

3.5.7.2 Ordered Fuzzy Relation

$\tilde{R}$ is defined as an ordered fuzzy relationship when it has reflexive, anti-symmetric and transitive properties.

Example 3.38 Investigate whether the fuzzy relationship $\tilde{R}$ in Table 3.20 is an equivalence relationship or not?

As for all a: $\tilde{R}(x, x) = 1$ it has reflexive property the main diameter is 1, the relationship matrix is symmetric so $\tilde{R}$ is symmetric. $\tilde{R}O\tilde{R}$ is not subset of $\tilde{R}$ thus $\tilde{R}$ is not transitive. The below fuzzy relationship is ordered (Table 3.21).

Table 3.20 Fuzzy relationship $\tilde{R}$ matrix representation

$\tilde{R}$	A	B	C	D
A	1	0.8	0.5	0.9
B	0.8	1	0.3	0.8
C	0.5	0.3	1	0.7
D	0.9	0.8	0.7	1

Table 3.21 An example of ordered fuzzy relation (Soleimani and Hajian 2017)

$\tilde{R}$	A	B	C	D
A	1	0.2	0	0.1
B	0.8	1	0	0
C	0.3	0.4	1	0.1
D	0	0	0	1

3.5.8 α-cut of a Fuzzy Relation

α-cut of fuzzy sets was defined in Sect. 3.4.4.

A Fuzzy relationship is a kind of fuzzy set thus its α-cut is defined similarly.

Definition—Assume $\tilde{R}$ is a fuzzy relationship over X × Y then α -cut of$\tilde{R}$:

$$R_\alpha = \{(x, y) : R(x, y) \geq \alpha; x \in X, y \in Y\} \tag{3.69}$$

Note that R_α will be a classical relation.

Example 3.39 Assume the below fuzzy relationship in Table 3.22, then calculate R_α for α = 0.2, 0.3, 0.5, 0.7 (Tables 3.23, 3.24, 3.25, 3.26 and 3.27).

Table 3.22 Fuzzy relationship in example 3.39

$\tilde{R}$	1	2	3
1	0.2	0.5	0.6
2	0.7	1	0.3
3	0.3	0.6	0.2

Table 3.23 R_α with α = 0.2 (Soleimani and Hajian 2017)

$R_{\alpha = 0.2}$	1	2	3
1	1	1	1
2	1	1	1
3	1	1	1

Table 3.24 R_α with α=0.3 (Soleimani and Hajian 2017)

$R_{0.3}$	1	2	3
1	0	1	1
2	1	1	1
3	1	1	0

Table 3.25 R_α with α = 0.5 (Soleimani and Hajian 2017)

$R_{0.5}$	1	2	3
1	0	1	1
2	1	1	0
3	0	1	0

Table 3.26 R_α with $\alpha = 0.7$

$R_{0.7}$	1	2	3
1	0	0	0
2	1	1	0
3	0	0	0

Table 3.27 Fuzzy relationship in example 3.48 (Soleimani and Hajian 2017)

$\tilde{R}$	a	b	C	d	e	f
a	1	0.8	0	0.4	0	0
b	0.8	1	0	0.4	0	0
c	0	0	1	0	1	0.5
d	0.4	0.4	0	1	0	0
e	0	0	1	0	1	0.5
f	0	0	0.5	0	0.5	1

3.5.9 *α-cut of Equivalent Fuzzy Relationship*

If $\tilde{R}$ is an equivalent fuzzy set over x. Then its α-cut will be an equivalent classical relationship, therefore, set x is an α-cut of an equivalent fuzzy relationship and is defined as:

$$R_\alpha(x, y) = \begin{cases} 1 & R(x, y) \geq \alpha \forall x, y \in X_i \\ 0 & O.W. \end{cases} \tag{3.70}$$

If the base set of fuzzy relationship $\tilde{R}$ is Δ so that $\wedge = \{\alpha_1, \alpha_2, \ldots\}$ is a α-cut for α_1-partition; $\prod(R_{\alpha 1})$ if $\alpha_1 \geq \alpha_2$ then $R_{\alpha 1} o R_{\alpha 2} \subseteq R_{\alpha 2}$ has more partitions more than $\prod(R_{\alpha 2})$.

Example 3.40 The fuzzy relationship shown in Table 3.27 is defined on universal set X = {a, b, c, d, e, f}.

This type of fuzzy relationship is named "similarity relationship". For α = 0.5, its α-cut is as shown in Table 3.28

As can be seen in Table 3.28: $\prod(R_{0.5}) = \{\{a, b\}, \{d\}, \{c, e, f\}\}$

Table 3.28 Matrix representation of $R_\alpha = 0.5$ for $\tilde{R}$

$\tilde{R}_{0.5}$	a	b	c	d	e	f
a	1	1	0	0	0	0
b	1	1	0	0	0	0
c	0	0	1	0	1	1
d	0	0	0	1	0	0
e	0	0	1	0	1	1
f	0	0	1	0	1	1

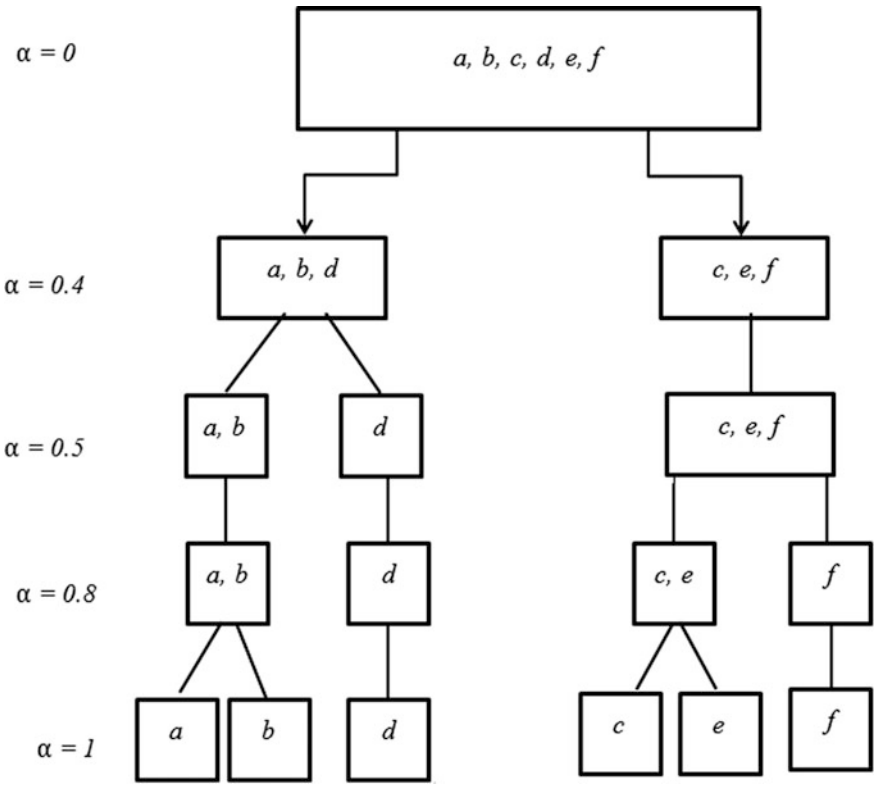


Fig. 3.48 Partition tree diagram for α-cut of fuzzy relationship $\tilde{R}$ with different values of α (Soleimani and Hajian 2017)

And similarly for other values of α, the partitions shown in the partition-tree diagram below (Fig. 3.48) presents these partitions. $R_{0.5}$ is an equivalent relationship at a level of α = 0.5.

3.6 Fuzzy Numbers

3.6.1 Further Description of the Extension Principle

In classical mathematics one of the important tools is the "function" tool. Function is a special relationship. For example if we consider the selection of pairs of (x,y), i.e. if (x,y) = (x,f(x)) = {(2,12), (4,1), (2,16), (9,9)}; then 'f' is not a function because x = 2 is repeated, as the first element, in two pairs of set(x,y), on other hand for x = 2 we have two values, so f can't be a function.

Usually a function is presented with a criterion such as: f(x) = 2x + 1.

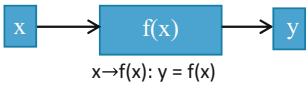


Fig. 3.49 Illustration of a function as a system with input and output

A function is like a system that takes x as the input and gives y as output (Fig. 3.49) i.e. if x = 2 then f (2) = 5.

Now, assume the input of the function is a fuzzy set i.e. "close to 2", then its output should be also a fuzzy set. For the above mentioned function the output will be "close to 5".

Assuming function f: x → y, when Ã is a fuzzy function over x, we want to calculate the output function for the input Ã. This is possible using the extension principle.

Assume f is a function from x to y and Ã is s fuzzy set over Y the f(Ã) is also a fuzzy set which is achieved using the extension principle as below:

$$\mathrm{f}(\tilde{\mathrm{A}})(\mathrm{y}) = \begin{cases} s \cup pA(\mathrm{x})\mathrm{f}^{-1} & (y) \neq \emptyset \\ x : f(\mathrm{x}) = y & \\ 0 & \mathrm{f}^{-1}(\mathrm{y}) = \emptyset \end{cases} \tag{3.71}$$

Example 3.41 Assume x = {−1, −2, 0, 1, 2} and f: x → Y
f (x) = x^2 and Ã is a fuzzy set over x:

Ã = {(−2, 0.5) (−1, 0.7), (0, 1), (1, 0.8), (2, 0.6)}
Calculate f (Ã).

Solve: First we calculate the values of y and then their membership $f_y(A)$.
y = f(x) = {0, 1, 4}
f (A)(1) = max{$A_{(x)}$: f(x) = 1} = max{A(−1), A(1)} = max{0.7, 0.8} = 0.8
x = $f^{-1}(1)$
f(A)(4) = max{A(x): f(x) = 4} = max{A(−2), A(2)} = max{0.5, 0.6} = 0.6
So:

$$\mathrm{f}(\tilde{\mathrm{A}}) = \{(0,1),(1,0.8),(4,0.6)\}$$

Example 3.42 Assume f(x) = 2x + 1, Ã is the below fuzzy set:
Ã = {(0, 0.2), (1, 0.7), (2, 1), (3, 0.6), (4, 0.1)}
Then what is f (Ã)?
Solve: First we calculate:
y = {1 3 5 7 9}, f (Ã) = {(1, 0.2), (3, 0.7), (5, 1), (7, 0.6), (9, 0.1)}
In this example Ã is the fuzzy set: "about 2" and the value of the fuzzy function is the fuzzy: "about 5" (Fig. 3.50).

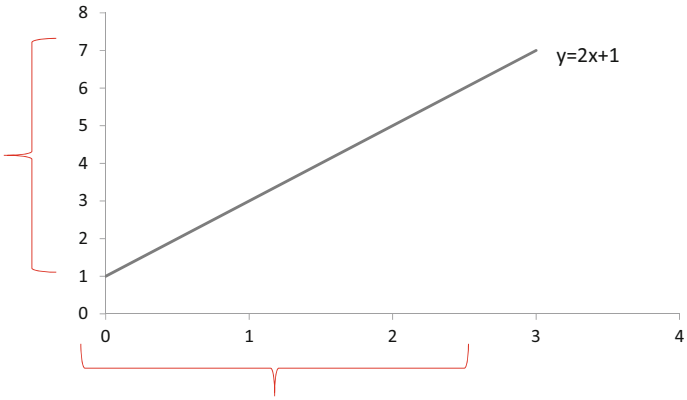


Fig. 3.50 Graphical presentation of extension principle for example 3.42

Example 3.43 Assume $\tilde{A}$ is a fuzzy set with membership A(x) as below and y = f(x) = 2x + 1 then calculate f ($\tilde{A}$):

$$A(x) = \begin{cases} x+1 & -1 < x \leq 0 \\ 1-x & 0 < x < 1 \end{cases} \tag{3.72}$$

Solve: First we calculate y versus x or $f^{-1}(y)$ and then f (A) (y) is calculated via the extension principle.

$$\begin{aligned} &y = x^2 + 1 \Rightarrow x = \pm\sqrt{y-1} \Rightarrow f^{-1}(y) \\ &= f(A)(y) = \sup[A(x) : x = f-1(y)] = \sup\begin{cases} -\sqrt{y-1}+1 : -1 < -\sqrt{y-1} < 0 \\ 1+\sqrt{y-1} : 0 < \sqrt{y-1} < 1 \end{cases} \\ &= \text{Sup}\begin{cases} 1-\sqrt{y-1} & 0 < y < 2 \\ 1+\sqrt{y-1} & 0 < y < 2 \end{cases} = \begin{cases} 1-\sqrt{y-1} & 0 < y < 2 \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{3.73}$$

3.6.2 *Generalized Extension Principle or Multi-variate Extension Principle*

Assume f is a function from $X_1 * X_2 * \ldots * X_n$ space to y, $\tilde{A}_i$ is a fuzzy set over X_i the f($\tilde{A}_1$, $\tilde{A}_2$, $\tilde{A}_n$) is as below:

$$f\left(\tilde{A}_1, \tilde{A}_2, \tilde{A}_n\right)_{(y)} = \begin{cases} \sup\{\min(A_1(x_1), A_2(x_2), \ldots, A_n(x_n)\} \\ (x_1, x_2, \ldots, x_n) : f(x_1, x_2, \ldots, x_n) = y f^{-1}(y) \neq \emptyset \\ 0 \qquad\qquad f^{-1}(y) = \emptyset \end{cases} \tag{3.74}$$

When the function is multi-variate the Cartesian production $\tilde{A}_1*\tilde{A}_2*\ldots*\tilde{A}_n$ is calculated first and then the value of the function is calculated according to the above formula.

Example 3.43 Assume $y = f(x_1, x_2) = x_1^2 + x_2$ and $\tilde{A}$, $\tilde{B}$ are fuzzy sets as below:

$$\tilde{A} = \{(-1, 0.3), (0, 0.7), (1, 1), (2, 0.5)\}$$

Solve: First we calculate $\tilde{A}*\tilde{B}$:

$\tilde{A}*\tilde{B}$= {((−1, 1), 0.3), ((−1, 2), 0.3), ((−1, 3), 0.3), ((0, 1), 0.4), ((0, 2), 0.7), ((0, 7), 0.3), ((1, 1), 0.4),

((1, 2), 0.7), ((1, 3), 1), ((2, 1), 0.4), ((2, 2), 0.3), ((2, 3), 0.5)}

Then we calculate $f(x_1, x_2)$ = {1, 2, 3, 4, 5, 6, 7},

$$f(\tilde{A}, \tilde{B}) = \{(1, 0.4), (2, 0.7), (3, 0.7), (4, 1), (5, 0.4), (6, 0.5), (7, 0.5)\}$$

i.e.: $f(\tilde{A}, \tilde{B})_{(2)}$ = max{minA(−1), B(1), min(A(0), B(2)), min(A(1), B(1))} = max {0.3, 0.7, 0.4} = 0.7

The bi-variate extension principle is used to carry out arithmetic operations on fuzzy numbers.

3.6.3 Philosophy of Fuzzy Numbers

Most qualitative phenomena are not representable with an absolute number, for example when we say that the velocity of a P wave is about 4.5 m/sec or the first peak is received to the geophone at about 3 s after the main shot the value of velocity and time are vague. Also in most of laboratory and also field measurements the value of the quantities are vague and non-precise; in all the cases mentioned we can use fuzzy numbers.

Usually for vague concepts with expressions like: about, approximately, close to, etc. fuzzy set are assigned which are indeed fuzzy numbers.

Fuzzy numbers are useful in various field-like decision-making situations, approximate reasoning, neuron-fuzzy network, fuzzy, control, etc. For example the below statement is related to a fuzzy controller: IF the temperature of room is about 40 °C then the rotation power of the cooler motor will be increased; About 40 °C can be shown with a fuzzy number.

3.6.4 Definition of a Fuzzy Number

A fuzzy set $\tilde{A}$ with membership function A(x) is named a fuzzy number when it has the properties below:

1. The fuzzy set $\tilde{A}$ is a convex set
2. $\exists x_0 \in X \text{ s.t.} A(x_o) = 1$
3. It is part by part sectional continuous.

Example 3.44 The below membership function is a fuzzy number shown in Fig. 3.51.

$$A(x) = \begin{cases} \frac{x-1}{2} & 1 \le x \le 3 \\ 7 - 2x & 3 < x < 3.5 \end{cases} \tag{3.75}$$

The membership is triangular and so it is convex. A (3) = 1, and is continuous, so the MF is a fuzzy number: "about 3" (Fig. 3.51).

An operation on a fuzzy number is possible through the extension principle, i.e. for f(x) = kx according to the extension principle $f(\tilde{m}) = k\tilde{m}$ where its membership function is achieved replacing y/k as x in mx.

Example 3.45 Calculate twice the fuzzy number $\tilde{3}$(about 3) in the example.
Solve:

$$f(A)_{(y)} = \sup = \begin{cases} \frac{\frac{y}{2}-1}{2} & 1 < y/2 \le 3 \\ 7 - 2\frac{y}{2} & 3 < \frac{y}{2} < 3.5 \end{cases} = \begin{cases} \frac{y-2}{4} & 1 < y \le 6 \\ 7 - y & 6 < y < 7 \end{cases} \tag{3.76}$$

Example 3.46 Assuming $\tilde{m}$, ñ are fuzzy numbers with membership functions defined below, calculate $\tilde{m}$ + ñ?

$$\tilde{m}(x) = \begin{cases} x - 1 & 1 < x \le 2 \\ 3 - x & 2 < x < 3 \end{cases} \quad \tilde{n}(x) = \begin{cases} 2x - 5 & 3.5 < x < 3 \\ 7 - 2x & 3 < x < 3.5 \end{cases} \tag{3.77}$$

To calculate the sum of two fuzzy numbers we can use the bivariate extension principle:

If: f(x, y) = x + y the $f(\tilde{m}, \tilde{n}) = \tilde{m} + \tilde{n}$, $f(m, n)_{(z)} = sup\{\min(m(x), n(y))\}$

$$z = x + y$$

or

$$(m + n)(z) = \vee(m(x) \wedge n(y)) \tag{3.78}$$

$$z = x + y$$

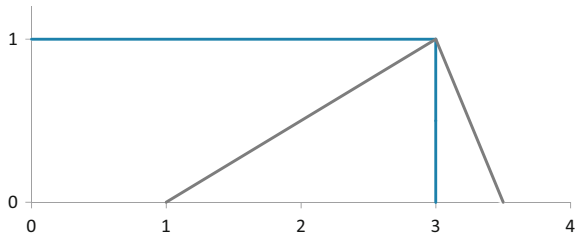


Fig. 3.51 Fuzzy number "about 3"

So:

$$(\mathrm{m}+\mathrm{n})_{(z)}=\begin{cases}\frac{27-7}{3} & 3.5<z\leq 5\\ \frac{13-2Z}{3} & 5<z<6.5\end{cases} \tag{3.79}$$

In general conditions when $\tilde{m}$, ñ are fuzzy numbers with membership function m (x), n(x) respectively, then:

$$\begin{gathered}(\mathrm{m}+\mathrm{n})_{(z)}=\vee(\mathrm{m}(\mathrm{x})\wedge\mathrm{n}(\mathrm{y}))\\ \mathrm{Z}=\mathrm{x}+\mathrm{y}\end{gathered} \tag{3.80}$$

$$\begin{gathered}(\mathrm{m}.\mathrm{n})_{(z)}=\vee(\mathrm{m}(\mathrm{x})\wedge\mathrm{n}(\mathrm{y})\\ \mathrm{Z}=\mathrm{x}.\mathrm{y}\end{gathered} \tag{3.81}$$

$$\begin{gathered}(\mathrm{m}-\mathrm{n})_{(z)}=\vee(\mathrm{m}(\mathrm{x})\wedge\mathrm{n}(\mathrm{y}))\\ \mathrm{Z}=\mathrm{x}-\mathrm{y}\end{gathered} \tag{3.82}$$

$$\begin{gathered}(\mathrm{m}/\mathrm{n})_{(z)}=\vee(\mathrm{m}(\mathrm{x})\wedge\mathrm{n}(\mathrm{y}))\\ \mathrm{Z}=\mathrm{x}/\mathrm{y}\end{gathered} \tag{3.83}$$

Note: Bell-shaped membership functions can also be membership functions of fuzzy numbers, i.e. $\mathrm{m}(\mathrm{x})=\frac{1}{1+(x-a)^2}$ can be the representation of 'about a'. Also a combination of linear and bell-shaped membership functions can be a membership function of a fuzzy number (Fig. 3.52).

Performing an arithmetic operation on fuzzy numbers is, as we have seen, sophisticated and time consuming. To solve this problem a general format is defined for all fuzzy numbers which is simpler than before and is famous as the 'LR' representation first suggested by Dubios and Prade (1978).

3.6.5 LR Representation of Fuzzy Numbers

Assume $\tilde{m}$ is a fuzzy number with membership function m(x). Its LR representation is as defined in the formula below (its graph is illustrated in Fig. 3.53:

$$\mathrm{m}(\mathrm{x})=\begin{cases}L\left(\frac{m-x}{\alpha}\right), x\leq m\\ R\left(\frac{x-m}{\beta}\right), x>m\end{cases} \tag{3.84}$$

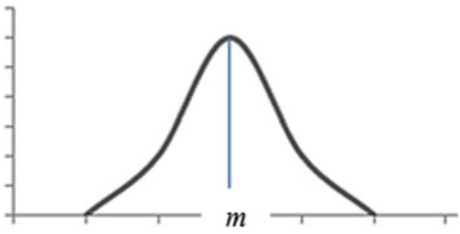


Fig. 3.52 Combination of linear and bell-shaped MFs as a fuzzy number $\tilde{m}$

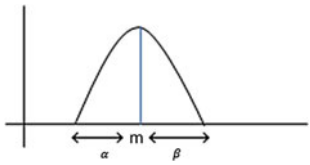


Fig. 3.53 LR fuzzy number

Where L and R, are functions with the properties below:

1. L and R, are continuous functions
2. L(0) = 1, R(0) = 1
3. L and R are non-increasing on [0; +∞].
4. L(x) = L(−x), R(x) = R(−x)
5. lim R(x) = 0, lim L(x) = 0
   $X \to \infty \quad X \to \infty$

(L and R membership function, and m characterizes the mean value of $\tilde{m}$).
Some examples of the functions which can be used as L, R are:

$$\max\{0, 1 - x\}, 1 - |x|^{P}: p > 0, e^{-x^2}, \frac{1}{1+x^2} \tag{3.85}$$

Symbolically we will present the LR fuzzy numbers with:

$$\tilde{m} = (m, \alpha, \beta)_{LR} \tag{3.86}$$

Where 'm' is the assumed number, 'α' is the left band width, 'β' is the right band width, L is left band function and R is the right band function (Fig. 3.54) namely the left and the right coefficients of "fuzziness" respectively.

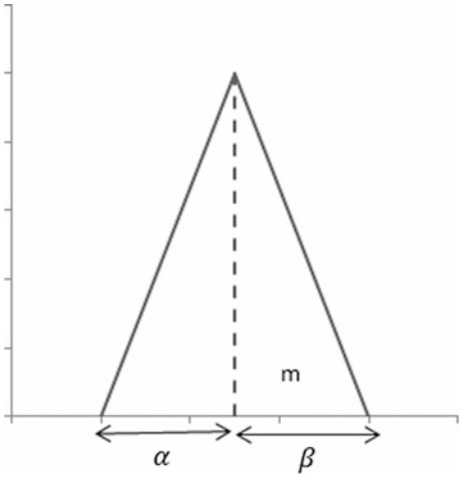


Fig. 3.54 A typical triangular fuzzy number

Example 3.47 The fuzzy number $\tilde{z} = (2\frac{1}{2}, \frac{1}{3})_{LR}$ with L(x) = 1 − |x|, $R_{(x)} = \frac{1}{1+x^2}$ has the membership function as below:

$$m(x) = \begin{cases} L\left(\frac{2-x}{\frac{1}{2}}\right), x \le 2 \\ R\left(\frac{x-2}{\frac{1}{3}}\right), x > 2 \end{cases} = \begin{cases} 2x - 5, x \le 2 \\ \frac{1}{1+(2x-6)^2}, x > 2 \end{cases} \tag{3.87}$$

3.6.6 Operations on LR Fuzzy Numbers

Assume $\tilde{m} = (m, \alpha, \beta)_{LR}, \tilde{n} = (n, \gamma, \delta)_{LR}$ then:

1. Sum of two LR fuzzy numbers:

$$(m, \alpha, \beta)_{LR} + (n, \gamma, \delta)_{LR} = (m + n, \alpha + \gamma, \beta + \delta)_{LR} \tag{3.88}$$

2. Scalar product of an LR fuzzy numbers

$$\lambda(m, \alpha, \beta)_{LR} = \begin{cases} (\lambda_m, \lambda_\alpha, \lambda_\beta)_{LR} \lambda > 0 \\ (\lambda_m, -\lambda_\alpha, -\lambda_\beta)_{LR} \lambda < 0 \end{cases} \tag{3.89}$$

3. Product of two LR fuzzy numbers:

$$(m, \alpha, \beta)_{LR} \times (n, \gamma, \delta)_{LR} = \begin{cases} (mn, m\gamma + n\alpha, m\delta + n\beta)_{LR} m, n > 0 \\ (mn, n\alpha - m\delta, n\beta - m\gamma) n \langle 0, m \rangle 0 \\ (mn, -n\beta - m\delta, -n\alpha - m\gamma)_{LR} n, m < 0 \end{cases} \tag{3.90}$$

4. Subtraction of two LR fuzzy numbers:

$$(m, \alpha, \beta)_{LR} - (n, \gamma, \delta)_{LR} = (m + n, \alpha + \gamma, \beta + \delta)_{LR} \tag{3.91}$$

Example 3.48 Assume: $\tilde{m} = (3, 1, 1)_{LR}$, $\tilde{n} = (2, 1/2, 1)_{LR}$ and L(x) = R(x) = 1 − |x|,

calculate $\tilde{m} + \tilde{n} = (3, 1, 1)$ and $\tilde{m} \times \tilde{n}$.

Solve:
$\tilde{a} = \tilde{m} + \tilde{n} = (3, 1, 1)_{LR} + (2, 1/2, 1)_{LR} = (5, 3/2, 2)_{LR}$

$$A(x) = \begin{cases} \frac{x-3.5}{1.5} x \leq 5 \\ \frac{7-x}{2} x > 5 \end{cases} \tag{3.92}$$

3.6.7 Triangular Fuzzy Numbers

A fuzzy number whose membership function shape is a triangle is called a triangular fuzzy number. Correspondingly when L(x) = R(x) = 1 − |x|, a fuzzy number will be triangular as shown in Fig. 3.54.

The fuzzy numbers are shown with triple (a, m, b) where m is the fuzzy number, 'a' is the start point and 'b' is the end point, if the domain (a, b) is the support of the fuzzy number. If $\alpha = 0.3$ then the fuzzy number is symmetric and m is the middle point between a, b.

Assume $\tilde{m}$, $\tilde{n}$ are triangular fuzzy numbers:
$\tilde{m} = (a, m, b)$, $\tilde{n} = (a_2, n, b_2)$ sum, subtraction and symmetry of each number are:

$$\tilde{m} + \tilde{n} = (a_1 + a_2, m + n, b_1 + b_2) \tag{3.93}$$

$$\tilde{m} - \tilde{n} = (a_1 - a_2, m - n, b_1 - b_2) \tag{3.94}$$

$$-(m) = (-b_1, -m, -a) \tag{3.95}$$

Example 3.49 Assume $\tilde{n} = (1, 2, 3)$, $\tilde{m} = (2, 3, 4)$ shown in Fig. 3.55a, b respectively, then:

$$\tilde{m} + \tilde{n} = (3, 5, 7),\ \tilde{m} - \tilde{n} = (-1, 1, 3) \tag{3.96}$$

3.6.8 α-cut of Fuzzy Number

The α-cut of a fuzzy number is a domain so the operation on fuzzy numbers can be calculated via operations on their domains (Fig. 3.56):

$$M_\alpha = [a^\alpha, b^\alpha] \tag{3.97}$$

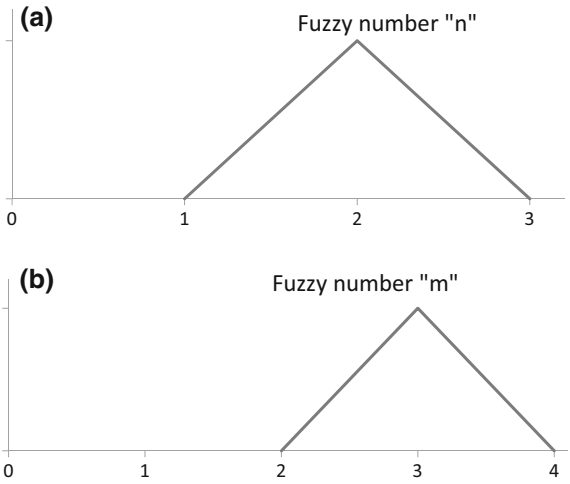


Fig. 3.55 **a** fuzzy number ñ **b** fuzzy number m̃

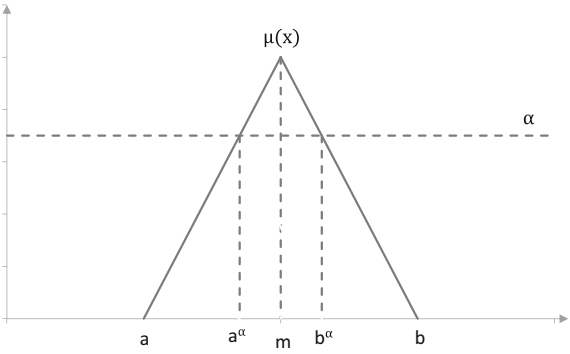


Fig. 3.56 Illustration of α-cut of a fuzzy number (Soleimani and Hajian 2017)

3.6.8.1 Operations on Domains

If [a, b] and [c, d] are domains then:

$$[a,b]+[c,d]=[a+c,b+d] \tag{3.98}$$

$$[a,b]-[c,d]=[a-d,b-c] \tag{3.99}$$

$$[a,b].[c,d]=[\min(ac,ad,bc,bd),\max\left(\frac{a}{d},\frac{a}{c},\frac{b}{d},\frac{b}{c}\right)] \tag{3.100}$$

$$[a,b]/[c,d]=[\min(\frac{a}{d},\frac{a}{c},\frac{b}{d},\frac{b}{c}),\max(\frac{a}{d},\frac{a}{c},\frac{b}{d},\frac{b}{c})] \tag{3.101}$$

where $d \neq 0$ and $c \neq 0$.

3.6.8.2 α-cut Method for Arithmetic Operations

In this method the α-cut of a fuzzy number is calculated first and then through the operations on domains, operations on α-cut are obtained via decomposition.

Assume $\tilde{m}$ = (a, m, b) is a triangular fuzzy number, in this case its α-cut is:

$$m_\alpha = [a^\alpha, b^\alpha] = [(m-a)\alpha + a, -(b-m)\alpha + b] \tag{3.102}$$

And if ñ = (c, n, d)
Then:

$$n_\alpha = [c^\alpha, d^\alpha] \text{ and } (m.n)\alpha = m_\alpha . n_\alpha \tag{3.103}$$

$$m.n = \cup \alpha (m.n)_\alpha \quad \alpha \in [0,1] \tag{3.104}$$

Note for fuzzy numbers the α-cut is a domain and so for transforming of a domain to a triangular fuzzy number it is sufficient to calculate the α-cut for α = 0, α = 1. α-cut with α = 1 is such height of fuzzy set and α-cut with α = 0 is its support.

Example 3.50 Calculate sum of the fuzzy numbers $\tilde{m}$ = (3, 4, 5), ñ = (1, 2, 3)

$$m_\alpha = [(4-3)\alpha + 3, -(5-4)\alpha + 5] = [\alpha + 3, -\alpha + 5] \tag{3.105}$$

$$n_\alpha = [(3.1)\alpha + 1, -(3.1)\alpha + 3] = [\alpha + 1, -\alpha + 3] \tag{3.106}$$

$$(m+n)_\alpha = [2\alpha + 4, -2\alpha + 8] \tag{3.107}$$

$$\begin{aligned} \alpha = 0 &\Rightarrow (m+n)_0 = (4,8) \\ &\Rightarrow \tilde{m} + \tilde{n} = (4,6,8) \\ \alpha = 1 &\Rightarrow (m+n)_1 = (6,6) = 6 \end{aligned}$$

3.6.8.3 Fuzzy Domain

A fuzzy domain is a trapezoidal shape fuzzy set (Fig. 3.57). Its LR representation is:

$$(\widetilde{m,n}) = (m, n, \alpha, \beta)$$

$$mn(x) = \begin{cases} L\left(\frac{m-x}{\alpha}\right) x < m \\ 1 m \le x \le n \\ R\left(\frac{x-n}{\beta}\right) x > n \end{cases} \tag{3.108}$$

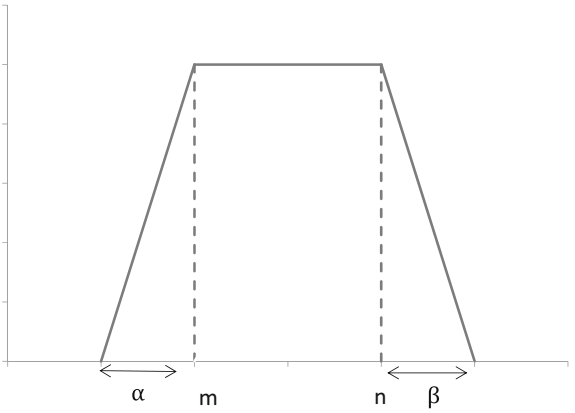


Fig. 3.57 Illustration of a typical trapezoidal fuzzy number

Some researchers know the fuzzy domain as a trapezoidal fuzzy number or flat fuzzy number. The fuzzy domains are used for fuzzy concepts such as close to the normal distribution or 'mediocre'.

3.7 Definition of Some Basic Concepts of Fuzzy Sets

Assume that set Ã is a fuzzy set of universal set x.

Definition 3.1—Height of fuzzy set The largest value of membership degree of fuzzy set Ã is called the height of Ã and is shown with hgt (Ã).

Definition 3.3—Normal set Fuzzy set Ã is called a normal set if hgt (Ã) = 1 or the height is 1.

Definition 3.3—Support of a fuzzy set The members of the universal set with membership degree greater than zero are shown with supp (Ã):

$$SUPP(\tilde{A}) = \{x \in X : A(x) > 0\} \tag{3.109}$$

Definition 3.4—Crossover point The point at which the membership degree is 1/2 is called the passing point.

Definition 3.5—Subset Given two fuzzy sets A, B defined on the same universal set X, A is said to be a subset of B if and only if: μA (x) $\leq$ μB (x).

For all $x \in X$. The usual notation A $\subseteq$ B is used to signify the subsethood relation.

Definition 3.6—Fuzzy power set of X The set of all fuzzy subsets of X is called the fuzzy power set of X and is denoted by F(x).

Definition 3.7—Degree of subsethood It stands for the degree that a given fuzzy set is a subset of another set (Dhar 2012) . It means the degree to which the set A is a subset of the set B, sub (A, B) to A in B: when the sets are defined on a finite universal set X we have:

$$\mathrm{Sub}(\mathrm{A},\mathrm{B}) = \frac{\sum_{x\in \mathrm{X}} A(x) - \sum_{x\in \mathrm{X}} \max[0, A(x) - B(x)]}{\sum_{x\in \mathrm{X}} A(x)} \tag{3.110}$$

The negative term in the numerator means that if the sum of the degree to which the subset inequality: A(x) $\leq$ B(x) is violated, the positive term implies the largest possible violation of the inequality, the difference in the numerator describes the sum of the degree to which the inequality is not violated, and the term in the denominator is a normalizing factor to obtain the range: 0 $\leq$ SUB (A, B) $\leq$ 1. Totally, the degree of subsethood defines the degree to which one set belongs to another.

When sets A and B are continuous (i.e. X is a closed interval of real numbers) then the three Σ terms in equation are replaced with integrals over X.

Definition 3.8—Scalar cardinality For any fuzzy set $\tilde{\mathrm{A}}$ defined on a finite universal set X, its scalar cardinality is defined by the formula:

$$\left|\tilde{\mathrm{A}}\right| = \sum\nolimits_{x\in \mathrm{X}} A(x) \text{ or } \left|\tilde{\mathrm{A}}\right| \int_{x\in \mathrm{X}} A(x)d(X) \tag{3.111}$$

and the relative cardinality is:

$$\left\|\tilde{A}\right\| = \frac{\left|\tilde{A}\right|}{|X|} \tag{3.112}$$

Definition 3.9—α-cut of a fuzzy set Given a fuzzy set A defined on α and a particular number α in the unit interval [0 1], the α-cut of A denoted by α_{A} is a crisp set that consists of all element of X whose membership degrees in A are greater than or equal to α:

$$\alpha_{\mathrm{A}} = \{\alpha | \mathrm{A}(\mathrm{x}) \geq \alpha\} \tag{3.113}$$

Definition 3.9.1—Strong α-cut The strong α-cut$\alpha_{+\mathrm{A}}$, has a similar meaning to α-cut, but the condition "greater than or equal to" is replaced with the stronger condition "greater than":

$$\alpha_{+\mathrm{A}} = \{\mathrm{x} | \mathrm{A}(\mathrm{x}) > \alpha\} \tag{3.114}$$

Definition 3.9.2—Core of fuzzy set The set 1_{A} (α-cut with α = 1) is called the core of A.

Definition 3.9.3 Level set of a fuzzy set:

The set of distinct values of A(x) for all x ∈ X is called the level set of A and is denoted by:

$$\Lambda_A : \Lambda_A = \{\alpha : A(x) = \alpha \quad x \in X\} \tag{3.115}$$

Some of the mentioned definitions are illustrated in Fig. 3.58.

Definition 3.10—Convex fuzzy set $\tilde{A}$ is a convex set if and only if (Fig. 3.59):

$$A(\lambda x_1 + (1-\lambda)x_2) \geq \min(A(x_1), A(x_2)) \forall x_1, x_2 \in X, \lambda \in [0,1] \tag{3.116}$$

Definition 3.11—Fuzzy partitioning Assume that $\tilde{A}_1$, $\tilde{A}_{2,\ldots,}\tilde{A}_n$ Are fuzzy sets over X so that:

$$\forall x \in X \sum_{i=1}^{n} A_i(x) = 1 \tag{3.117}$$

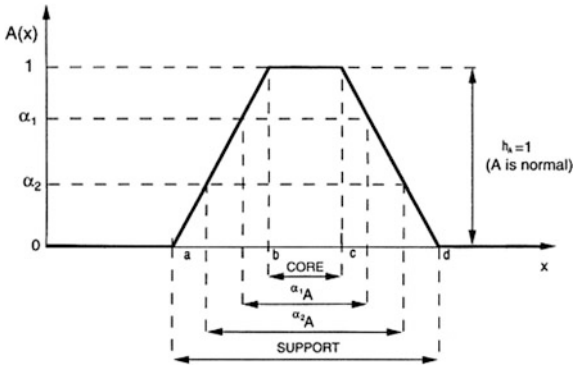


Fig. 3.58 Illustration of some basic characteristics of fuzzy sets (Demicco and Klir 2004)

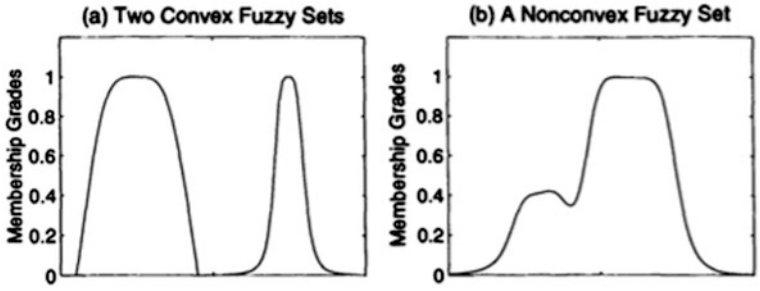


Fig. 3.59 **a** Convex and non-convex **b** fuzzy sets (http://researchhubs.com/post/engineering/fuzzy-system/linguistic-variables.html)

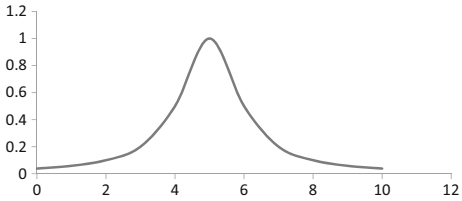


Fig. 3.60 An example of a normal and convex fuzzy set

In this case $\tilde{A}_1$, $\tilde{A}_2$, …,$\tilde{A}_n$ are called the partitions of universal X.

Example 3.51 Assume
$\tilde{Y} = \{(5,0),(10,0.2),(15,0.4),(20,.9),(25,1),(30,0.9),(35,0.5),(40,0.3),(45,0)\}$.
In this fuzzy set the height is 1 and so it is a normal set. Also:

$$SUPP(\tilde{Y}) = \{40, 35, 30, 25, 20, 15, 10\}$$

The passing point of $\tilde{Y}$ is 35. The cardinality of $\tilde{Y}$:

$|\tilde{Y}| = 0 + 0 + 0.2 + 0.4 + 1 + 0.9 + 0.9 + 0.5 + 0.3 = 4.2$

The relative cardinality $||\tilde{Y}|| = 4.2/9 \tilde{=} 0.48$

And the level set: $\Lambda_A = \{0, 0.2, 0.3, 0.4, 0.5, 0.9, 1\}$

Example 3.52 If the universal set X = {1, 2, 3, 4, 5}
$\tilde{A}_1$, $\tilde{A}_2$ are partitions of X:
$\tilde{A}_1$ = {(1, 0), (2, 0.4), (3, 0.7), (4, 1)}
$\tilde{A}_2$ = {91, 1), (2, 0.6), (3, 0.3), (4, 0)}

Example 3.53 The fuzzy set 'real numbers close to 5' with membership function:

$A(x) = \frac{1}{1+(x-5)^2}$ is a normal and convex fuzzy set (Fig. 3.60).

3.8 T-Norm

A t-norm is a bivariate function:

$$t : [0,1] \times [0,1] \rightarrow [0,1] \tag{3.118}$$

With the properties below:

1. $t(0,0) = 0$ (3.119)
2. $t(x,1) = x$ (3.120)

3. $t(u, v) \leq t(w, z)$ if $u \leq w$ and $u \leq z$ (Monotonicity) (3.121)

4. $t(x, y) = t(y, x) : (\text{symmetry})$ (3.122)

5. $t(x, t(y, t)) = t(t(x, y), z)(\text{associativity})$ (3.123)

A t-norm can be considered to be an intersection operator.

Applied to the membership functions of A and B, let:

$$C = A \cap B \text{ then } : \mu_c(x) = t(\mu_A(x), \mu_B(x)) \tag{3.124}$$

The properties of a t-norm describe the natural properties of an intersection.

The first property states that if an element does not belong to either one of the sets, that it does not belong to the intersection.

The second property states that if an element if surely belongs to one of the sets it also belongs to the intersection at the same level as it belongs to the other set.

The third property means that if an element belongs to both intersecting sets more than the other one, it also belongs to the intersection more than the other one.

Properties 4 and 5 assure that the intersection is independent of the order in which the intersecting sets are considered.

T-norms define generalizations of the intersection of ordinary sets. The union of fuzzy sets can be defined with a t-norm function which assigns the membership value to an element in the union depending on the individual membership values in their respective sets.

3.9 S-Norm

Definition—A s-norm is a bivariate function:

$$S : [0, 1] \times [0, 1] \rightarrow [0, 1] \tag{3.125}$$

With below properties:

1. $S(1, 1) = 1$ (3.126)

2. $S(x, 0) = x$ (3.127)

3. $S(u, v) \leq S(w, z)$ if $u \leq w$ and $v \leq z$ (monotonicity) (3.128)

4. $S(x, y) = S(y, x)$; (symmetry) (3.129)

5. $S(x, s(y, z)) = S(s(x, y), z)$ (Associativity) (3.130)

A S-norm can be taken as a union operator applied to the membership function of $\tilde{A}$ and $\tilde{B}$ let $\tilde{C} = \tilde{A} \cup \tilde{B}$ then:

$$\mu_c = S(\mu_A(x), \mu_B(x)) \tag{3.131}$$

The properties of S-norm ensure that the union is independent of the order in which the arguments are taken. The union so defined is also a generalization of the union of ordinary sets.

There is a close relationship between t-norm and s-norm as shown in Schweizer and Sklar (1961):

$$S(x, y) = 1 - t(1 - x, 1 - y) \tag{3.132}$$

And also:

$$T(x, y) = 1 - S(1 - x, 1 - y) \tag{3.133}$$

3.10 If-then Fuzzy Rules

If-then fuzzy rules are the main base of the fuzzy-rule based modeling with various applications in geophysical problems from forward modeling to interpretation.

In this section we first explain the concept of "fuzzy statements" and then describe the principles of if-then fuzzy rules.

3.11 Fuzzy Statement

A fuzzy statement is a statement with uncertainty and vagueness, i.e. "The residual anomaly is small" or "The velocity of P-wave is very high".

In fuzzy statements verbal or linguistic variables are used. A linguistic variable is a variable with an equivalent verbal expression and is defined as below.

3.12 Linguistic Variable

A linguistic variable is defined with a quintuple (x, T(x), U, G, M) where:

X is the name of variable, U is the universal set, T(x) is the set of terms related to variable x (Term is a fuzzy set).

A semantic rule produced through syntactic rule G and M which relates to each of T(x) its semantic or on the other hand, determines its membership function.

Example 3.54 Assume x is a linguistic variable for tallness of people and U = [0.250]. Each of the terms of this linguistic variable is a fuzzy set over U i.e.: tall, short, not-very tall, very tall, etc., and in general condition can be produced regularly through rule G(x).

$$\mathrm{T(tallness)} = \{\text{tall, short, very tall, not} - \text{very tall, to some extent tall}, \ldots\}$$

M(x) is a rule that gives to each terms a semantic through a membership function of U. For example for term A: "tall" the below membership function can be defined:

$$\mathrm{M(tall)} = \{(\mathrm{u}, \mathrm{A(u)} : \mathrm{A} : \mathrm{U} \rightarrow [0, 1]\} \tag{3.134}$$

Where:

$$\mathrm{A(u)} = \begin{cases} 0 & 0 \leq u \leq 160 \\ \frac{u-160}{50} & 160 \leq u \leq 210 \\ 1 & 210 \leq u \leq 250 \end{cases} \tag{3.135}$$

So the proposition "peter is tall" can be written is:

$$\text{`x is } \tilde{\mathrm{A}}\text{'} \tag{3.136}$$

Where x is the stature of peter and $\tilde{\mathrm{A}}$ is the fuzzy set of "tall".

3.13 Fuzzy Conditional Proposition (Fuzzy if-then Rule)

3.13.1 Definition with Example in Geophysics

A fuzzy if-then rule is defined as:

$$\text{"if} < \langle \text{fuzzy statement} \rangle \text{ then } \langle \text{fuzzy statement} \rangle \text{"} \tag{3.137}$$

This conditional fuzzy proposition is represented as:
If x is $\tilde{\mathrm{A}}$ then y is $\tilde{\mathrm{B}}$ or:
If A(x) then B(y)
where x is a linguistic variable for the Prior statement and y is a linguistic variable for the following statement.

Example 3.55 Assume the relationship between pressure and temperature of a reservoir is as below:

If "pressure is high "then" Temperature is very high".

In this example temperature and pressure are linguistic variables x, y respectively. If we assume 'High' and 'very high' as fuzzy sets F, a, respectively then this can be re-written as:

$$\text{If x is } \tilde{\mathrm{F}} \text{ then y is } \tilde{\mathrm{G}} \tag{3.138}$$

Fuzzy rules are used widely in fuzzy controllers, fuzzy expert systems, fuzzy approximate reasoning, pattern recognition and fuzzy decision making.

Using if-then fuzzy rules we can model the behavior of nonlinear systems or complex systems which can't be modeled easily with classical algorithms of modeling.

In geophysical aspects we can use if-then fuzzy rules especially for interpretation of geophysical data i.e. well-logs, conductivity, gravity, geomagnetic anomalies, etc. For example the statement "If the amplitude of the residual gravity is high then the size of the object is big or the depth is shallow". Variables 'big' and 'shallow' are linguistic (Fig. 3.61).

Example 3.56 The shape factor 'q' is used to calculate the gravity anomaly of simply shaped objects such as: sphere (q = 0.5), horizontal cylinder (q = 1), vertical cylinder (q = 5):

$$g(x) = \frac{A}{(x^2 + z^2)^q} \tag{3.139}$$

When we calculate the value of q for real bodies from residual gravity data of a principle profile the variable q can take values which are not exactly the above mentioned but 'closer'.

Then we can have the fuzzy if-then rule for the shape factor as below:

- If q is close to 1.5 then the buried object shape is close to a sphere.
- If q is close to 5 then the buried object shape is close to a vertical cylinder.
- If q is close to 1 then the buried object shape is close to a horizontal cylinder.

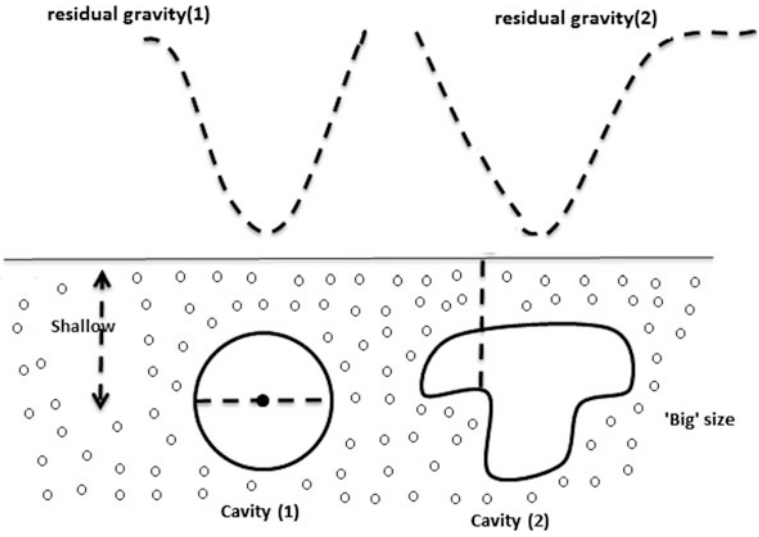


Fig. 3.61 The 'linguistic' variables used in gravity data interpretation ('big', 'shallow' are linguistic variables)

The most important problem for if-then fuzzy rules is the quality of their interpretation. In bi-value logic or classical logic when the rule P $\Longrightarrow$ q is defined it is very easy to find its value because the final interpretation is zero or one.

The conditional proposition value-table is demonstration of the relationship between the prior and the following. On the other hand a conditional Proposition is false when its prior is true but its following is false.

This relationship can be presented as a value-function as below:

$$\mathrm{R}(\mathrm{p},\mathrm{q}) = \begin{cases} 1 & p \le q \\ 0 & p > q \end{cases} \tag{3.140}$$

Similarly in fuzzy logic the fuzzy if-then is interpreted from the fuzzy point of view.

3.13.2 Interpretation of Fuzzy if-then Rule

Assume the fuzzy rule: if x is $\tilde{\mathrm{F}}$ then y is $\tilde{\mathrm{G}}$. The interpretation of this rule is a fuzzy relationship such as $\tilde{\mathrm{R}}$(x, y) which is named the fuzzy implication. There are different various necessities and some of them are listed in Table 3.29.

Example 3.57 Assume the fuzzy rule: "If it rains heavily then the humidity is high". Assume X is the linguistic variable for strength of raining, H is the fuzzy set 'High', y is the linguistic variable for amount of humidity, HN is the fuzzy set 'High', then the if-then rule is defined as: if x is $\tilde{\mathrm{H}}$ then y is $\tilde{\mathrm{H}}$N. Assume $\tilde{\mathrm{H}}$, $\tilde{\mathrm{H}}$N are as below (the values are in the denominator and its membership degree is in the numerator; note this format is also a way to show fuzzy set members):

$$\widetilde{\mathrm{HN}} = \frac{0.1}{25} + \frac{0.6}{50} + \frac{0.8}{75} + \frac{1}{100} \tag{3.141}$$

$$\tilde{\mathrm{H}} = \frac{0.2}{20} + \frac{0.4}{40} + \frac{0.6}{60} + \frac{0.8}{80} + \frac{1}{100}$$

Thus the interpretation of the rule through Mamdani implications (Table 3.30):

Table 3.29 Definition of some of fuzzy implications

Name of relation	R(x, y)
Zadeh	$\max\{\min(F(x), G(x)), 1 - F(x)\}$
Denis-Rescher	$\max\{1 - F(x), G(y)\}$
Mamdani	$\min\{1 - F(x), G(y)\}$
Łukasiewicz	$\min\{1, 1 - F(x) + G(y)\}$
Larsen	$F(x).G(y)$
Gödel	$\begin{cases} 1 & F(x) \le G(y) \\ 0 & otherwise \end{cases}$

Table 3.30 Interpretationof if-then fuzzy rule through Mamdani implication

R	25	50	75	100
20	0.1	0.2	0.2	0.2
40	0.1	0.4	0.4	0.4
60	0.1	0.6	0.6	0.6
80	0.1	0.6	0.8	0.8
100	0.1	0.6	0.8	1

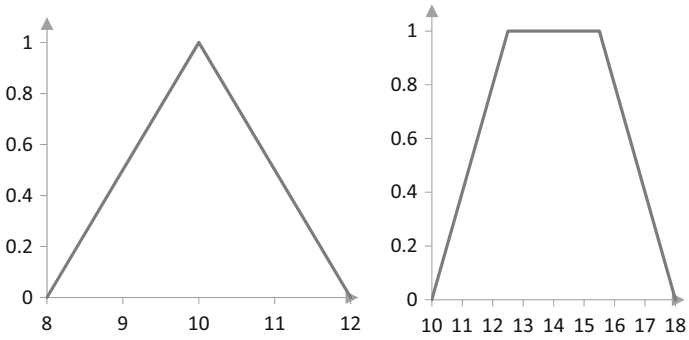


Fig. 3.62 Fuzzy sets left: $\tilde{A}$, right: $\tilde{B}$

$$R(x, y) = \min(H(x), HN(y)) \tag{3.142}$$

Example 3.58 Assume if-then fuzzy rule as below:

$$\text{if } x \text{ is } \tilde{A} \quad \text{then} \quad y \text{ is } \tilde{B} \tag{3.143}$$

$\tilde{A}$ and $\tilde{B}$ are shown in Fig. 3.62. The interpretation of this rule is depicted in Fig. 3.63 through the Denis-Richer implication and in Fig. 3.64 using Mamdani implication.

3.14 Approximate Reasoning

3.14.1 Fuzzy Inference

As mentioned before the human brain automatically uses approximate reasoning which is not provable via classical explicit reasoning or through exceptional deduction rules like the inference below.

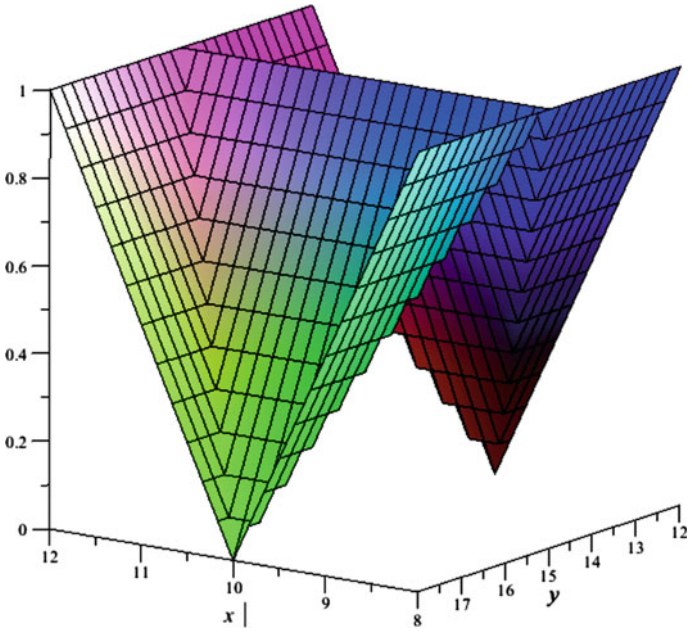


Fig. 3.63 Denis-Rescher implication

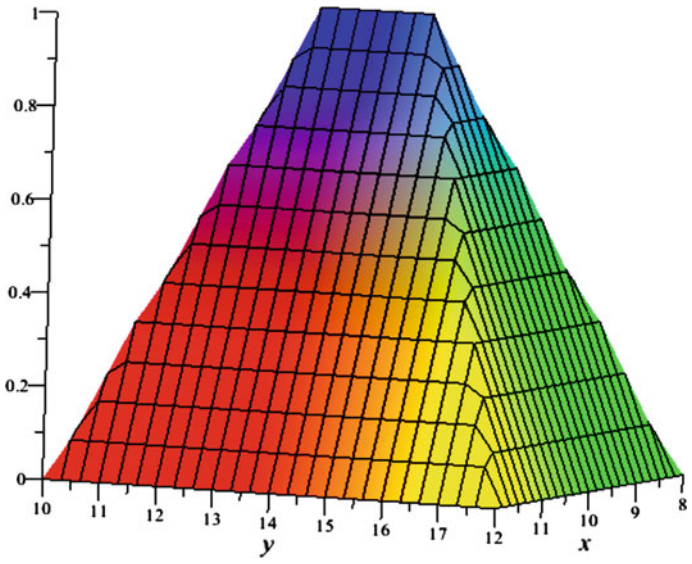


Fig. 3.64 Mamdani implication

– If the amount it is raining is high then humidity is high.
– The amount of rain is high.

This inference is explained through fuzzy logic. Using the exceptional deduction rule in fuzzy state (which is named the fuzzy extended exceptional deduction rule).

3.14.2 Fuzzy Extended Exceptional Deduction Rule

Assume the rule: if x is $\tilde{A}$ then y is $\tilde{B}$. If the observation is: x is $\tilde{A}^*$ then the result is: y is $\tilde{B}^*$ represents as below:

– If x is $\tilde{A}$ then y is $\tilde{B}$
– $\underline{\text{X is } \tilde{A}^*}$
– Y is $\tilde{B}^*$

This fuzzy interpretation is a fuzzy relationship named implication thus the result is obtained through combination of the relationship with observation:

$$\widetilde{B}^* = \widetilde{A}^* o \widetilde{R}^* \Rightarrow B^*(y) = \sup_x \{\min(A^*(x), R(x, y))\} \tag{3.144}$$

The above operations are named **fuzzy inference**. The fuzzy inference is like a system with an input A* and output B* (Fig. 3.65).

Example 3.59 Assume the fuzzy-rule in the previous example (example 3.58). If the amount of raining is observed as very high what will be the amount of humidity?

The fuzzy set, "very high" is:

$$\widetilde{H}^* = \frac{0.05}{20} + \frac{0.3}{40} + \frac{0.5}{60} + \frac{0.7}{80} + \frac{1}{100} \tag{3.145}$$

The approximate reasoning is as below:

$$\begin{aligned} &- \quad \text{If x is } \tilde{H} \text{ then y is } \tilde{HN} \\ &- \quad \underline{x \text{ is } \tilde{H}*} \\ &- \quad \text{y is } \tilde{HN}^* \end{aligned} \tag{3.146}$$

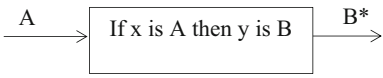


Fig. 3.65 Fuzzy interference

According to fuzzy exceptional inference deduction we have:

$$\widetilde{HN}^* = \widetilde{H}^* \circ \widetilde{R}, \text{(R was calculated in the privious example), thus :}$$

$$\widetilde{HN}^* = [0.05 \quad 0.3 \quad 0.5\ 0.7 \quad 1]\text{o}\begin{bmatrix} 0.1 & 0.2 & 0.2 & 0.2 \\ 0.1 & 0.4 & 0.4 & 0.4 \\ 0.1 & 0.6 & 0.6 & 0.6 \\ 0.1 & 0.6 & 0.8 & 0.8 \\ 0.1 & 0.6 & 0.8 & 1 \end{bmatrix} = [0.1\,0.6\,0.8\,1]$$

Now, assume the membership functions of fuzzy sets are continuous then the fuzzy inference is the same as before. If the Mamdani implication is used it has an interesting geometrical interpretation. Assume the below fuzzy if-then rule:

$$\text{if x is } \widetilde{A} \text{ then y is } \widetilde{B}$$

Where A(x) and B(x) are continuous membership functions and the input of this rule A*(x) is also continuous then the output of the above fuzzy rule will be as below:

$$\begin{aligned} B^*(y) &= A^* \circ R = \sup_x\{\min(A^*(x), R(x,y))\} = \sup_x\{\min(A^*(x), \min(A(x), B(y)))\} \\ &= \sup_x\{\min(\min(A(x), A^*(x)), B(y))\} = \sup_x\{\min((A \cap A^*)(x), B(y))\} \\ &= \min\left\{\sup_x((A \cap A^*)(x), B(y))\right\} = \min\{\text{hgt}(A \cap A^*), B(y)\} = \min(\alpha, B(y)) \end{aligned} \tag{3.147}$$

Where α = hgt (A $\cap$ A*) is the height of the intersection of the prior fuzzy set and input fuzzy set. Its geometrical interpretation is depicted in Fig. 3.66.

Example 3.60 Ina cooler system one of the fuzzy rules is: if the 'room temperature is high' then 'the motor power is very high'. Assume the linguistic variable 'high'

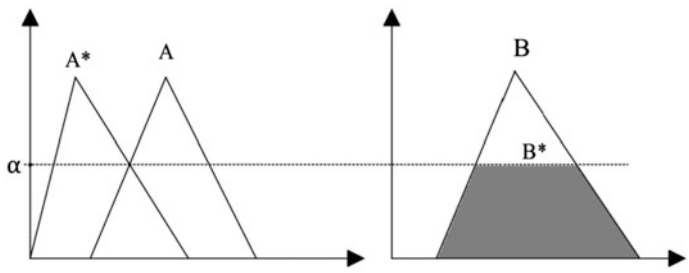


Fig. 3.66 Geometrical interpretation of if-then fuzzy rule: if x is $\tilde{A}$ then y is $\tilde{B}$

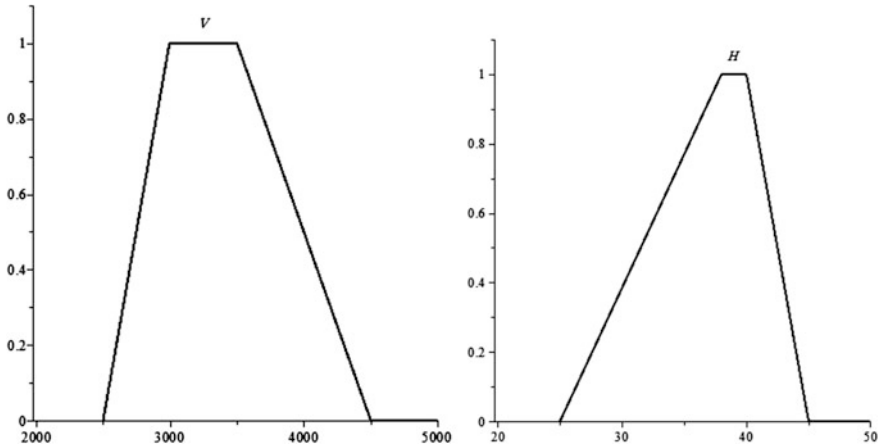


Fig. 3.67 Fuzzy set for ($\tilde{H}$) and $\tilde{V}$

as the $\tilde{H}$ fuzzy set and Assume the linguistic variable 'very high' as the $\tilde{V}$ fuzzy set (Fig. 3.67) then the fuzzy rule is:

$$\text{If x is } \tilde{H} \text{ then y is } \tilde{V} \tag{3.148}$$

where x is the linguistic variable for room temperature and y is the linguistic variable for rotational power of motor. The fuzzy sets are shown below. If the room temperature (T) is about 30 °C, then how much will the rotational power be? Solve: The input is T* = {T| T is about 33} so the output will be V* = ROT* and V* is shown in Fig. 3.68.

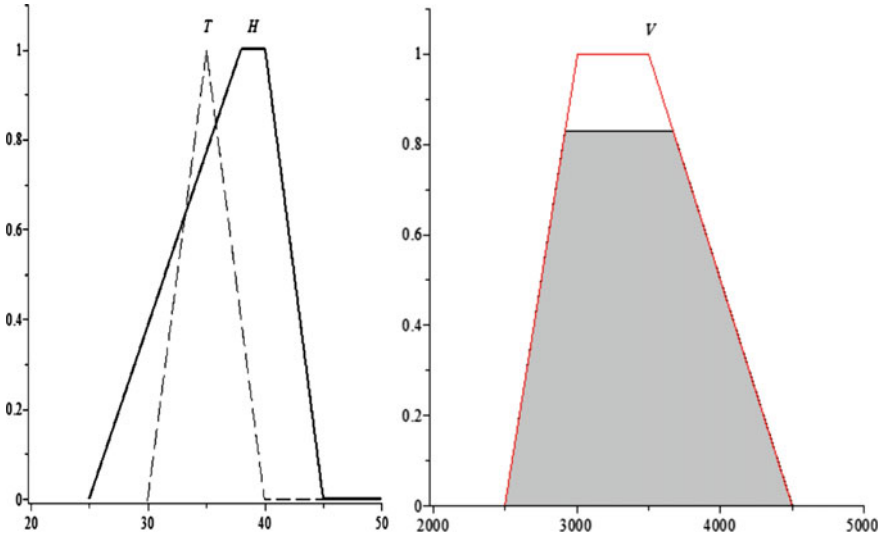


Fig. 3.68 The fuzzy output V* (Soleimani and Hajian 2017)

3.15 Fuzzy Rules Base

3.15.1 Definition Assume F_1, G_1, I = 1, 2,…, N Are Fixed Fuzzy Sets Over Set U then

$$RB\begin{cases} if\ F_1\ then\ G_1 \\ . \\ . \\ if\ F_n\ then\ G_n \end{cases} \tag{3.149}$$

is named a fuzzy if-then rules base.

In fuzzy systems the fuzzy rule base is used, thus the interpretation of this type of base plays an important role in fuzzy systems. There are two common methods for inference of this fuzzy-rules base. The first method is named FATI which means First Associate Then Inference. The second method is named FITA which means: First Inference Then Associate. In the first method first all the rules are satisfied and then inference is done but in the second method each inference is applied separately and then the results are associated.

3.15.2 FATI Method

Assume RB is a fuzzy if-then rules base and $\tilde{R}_i$ is a standard relationship base on $\tilde{F}_i$, $\tilde{G}_i$ respect to ith rule:

$R_i(x,y) = \min(F_i(x), G_i(y))$, if the input of this rules base is $\tilde{F}^*$ the output is:

$$B*(y) = \sup_{x\in U}\{\min(F*(x), \max\{R_1(x,y),\ldots,R_n(x,y)\})\} \tag{3.150}$$

This method means that if a rules base is defined as in this equation, the fuzzy standard relationship of each rule is calculated first: $R_i(x,y) = \min(F_i(x), G_i(y))$, then the standard relationship of the base is defined as: $R(x,y) = \max_i R_i(x,y)$. So if $\tilde{F}^*$ is the input of the base then the output is:

$$FATI^{RB}(F*)(y) = \sup_{x\in U}\{\min(F*(x), R(x,y))\} \tag{3.151}$$

where $FATI^{RB}$ (F*) (y) is B*(y).

3.15.3 FITA Method

Assume RB is a fuzzy if-then rule base, whereas the input of this base is F*, in FITA method the input is applied to each of the rules and the output is calculated: $G^*_i(y) = \sup\{\min(F^*(x), R_i(x, y))\}$ then these outputs are associated:

$$G*(y) = \max\{G*_1(y), G*_2(y), \ldots, G*_n(y)\} \tag{3.152}$$

In fuzzy systems the input is usually a crisp number thus at the first stage it is necessary to fuzzificate the input, on the other hand if the input is for example x_0, fuzzification of x_0 means calculating its membership degree to the fuzzy set of the prior proposition.

Example 3.61 Assume a fuzzy rules base with two rules, x_0 is the input of the base, as shown in Fig. 3.69, calculate the output of this base.

$$\text{If x is } \tilde{A}_1 \text{then y is } \tilde{B}_1 \tag{3.153}$$

$$\text{If x is } \tilde{A}_2 \text{then y is } \tilde{B}_2$$

Fuzzy rules base plays a basically role in fuzzy controllers. A fuzzy rule-base model is shown in Fig. 3.70.

3.16 Defuzzification

As can be seen in the Fig. 3.70, one of the final stages for fuzzy rule based modeling is defuzzification. Some of the most useful defuzzification methods are: method of center of mass, center of totals, means of max and height method and these are investigated in the next sections.

Assume $\tilde{A}$ is a defined fuzzy set over x with membership function A(x) and its defuzzification is represented by x*. The value of x* can be obtained using one of the above mentioned methods that will explain in the following sections.

3.16.1 Center of Gravity (Centroid of Area) Defuzzification

A—if x is a discrete set $X = \{x_1, x_2, \ldots, x_n\}$ its defuzzification is obtained via the formula below:

$$x* = \frac{\sum_{i=1}^{n} x_i A(x_i)}{\sum_{i=1}^{n} A(x_i)} \tag{3.154}$$

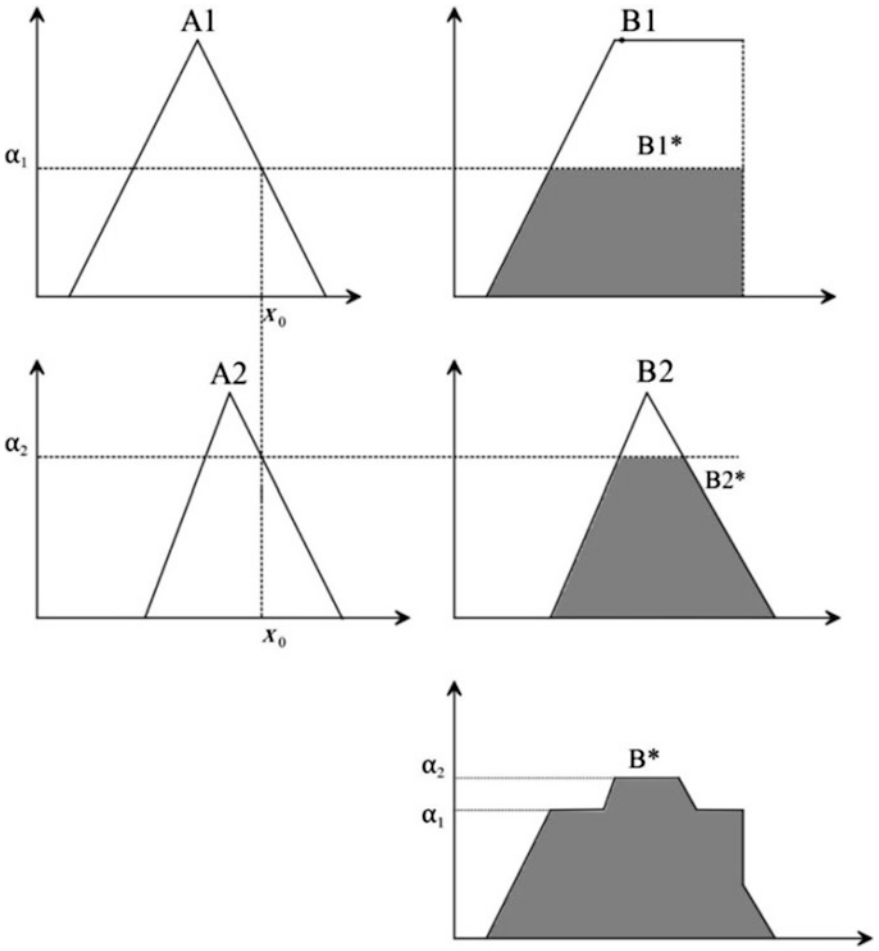


Fig. 3.69 Calculation output of a fuzzy rules base for a crisp input

B—if x is a continuous set then:

$$x* = \frac{\int_x x\ A\ (x)dx}{\int_x A(x)dx} \tag{3.155}$$

A schematic of a center of gravity defuzzificator is shown in Fig. 3.71.

Example 3.62 Defuzzificate the fuzzy set below:

$$\tilde{A} = \frac{0.2}{1} + \frac{0.5}{2} + \frac{0.7}{3} + \frac{0.9}{4} + \frac{1}{5} \tag{3.156}$$

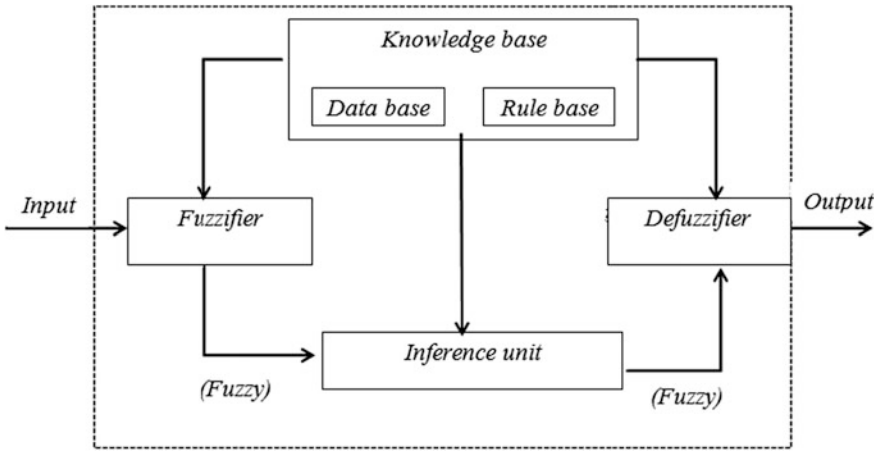


Fig. 3.70 Schematic diagram of a fuzzy rule-base model (fuzzy system controller)

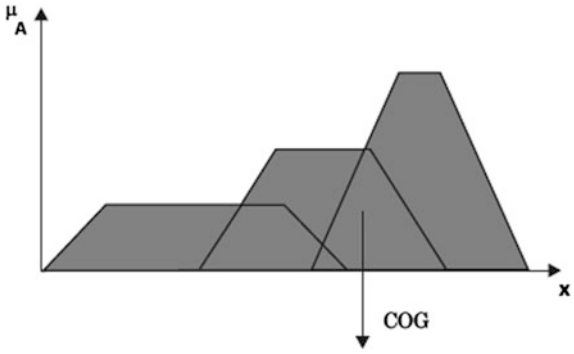


Fig. 3.71 Schematic of a defuzzificator base on center of gravity method

$$x* = \frac{0.2 \times 1 + 0.5 \times 2 + 0.7 \times 3 + 0.9 \times 4 + 1 \times 5}{0.2 + 0.5 + 0.7 + 0.9 + 1} = 3.6 \qquad (3.157)$$

Here as the set numbers is integer and the nearest integer number to 3.6 is selected as the defuzzification of $\tilde{A}$.

3.16.2 Center of Sum Method

In a fuzzy system, if $\tilde{B}_1$, $\tilde{B}_2$, …,$\tilde{B}_m$ are output fuzzy sets, defuzzification of the final output is calculated via the formula below:

$$x* = \frac{\sum_{i=1}^{n} xi \sum_{x=1}^{m} B_k(x_i)}{\sum_{i=1}^{n} \sum_{x=1}^{m} B_x(x_i)} \quad \text{(for discrete sets)} \tag{3.158}$$

and

$$x* = \frac{\int_x x \sum_{x=1}^{m} B_{x(x)dx}}{\int_x \sum_{x=1}^{m} B_x(x)dx} \quad \text{(for continuous sets)} \tag{3.159}$$

3.16.3 Mean of Max Method

Assume $M = \{x_i | A(x_i) = \text{hgt}(\tilde{A})\}$, in this case:

$$x* = \frac{\sum_{x_{i \in M}} x_i}{|M|} \tag{3.160}$$

This method is depicted in Fig. 3.72.

3.16.4 Height Method

In a fuzzy system, if $\tilde{B}_1, \tilde{B}_2, \ldots, \tilde{B}_m$ are output fuzzy sets and $H_k = \text{hgt}(\tilde{B}_k)$ then:

$$x* = \frac{\sum_{k=1}^{m} H_k x_k}{\sum_{k=1}^{m} H_k} \tag{3.161}$$

In Fig. 3.73 the height defuzzification method is shown for a four fuzzy output.

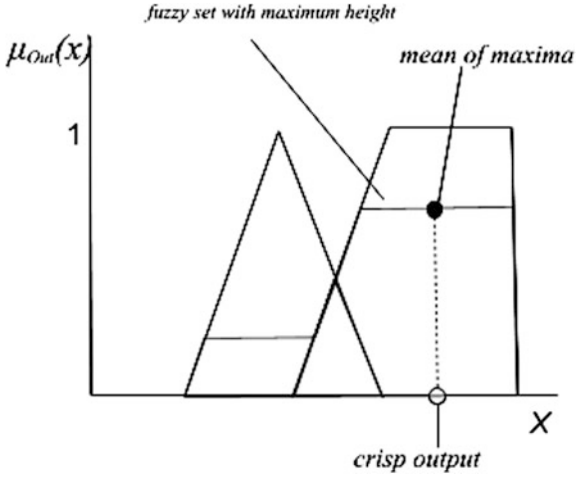


Fig. 3.72 Mean of max defuzzification method. *Source* http://control.ee.ethz.ch/~ifa-fp/wiki/index.php?n=Main.FuzzyHeli

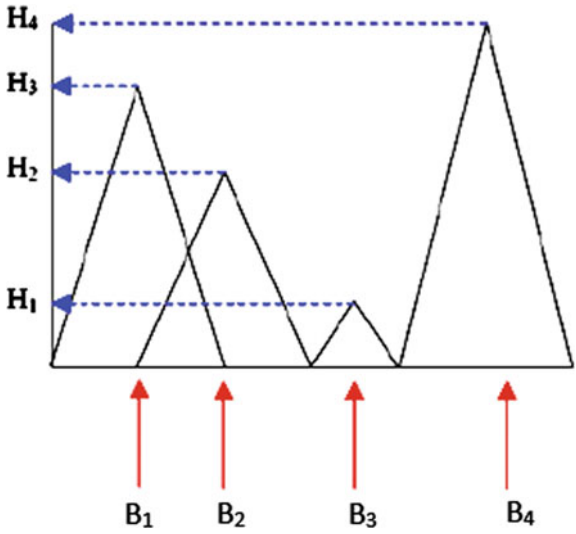


Fig. 3.73 Height defuzzification method for four fuzzy sets

Example 3.63 Assuming the fuzzy rules base below. If this base input $x_0 = 3.5$ then calculate the output through center of gravity defuzzification method.

- If x is $(2, 1, 1)_{LR}$ then y is $(12, 1, 1)_{LR}$
- If x is $(3, 0.75, 1)_{LR}$ then y is $(14, 2, 1)_{LR}$

where $L(x) = R(x) = 1 - |x|$.

$$B*(y) = \begin{cases} x - 11 & 11 < x \leq 11.5 \\ 0.5 & 11.5 \leq x \leq 13.5 \\ 13 - x & 13.6 \leq x \leq 13.67 \\ 0.33 & 13.67 < x \leq 14.67 \\ 15 - x & 14.6 < x \leq 15 \end{cases} \tag{3.162}$$

B*(y) is shown in Fig. 3.74. As it can be seen the output for $x_0 = 3.5$ is 0.5.

3.16.5 Bisector Defuzzification

This method uses a vertical line that divides the area under the membership curve into two equal areas:

$$\int_{\alpha}^{z} A(x)dx = \int_{z}^{\beta} A(x)dx \tag{3.163}$$

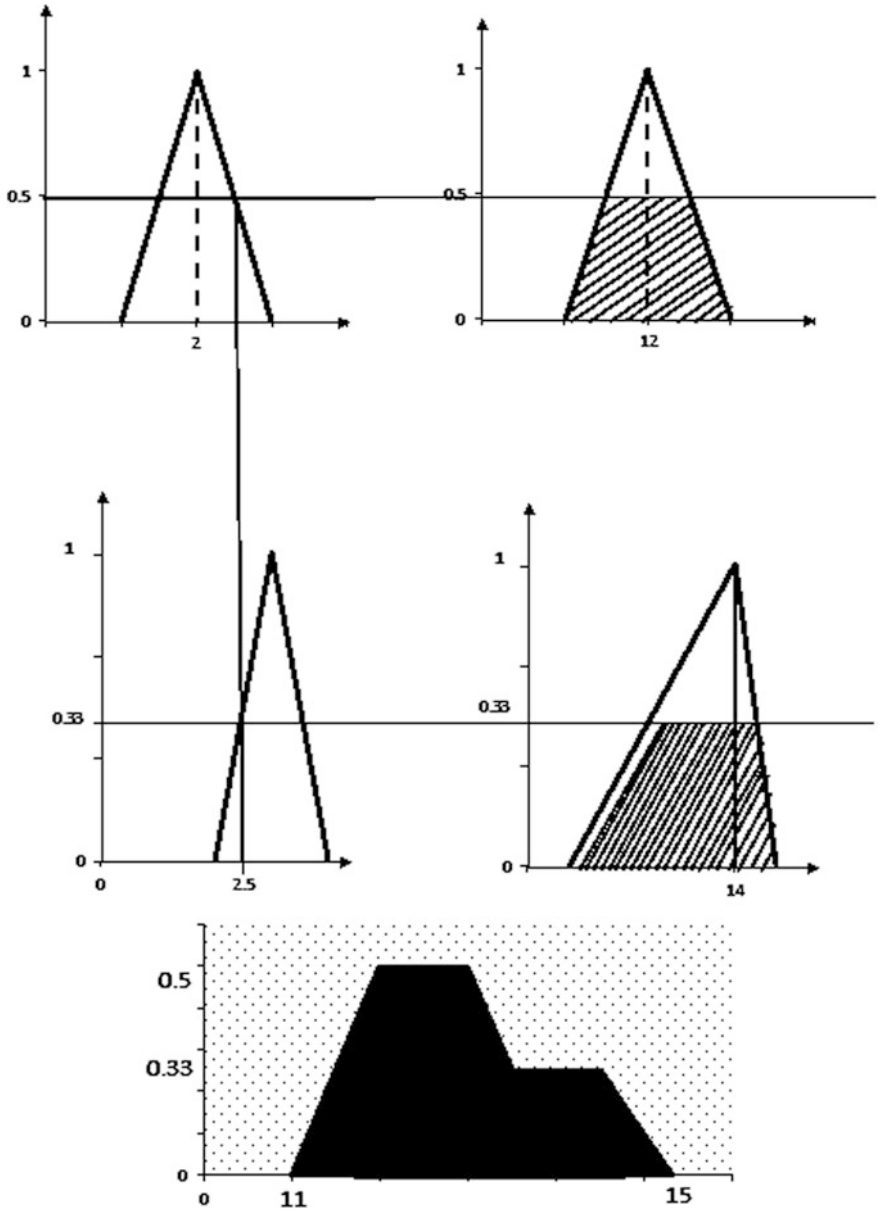


Fig. 3.74 Center of gravity defuzzification method for example 3.63 (Soleimani and Hajian 2017)

3.16.6 Smallest of Maximum Defuzzification

This method uses the minimum value of the aggregated membership function outputs.

$$z_0 \in \left\{ x | \mu(x) = \min_{\omega} \mu(\omega) \right\} \tag{3.164}$$

3.16.7 Largest of Maximum Defuzzification

This method uses the maximum value of the aggregated membership function outputs.

$$z_0 \in \left\{ x | \mu(x) = \max_{\omega} \mu(\omega) \right\} \tag{3.165}$$

3.16.8 Weighted Average Defuzzification Method

This method is based on the peak value of each fuzzy set, and calculates weighted sum of these peak values (Kaehler 1998). Depending on these weight values and the degree of membership for fuzzy output, the crisp value of the output is calculated by the following formula:

$$\mathrm{x}* = \frac{\sum_{i=1}^{m} A(x)_i W_i}{\sum_{i=1}^{m} W_i} \tag{3.166}$$

Where $\mathrm{A(x)_i}$ is the degree of membership in output singleton i, W_i and is the fuzzy output weight value for the output singleton i (Ross et al. 2010). This method is valid for symmetrical output membership functions.

A graphical comparison among the smallest of max, largest of max, bisector of area centroid of area and mean of max defuzzification methods is illustrated in Fig. 3.75.

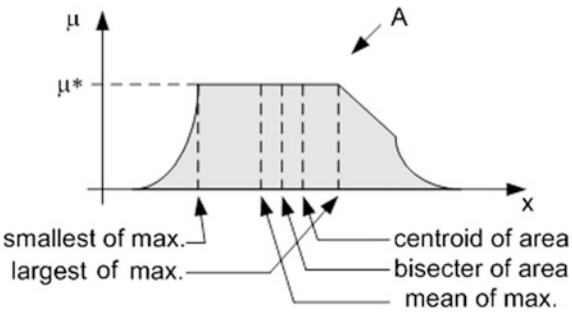


Fig. 3.75 Graphical demonstration of defuzzification methods (Ross et al. 2010). *Source* http://access.feld.cvut.cz/rservice.php?akce=tisk&cisloclanku=2012080002

3.17 Fuzzifiers

A Fuzzifier is a mapping of a point x* ∈ u CR^n to a fuzzy set A′ defined in U. The criterions to design a fuzzifier are:

a. Fuzzifier should satisfy the fact that x* input is crisp, hence the fuzzy set A′ should have the maximum degree of membership.
b. If the input signal of the fuzzy system is disturbed with noise, the fuzzifier should be able to attenuate the effect of noise or eliminate it.
c. Fuzzifier should simplify the calculations of the fuzzy inference engine.

Here we introduce three of the most common fuzzifiers:

3.17.1 Singleton Fuzzifier

This fuzzifier maps a point x∈U to a singleton fuzzy set A′ over U and its membership degree is equal to 1 at x* and equal to zero for other points, it means that:

$$\mu_{A'}(x) = \begin{cases} 1 & x = x^* \\ 0 & otherwise \end{cases} \tag{3.167}$$

This fuzzifier maps x*∈ U to a fuzzy set over U with a Gaussian membership function as below:

$$\mu_{A'}(x) = e^{\frac{-(x-x_1^*)^2}{a_1}} * \ldots * e^{\frac{-(x-x_n^*)^2}{a_n}} \tag{3.168}$$

where a_i is positive and '*' represents a t-form operator which is usually selected as either min or product operators.

3.17.2 Triangular Fuzzifier

This fuzzifier maps x* ∈ U to a fuzzy set A′ over U with a triangular membership function as below:

$$M_{A\prime}(x) = M_{A\prime}(x) = \begin{cases} \left(1 - \frac{|x-x_1^*|}{b_1}\right) * \ldots * \left(1 - \frac{|x-x_n^*|}{b_n}\right) : |x - x_i^*| \le b_i \\ 0 \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad O.W. \end{cases} \tag{3.169}$$

Where b_i is positive and * shows a t-norm which is selected usually as an algebraic product or min. operator.

Note that:

- Singleton fuzzifiers simplify the calculations of fuzzy inference engine for any kind of membership functions in if-then fuzzy rules.
- Gaussian and triangular fuzzifiers simplify the calculation when the membership functions are Gaussian or triangular.
- Guassian and triangular fuzzifiers can attenuate the noise available in the input whereas singleton fuzzifiers do not have this ability.

3.18 Fuzzy Modeling Using the Matlab Toolbox

The fuzzy logic toolbox enables powerful fuzzy interference system design (FIS) in a graphical user friendly space. This toolbox has two main parts: part one includes editing tools: the FIS editor, and the Membership Function Editor and part two includes read only tools: the Rule viewer and the surface viewer (Fig. 3.76).

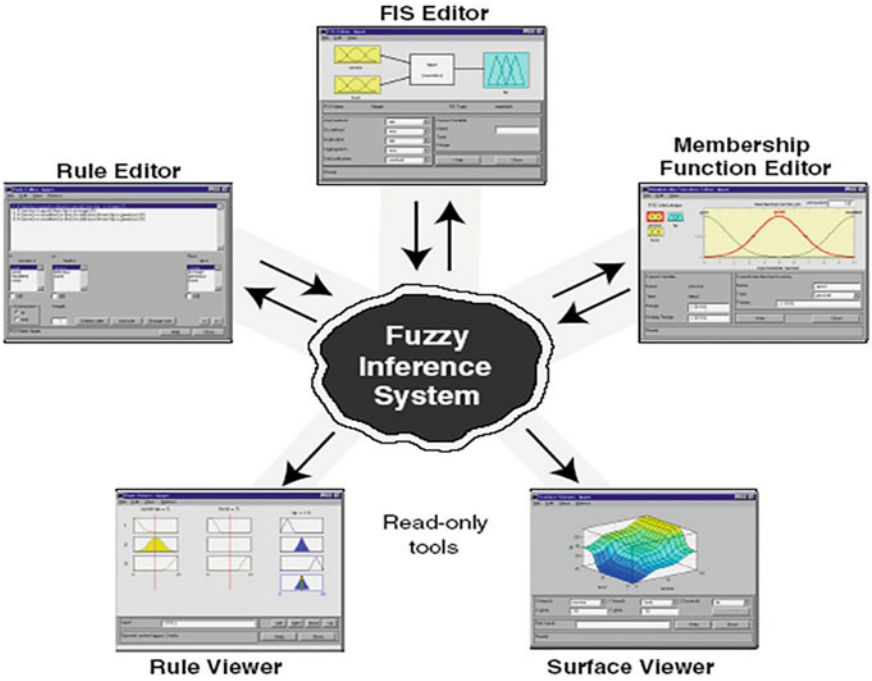


Fig. 3.76 Editing and Read-only tools of Fuzzy modeling toolbox in Matlab. *Source* Matlab Fuzzy Toolbox Tutorial Help, Mathworks (2009)

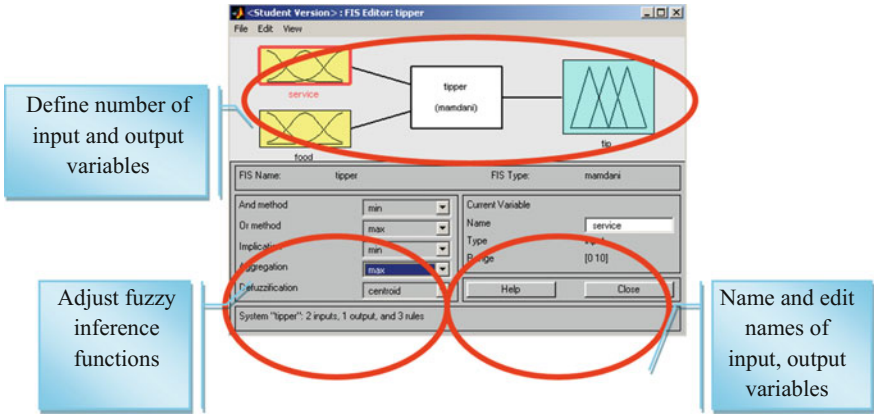


Fig. 3.77 FIS editor tool. *Source* Matlab Fuzzy Toolbox Tutorial Help, Mathworks (2009)

To access the fuzzy logic designer it is enough to enter 'fuzzy' in command window:

$$\gg \textbf{fuzzy}$$

On applying this, a command window named 'fuzzy logic designer: untitled':

is opened as shown in Fig. 3.77.

The fuzzy designer is basically used to design FIS which is based on if-then fuzzy rules and it has two main tools: Editor and viewer tools. With the editor tools you can edit FIS input and outputs structure. Rules and membership functions and in viewer tool, are read-only tools.

3.18.1 Fuzzy Inference System (FIS) Editor

In this section you can define the number of input and output variables, name and edit names of input/output and also adjust fuzzy inference functions.

3.18.2 Membership Function Editor

This tool allows you to save, open, or edit a fuzzy system using any of the five basic GUI tools. It covers the items (Fig. 3.78):

- Select & edit attributes of membership function
- Display & edit values of current variable
- Name & edit parameters of membership function

More details about this tool are shown in Fig. 3.79 (Mathworks 2009).

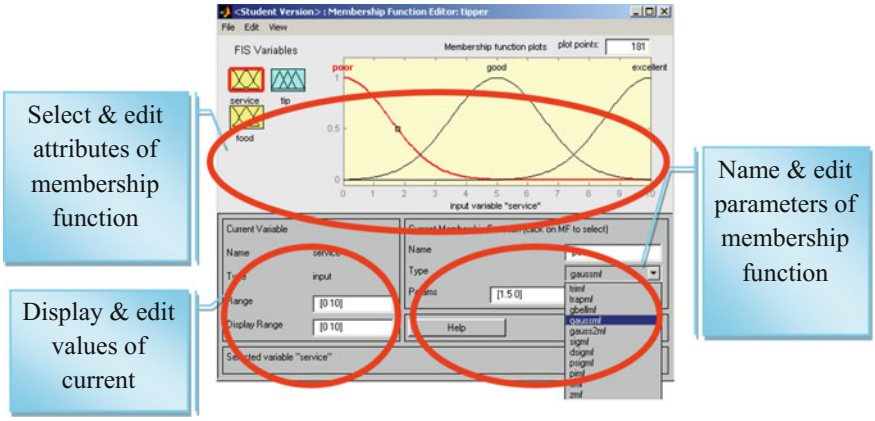


Fig. 3.78 Membership function editor. *Source* Matlab Fuzzy Toolbox Tutorial Help, Mathworks (2009)

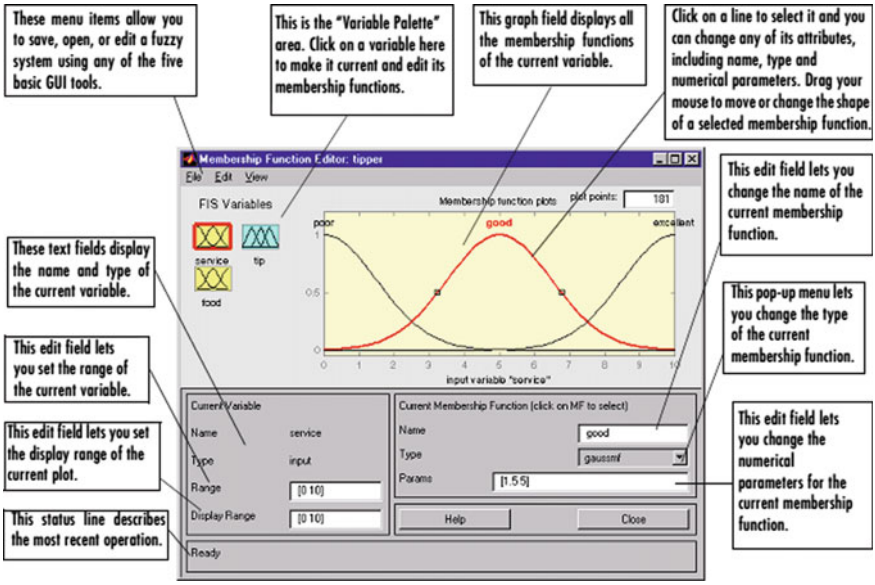


Fig. 3.79 Membership function editor with more details. *Source* Matlab Fuzzy Toolbox Tutorial Help, Mathworks (2009)

3.18.3 Rule Editor

This tool helps you to create and edit rules. When you add a new rule or change a rule parameters, rules will automatically be updated (Fig. 3.80).

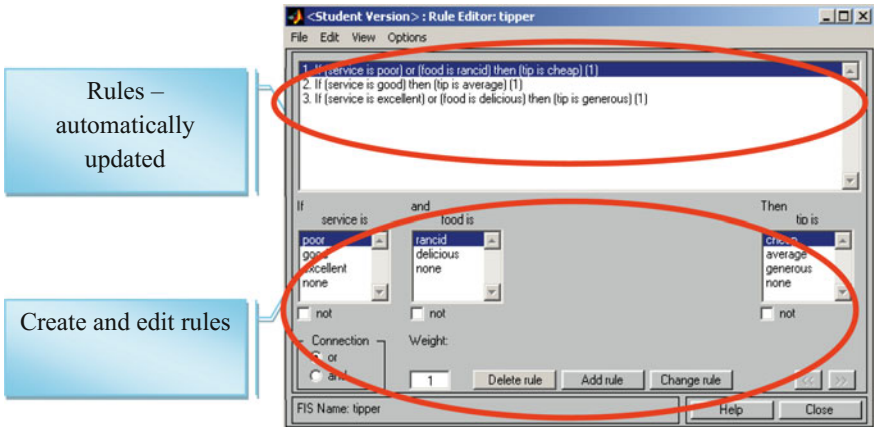


Fig. 3.80 Rule editor. *Source* Matlab Fuzzy Toolbox Tutorial Help, Mathworks (2009)

3.18.4 Rule Viewer

As mentioned before, the rule viewer is a read-only tool to show how input variables are used in rules and how output variables are used in rules and finally the output value of the fuzzy system (Fig. 3.81).

3.18.5 Surface Viewer

This is a read-only tool that when you specify input and output variables, displays the output surface for any system output versus any one (or two) inputs (Fig. 3.82).

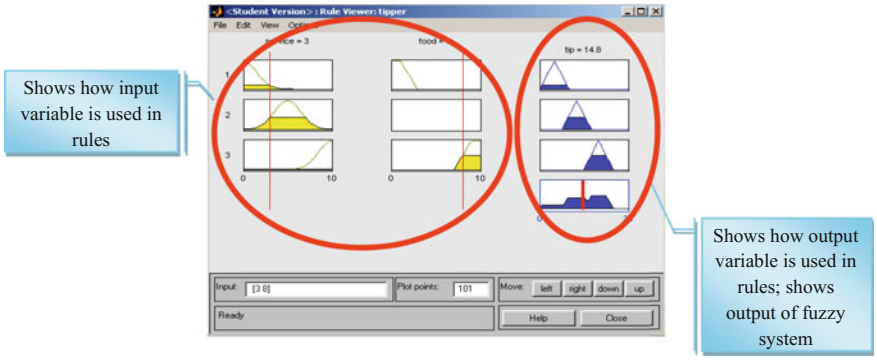


Fig. 3.81 Rule viewer. *Source* Matlab Fuzzy Toolbox Tutorial Help, Mathworks (2009)

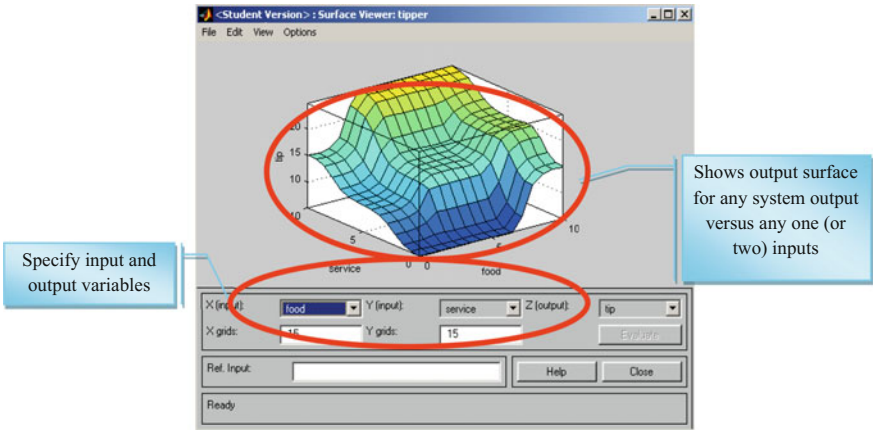


Fig. 3.82 Surface viewer. *Source* Matlab Fuzzy Toolbox Tutorial Help, Mathworks (2009)

References

Adams R.D., Collister J.W., Ekart D.D., Levey R.A. and Nikravesh M., 1999a, Evaluation of gas reservoirs in collapsed paleo cave systems. AAPG Annual Meeting, 11–14 April, San Antonio, Texas, USA, Expanded Abstracts.

Adams R.D., Nikravesh M., Ekart D.D., Collister J.W., Levey R.A. and Siegfried R.W., 1999b, Optimization of well locations in complex carbonate reservoirs. GasTIPS 5.

Aminzadeh F., 1989, Future of expert systems in the oil industry, Proceedings of the HARC Conference on Geophysics and Computers, A Look at the Future.

Aminzadeh F. and Chatterjee S., 1984/1985, Applications of clustering in exploration seismology, Geoexploration, 23, 147–159.

Aminzadeh F. and Jamshidi M., 1995, Soft Computing, Prentice Hall.

An P. , Moon W. M., Rencz A., 1991, Application of fuzzy set theory to integrated mineral exploration ,Canadian Journal of exploration geophysics, 27(1), 1-11.

Bloch I., 1999, On fuzzy distances and their use in image processing under imprecision, 32(11), 1873-1895, doi.org/https://doi.org/10.1016/s0031-3203 (99)00011-4.

Bouchon-Meunier B. and M. Rifqi and S. Bothorel, 1986, Towards general measures of comparison of objects, Fuzzy Sets and Systems, 84(2), 143–153.

Cagnoli B., 1998, Fuzzy logic in volcanology, Episodes, Journal of international geosciences, 21(2), 94–96.

Cas R.A.F., and Wright, J.V., 1988, Volcanic successions, modern and ancient: Chapman and Hall, London, 528.

Demicco R. V. and Klir G.J., 2004, Fuzzy Logic in Geology, Elsevier Science (USA).

Dhar M., 2012, A Note on Subsethood Measure of Fuzzy Sets, International Journal of Energy, Information and Communications, 3(3), 55–62.

Dubios D. and Prade H., 1978, Operation on fuzzy numbers, International Journal of Systems Science, 9(6), 613–626.

Ghasem-Aghae N., Kaedi M., Ören TI. 2005, Effects of Cognitive Complexity in Agent Simulation: Fuzzy Rules and an Implementation, Proceedings of: CM&SC-Conceptual Modeling and Simulation Conference, 20–22.

Grandjean G., Malet J., Bitri A., and Méric O., 2006, Geophysical data fusion by fuzzy logic for imaging the mechanical behavior of mudslides, Bull. Soc. géol.Fr., t. 177(2), 127–136.
Hajian A., Zomorrodian H., Styles P., Greco F. , Lucas C., 2012, Depth estimation of cavities from microgravity data using a new approach: the local linear model tree (LOLIMOT), Near Surface Geophysics, 10, 221-234, https://doi.org/10.3997/1873-0604.2011039
http://access.feld.cvut.cz/rservice.php?akce=tisk&cisloclanku=2012080002.
http://control.ee.ethz.ch/∼ifa-fp/wiki/index.php?n=Main.FuzzyHeli.
http://researchhubs.com/post/engineering/fuzzy-system/linguistic-variables.html.
http://slideplayer.com/slide/5336605.
http://www.jsjgeology.net/San-Salvador,Bahamas-caves-karst.htm.
http://www.slideshare.net/oncel/geophysics-overview2, slide17.
https://en.wikipedia.org/wiki/Infimum_and_supremum.
Jongmans D. and Garambois S., 2007, Geophysical investigation of landslides: a review, Bull. Soc. géol. Fr., 178(2), 101–112.
Klir G. J. and Yuan, B., 1995, Fuzzy Sets and Fuzzy Logic Theory and Applications. Prentice Hall, Upper Saddle River, NJ.
Klir G.J. and Folgert. A., 1988, Fuzzy sets, uncertainty and information, Prentice Hall, Englewood, Cliffs, USA, 257.
Mamdani E. H., Assilian S., 1975, An Experiment in Linguistic Synthesis With a Fuzzy Logic Controller, in International Journal of Man Machine Studies, 7, 1–13.
Mathworks, 2009, MatlabR2009a Help Guide, www.mathworks.com.
Nelles O., 2001, Nonlinear Systems Identification, Springer.
Nikravesh M. and Aminzadeh F., 2003, Soft computing for intelligent reservoir characterization and modelling. Developments in Petroleum Science 51, 3–32.
Nikravesh M., Zadeh L.A., Korotkikh V., 2004, Fuzzy Partial Differential Equations and Relational Equations Reservoir Characterization and Modeling, Springer, ISBN: 978-3-642-05789-2 (Print) 978-3-540-39675-8 (Online), https://doi.org/10.1007/978-3-540-39675-8.
PourMohammadBagher L., Kaedi M., Ghasem-Aghaee, N., Ören, TI., 2009, Anger evaluation for fuzzy agents with dynamic personality, Mathematical and Computer Modelling of Dynamical Systems 15 (6), 535–553.
Ross T.J., Hassanein H., Ali A.N., 2010, Fuzzy logic with engineering applications: design and stability analysis. 3rd ed. Chichester (Royaume Uni): Wiley, xxvii, 275 p. ISBN 978-047-0748-510.
Schweizer B. and Sklar A., 1961, Associative functions and statistical triangle inequalities, Publ. Math.Debrecen, 8, 169–186.
Soleimani Kh., and Hajian A., 2017, Fundamentals of fuzzy logic and its applications in Geophysics ,Civil Engineering, Mechanical Engineering and Computer Engineering, Jahad Daneshgahi Isfahan University of Technology Publisher (Published in Persian Language), ISBN: 978-600-8157-24-3.
Takagi T. and Sugeno M., 1985, Fuzzy identification of systems and its applications to modeling and control. IEEE Trans. on Systems, Man, and Cybernetics 15, 116–132.
Wilson W.L., Mylroie, J.E. and Carew J.L., 1995, Caves as a geologic hazard: A quantitative analysis from San Salvador Island, Bahamas. In: Beck, B.F. (Editor), Karst Geohazards: In: Engineering and Environmental Problems in Karst Terrane. Balkema, Rotterdam, 487–495.
Yager R.R., 1988, On ordered weighted averaging aggregation operators in multicriteria decision making, IEEE Transactions on Systems, Man and Cybernetics, 18(1), 183-190, https://doi.org/10.1109/21.87068
Zadeh L.A., 1965, Fuzzy sets. Information and Control 8, 338–353.
Zwick R., Carlstein E. and Budescu D. V., 1987, Measures of similarity among fuzzy concepts: A comparative analysis. International Journal of Approximate Reasoning, 1(2), 221–242, doi: https://doi.org/10.1016/0888-613x (87)90015-6.

Chapter 4
Applications of Fuzzy Logic in Geophysics

4.1 Introduction

As most geophysical data are imprecise they implicitly have a level of uncertainty which makes them a good choice for using fuzzy methods to model and/or interpret them. In this chapter we investigate the use of fuzzy methods in various geophysical disciplines.

4.2 Fuzzy Logic for Classification of Volcanic Activities

The table below (Fig. 4.1) with the criteria for the volcanic explosivity index (VEI) was published in Fisher and Schmincke (1984), (after Newhall and Self 1982), and can be easily transformed into a fuzzy system. In this case the volumes of ejecta, column heights, durations (hours of continuous blast), tropospheric and stratospheric injections can be considered as the input fuzzy sets (whose values can be measured in the field) while the classification of the volcanic activities (Hawaiian, Strombolian, Vulcanian, Plinian and Ultraplinian) can be considered as the output fuzzy sets. Figure 4.1 shows a portion of this table that is relatively easy to rewrite using fuzzy sets. Of course, triangular membership functions are arbitrary, and again their shape and size can be improved using observations, deductions and experiments.

The fuzzy rules connecting inputs to outputs in this system are represented by the graphical, vertical correspondence between sets in the table of Fig. 4.1 and are equivalent to IF-THEN sentences. In this very simple system IF the volume of ejecta, column height, duration, tropospheric and stratospheric injection of a certain eruption are small, negligible, none, etc., THEN the activity is for example Hawaiian. Fuzzy systems can be used to relate causes and effects in extremely complex volcanological systems (that mathematics is unable to handle) using

A. Hajian and P. Styles, *Application of Soft Computing and Intelligent Methods in Geophysics*, Springer Geophysics,
https://doi.org/10.1007/978-3-319-66532-0_4

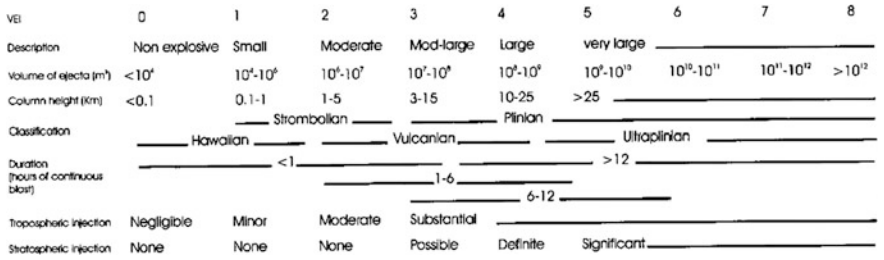


Fig. 4.1 Classical classification of volcanoes activities (Cagnoli 1998)

experiments to assess membership functions, fuzzy rules, fuzzy sets, etc. Then to obtain just a single answer some (defuzzification) technique of the output fuzzy sets is required (Jamshidi et al. 1993).

This table can be rewritten in fuzzy terms. The vertical correspondences between sets represent the fuzzy rules (in Fisher and Schmincke 1984 after Newhall and Self 1982).

The inputs and the output fuzzy sets are obtained from Fig. 4.2. Again the shapes of the membership functions are arbitrary and used only as an example.

4.3 Fuzzy Logic for Integrated Mineral Exploration

An et al. (1991) used the *algebraic sum and the y*-fuzzy operators to integrate geological and geophysical data sets (Fig. 4.3) from the Farley Lake area (Fig. 4.4), digitally using fuzzy memberships. The two exploration targets tested for, pursuing integration and subsequent identification of the favorable areas were "existence of a base metal deposit "or "existence of an iron formation deposit".

The algebraic-sum operator is defined:

$$C = A + B = \{[x, \mu_{A+B}(x)] | \, x \in X\} \tag{4.1}$$

where:

$$\mu_{A+B}(x) = \mu_A(x) + \mu_B(x) - \mu_A(x) \cdot \mu_B(x) \tag{4.2}$$

The algebraic sum may be interpreted as the logic "OR" but not only does it assume full compensation, it is also Increasive, because the membership increases whenever it is combined with a non-zero membership. The γ-operator is defined by Zimmermann and Zysho (1980) as a combination of algebraic product and algebraic sum (Table 4.1).

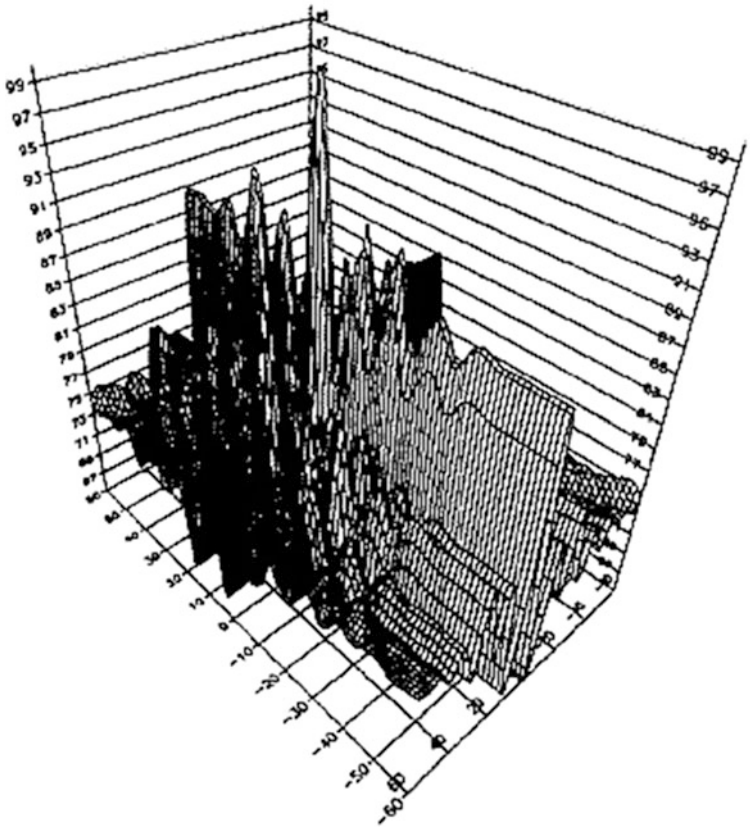


Fig. 4.2 Fuzzy classification of volcanoes activities (Cagnoli 1998)

For example in the aeromagnetic map, an iron ore deposit often produces a prominent magnetic anomaly and this knowledge is useful to scale and rank the information contained in the aeromagnetic map in the integration of mineral exploration. When the exploration target changes from "iron ore deposit" to another different target, the aeromagnetic map information has to be reprocessed. In this way An et al. (1991) separately assign membership functions for iron ore deposit and for base metal deposits for each type of geophysical data. Again for example pixels with a magnetic field anomaly greater than 3000γ are assigned $\mu_I(i, j) = 0.35$ and $\mu_B(i, j) = 0.1$, pixels with an anomaly range of 500–3000γ are assigned $\mu_I(i, j) = 0.2$ and $\mu_B(i, j) = 0.13$, etc. (An et al. 1991).

An et al. (1991) pointed out that determination of the initial fuzzy membership function depends critically on the exploration target and related geological deposit characteristics. Generally, the initial fuzzy function representation also depends heavily on the expertise of the exploration geophysicists and can sometimes be very qualitative.

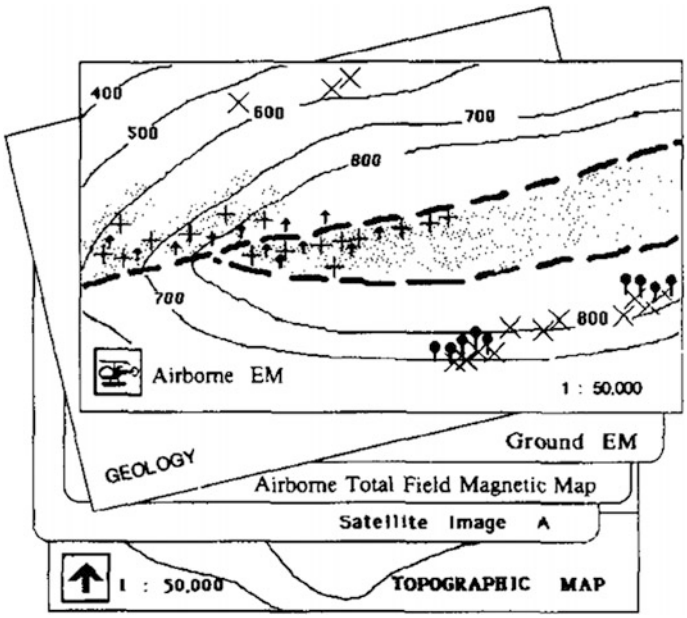


Fig. 4.3 Schematic diagram of spatial information layers (An et al. 1991)

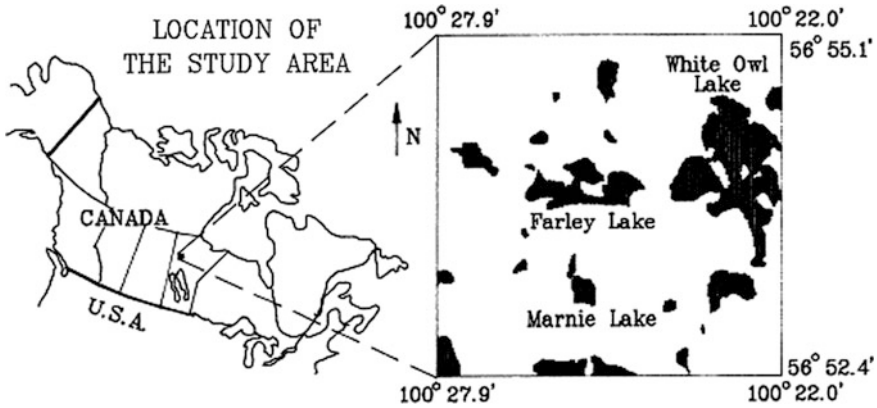


Fig. 4.4 Location of the study area in Farley Lake (An et al. 1991)

The membership function $\mu_A(x)$ of the γ aggregation of fuzzy sets A_1, A_2, …, A_m is defined as:

$$\mu_A(x) = \left(\prod_{i=1}^{m} \mu_i(x)\right)^{1-\gamma} \cdot \left(1 - \prod_{i-1}^{m} (1 - \mu_i(x))^{\gamma} \quad (x \in X, 0 \leq \gamma \leq 1) \right. \tag{4.3}$$

Table 4.1 Table of membership functions for iron ore deposits: $\mu_I(x)$ and membership functions for base metal deposits: $\mu_B(x)$ Redrawn after An et al. (1991)

Aeromagnetic data (ɤ)	$\mu_I(x)$	$\mu_B(x)$
>3000	0.35	0.10
500–3000	0.2	0.13
<500	0.05	0.15
Grand EM data[a]		
>20	0.25	0.23
10–20	0.20	0.18
4–10	0.15	0.14
<4	0.05	0.06
Air borne EM[b]		
B	0.2	0.15
C	0.18	0.13
D	0.15	0.11
E	0.13	0.10
Band	0.10	0.08
No anomaly	0.05	0.05
Ground resistivity data (ohm/m)		
<100	0.30	0.27
100–500	0.25	0.20
500–1000	0.20	0.13
>1000	0.05	0.06
Ip chargeability data[c] (mv/v)		
>40	0.25	0.27
20–40	0.20	0.20
6–20	0.15	0.13
<6	0.05	0.06
VLF EM data[d] (Annapolis)		
>80	0.20	0.20
50-80	0.15	0.15
20-50	0.13	0.10
<20	0.10	0.06
VLF EM data[e] (Seattle)		
>80	0.20	0.20
50–80	0.15	0.15
20–50	0.13	0.10
<20	0.10	0.06
Airborne input EM data[f]		
No anomaly	0.05	0.07
Anomaly area	0.15	0.09
2	0.17	0.11

(continued)

Table 4.1 (continued)

Aeromagnetic data (x)	$\mu_I(x)$	$\mu_B(x)$
3 and 4	0.19	0.13
5 and 6	0.22	0.15
Geological map		
Felsic intrusive	0.05	0.20
Basalt—Andesite	0.18	0.15
Iron rich rocks	0.35	0.10
Picrate	0.20	0.10
Mafic intrusive	0.25	0.10

[a]The survey parameters for the ground EM survey were: operating frequency ≈ 2400 Hz, coil spacing = 300 m and the coil configuration-horizontal loop. The anomaly was estimated as percentage of the ratio Hs/Hp which is equal to $[(\text{in phase/Hp})^2 + (\text{out of phase/Hp})^2]^{1/4}$

[b]The airborne EM anomaly map used in this study was graded A, B, C, … in the original map to represent the relative amplitude ranges. The "Band" represent low and wide band of weak anomalies

[c]Time domain IP with pole-dipole array configuration

[d]VLF H-field (21.4 kHz), station NSS, Annapolis USA

[e]VLF H-field (24.8 kHz), station NLK, Seattle, USA

[f]The INPUT decay curve was sampled six points but the sample interval was not specified. The anomaly# represents real anomaly strength

Different values of γ allow different degrees of compensation. For two or more sets of information, selecting the optimum valve for γ leads to an optimized compensatory process to aggregate subjective information categories.

Suppose over an exploration target one finds a very high possibility in one data set while another data set doesn't show any significance in the context of the exploration target or may even show negative possibility. In these cases the confidence level of estimation by combining the two data sets lies between high confidence and low confidence and the degree of compensation between the two extreme confidence levels depends on the value of γ (An et al. 1991). When $\gamma = 0$ there is no compensation and when $\gamma = 1$ there is full compensation (Figs. 4.5 and 4.6). An et al. (1991) found $\gamma = 0.975$ as the optimized value for fusion of geophysical data. Plot of the Zimmermann operator with $\gamma = 0.975$ using Matlab is illustrated in Fig. 4.7.

```
a=0:0.01:1;
b=[0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1];
gama=1;
for j=1:length(b)
for i=1:length(a)
d=(a(i)*b(j))^(gama)
e=(1-b(j))*(1-a(i));
f=(1-e)^(1-gama);
c(i)=d*f;
end
```

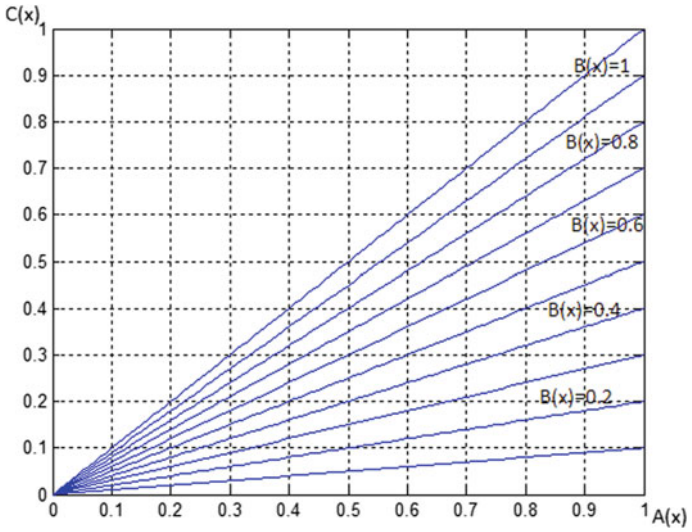


Fig. 4.5 Plot of the Zimmermann operator with $\gamma = 0$ using Matlab. The vertical axis represents the membership functions for the integration of two data sets, A and B with memberships A(x), B (x) respectively, C(x) represents membership of integrated information

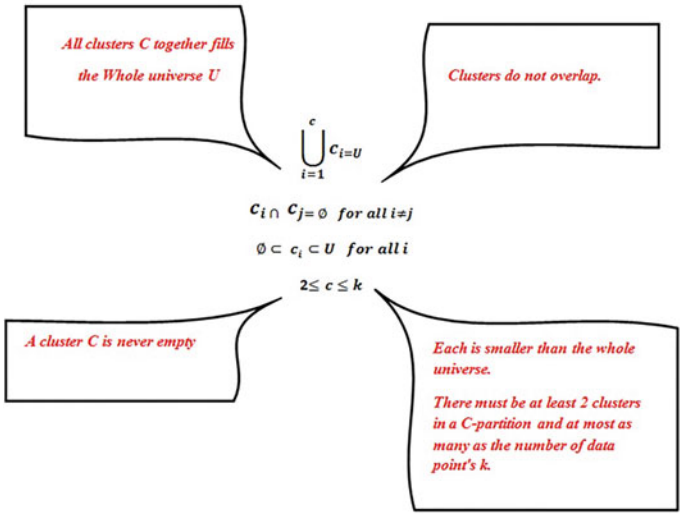


Fig. 4.6 Plot of the Zimmermann operator with $\gamma = 1$ using Matlab. The vertical axis represents the membership functions for the integration of two data sets, A and B with memberships A(x), B (x) respectively, C(x) represents membership of integrated information

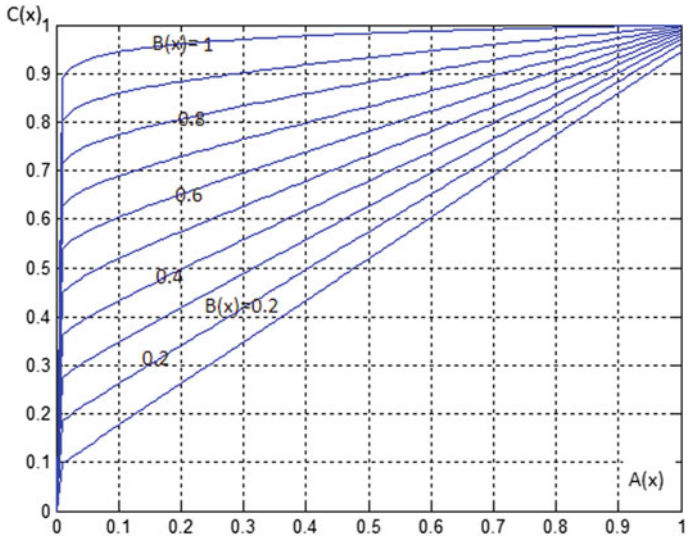


Fig. 4.7 Plot of the Zimmermann operator with $\gamma = 0.975$ using Matlab. The vertical axis represents the membership functions for the integration of two data sets, A and B with memberships A(x), B(x) respectively, C(x) represents membership of integrated information

```
plot(a,c)
hold on
end
grid
Matlab code for Zimmerman Operator with γ = 0.975
a=0:0.01:1;
b = [0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1];
for j=1:length(b)
fori=1:length(a)
d=(a(i)*b(j))^(1-0.975)
e=(1-b(j))*(1-a(i));
f=(1-e)^(0.975);
c(i)=d*f;
end
plot(a,c)
holdon
end
grid
```

The general properties of Max Operator and Algebraic sum are depicted in Table 4.2.

An et al. (1991) used both the algebraic-sum and the γ operator to combine various data sets of geophysical and geological information (Fig. 4.8) over the Farley lake area, Manitoba Canada. They derived possibility distribution maps

Table 4.2 Comparison of max and algebraic-sum operators

Combination method	Results
Max-operator	The lower confidence is usually ignored and the higher one is chosen as the combination of the confidence, as if there does not exist a data set which actually can produce a low or negative possibility
Algebraic-sum	The total confidence level increases regardless of low or negative confidence

using both the afore-mentioned fuzzy operators and successfully outlined favorable areas for two top hypotheses: "base metal deposit" and "iron formation deposits". The results are shown in Figs 4.8, 4.9, 4.10, and 4.11a, to evaluate the fuzzy operator method they used the jackknife estimation approach (for more details read paper An et al. 1991). The comparison of the results indicated that the fuzzy logic approach provides a more effective tool for integrating geological, geochemical and geophysical data sets for resource exploration. Figure 4.11b, c shows an overlay of the final possibility map computed for a base metal deposit and iron formation respectively, both are co-registered over Bands l and 2 of the MEIS-II image (Singh et al. 1989).

Beside the benefits achieved through using fuzzy logic for integrating geophysical data there are some problems in using the fuzzy logic approach to integrate geological and geophysical data as An et al. (1991) noted in their work:

(a) There is no adequate way of representing ignorance.
(b) The Fuzzy logic approach does not allow one to analyze the nature of low possibilities.
(c) There is no standard aggregation operator, although this feature provides some flexibility but more often provides confusion and non-uniqueness to the solution.

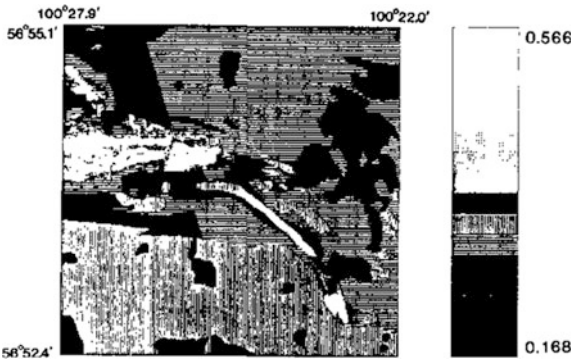


Fig. 4.8 Possibility map for base metal computed using Zimmerman operator with $\gamma = 0.975$ (An et al. 1991)

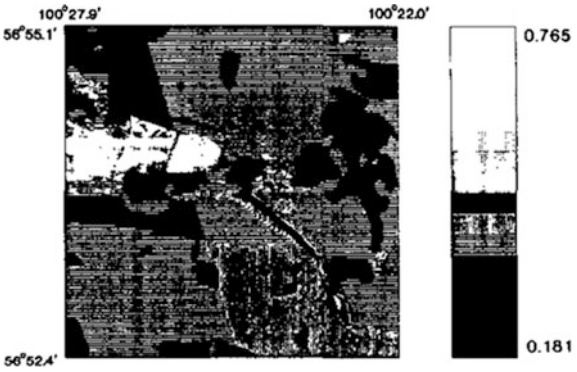


Fig. 4.9 Possibility maps for base metal computed using algebraic-sum operator (An et al. 1991)

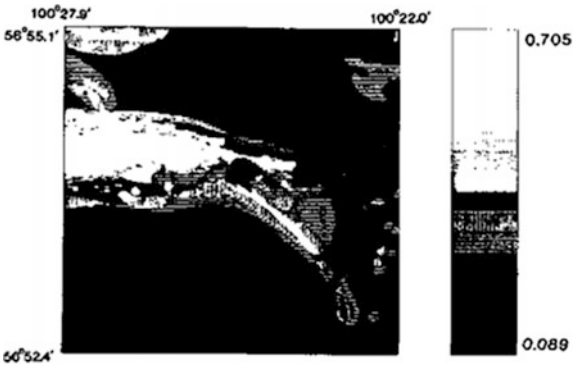


Fig. 4.10 Possibility map for iron formation Zimmerman operator with $\gamma = 0.975$ (An et al. 1991)

4.4 Shape Factors and Depth Estimation of Microgravity Anomalies via Combination of Artificial Neural Networks and Fuzzy Rules Based System (FRBS)

In this section a new method is presented for shape factor estimation of microgravity anomalies (Hajian and Styles 2012). The method is based on training of a designed neural network (NN) with a vast training data set of several objects with different shapes such as: Vertical Cylinder, Sphere and Horizontal Cylinder. The input of the NN is the residual anomaly of the selected principal anomaly profile and the output is the depth and shape factor of the related object. To extract the most probable shape of the object, three main **If-then** fuzzy rules are used and the membership degree of the shape to the fuzzy set of {near to: Vertical Cylinder, Sphere and Horizontal Cylinder} is achieved. The method is tested for synthetic data and also real data measured on an abandoned mine shaft within an open-cut excavation in Kalgoorlie Gold Mine in Australia. The main advantage of the method is its ability to specify how close the gravity target to the estimated shape and depth with no pre-assumptions about its shape.

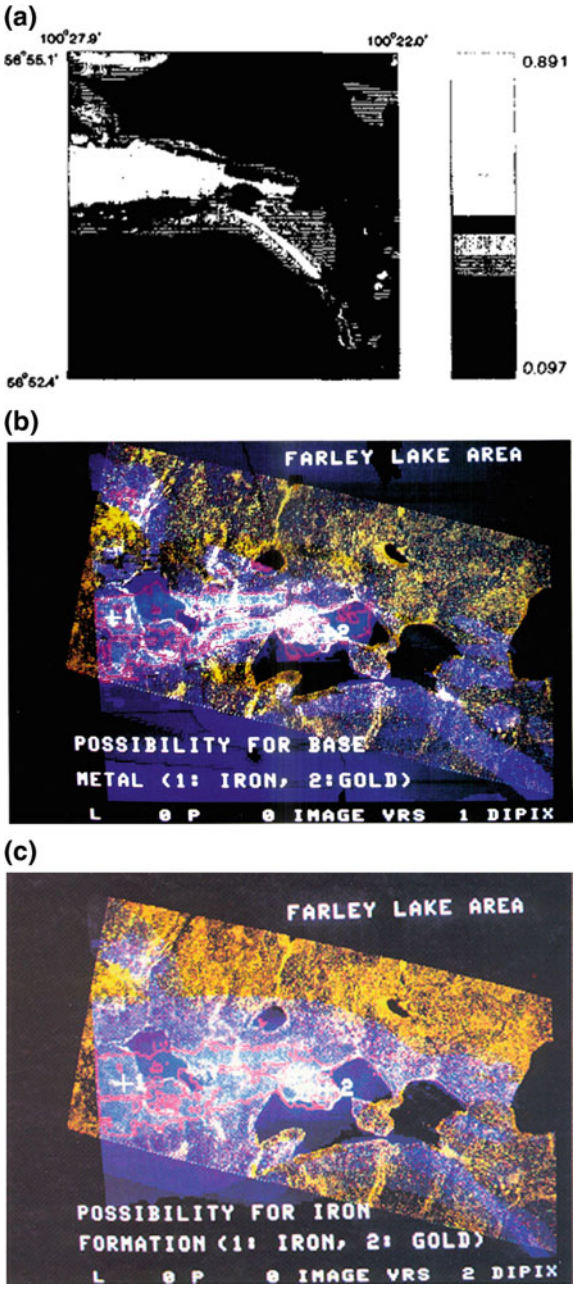


Fig. 4.11 **a** Possibility map for iron formation using algebraic-sum operator. **b** The final possibility map computed using γ-operator for a base metal deposit is displayed over the Bands 1 and 2 of the MEIS-II image. **c** The final possibility map computed using γ-operator for a metal formation (Moon and An 1991)

4.4.1 Introduction

The aim of gravity interpretation is to discover how masses producing a given gravity anomaly are distributed. Although great efforts have been devoted to this topic, it is clear that no ideal theoretical or practical solution to completely solve the problem exists. In applied geophysics, frequently used elementary shapes for cavities, namely sphere, horizontal cylinders and vertical cylinders, are considered accurate enough for representing bodies (Grêt et al. 2000). Several methods have been developed to interpret gravity anomalies assuming simple causative bodies; these methods include depth rules (Smith 1959), Fourier transformation (Odegard and Berg 1965; Sharma and Geldrat 1968), Euler deconvolution (Thompson 1982; Reid et al. 1990), Mellin transforms (Mohan et al. 1986; Shaw and Agarwal 1990), least-squares minimization approaches (Gupta 1983; Abdelrahman et al. 2001), methods of inverting gravity data to determine model parameters (Li and Oldenburg 1998; Li and Chouteau 1998; Boulanger and Chouteau 2001).

Recently, considerable effort has been devoted to the use of soft computing approaches for automatic interpretation of gravity data using artificial neural networks. Elawadi et al. (2001) used back-propagation neural networks for detection of cavities from gravity data. Salem et al. (2003) used a Hopfield neural network for imaging subsurface cavities from microgravity data. Osman et al. (2007) used Forced Neural Networks for forward modeling of gravity anomaly profiles. In all the mentioned methods using neural computation, a pre-assumption about the cavity shape was made and then the neural network was trained with the properties of the supposed shape. We have combined neural networks with fuzzy sets and if-then fuzzy rules to achieve a method where no pre-assumption about the cavity shape is needed and where it is also possible to estimate from the calculated shape factor how close the gravity source is to sphere, horizontal cylinder or vertical cylinder ideal bodies using if-then fuzzy rules.

As an example used in this study, a cavity can be a member of the set

$$C = \{\text{cavity} \mid \text{cavity shape is close to sphere}\} \tag{4.3}$$

A membership degree $\mu c = 0.7$, means that the cavity is not exactly a sphere but is close to a sphere with a membership degree of 0.7.

4.4.2 Extracting Suitable Fuzzy Sets and Fuzzy Rules for Cavities Shape Estimation

The gravity effect of a cavity depends on its size, depth and shape. Abdelrahman (2001) presented the analytical formulation for the gravity anomaly of objects with different shapes as in equation below:

Table 4.3 The values of amplitude factor (A) and shape factor (q) in Eq. 4.4, for different shapes of objects

Shape	q (shape factor) A (amplitude factor)
Sphere	1.5 $\frac{4}{3}\pi G\rho R^3$
Horizontal cylinder	1 $2\pi G\rho R^3$
Vertical cylinder	0.5 $\pi G\rho R^3$

Where R is the radius of the object, G is the universal gravity constant and ρ is the density contrast

$$g(x) = \left[\frac{A}{(x^2 + z^2)^q}\right] \tag{4.4}$$

where Z is the depth of the subsurface object, x is the horizontal distance over the measured point, A is the amplitude factor, which depends on the size and density contrast of the object, and q is the shape factor which depends on the shape of the object. The values of amplitude factor (A) and shape factor (q) are listed in Table 4.3.

As mentioned above and in Table 4.3, q is the shape factor, which describes the source geometry and is equal to 0.5, 1 or 1.5 for a horizontal cylinder, a vertical cylinder and a sphere, respectively. In practice, the value of the shape factor is not exactly equal to the mentioned values because of the nature of the real natural cavities which are almost, but not exactly, near to a sphere or a cylinder (vertical or horizontal), Fig. 4.12.

Fig. 4.12 Practical shapes of natural cavities: **a** Near to vertical cylinder subsurface cavity, **b** near to sphere subsurface cavities (the location is Hosseinabad Jarghoye village, Isfahan, Iran, the reason for the occurrence these cavities is drying of subterranean tunnels used in ancient times for Qanats, Photo by A. Hajian)

After normalizing the gravity data by dividing the residual gravity by its value at $x = 0$ (where the residual gravity has its maximum amplitude), the residual values obtained will only depend on the depth and shape factor of the object. The details are deduced from Eq. 4.5 as follows (Abdelrahman 2001):

$$g_n(x) = \frac{g(x)}{g_o(x)} = \frac{g(x)}{g(x)|_{x=0}} = \left(\frac{z^2}{x^2 + z^2}\right)^q \tag{4.5}$$

where g_n is the normalized residual gravity value. Note that for cavities the density contrasts are negative and both $g(x)$ and $g_o(x)$ are also negative, and consequently the normalized residual gravity will be positive.

In real cases, with regard to the most probable shapes of the cavities, they are classified into three main fuzzy sets:

$\tilde{A}_1$ = {cavity | the cavity shape is near to sphere}
$\tilde{A}_2$ = {cavity | the cavity shape is near to vertical cylinder}
$\tilde{A}_3$ = {cavity | the cavity shape is near to horizontal cylinder}.

Also to extract the shape of a cavity from the calculated shape factor via its gravity effect, three main if-then Fuzzy Rules are considered:

Rule1: If q is near to 1.5 then the cavity shape is near to a sphere
Rule2: If q is near to 1 then the cavity shape is near to a vertical cylinder
Rule3: If q is near to 0.5 then the cavity shape is near to a horizontal cylinder.

Other secondary fuzzy rules can be deduced using the extension principle (see Appendix A). So from the above main fuzzy if-then rules the below extended fuzzy rules can be extracted:

Rule1.1: If q is very near to 1.5 then the cavity shape is very near to sphere.
Rule1.2: If q is to some extent near to 1.5 then the cavity shape is to some extent near to sphere.
Rule4.1: If q is very near to 1 then the cavity shape is very near to vertical cylinder.
Rule4.2: If q is to some extent near to 1 then the cavity shape is to some extent near to vertical cylinder.
Rule3.1: If q is very near to 0.5 then the cavity shape is very near to horizontal cylinder.
Rule3.2: If q is to some extent near to 0.5 then the cavity shape is to some extent near to horizontal cylinder.

The three main fuzzy rules can be rewritten via mathematical sets language as:

If q $\approx$ 1.5 then the cavity $\in \tilde{A}_1$
If q $\approx$ 1 then the cavity $\in \tilde{A}_2$
If q $\approx$ 0.5 then the cavity $\in \tilde{A}_3$.

Where the sign "$\approx$" means near to or approximately equal.

So we need to define fuzzy sets for: q near to 0.5, q near to 1 and q near to 1.5. In general we are required to define fuzzy sets $\tilde{A}_i$ = {real number near to q_0}, where q_0 is equal to 0.5, 1 or 1.5 respectively for i = 1, 2, 3. The boundary for set "real number near to q_0" is pretty ambiguous. There are lots of MFs like: Bell-Shaped, Triangular-shaped, Trapezoidal-shaped, Gaussian or Exponential-shaped, Z-shaped, Pi-shaped, S-shaped, Parabolic-shaped, Sigmoid, etc. We found that the Generalized Bell-shaped, Z-shaped, Pi-shaped, S-shaped and Trapezoidal-shaped Membership Functions are not good enough for our purpose because in this study the only point that the membership degree equals to one is the state that the cavity shape is exactly one of the probable shapes of sphere, horizontal cylinder or vertical cylinder. On the other hand for $\mu_{Ai}(q)$ there is only one point where the MF = 1 so the useful MF's in this way are triangular-shaped, Gaussian shaped and Parabolic-shaped templates. Consequently, the possibility of real number q to be a member of prescribed set can be defined by the following common membership functions:

A. Parabolic-shaped MF:

$$\mu_{Ai}(q) = \frac{1}{1 + k(q - q_0)^2} \text{ where k is a constant positive variable;} \tag{4.6}$$

An example of this type of MF's with q_0 = 1 and k = 100 is illustrated in Fig. 4.13.

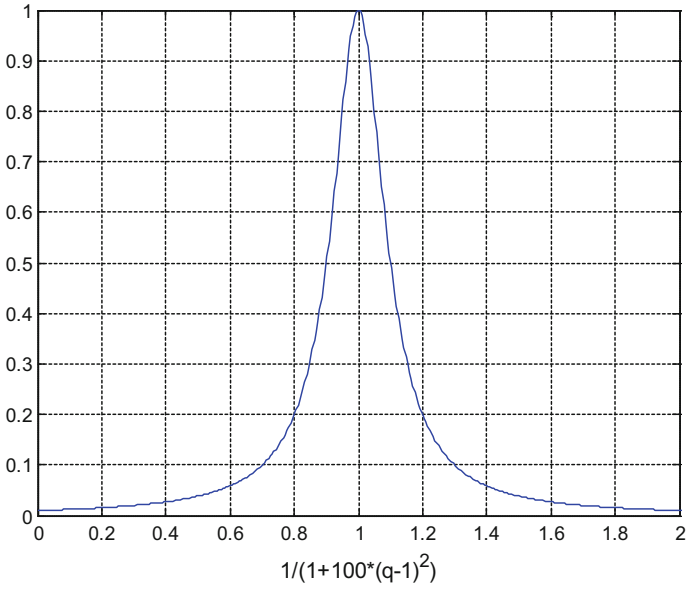


Fig. 4.13 A parabolic-shaped MF with q_0 = 1 and k = 100

B. Triangular-shaped MF:

$$\mu_{Ai}(q) = \begin{cases} 1 - \left|\frac{q-q_0}{d}\right| q_0 & -d < q < q_0 + d \\ 0 & otherwise \end{cases} \text{ where d is a constant variable} \quad (4.7)$$

A triangular-shaped MF for fuzzy set 'real numbers near to 1' with d = 0.2 and q_0 = 1, is presented in Fig. 4.14.

C. Gaussian Membership: The symmetric Gaussian function depends on two parameters q_0 and σ as given by equation below:

$$\mu_{Ai}(q) = e^{\frac{-(q-q0)^2}{2\sigma^2}} e^{-k(q-q_0)^2}; \quad (4.8)$$

where k is a positive constant real number and σ is.

A Gaussian membership function with σ = 0.3 (K = 5.5556), q_0 = 1.5 is depicted in Fig. 4.15. For all the above MFs four characterizations are considered which are in agreement with common sense:

1. For real numbers far from q_0, the membership degree is zero.
2. For q = q_0 the membership degree takes its maximum value which is equal by one.
3. More far from q_0 more the membership degree decreases.
4. For q < q_0 the MF is increasing and for q > q_0 the MF is decreasing.

In order to find the optimum MF for q we tested the mentioned MF's with an algorithm based on a trial and error procedure with different constants for the above

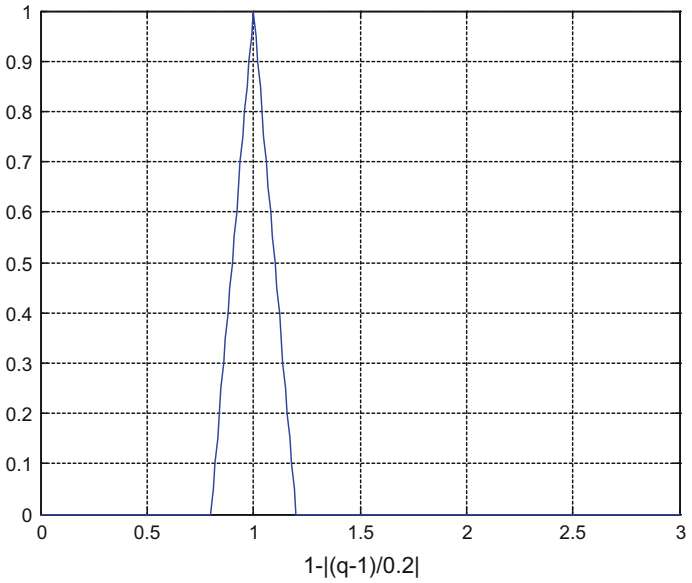


Fig. 4.14 A triangular-shaped MF with q_0 = 1 and d = 0.2

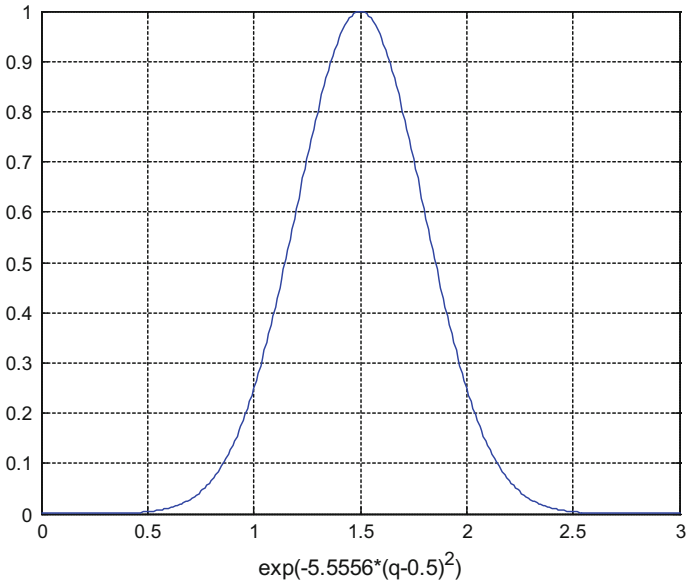


Fig. 4.15 A Gaussian membership function with $\sigma = 0.3$ (K = 5.5556), $q_0 = 1.5$

MF's and finally found that the best MF type for our study is the Gaussian MF with an optimized σ (for each of q_0) listed in Table 4.4.

The problem then is that while we know from Rule1 that if q is near to 1.5 then the cavity shape will be near to a sphere but we can't say exactly how near it is to an exact sphere? On the other hand, an attempt can be made to specify the membership function for set $\tilde{A}_i$, from the calculated q. This is also a mapping from the real number (q) space to the cavity's shape space, so that by knowing the value of q, the MF of the cavity in set $\tilde{A}_i$ Ai is specified properly. To evaluate the MF, we considered the fact that more similar the cavity shape is to an exact sphere, the more similar its gravity effect will be to the residual anomaly of the exact sphere, and consequently the shape factor will be near to the related exact shape. In order to measure the similarity of the object anomaly to the Exact-Shapes (ES) anomaly, ES: sphere, vertical cylinder or horizontal cylinder, its gravity effect is compared with the gravity effect of ES. In this way, three main indexes are considerable:

Table 4.4 Results of FRBS for the selected gravity profile (Hajian and Styles 2012)

Selected gravity profile	Real cavity shape	Real depth	Real shape factor	Estimated depth using FRBS	Estimated shape factor using FRBS	Relative error for depth (%)	Relative error for shape factor (%)
47900 North	Near to vertical cylinder	4 m	0.5	3.5	0.43	14.28	0.14

$$\text{Mean Absolute Error:MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|g_i(x_i, q_0, z) - g_i(x_i, q, z)\right| \tag{4.9}$$

where $g_i(x_i, q, z), g_i(x_i, q_0, z)$ are the gravity effects of cavities with depth z and shape factors q, q_0 on the point with horizontal distance from its center, respectively and n is the total number of gravity points along the selected profile.

$$\text{Correlation Coefficient:R} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left[g_i(x_i, q_0, z) - \bar{g}_i(x_i, q_0, z)\right]\cdot\left[g_i(x_i, q, z) - \bar{g}_i(x_i, q, z)\right]} \tag{4.10}$$

where $\bar{g}_i(x_i, q_0, z)$ is the mean value of $g_i(x_i, q_0, z)$equales to $\frac{1}{n}\sum_{i=1}^{n} g_i(x_i, q_0, z)$ and $\bar{g}_i(x_i, q, z)$ is the mean value of $g_i(x_i, q, z)$ equals to $\frac{1}{n}\sum_{i=1}^{n} g_i(x_i, q, z)$.

Root-Mean Square Error: RMSE

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left[g_i(x_i, q, z) - g_i(x_i, q_0, z)\right]^2} \tag{4.11}$$

In the present study, we found the RMSE condition to be good enough to evaluate the similarity of the anomaly of the Exact-Shapes (ES) and Near-Shapes (NS) cavity. For the ES cavity with depth z_0:

$$g_i(x_i, q, z_0) = g_i(x_i, q_0, z_0): \quad \text{for all } i = 1, 2, \ldots, n, q_0 \in \{0.5, 1, 1.5\} \tag{4.12}$$

In this state from Eq. (4.12) RMSE = 0 but as the cavity is ES the membership degree of μ_{Ai} should be one. Also for the maximum value of the RMSE the membership degree μ_{Ai} should be zero. In our study from prior geological information we understood that the maximum depth for probable cavities was about 15 ms, so the RMSE values for each of fuzzy sets $\tilde{q}$ are calculated for cavities in different depths from Z = 0.5 m to Z = 15 m.

The RMSE values are then normalized for each set, because for the maximum value of RMSE, the similarity of the cavity shape to ES reaches to its minimum value and so:

$$\mu_{Ai} = 1 - (\text{RMS})_n \tag{4.13}$$

$$\text{RMS}_n = \frac{RMSE - RMSE_{\min}}{RMSE_{\max} - RMSE_{\min}} \tag{4.14}$$

As mentioned in last paragraph $\text{RMSE}_{\min} = 0$, so:

$$\text{RMS}_n = \frac{RMSE}{RMSE_{\max}} \tag{4.15}$$

MFs are extracted using Eqs. 4.12 and 4.13.

4.4.3 The Fuzzy Rule Based System (FRBS) for Depth and Shape Estimation with Related Membership Degree

A schematic view of the algorithm used in this research is depicted in Fig. 4.16.

4.4.4 Test of the Fuzzy Rule-Based Model with Real Data

As a test for real data we used microgravity data measured for Kalgoorlie Gold mine located in Western Australia. The gravity anomaly map is shown in Fig. 4.17 with the selected profile highlighted with line of red-stars. The results showed the FRBS accuracy to be useful for 2D gravity interpretation (Fig. 4.18).

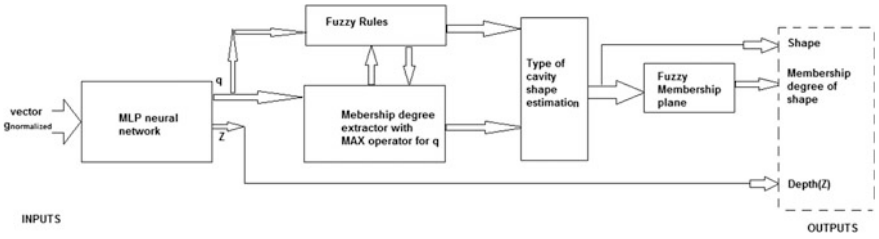


Fig. 4.16 Block diagram of the combination of fuzzy rules and neural network in order to shape and depth estimation of cavities (Hajian and Styles 2012)

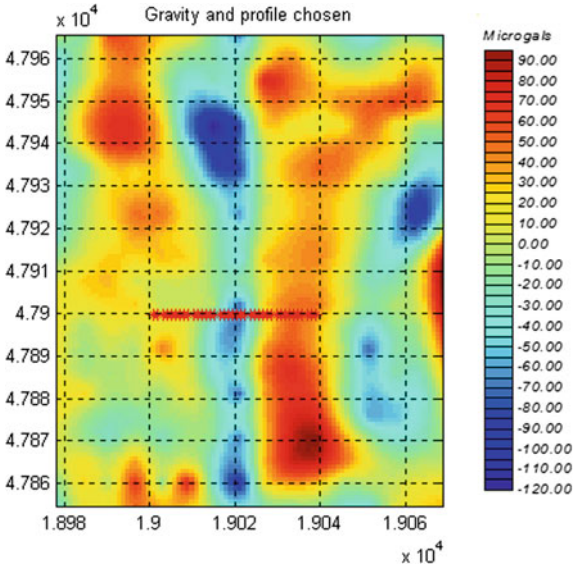


Fig. 4.17 Gravity anomaly map of Kalgoorlie Gold Mine, Western Australia (Hajian and Styles 2012)

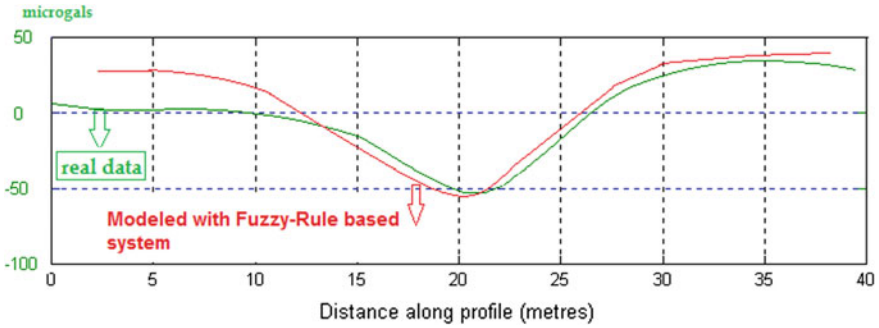


Fig. 4.18 The result of fuzzy-rule based system (FRBS) for the chosen profile (red) in comparison with real data (Hajian and Styles 2012)

4.5 Application of Fuzzy Logic in Remote Sensing: Change Detection Through Fuzzy Sets Using Multi Temporal Landsat Thematic Mapper Data

Madanian et al. (2014) used a fuzzy approach for change detection in remote sensing images using multi-temporal Landsat Thematic mapper data.

They assessed the applicability of using remote sensing and fuzzy sets in detecting changes that have occurred in the Falavarjan area, Isfahan, Iran. The images of the multi-temporal Landsat Thematic Mapper (TM) data on 17 September, 1990 and 13 August, 2010 were used to apply change analysis. Images were radiometrically and geometrically corrected. The root mean square errors were less than 0.5 pixels for each image. Then, the image differencing method and NDVI (Normalized Difference Vegetation Index) image differencing technique were used to produce change images. Fuzzy modeling was implemented to compute the amount of change. The input parameters of the related fuzzy membership function were optimized so that they fitted the shape of the change image histogram. The results were also compared with the classical method of using a thresholding algorithm. The results confirmed the usefulness of fuzzy sets not only in indicating where changes had happened but also indicating the degree of change in the study area. Here we briefly explain how Madanian et al. (2014) used the fuzzy method to estimate the degree of change in remote sensing images.

4.5.1 Introduction

Change detection is a process showing the differences of an object or phenomenon at different times (Singh 1989). Remote sensing provides repetitive data such as Thematic Mapper (TM), which is a very useful tool for change detection processing because of its synoptic view, digital format and cost-effective potential (Nelson et al.

2003; Lu et al. 2004; Chen et al. 2005; Serra et al. 2008). Although different satellite programs are used for change detection analysis, Landsat data are distinctive because of the availability of continual and historical images (Wulder et al. 2008).

A critical element of the numerous change detection procedures is the selection of thresholds to distinguish between change and 'no change' areas (Fung and Ledrew 1988). In order to improve the change detection results, Metternicht (1999) used fuzzy sets and fuzzy membership functions to replace the thresholds. The fuzzy set is a useful tool to have in order to handle classification problems in an ambiguous environment (Tso and Mather 2009). The outcome of a fuzzy classification is a record for every object being analyzed of the degree to which that objects belong to every single class being considered (Fisher 2010).

Madanian et al. (2014) selected the Falavarjan area, located in the western part of Isfahan city which covers about 17550.6 ha, to apply the fuzzy classification method. It is located between 32° 29′–32° 37′N and 51° 20′–51°35′E (Fig. 4.19). Falavarjan city, located in the center of the study area, is on the bank of the Zayandehrud River which has its source in Zardkuh Mountain and flows in eastern Falavarjan. The climate is hot and dry with an average temperature of about 16.4 °C and an average annual rainfall of 162 mm/year. The study area includes agricultural fields, the Zayandehrud River, the ZobAhan Highway, bare lands, rocky outcrops, and urban areas. If water and land are suitably managed, this area has good potential

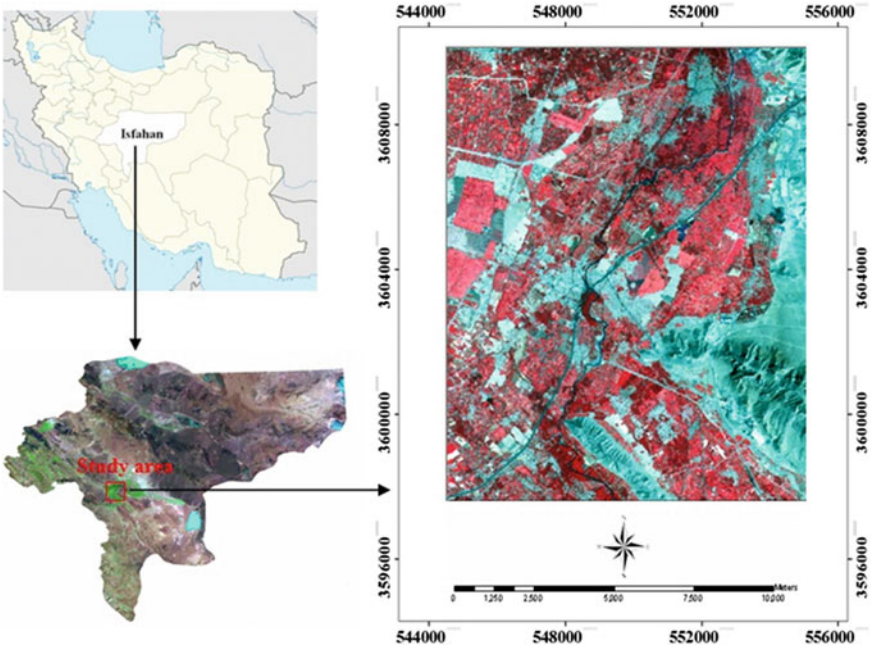


Fig. 4.19 Study area: Landsat TM image of Falavarjan area, collected on 17 September 1990 (right) in the west of Isfahan city (below left) and Iran (above left) (Madanian et al. 2014)

for agriculture expansion. In this study, two Landsat-5 Thematic Mapper (TM) images acquired in September 17, 1990 and August 13, 2010 (path 164, row 37) were utilized to detect changes over a period of 20 years, respectively (Madanian et al. 2014).

In the Madanian et al. (2014) work, digital images were georeferenced to the UTM (Universal Transverse Mercator) projection (zone 39) with a spatial resolution of 28.5 m using approximately 40 ground control points (GCPs) per image. Ground control points were extracted from the digitized map for the earlier image. For the later image, GCPS were acquired in the field. Rectification was carried out using a first order polynomial model and a nearest neighborhood resampling method. The root mean square (RMS) errors were less than 0.5 pixels for each image, which is acceptable. All reflection bands, excluding the thermal band, were used in change detection. The process of change detection using a fuzzy model is depicted in Fig. 4.20.

4.5.1.1 The Change Detection Process

Various techniques are used to identify binary change and non-change information (Lu et al. 2004). This study employed image differencing and NDVI differencing methods to produce change images.

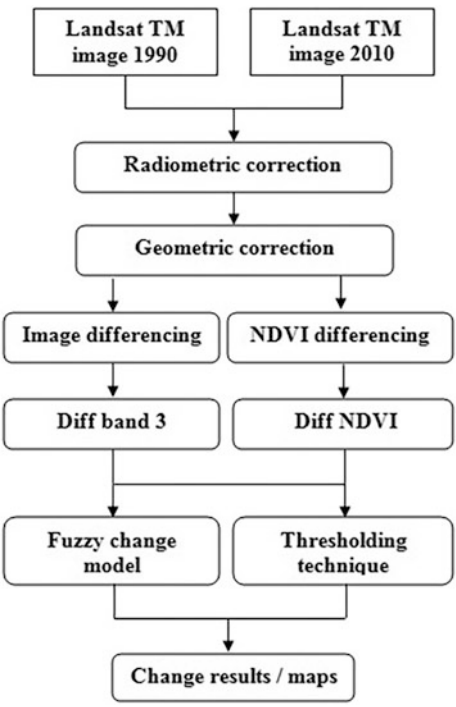


Fig. 4.20 General sequence of change detection process

4.5.1.2 Image Differencing

The image differencing method produces a change image by subtracting two or more datasets. Digital numbers in the resultant difference image are often considered to be normally distributed where pixels with small change are observed around the mean. Pixels which have been changed a great deal are distributed in the tails of histogram (Singh 1989) (Fig. 4.21). The main advantages of this technique are its simplicity and the ease of interpretation of change images. However, selecting the best threshold which distinguishes change from non-change areas is crucial (Lu et al. 2005). In this study, the change image was produced by subtracting the 1990 image band 3 from the 2010 image band 3 pixel-by-pixel.

4.5.1.3 Optimal Threshold Determination

Threshold levels ranging from 0.1 to 3.0 standard deviations from the mean were tested on the change image in order to determine the most suitable threshold values. Consequently, 1σ was identified as the most accurate value among others as determined from the aerial photographs of 1990 and ground data. The change image was then, reclassified into two classes. The value '0' was assigned for 'no change' areas and '1' for change areas.

4.5.1.4 NDVI Differencing Method

Normalized Difference Vegetation Index (NDVI) differencing method is an effective and a very popular method in change detection (Lu et al. 2004). For this study, the NDVI image was derived using Eq. (4.16) and the NDVI difference image (ΔNDVI) was created using Eq. (4.17).

$$NDVI = (NIR - R)/(NIR + R) \tag{4.16}$$

$$\Delta NDVI = NDVI_{pre} - NDVI_{post} \tag{4.17}$$

where NIR and R represent the digital values of the near infrared and red channel images, respectively. Digital numbers of ΔNDVI were normally distributed. The threshold value 1.2σ was applied as an optimal threshold.

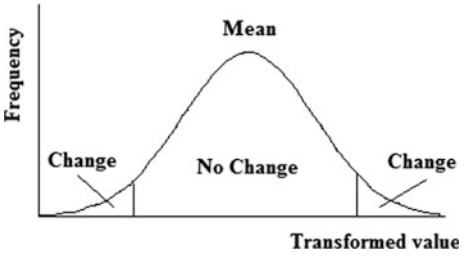


Fig. 4.21 Threshold application (Singh 1989)

4.5.1.5 Fuzzy Application to Change Detection

Having produced the change image, a fuzzy model was applied to compute the degree of change which has happened within the Falavarjan area. The methodological approach includes the following steps (Metternicht 1999; Madanian et al. 2014):

1. As the first step, a fuzzy membership function which has been adopted for the change images histogram was defined. A bell-shaped membership function proposed by Dombi (1990) was used in the Madanian et al. (2014) study. This membership function is divided into two parts: a monotonically increasing part and a monotonically decreasing part. The monotonically increasing function is:

$$\mu A(x) = \left[(1-v)^{\lambda-1}(x-a)^{\lambda}\right] / \left[(1-v)^{\lambda-1}(x-a)^{\lambda} + v^{\lambda-1}(b-x)^{\lambda}\right]; \quad x \in [a,b] \tag{2.18}$$

   and the monotonically decreasing function is:

$$\mu A(x) = \left[(1-v)^{\lambda-1}(c-x)^{\lambda}\right] / \left[(1-v)^{\lambda-1}(c-x)^{\lambda} + v^{\lambda-1}(x-b)^{\lambda}\right]; \quad x \in [b,c] \tag{4.19}$$

   where λ is sharpness, v determines the inflection point of function, a and c are the typical points of the function with a membership degree of zero to the fuzzy set considered, and b represents the standard point of the variables at the central concept, which is a grade of membership equal to 1 (Dombi 1990).
2. To determine the form of the membership function that fits the shape of the change images histogram, the two parameters, sharpness and inflection, were manipulated. In fact, by varying the input parameters, the resulting membership function is changed. Therefore, if the resulting membership functions fit the shape of change image, discrimination of change and 'no change' areas will be performed with a high accuracy.
3. In order to describe changes, linguistic constructs were used. In this case, possibility of change, ranked within the range of 0–1, was expressed using some linguistics terms.
4. Finally, the change image was visualized in a gray scale ranging from white to black. In this way, pixels with an absolute certainty of 'no change' were indicated by white while black represents areas where changes have occurred with absolute certainty.

4.5.1.6 Image Differencing and NDVI Differencing Methods Applied to Case Study

For change detection analysis, it is critical to choose appropriate image bands (Lu et al. 2005). In this study, band 3 and 4 were chosen for two reasons: (1) the

dominant land-cover in the study site is vegetative cover and band 3 and 4 have the potential to indicate changes that have occurred in this class. Ridd and Liu (1998) used different techniques including image differencing for urban land-use change detection in the Salt Lake Valley area using Landsat TM data. They concluded that band TM 3 differencing was the best method (Lu et al. 2004). Similarly, Hame (1986) and Fung (1990) concluded that the differencing image produced by TM 3 was the most accurate one in detecting vegetation change. Pilon et al. (1988) concluded that visible red band data provided the most accurate identification of spectral change for their semi-arid study area of north-western Nigeria in sub-Sahelian Africa. Jensen and Toll (1982) found the usefulness of visible red band data in change detection analysis in both vegetated and urban environments; (2) Digital numbers of the difference images were normally distributed.

The error matrix was used to analyze the accuracy of change images classified using thresholds. Ground truth data and aerial photographs were applied to generate the error matrix. In order to estimate the overall accuracy, the total correct pixels were divided by the total number of pixels. The image differencing using band 3 yielded more accurate results with the kappa coefficient and an overall accuracy of 69 and 86.2%, respectively. The resultant image demonstrated that 74.3% of the study area has remained unchanged and 27.7% of the area has undergone significant change (Table 4.5). The NDVI differencing method resulted in a smaller kappa coefficient and an overall accuracy of 36.4% and 74.9%, respectively. The results of change detection by image differencing with band 3 and the NDVI image differencing using threshold technique are shown in Fig 4.22a, b, respectively.

4.5.1.7 Fuzzy Applications Applied to Case Study

Diff3 and Diff NDVI were used for this fuzzy application. The next step was applying a membership function, which fitted the shape of the change image histograms. Figure 4.23 shows the optimized membership functions for change images. In order to fit the shape of the change images to Diff3 and Diff NDVI histograms, two parameters, sharpness and inflection were modified as the optimized values were selected. In this way, the mean is considered as a standard point that represents pixels of 'no change' with a membership degree of 1. Typical points are the tail values of the histogram that signify change pixels. The membership degree of 0 is assigned to these change pixels. Table 4.6 shows the optimized values for the membership functions.

Table 4.5 The number of the change/no-change pixels and the area of each class detected by the image differencing method with band 3 and the NDVI image differencing

Class	Image differencing with band 3		NDVI differencing	
	Pixels	Area (ha)	Pixels	Area (ha)
Change	62,104	4868.95	53,090	4164.26
No-change	161,756	12681.67	170,770	13388.36

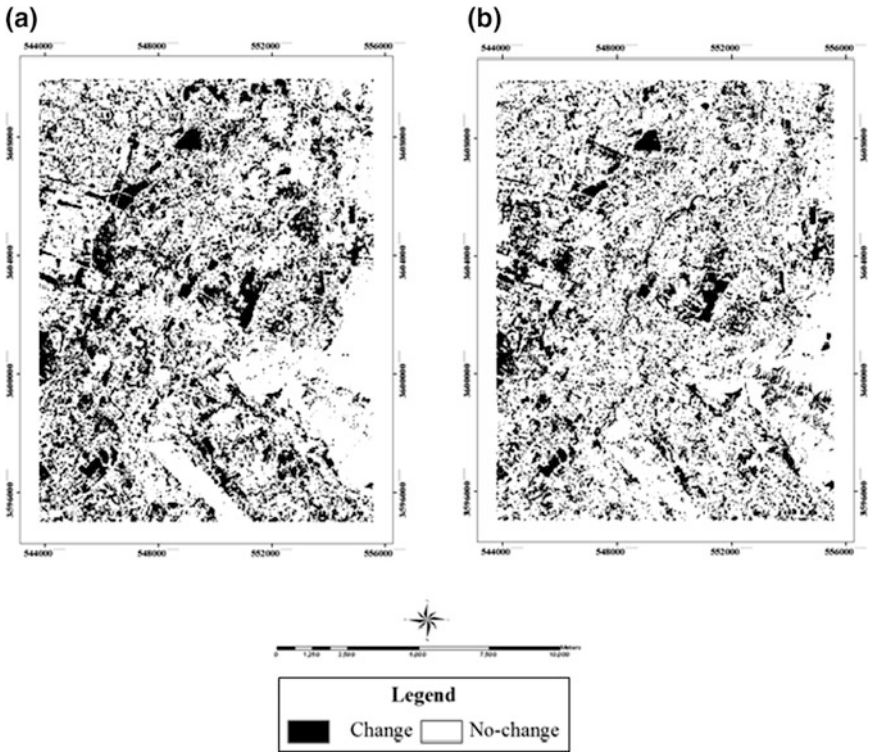


Fig. 4.22 The resultant images obtained by the thresholding technique with the **a** Diff3, **b** Diff NDVI (Madanian et al. 2014)

Then, linguistic constructs were applied to describe the changes that were expressed by fuzzy quantifiers. In this way, each class was described by a special linguistic term. Next, the area of each class was calculated using the number of pixels. Tables 4.7 and 4.8 show the results of the area calculations.

As the final step, sliced change images were visualized using a gray scale. In such a way, a spectrum of white shows pixels which have never changed while black depicts areas that have definitely undergone changes. Figure 4.24a, b represent this concept.

The process of accuracy assessment was based on visual assessment. The accuracy of the fuzzy change image was assessed using ground truth data. It was observed that there was an acceptable agreement between the real and the fuzzy change images.

The results presented in Table 4.8 indicate that 55% of the area is labeled as 'no change'. This class mainly covers eastern and southern parts of the study area, which are rocky outcrops where no changes had occurred during 1990–2010. The ZobAhan Highway is another part of the Falavarjan area that has been remained unchanged. 10.9% of the area is categorized as very likely to extremely likely to

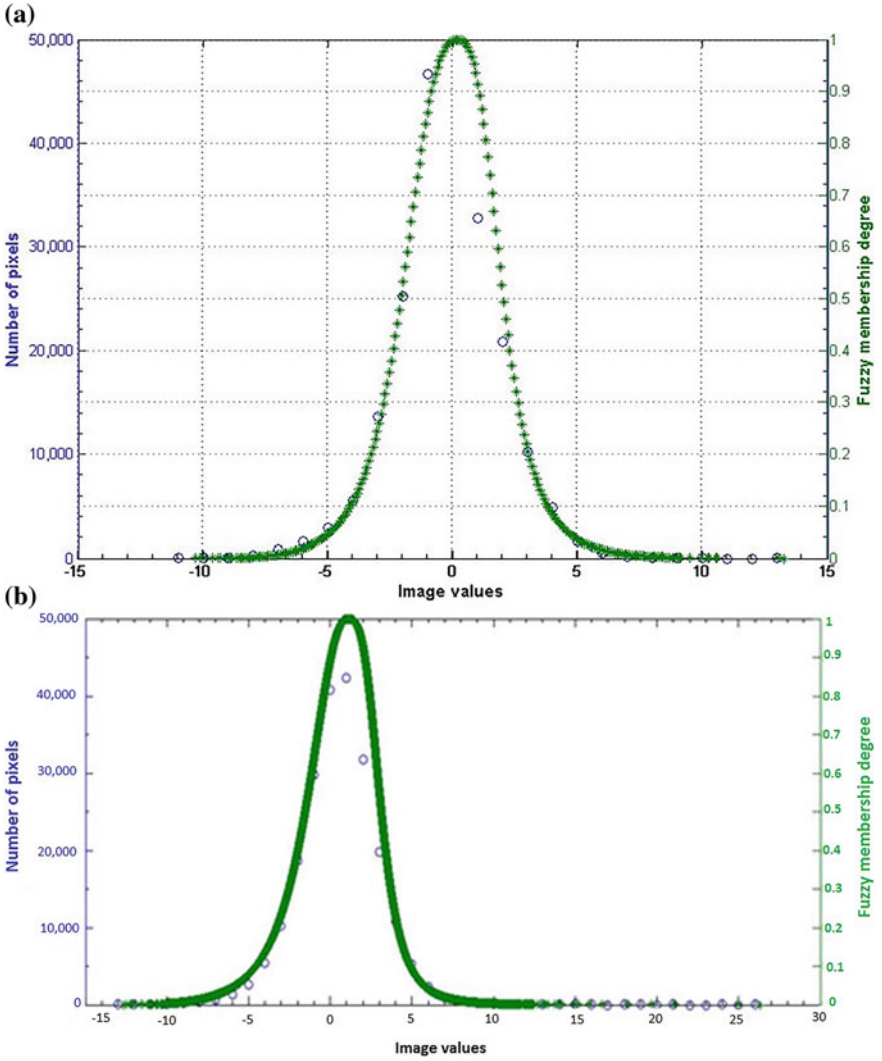


Fig. 4.23 Optimized membership function for the Falavarjan change images, where μ is the membership degree to set of 'no change'. **a** Image differencing with band 3, **b** NDVI image differencing

have changed. 6.2% of the study area has absolutely changed. This part of the area includes bare lands and agricultural areas. These changes are also observable around the urban area, which has expanded since 1990. 27.6% of the area is ranked as likely to very unlikely to have changed. The results presented in Table 4.9 represents noticeable similarities to Table 4.10. It shows that the areas of change and no-change classes estimated by applying the fuzzy method with Diff NDVI are 49.6 and 6.02%, respectively. These values are near to the values derived from

Table 4.6 Values for parameters of membership functions of the change images

Change image	Sharpness (λ)		Inflection (ν)		Standard point	Typical points
–	MI[a]	MD[b]	MI[a]	MD[b]	–	
Diff3	4.4	4.6	0.9	0.95	0.24	−10.23 and 13.3
Diff NDVI	2	3.4	0.95	0.96	1	−14.64 and 26.3

[a]*MI* Monotonically increasing part of the function. [b]*MD* Monotonically decreasing part of the function

Table 4.7 Results of area estimation using fuzzy technique via image differencing method

Code	Fuzzy linguistic terms	Degree of membership	Number of pixels	Area (ha)	percentage
1	No changes	0.81 to 1.00	123881	9714.3	55.3
2	Very unlikely changes	0.61 to 0.80	27671	2169.4	14.4
3	Unlikely changes	0.51 to 0.60	12714	996.8	5.7
4	Neither nor	0.41 to 0.50	10805	847.1	4.8
5	Likely changes	0.31 to 0.40	10446	818.9	4.7
6	Very likely changes	0.21 to 0.30	12051	944.8	5.4
7	Extremely likely changes	0.11 to 0.20	12412	973.1	5.5
8	Changes	0.00 to 0.10	13880	1088.2	6.2

[a]Area (ha) = 0.0784 ha × number of pixels, as the TM image has a 28.5 m resolution

Table 4.8 Results of area estimation using fuzzy technique via NDVI differencing method

Code	Fuzzy linguistic terms	Degree of membership	Number of pixels	Area (ha)	Percentage
1	No changes	0.81–1.00	111,189	8717.22	49.67
2	Very unlikely changes	0.61–0.80	34,114	2674.54	15.24
3	Unlikely changes	0.51–0.60	13,983	1096.27	6.25
4	Neither nor	0.41–0.50	12,937	1014.26	5.78
5	Likely changes	0.31–0.40	12,483	978.66	5.58
6	Very likely changes	0.21–0.30	12,549	983.83	5.6
7	Extremely likely changes	0.11–0.20	13,134	1029.70	5.86
8	Changes	0.00–0.10	13,471	1056.12	6.02

[a]Area (ha) = 0.0784 ha × number of pixels, as the TM image has a 28.5 m resolution

applying a fuzzy model with Diff3. The accuracy of the changed map generated from the applying thresholding technique on Diff NDVI was less than the change image accuracy of image differencing. But the maps produced through applying the fuzzy method with Diff3 and Diff NDVI are very similar to each other. This means that the fuzzy method is more effective and accurate in identifying change/no-change areas than the thresholding technique.

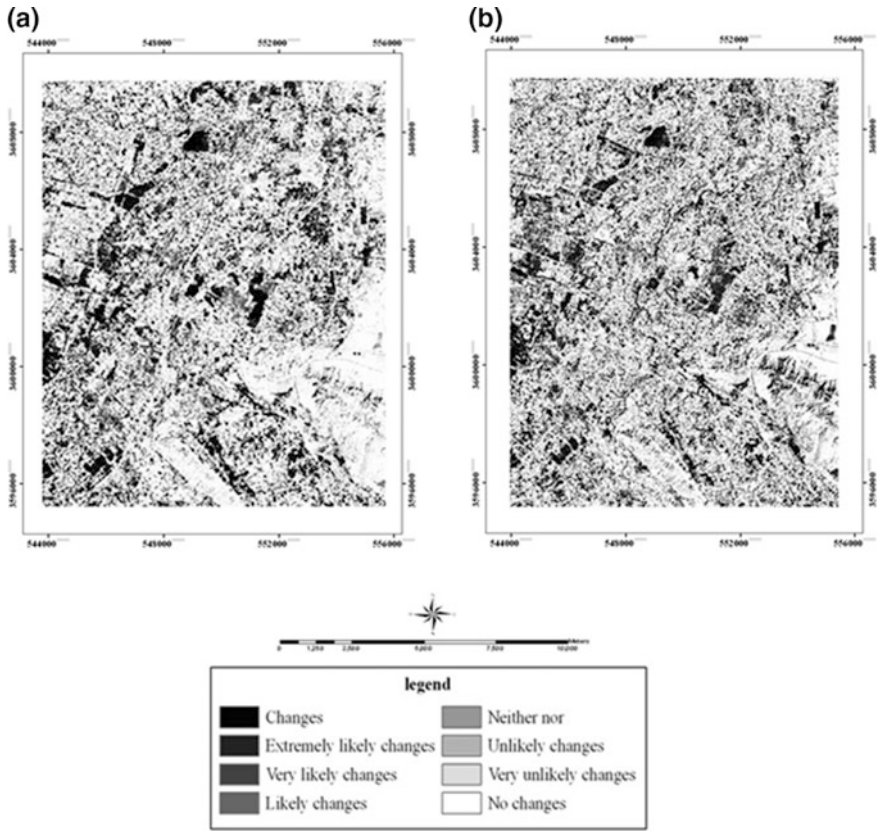


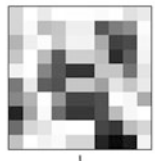
Fig. 4.24 The resultant images using the fuzzy method with the **a** Diff3, **b** Diff NDVI (Madanian et al. 2014)

As a final comparison, the area of 'no change' class in the image classified using threshold values was more than the area of the same class in the fuzzy technique. In the image differencing method using band 3, 74.3% of the study area was classified as 'no change' but this value was 55% in the fuzzy technique. It shows that the fuzzy change model can identify a continuum of changes. As the fuzzy logic method based on inaccuracy modeling with a membership function, it not only improves the ability of the results to distinguish change and 'no change' areas but also it is able to help the interpreter to know the degree of change in areas of change. Each pixel has been labeled with code '0' ('no change' areas) or '1' (change areas) in both techniques (Fig. 4.25). In order to make the interpretation of change images produced by two methods easier, the membership degree of the pixel to the set of 'change' is represented so that the more the membership degree changes, the more the area has changed.

(a)

Col/row	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
2	0	0	0	0	0	0	0	1	1	0
3	0	0	0	1	0	0	0	1	1	0
4	0	0	0	1	1	1	0	0	0	0
5	1	0	1	1	0	0	0	0	0	0
6	1	0	0	0	1	1	1	0	0	0
7	1	0	0	1	1	1	1	0	0	0
8	1	0	0	0	0	0	1	1	0	0
9	0	0	0	0	0	0	1	1	1	0

(b)



Col/row	0	1	2	3	4	5	6	7	8	9
0	0.17	0.01	0.09	0.01	0.01	0.01	0.03	0.01	0.22	0.09
1	0.17	0.03	0.27	0.23	0.07	0.07	0.66	0.51	0.44	0.02
2	0.30	0.07	0.02	0.21	0.05	0.05	0.34	0.75	0.71	0.04
3	0.23	0.01	0.23	0.64	0.19	0.19	0.27	0.83	0.63	0.05
4	0.06	0.01	0.47	0.66	0.80	0.80	0.23	0.23	0.54	0.11
5	0.53	0.09	0.64	0.68	0.23	0.23	0.37	0.11	0.23	0.09
6	0.47	0.30	0.01	0.41	0.73	0.73	0.68	0.01	0.04	0.34
7	0.86	0.36	0.03	0.58	0.78	0.78	0.68	0.27	0.04	0.01
8	0.58	0.17	0.11	0.02	0.01	0.01	0.73	0.78	0.27	0.17
9	0.24	0.01	0.05	0.03	0.17	0.17	0.86	0.96	0.88	0.05

Fig. 4.25 Part of the change image matrix achieved through **a** classical statistical method **b** fuzzy method. In the thresholding technique, the code '0' was assigned for 'no change' areas and '1' for changed areas. In the fuzzy method, the degree of change (μ) has been calculated for each pixel (Madanian et al. 2014)

4.6 Fuzzy Transitive Closure Algorithm for the Analysis of Geomagnetic Field Data

Sirdharan (2009) introduced a fuzzy mathematical model for the analysis of geomagnetic field data using the geomagnetic field variations measured during magnetic storms. The data set was collected from the Indian network of magnetometers. The patterns of variation differ at the non-equatorial, equatorial latitudes and the observatory situated nearer to the geomagnetic S_q focus. The fuzzy method that Sirdharan (2009) used was the "fuzzy transitive closure" analysis, which is a powerful technique among various recognition methods. Here, we first describe the principles of fuzzy-based clustering and then explain how Sirdharan (2009) used this approach for pattern recognition of geomagnetic storms.

4.6.1 Classical and Fuzzy Clustering

A number of similar individuals that occur together are like two or more consecutive constants or vowels in a segment of speech, a group of houses or an aggregation of stars or galaxies that appear close together in the sky and are gravitationally associated (Jantzen 2004). Cluster analysis is a statistical technique for discovering whether the individuals of a population fall into different groups by making quantitative comparisons of multiple characteristics thereby producing concise representation of a system's behavior.

Generally for a given set X, clustering in X means to find several cluster centers (at least two) which can properly characterize the relevant classes of X.

The classification is based on the feature(s) of the population and is the main basis of distinguish whether a member belong to the assumed group.

In classical clustering the classes are partitions of X and all the clusters together fills the whole universal U. Also the clusters don't have any overlaps. Cluster (C) is never empty and it is smaller than the whole universe U. Furthermore, there must be at least 2 and at most k clusters. Their important properties are shown in Fig. 4.26.

The classical clustering method partitions the universe such that any element only belongs only to one of the classified groups, on the other hand the membership degree of an element is '1' for a group and zero for others, but in the fuzzy clustering approach an element can belong to two or more groups and this property is very useful in many applications specially when the data are chaotic in nature such as geomagnetic storm time data. So in these cases it is better to replace the classical clustering by fuzzy partitioning or fuzzy clustering. Using fuzzy clustering, the compactness and separation validity are more accurate and can produce a better segmentation result while unknown parameters between one class of data and the other classes can be estimated by a fuzzy model (Sirdharan 2009).

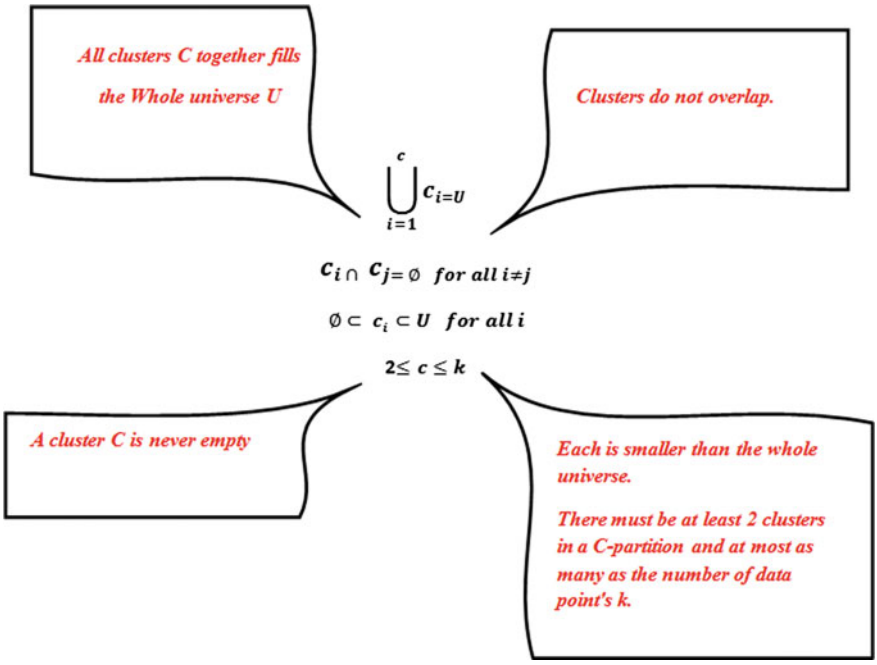


Fig. 4.26 Main properties of a classical clustering

In classical clustering the membership of point 'k' to cluster 'i' is:

$$m_{ik} = \begin{cases} 1 & if \parallel u_k - c_i \parallel^2 \leq \parallel u_k - c_y \parallel^2 \\ 0 & otherwise \end{cases} \tag{4.20}$$

where c_i is the cluster center I, u_k is data point k, c_j is the cluster center j, and $\parallel u_k - c_i \parallel$ is the distance between point k and cluster center i.

The object function j is:

$$J = \sum_{i=1}^{c} J_i = \sum_{i=1}^{c} \left(\sum_{k, U_k \in c_i} \parallel u_k - c_i \parallel^2 \right) \tag{4.21}$$

The optimized clustering is that which minimizes the total sum of all distances.

In the fuzzy clustering method the degree of association is strong for data and weak for data in different classes. The distance of an input sample to the cluster to which the input sample belongs is used as a criterion to measure the cluster compactness. To apply the fuzzy approach to a set classical of data three main steps are used:

1. Fuzzification: Conversion of numeric data to fuzzy numbers.
2. Aggregation: Computation of fuzzy numbers in the fuzzy domain.
3. Defuzzification: Conversion of the obtained fuzzy numbers into numeric data.

To subjectively estimate the resemblance between pairs, a numerical proximity matrix is defined such that each of its elements [i, j] is the score of the proximity relation (subjective similarity) between data points i and j. The numerical values in the proximity matrix are only a quantitatively descriptive number and their significance can't be evaluated through conventional statistical methods and thus are determined subjectively. The proximity relation is not necessarily transitive, hence the theory of inexact matrices was utilized by Sridharan (2009) to formulate a transitive closure structure that provides the possibility of separating the data set into mutually exclusive cluster which are, in essence, equivalent in class. In all classical clustering methods, proximity relations defined by arbitrary similarity measure are not necessarily similarity relations.

4.6.2 *Fuzzy Transitive Closure Method*

In the method of fuzzy transitive closure, cluster variables are classified based on the Minkowski formula. This technique requires accuracy and precision. An α-cut ($\alpha \in (0, 1]$) of a fuzzy relation R is defined as the binary relation R_α. As α runs through (0, 1], α-cuts of R form a nested sequence of crisp relations such that when $\alpha_1 \geq \alpha_2$ then $R\alpha_1$ is a subset of $R\alpha_2 (R\alpha_1 \leq R\alpha_2)$. The scalar '$\alpha$' is an indication of the validity of each clustering ($0 \leq \alpha \leq 1$).

One of the common fuzzy clustering methods is fuzzy c-means clustering which has been applied for various atmospheric and geophysical studies (e.g. Dekkerse et al. 1994; Kruiver et al. 1999; Hajian et al. 2016). This method needs the desired number of clusters to be specificated which is a disadvantage when the clustering problem does not specify any desired number of clusters. Furthermore when the population is known to contain k groups, the sampling method may be such that the data from the rarest group do not appear in the sample, therefore forcing the data into k groups may lead to nonsensical clusters. In these types of problems the structure of the given data should be reflected through the number of clusters, in a natural way. On the other hand, no pre-assumption about the number of clusters is required and this will be determined through applying the clustering technique depending on the structure of data. The fuzzy transitive closure works based on this fact.

The fuzzy transitive closure method is based on three fundamentals:

- Fuzzy equivalence relations
- Minkowski classifier
- α-cut.

4.6.3 *Fuzzy Equivalence Relations*

A Fuzzy equivalence relation as defined in Chap. 3, is a relation defined on a set which is reflexive $R(a_i, a) = 1$, symmetric, $R(a_i, b_i) = R(b_i, a_i)$ and maximum-minimum transitive. Similar to an ordinary equivalence relation, a fuzzy equivalence relation induces a partition in each of its α-cuts (Zadeh 1965).

A meaningful fuzzy equivalence relation is defined on the transitive closure of the fuzzy compatibility (Anderberg 1973). A fuzzy compatibility relation R on a set S consisting of n data items can be defined by an appropriate distance function (Klir and Folger 2000) of Minkowski class by:

$$R(x_i, x_k) = 1 - \delta \left[\sum_{j=1}^{n} (x_{ij} - x_{kj})^q \right]^{1/q} \tag{4.22}$$

For all: $(x_i, x_k) \in S$.

Where q is a positive real number and δ is a constant that ensures that $R(x_i, x_k) \in [0, 1]$. The value of δ is the inverse value of the longest distance in S.

As α-cut of a fuzzy set was defined in Sect. 4.4.4, α-cut of a fuzzy set "I" is a crisp set I_α:

$$I_\alpha = \{x \in X | \mu_{I(x)} \geq \alpha\} \tag{4.23}$$

4.6.4 Fuzzy Transitive Closure Algorithm

Let R^1 be the square matrix of order k obtained from the given data matrix derived by the Minkowski class. The relational matrix $R^{(2)} = R^1$ o R^1 where an element of R^1 o R^1 is the maximum-minimum (k is the number of observations) and x_{rs} is an element in the rth row and sth column of the matrix $R^{(2)}$ similarly:

$$\begin{aligned} R^{(4)} &= R^{(2)} \text{o } R^{(2)} \\ R^{(2k)} &= R^{(2k-1)} \end{aligned} \tag{4.24}$$

This procedure is continued until no new relationship is produced. Thus the max-min transitive closure R is the relation R^{n-k} which is denoted by R_τ.

This relation R_τ induces a partition called an α-cut (Zadeh 1965). In different interval S is the distance matrix R defined as:

$$\text{Distant} = \left[(x_{11}-x_{12})^2 + (x_{21}-x_{22})^2 + \cdots + (x_{n1}-x_{n2})^2\right]^{1/2} \tag{4.25}$$

As we mentioned in Sect. 3.4.7.5 this kind of distance is called the Euclidean distance. The operator 'O' is applied to define the maximum of the minimum values obtained from the corresponding rows and columns of a particular element in the matrix R^1, R^2, R^4, etc.

4.6.5 Application to for Geomagnetic Storm Data

Sridharan (2009) used the simultaneous data of geomagnetic storm time ranges of the horizontal component (H) of observatory stations: Alibag, Hyderabad, Kodaikanal, Nagpur, Pondicherry, Sabhawala, and Tirunelveli for the period between 2001 and 2003 to apply the fuzzy clustering technique through the Fuzzy Transitive Closure Algorithm (FTCA). The total range from maximum to minimum values of the recorded magnetogram observations were considered for this method. The station names and related dipole coordinates are listed in Table 4.9. The location map of the observatories is depicted in Fig. 4.27.

Sridharan (2009) used the data taken from the data book published by the Indian Institute of geomagnetism and the values are shown in Table 4.10 for each of the stations.

To apply the FTCA method, it is first necessary to calculate the distance matrix (R) through Eq. 4.22 as follows (Sridharan 2009);

Table 4.9 Location of observatories and the related dipole coordinates Redrawn after Sridharan (2009)

Serial No.	Station	Dipole
1	Sabhawala (SAB)	21.2°N 151.9°
2	Nagpur (NGP)	11.96°N 154.1°
3	Alibag (ABG)	10.2°*N* 145.9°
4	Hyderabad (HYB)	8.3°N 151.3°
5	Pondicherry (PON)	4.7°N 154.1°
6	Kodaikanal (KOD)	1.23°N 149.6°
7	Tirunelveli (TIR)	0.33°S 149.76°

$$\mathrm{R} = \begin{matrix} ABG \\ HYB \\ KOD \\ NGP \\ PON \\ SAB \\ TIR \end{matrix} \begin{pmatrix} ABG & HYB & KOD & NGP & PON & SAB & TIR \\ 0 & 109.86 & 371.87 & 84.469 & 134.97 & 289.02 & 431 \\ 109.86 & 0 & 379.31 & 117.32 & 148.31 & 275.73 & 435.85 \\ 371.87 & 379.31 & 0 & 347.89 & 304.89 & 461.38 & 264.46 \\ 84.496 & 117.32 & 347.89 & 0 & 130.92 & 284.01 & 401.09 \\ 134.97 & 148.31 & 304.89 & 130.92 & 0 & 319.18 & 394.08 \\ 289.02 & 275.73 & 461.38 & 284.01 & 319.18 & 0 & 536.07 \\ 431 & 435.85 & 264.46 & 401.09 & 334.08 & 536.07 & 0 \end{pmatrix}$$

The matrix R elements are calculated using the Euclidean distance formula, for example the element R(ABG, HYB) is calculated as below:

$$\begin{aligned} \mathrm{R}(\mathrm{ABG}, \mathrm{HYB}) &= \left[(149-146)^2 + (271-273)^2 + (146-142)^2 + \cdots + (2541)^2 + (124-122)^2\right]^{1/2} \\ &= 109.86 \end{aligned} \tag{4.26}$$

The $R^{(1)}$ relational matrix is derived using the Minskowski class formula:

$$\delta = \frac{1}{largest\ distance} = \frac{1}{distance(SAB, TIR)} = \frac{1}{536.07} \tag{4.27}$$

The 'q' value for Minkowski class is taken: q = 2.4.

$$\text{So}: \mathrm{R}(\mathrm{x_i}, \mathrm{x_k}) = 1 - \frac{1}{536.07}\left[\sum_{j=1}^{7}(x_{ij} - x_{kj})^2\right]^{1/2} \tag{4.28}$$

Thus:

$$\mathrm{R}^{(1)} = \begin{matrix} ABG \\ HYB \\ KOD \\ NGP \\ PON \\ SAB \\ TIR \end{matrix} \begin{pmatrix} ABG & HYB & KOD & NGP & PON & SAB & TIR \\ 1 & 0.795 & 0.3603 & 0.8424 & 0.7519 & 0.4609 & 0.196 \\ 0.795 & 1 & 0.2925 & 0.7812 & 0.7234 & 0.4857 & 0.1869 \\ 0.3063 & 0.2925 & 1 & 0.3511 & 0.435 & 0.1393 & 0.5104 \\ 0.8424 & 0.7812 & 0.3511 & 1 & 0.7556 & 0.474 & 0.252 \\ 0.7519 & 0.7234 & 0.435 & 0.7556 & 1 & 0.4046 & 0.3768 \\ 0.4609 & 0.4857 & 0.1393 & 0.474 & 0.4046 & 1 & 0 \\ 0.196 & 0.1869 & 0.5104 & 0.252 & 0.3768 & 0 & 1 \end{pmatrix} \tag{4.29}$$

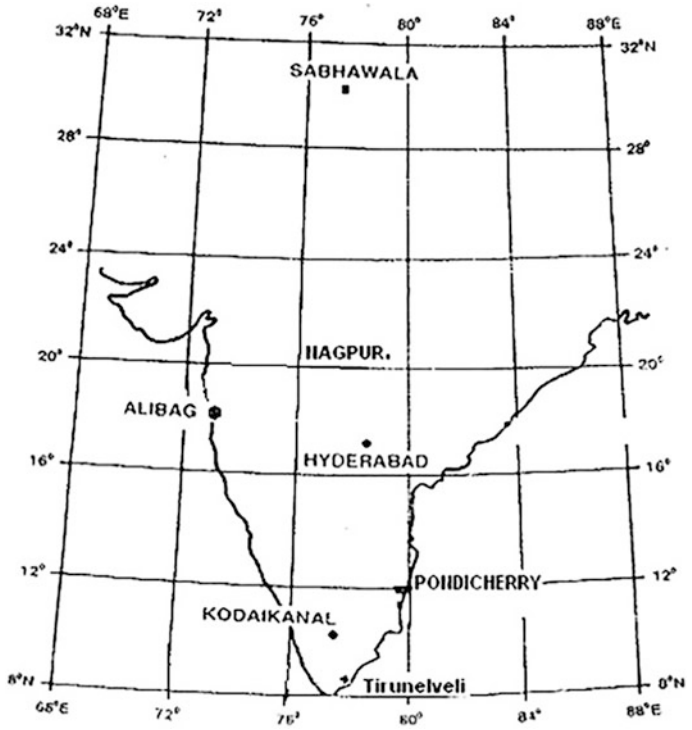


Fig. 4.27 Location map of the observatories (Sridharan 2009)

Table 4.10 Geomagnetic storm time ranges for the horizontal (H) component in nT Redrawn after Sridharan (2009)

Serial No.	Date	ABG-H	HYB-H	KOD-H	NGP-H	PON-H	SAB-H	TIR-H
1	31.01.2001	149	146	163	158	157	128	194
2	19.03.2001	271	273	275	266	279	271	291
3	24.03.2001	146	142	211	145	170	138	267
4	27.03.2001	202	223	271	208	206	131	280
5	08.04.2001	177	177	194	173	234	228	232
6	11.04.2001	332	338	323	347	337	382	323
7	18.04.2001	118	117	189	115	147	160	240
8	28.04.2001	222	226	246	237	240	194	293
9	27.05.2001	142	145	133	154	134	128	144
10	18.06.2001	158	172	225	153	198	125	265
11	17.08.2001	215	207	219	223	213	220	230
12	25.09.2001	185	203	184	191	185	255	207
13	11.10.2001	136	138	181	145	167	153	215
14	21.10.2001	284	295	270	297	286	347	267
15	28.10.2001	219	238	292	213	272	224	315
16	29.14.2001	188	167	228	184	200	170	276
17	17.04.2002	108	113	222	110	139	51	246
18	28.04.2002	176	193	251	193	219	100	211
19	18.03.2002	119	121	133	135	123	158	187
20	23.03.2002	181	176	243	177	180	173	225
21	19.04.2002	183	193	222	200	203	164	272
22	23.04.2002	179	191	234	187	216	144	292
23	11.05.2002	157	158	241	163	165	189	178
24	23.05.2002	255	220	147	235	244	65	239
25	01.08.2002	138	152	186	200	163	155	228
26	18.08.2002	81	98	221	102	113	110	236
27	07.09.2002	162	76	275	166	169	133	252
28	30.09.2002	147	150	308	150	146	134	174
29	11.11.2002	91	90	127	93	102	65	145
30	26.11.2002	83	95	92	81	95	114	94
31	20.03.2003	161	158	158	169	169	126	223
32	29.05.2003	169	180	207	182	171	241	253
33	17.08.2003	254	261	251	258	265	258	248
34	15.11.2003	122	122	119	127	123	122	150

Then R^2 is calculated using 'o' operator, mentioned before, as:

$$R^2 = R^1 \, o \, R^1$$

$$= \begin{matrix} ABG \\ HYB \\ KOD \\ NGP \\ PON \\ SAB \\ TIR \end{matrix} \begin{pmatrix} ABG & HYB & KOD & NGP & PON & SAB & TIR \\ 1 & 0.795 & 0.435 & 0.8424 & 0.7519 & 0.4837 & 0.3768 \\ 0.795 & 1 & 0.435 & 0.795 & 0.7556 & 0.4857 & 0.3768 \\ 0.435 & 0.435 & 1 & 0.435 & 0.435 & 0.4046 & 0.5104 \\ 0.8424 & 0.795 & 0.435 & 1 & 0.7556 & 0.4857 & 0.3768 \\ 0.7519 & 0.7556 & 0.435 & 0.7556 & 1 & 0.4857 & 0.435 \\ 0.4857 & 0.4857 & 0.4046 & 0.4857 & 0.4857 & 1 & 0.3768 \\ 0.3768 & 0.3768 & 0.5104 & 0.3768 & 0.435 & 0.3768 & 1 \end{pmatrix} \tag{4.30}$$

As an example x_{12} in the matrix R^2 is calculated as below (Sridharan 2009). Take the elements of first row and second column in R^1:

First row	1	0.795	0.2925	0.7812	0.7234	0.4609	0.1869
Second column	0.795	1	0.2925	0.7812	0.7234	0.4857	0.1869
Minimum	0.795	0.795	0.2925	0.7812	0.7234	0.4609	0.1869

At the next stage R^4 is obtained:

$$R^4 = R^2 \, o \, R^2$$

$$= \begin{matrix} ABG \\ HYB \\ KOD \\ NGP \\ PON \\ SAB \\ TIR \end{matrix} \begin{pmatrix} ABG & HYB & KOD & NGP & PON & SAB & TIR \\ 1 & 0.795 & 0.435 & 0.8424 & 0.7556 & 0.4857 & 0.435 \\ 0.795 & 1 & 0.435 & 0.795 & 0.7556 & 0.4857 & 0.435 \\ 0.435 & 0.435 & 1 & 0.435 & 0.435 & 0.4046 & 0.5104 \\ 0.8424 & 0.795 & 0.435 & 1 & 0.7556 & 0.4857 & 0.435 \\ 0.7556 & 0.7556 & 0.435 & 0.7556 & 1 & 0.4857 & 0.435 \\ 0.4857 & 0.4857 & 0.4046 & 0.4857 & 0.4857 & 1 & 0.435 \\ 0.435 & 0.435 & 0.5104 & 0.435 & 0.435 & 0.435 & 1 \end{pmatrix} \tag{4.31}$$

Again R^8 is calculated:

$$R^8 = R^4 \, o \, R^4 = \begin{matrix} ABG \\ HYB \\ KOD \\ NGP \\ PON \\ SAB \\ TIR \end{matrix} \begin{pmatrix} ABG & HYB & KOD & NGP & PON & SAB & TIR \\ 1 & 0.795 & 0.435 & 0.8424 & 0.7556 & 0.4857 & 0.435 \\ 0.795 & 1 & 0.435 & 0.795 & 0.7556 & 0.4857 & 0.435 \\ 0.435 & 0.435 & 1 & 0.435 & 0.435 & 0.4046 & 0.5104 \\ 0.8424 & 0.795 & 0.435 & 1 & 0.7556 & 0.4857 & 0.435 \\ 0.7556 & 0.7556 & 0.435 & 0.7556 & 1 & 0.4857 & 0.435 \\ 0.4857 & 0.4857 & 0.4046 & 0.4857 & 0.4857 & 1 & 0.435 \\ 0.435 & 0.435 & 0.5104 & 0.435 & 0.435 & 0.435 & 1 \end{pmatrix} \tag{4.32}$$

As i can be seen $R^8 = R^4$ o $R^4 = R^4$ and as mentioned before this is the stopping criterion for the FTCA method. The final matrix is R_τ and the FTCA leads to α-cut with:

$$\alpha = 0.842,\ 0.795,\ 0.755,\ 0.510,\ 0.485,\ 0.435$$

Finally the following α-cut are formed by R_τ (Sridharan 2009):

$$\alpha \in (0.842,\ 1] : \{(\text{ABG}),\ (\text{HYB}),\ (\text{KOD}),\ (\text{NGP}),\ (\text{PON}),\ (\text{SAB}),\ (\text{TIR})\}$$
$$\alpha \in (0.755,\ 0.842] : \{(\text{NGP} - \text{ABG}),\ \text{HYB},\ \text{KOD},\ \text{PON},\ \text{SAB},\ \text{TIR}\}$$
$$\alpha \in (0.755,\ 0.795] : \{(\text{HYB} - \text{ABG}), (\text{NGB} - \textit{HYB}), (\ \text{KOD},\ \text{PON},\ \text{SAB},\ \text{TIR})\}$$
$$: \{(\text{NGP} - \text{ABG} - \text{HYB}),\ (\text{KOD},\ \text{PON,SAB},\ \text{TIR})\]$$
$$\alpha \in (0.510,\ 0.755) : \{(\text{ABG} - \text{PON}), (\text{HYB} - \text{PON}), (\text{NGP} - \text{PON}),\ ), (\text{KOD},\ \text{SAB},\ \text{TIR})$$
$$: \{(\text{NGP} - \text{ABG} - \text{HYB} - \text{PON}), (\text{KOD},\ \text{SAB},\ \text{TIR})\]$$
$$\alpha \in (0.485,\ 0.510] : \{(\text{KOD} - \text{TIR}), (\text{NGP} - \text{ABG} - \text{HYB} - \text{PON}), (\text{SAB})\]$$
$$\alpha \in (0.435,\ 0.465] : \{(\text{ABG} - \text{KOD}), (\text{ABG} - \text{TIR}),$$
$$(\text{HYB} - \text{KOD}), (\text{HYB} - \text{TIR}), (\text{KOD} - \text{NGP}), (\text{KOD} - \text{PON}),$$
$$(\text{NGP} - \text{TIR}),\ (\text{PON} - \text{TIR}),\ (\text{SAB} - \text{KOD}), (\text{SAB} - \text{TIR})$$
$$: [(\text{NGP} - \text{ABG} - \text{HYB} - \text{PON}), (\text{KOD} - \text{TIR}),\ \text{SAB}]$$

To represent the hierarchical clustering analysis a graphical drawing called a 'dendogram' is used. It is a tree-like plot where each step of hierarchical clustering is shown as a fusion of two branches of tree into single one. Each branch represents clusters obtained at each step of the clustering (Sridharan 2009). The dendogram is shown in Fig. 4.28. As it can be deduced from Fig. 4.27, the horizontal (H) component is divided into three clusters. The first cluster is (NGP, ABG, HYB, PON), the second cluster is (KOD, TIR) and the third is SAB which is isolated from other two cluster. The graphs related to the dendrogram shown in Figs. 4.29, 4.30 and 4.31 implies the validity of the FTCA method for clustering of geomagnetic storms based on their horizontal component, compared to classical methods of clustering (Sridharan 2009).

4.7 Geophysical Data Fusion by Fuzzy Logic to Image Mechanical Behavior of Mudslides

Grandjean et al. (2006a, b) used fuzzy logic for geophysical data fusion in order to image the mechanical behavior of mudslides. They used two kinds of geophysical data: Seismicity data consisting acoustic (P) and shear (S) wave velocity, resistivity data consisting of electrical resistivity. These geophysical data when combined together are well adapted for investigation of landslide structure and understanding the geotechnical related mechanisms. These three physical parameters were considered to define the properties of reworked moving materials. The data was collected over the "super-Sauze" site located in the French South Alps, where intra-material mudslides are available (Grandjean et al. 2006a, b) (Fig. 4.32).

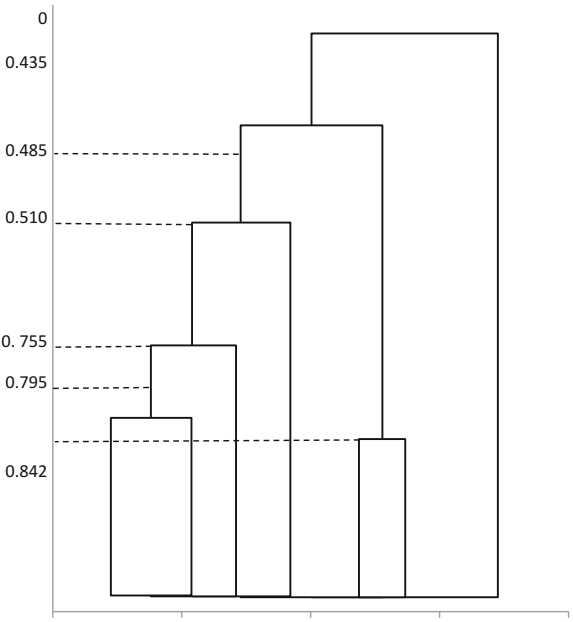


Fig. 4.28 Dendrogram for geomagnetic storm time ranges. Redrawn after Sridharan (2009)

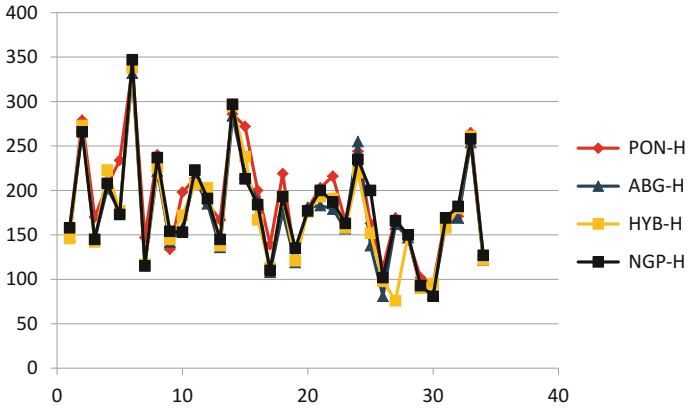


Fig. 4.29 Storm time ranges at non-equatorial regions. Redrawn after Sridharan (2009)

The geophysical data were measured simultaneously along a 325 m profile which was perpendicular to the axis of the mudslide. Grandjean et al. (2006a, b) showed that there is a correlation between the seismic velocities and electrical resistivity data and confirmed that simultaneously using both seismic velocities and electrical resistivity gives more complementary information on the geomechanical behavior of the landslide. The variations of fissure density and the presence of

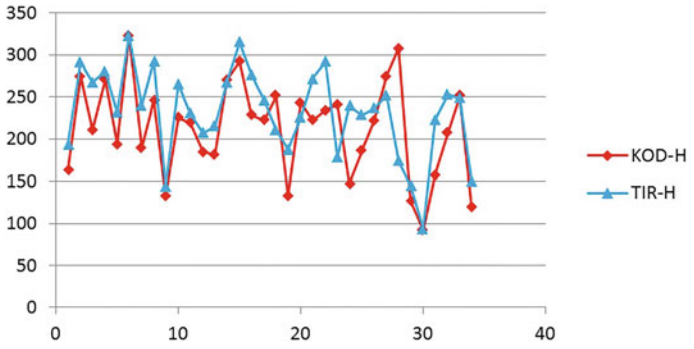


Fig. 4.30 Storm time ranges at equatorial region. Redrawn after Sridharan (2009)

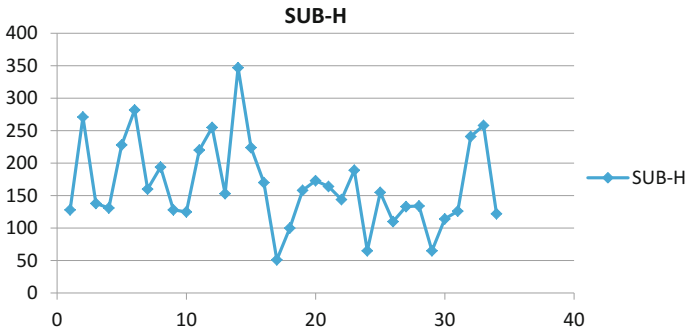


Fig. 4.31 Storm time ranges nearer to Sq focus. Redrawn after Sridharan (2009)

deformed material create the variation of the P and S wave velocity and this variation provides information on the state of compaction of the layers.

The variations of water content within the mudslide lead to changes in electrical resistivity. Therefore, fusion of these data can help the interpreter to go deeper into the interpretation of the geomechanical behavior of the mudslides. One of the useful methods of data fusion is fuzzy data fusion, which is based on fuzzy subset theory. Grandjean et al. (2006a, b) used this strategy for fusion of P-S wave velocity and resistivity to image the mechanical behavior of mudslides. Two main functions were used for the fusion of data, the likelihood function and the possibility function. The reliability of the geophysical tomographically derived (V_p, V_s and ρ) was quantified by the mean of likelihood functions as below:

$$L_{Vp} = \exp\left(\frac{\sum_N \left(\frac{t^c - t^0}{\sigma}\right)^2}{2}\right) \tag{4.33}$$

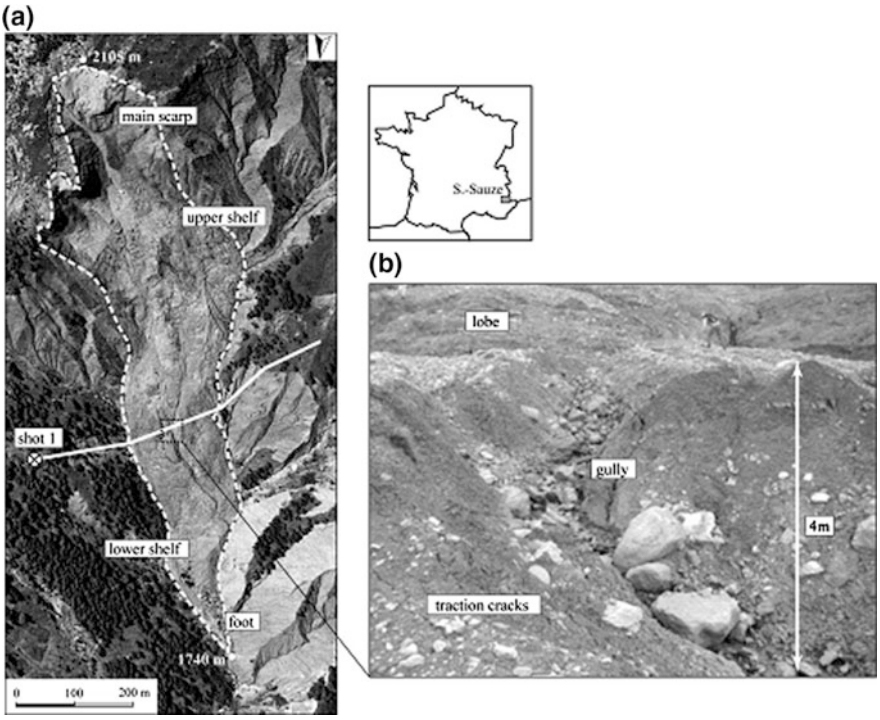


Fig. 4.32 Geomorphology of the "Super-Sauze" mudslide. **a** Location and aerial photograph of the mudslide. The white line represents the studied geophysical profile; **b** morphology of the mudslide with gullies, cracks and topographic variation (Grandjean et al. 2006a, b)

where N is the number of Fresnel wave paths, t^c and t^0 are computed and observed travel times, respectively and **σ** is the a priori uncertainty on the observations.

The L_{Vp} likelihood function represents the quantitative reliability of each part of the V_p tomogram.

$$L_{Vs} = \mathrm{diag}(R); R = W^{-1}V\left(\wedge^2 + \partial^2 I\right)^{-1}\wedge^2 V^T w \tag{4.34}$$

where W is the weight matrix, V and ∧ are the singular value decomposition of the inverse generalized function $G = U\wedge V^{-1}$ and ∂ is the damping factor used for the regularization. For electric resistivity the likelihood function is:

$$L_\rho = \partial\sqrt{2\pi\varphi}, \phi(\varepsilon) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(\frac{-\varepsilon^2}{2\sigma^2}\right) \tag{4.35}$$

where φ is a standard Gaussian function.

P-wave (V_p) velocity and its likelihood, Resistivity and its likelihood, S-wave (V_s) and its likelihood are depicted in Fig. 4.33a–f.

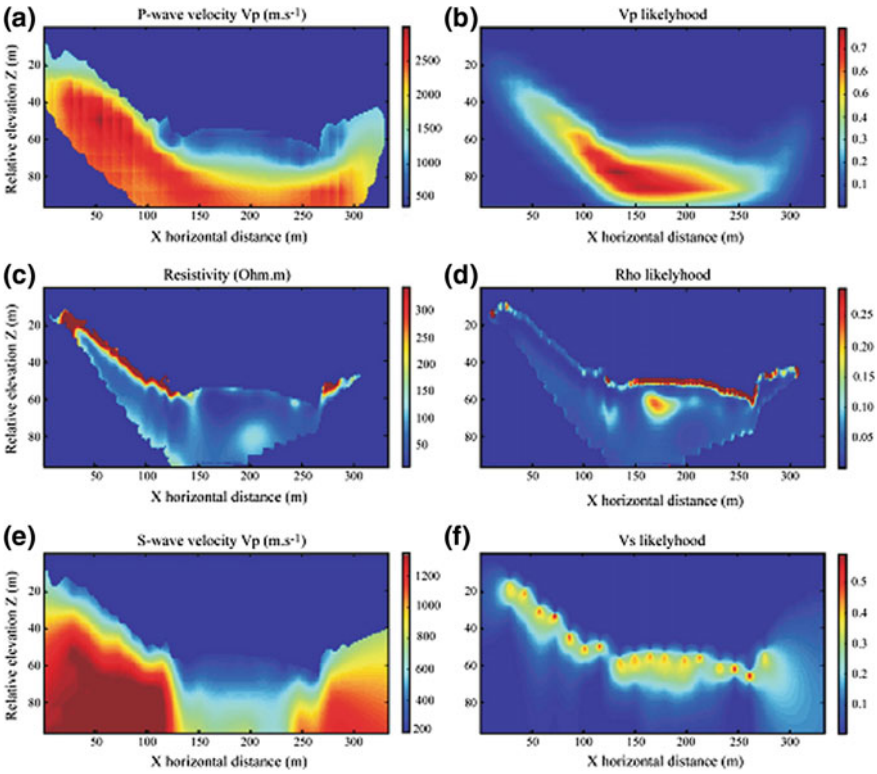


Fig. 4.33 Interpretation of the geophysical tomographic images. **a** (P)-wave velocity tomography inverted from the seismic first arrivals; **b** Vp likelihood function indicating the places where the inverted velocities are reliable (highest values); **c** electrical tomography inverted from apparent (resistivity values). **d** Likelihood function indicating the places where the inverted resistivity values are reliable (highest values); **e** (S)-wave tomography inverted from SASW; **f** Vs likelihood function indicating the places where the inverted velocity values are reliable (highest values) (Grandjean et al. 2006a, b)

The possibility function (π) means the possibility distribution which is a function defining the dependency of each element of set (S) for a given hypothesis, as below (sup means supremum):

$$\pi : S \rightarrow \{0,\ 1\] \quad \sup(\pi(x)) = 1 \tag{4.36}$$

This means that at least one element of S is possible. If C is a subset of S, then the possibility distribution can constructed from the possibility measurement π as below:

$$\forall\ X \in C,\ \pi_{(a)} = \sup\{\ \pi_{(x)},\ x \in X\} \Leftrightarrow \forall_x \in S,\ \pi_{(x)} \in \pi\{x\} \tag{4.37}$$

Also, the combination of two possibility distributions is calculated via the equation below:

$$\pi_{(x)} = \pi_1 \oplus \pi_2 = \frac{\pi_1(x) \bigwedge \pi_2(x)}{\sup\{\pi_1(x) \bigwedge \pi_2(x)\}} \tag{4.38}$$

where '∧' is the fuzzy and operator. The flowchart for calculating the possibility function is shown in Fig. 4.34.

For the fusion of super-Sauze geophysical datasets, the goal that Grandjean et al. (2006a, b) followed was to combine data to increase the amount of information from each tomography image without overestimating the quality and reliability of the results. Therefore, data sets, hypotheses and meta-hypotheses were distinguished as below:

- **Data sets**: V_p, V_s, R_{ho} are featured by likelihood distributions which are quantitative and modeled by a probabilistic approach. The likelihood value of each point of the tomograms means the inaccuracy of the inverted Vp, V_s and R_{ho} for that point.
- **Hypotheses**: Several interpretations derived from the geophysical data define hypotheses. In this case Grandjean et al. (2006a, b) investigated three of main hypotheses as below:

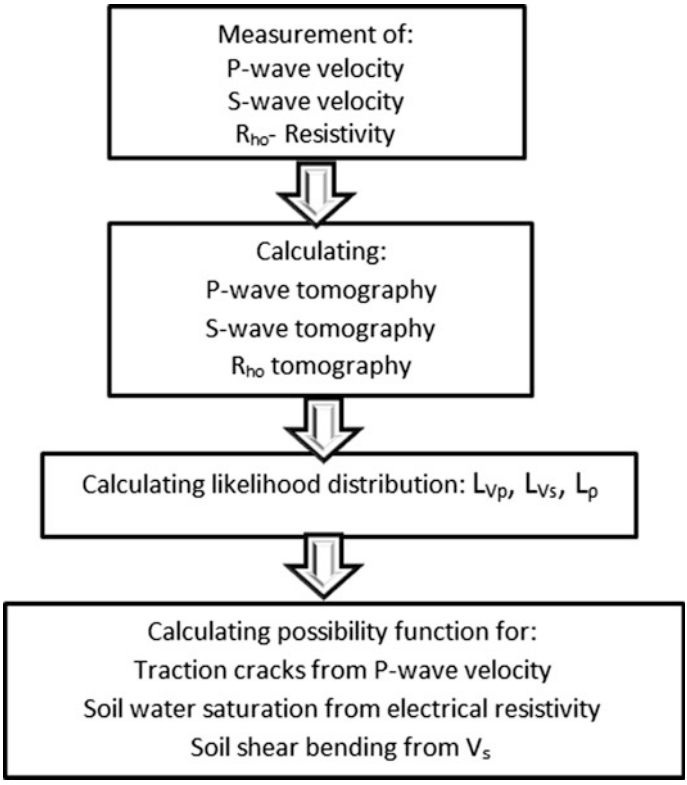


Fig. 4.34 The flowchart for calculating possibility functions

Hypothesis $\mathbf{h}_1$: The possibility π_1 of the soil structure being strongly affected by cracking due to the traction deformation which occurs during the sliding process (Fig. 4.34a). The density of cracking is correlated with P-wave velocity variations. If V_p is lower than 500 m/s the soil strata is fissured and if V_p is greate$_r$ than 1500 m/s the soil strata is not fissured. The uncertainty between these values is assumed to be linear.

Hypothesis $\mathbf{h_2}$: the possibility π_2 of the soil strata to be saturated by water is correlated with the electrical resistivity (Fig. 4.33b). According to mudslide geomorphological knowledge, if electrical resistivity R_{Ho} is lower than 150 Ω m it is saturated and when the $\overline{R_{Ho}}$ is greater than 300 Ω m the soil strata is not saturated. The uncertainty between these values is also assumed to be linear.

Hypothesis $\mathbf{h_3}$: Defines the possibility π_3 of the soil strata to be sheared due to the friction forces which occur during the slope surface failure (Fig. 4.35c). If the S-wave velocity is lower than 500 m/s the soil strata is sheared and if V_s is greater than 1000 m/s the soil strata is not sheared.

The likelihood function has to be integrated. Nifle and Reynaud (2000) demonstrated the data fusion between possibility and probability functions has a mathematical sense only in the framework of evidence theory. Thus, the fusion of a possibility function and likelihood function is expressed via the equation below:

$$\pi^*(x) = \pi(x) v (1 - L(x)) = \max(\pi(x), 1 - L(x)) \tag{4.39}$$

where 'v' is the union operator, $\pi(x)$ is the possibility function and $L(x)$ is the distributions of likelihood values computed for each inversion process.

So:

$$\begin{aligned} \pi_1^*\left(V_p(x,z)\right) &= \pi_1\left(V_p(x,z)\right) v \left(1 - LV_p(x,z)\right) \\ &= \max\left(\pi_1\left(V_p(x,z)\right), 1 - LV_p(x,z)\right) \end{aligned} \tag{4.40}$$

$$\begin{aligned} \pi_2^*(V_s(x,z)) &= \pi_2(V_s(x,z)) v (1 - LV_s(x,z)) \\ &= \max(\pi_2(V_s(x,z)), 1 - LV_s(x,z)) \end{aligned} \tag{4.41}$$

$$\begin{aligned} \pi_3^*(R_{ho}(x,z)) &= \pi_3(R_{ho}(x,z)) v (1 - LR_{ho}(x,z)) \\ &= \max(\pi_3(R_{ho}(x,z)), 1 - LR_{ho}(x,z)) \end{aligned} \tag{4.42}$$

The fusion of the hypotheses produces meta-hypotheses and assigns the soil data to different geomechanical behavior interpretations. For the 'super-Sauze' site geophysical data, Grandjean et al. (2006a, b) assumed two meta-hypotheses: H_1^*, H_4^*.

H_1^*: Define the possibility for the material related to a rigid mechanical behavior.

The rigid mechanical behavior is supposed to exhibit lower water saturation and to be fractured by traction forces but not by shear forces. Therefore the possibility function for this meta-hypothesis was expressed by:

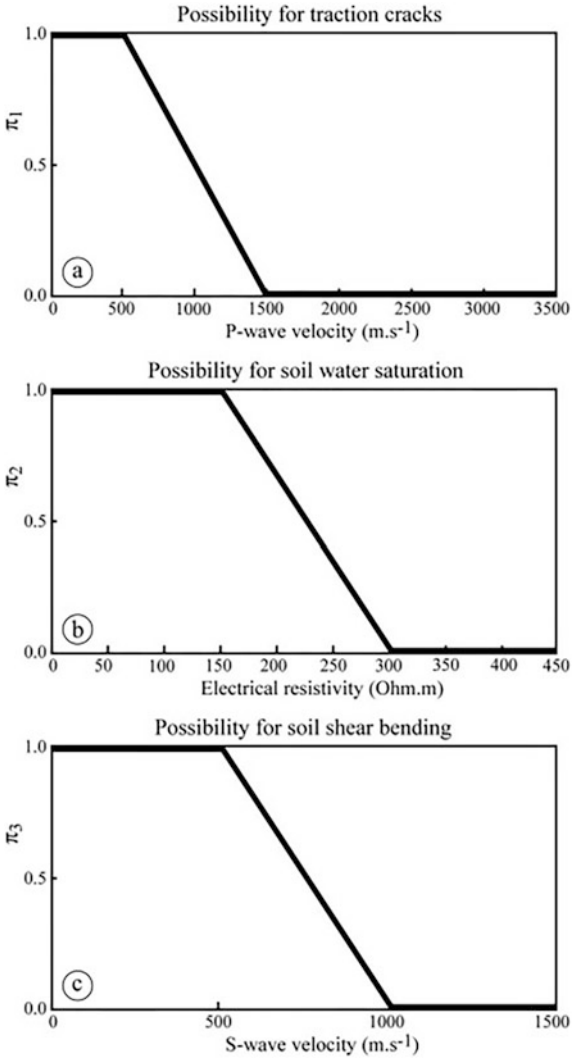


Fig. 4.35 Possibility distributions 1, 2 and 3 corresponding to three hypotheses **a** h_1: "possibility for traction cracks"; **b** h_2: "possibility for soil water saturation"; **c** h_3: "possibility for soil shear bending" (Grandjean et al. 2006a, b)

$$\boldsymbol{\pi_1} = \pi_1^* \oplus \bar{\pi}_2^* \oplus \bar{\pi}_3^* \tag{4.43}$$

H_2^*: This meta-hypothesis defines the possibility for a plastic mechanical behavior which has been subjected to shear forces but not to traction forces and to have a high degree of water saturation. Therefore the possibility function for this meta-hypothesis was expressed by:

$$\boldsymbol{\pi_2} = \bar{\pi}_1^* \oplus \pi_2^* \oplus \pi_3^* \tag{4.34}$$

Using equations above, the possibility distribution for both the solid-state behavior and plastic-state behavior of the mudslide was calculated for each of the points, and illustrated in Fig. 4.36. The computed fuzzy cross-sections show the possibility of the geomechanical hypotheses to be realized in specific areas of the tomographic cross-sections highlighting the places where plastic or solid-body deformations could occur (Grandjean et al. 2006a, b).

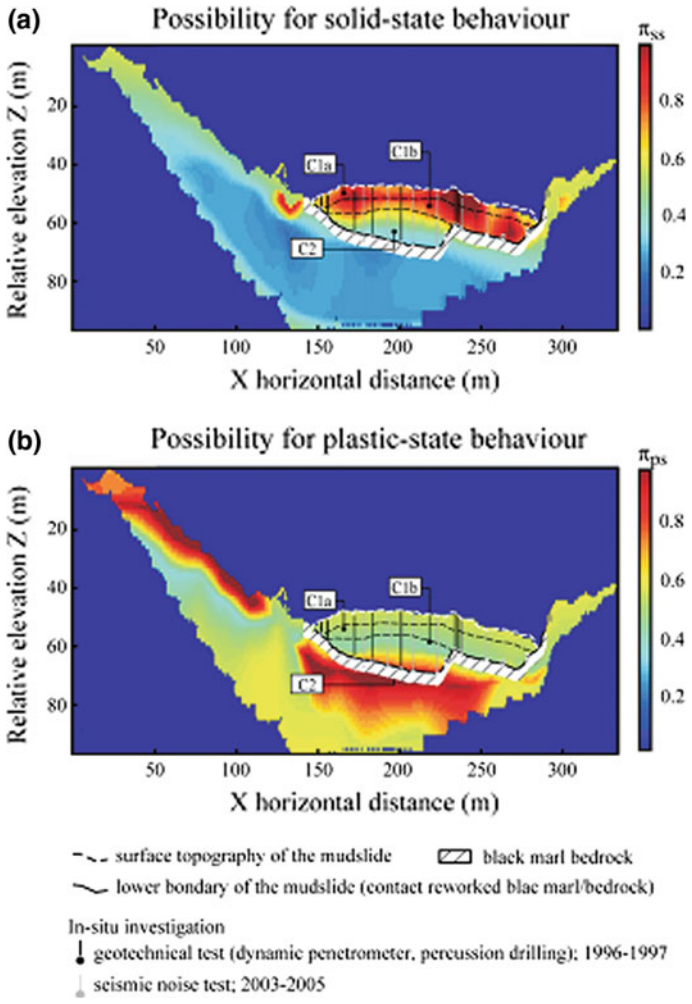


Fig. 4.36 Possibility distributions ($\boldsymbol{\pi_1}$ and $\boldsymbol{\pi_2}$) for **a** the meta-hypothesis (H1*) "solid-state behavior"; **b** the meta-hypothesis (H2*) "plastic-state behavior" (Grandjean et al. 2006a, b)

Grandjean et al. (2006a, b) comprehensively showed that data fusion through fuzzy set theory is an efficient tool for interpreting geophysical tomograms and for proposing hypotheses about the geomechanical behavior of materials as well as on the mechanisms of deformation.

4.8 Automatic Fuzzy-Logic Recognition of Anomalous Activity on Geophysical Log Records

4.8.1 Description of the Research

Zlotnicki et al. (2005) used a fuzzy approach to recognize anomalous activity on long geophysical records of electric signals associated with the volcanic activity of Piton La Fournaise volcano, Reunion Island (Fig. 4.37a). They focused on electric signals recorded on Piton de la Fournaise (Reunion Island) volcano by five stations (Fig. 4.37b) measuring two horizontal components of the electric field during a renewal of activity in 1997–1998. The main goal of their study was to detect and describe the anomalous electric variation specific to the period of effusive activity from March 9 until September 1998. To achieve this goal they built a pattern recognition algorithm based on fuzzy-logic which can be efficiently turned for the identification of an electric anomalous signal. They used the Difference Recognition Algorithm for Signal (DRAS) method, which is a new recognition algorithm for the identification of periods of activity in geophysical time series. The algorithm belongs to the family of the fuzzy sets based anomaly recognition and clustering algorithm introduced by Gvishiani et al. (2003).

One of the biggest advantages of the DRAS method is that it depends on free parameters, which allow the user to tune it for recognition of different type of anomalies in time series. An important feature of the DRAS algorithm, which makes it suitable for automatic recognition, is that the notion of anomaly is strictly defined in terms of fuzzy set mathematics. This feature is especially important in the case where analyzing long time series should be done in the most objective and homogeneous way. DRAS is aimed at significantly increasing the reliability of the identification of anomalous time segments (Zlotnicki et al. 2005). Self-potential anomalies on volcanoes are mostly attributable to streaming or electro-kinetic potentials. The electro-kinetic potential can have a large amplitude (Zlotnicki and Nishida 2003); for example, a 2700 mVSp anomaly measured on Agadak volcano, Adak Island, Alaska, has been attributed to the streaming potential (Nyquist and Corry 2002).

A streaming potential appears when water or other fluid carrying electric charges flow through a porous medium (Uyeda 1996). The water flow is the sum of a gravity flow of meteoric water and a convective flow generated by heat sources (Zlotnicki and Nishida 2003).

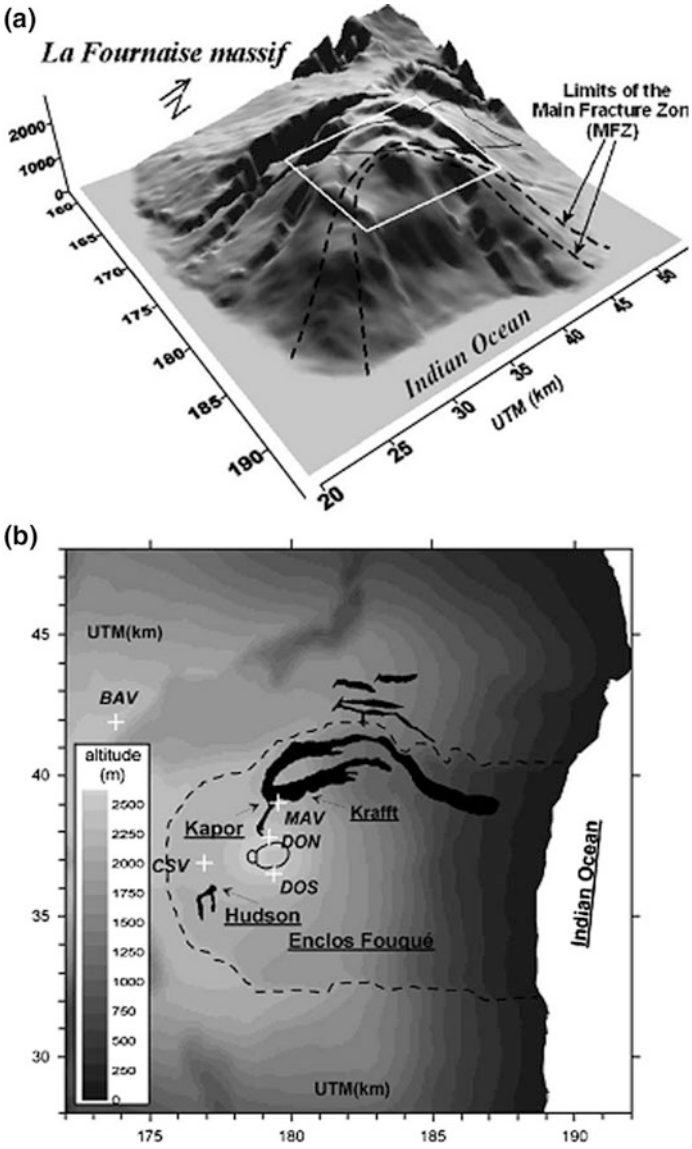


Fig. 4.37 Topography of Piton de La Fournaise volcano. **a** The two dashed lines represent the extension of the Major Fracture Zone (MFZ). **b** Sketch of Piton de la Fournaise volcano with lava flow contours from the March–September 1998 eruption. Electric stations are indicated by white crosses (Zlotnicki et al. 2005)

The SP anomalies observed at the ground surface of volcanoes are generated by different mechanisms:

- Variations of the fluid's pressure gradient.
- Variations of temperature.
- Variations of the chemical composition.
- Variations of the water content in the porous medium.

For example, the pressure variations generate SP signals with characteristic electro-kinetic coefficients between 1 and 100 mv/10 Mpa.

The main purpose of the DRAS fuzzy approaches that Zlontnicki et al. (2005) used was to automatically recognize anomalous SP variations so that they can be briefly analyzed for their possible sources. Before we describe how Zlontnicki et al. (2005) applied the DRAS method for automatic recognition of SP anomalies generated during volcanic activity of Piton de l la Fournaise volcano (Fig. 4.37), we first need to explain Detour's DRAS algorithm based on Zlontnicki et al. published in 'Earth and Planetary Science Letters', 2005. In the next section we first introduce the DRAS method from the appendix of the paper by Zlotnicki et al. (2005).

4.8.2 Difference Recognition Algorithm for Signals (DRAS)

4.8.2.1 The Goal of the Algorithm

We consider a time series (discrete function; recordings) $y = \{y_k = y(kh), k = 1, 2, 3,\ldots\}$, which is defined on a coherent set Y that belongs to the discrete positive semi-axis R_h^+. DRAS represents Y in the form of the two following unions (here and below, as usual, $A \coprod B = A \cup B$, if $A \cap B = \phi$):

$$Y = \sum_{n=1}^{N} A'_n \coprod S' \tag{4.45}$$

where A'_n are the coherent subsets $A'_n \subset Y$ on which the time series y has potential (possible) anomalies perturbing its regular behavior, S′ is the complement, and:

$$Y = \sum_{n=1}^{N} A_n \coprod S, \tag{4.46}$$

where A_n are the coherent subsets $A_n \subset A'_n \subset Y$, on which the time series y has real (genuine) anomalies. It is obvious that the decompositions (4.45) and (4.46) are linked by the conditions $S \supset S'$ and $\forall A_n \exists A'_m : A^n \subset A'_m$.

4.8.2.2 Preparatory Phase of the Algorithm, Rectification of the Time Series

We call Δ, Δ = l.h, l = 1,2, …,k the "parameter" of local "observation". The following part of the time series y with center at the point k.h

$$\Delta^{\mathrm{k}}\mathrm{y} = \left\{ \mathrm{y_k} - \Delta/\mathrm{h}, \ldots, \mathrm{y_k}, \ldots, \mathrm{y_{k+\Delta/h}} \right\} \in \Re^{\frac{2\Delta}{h}+1} \tag{4.47}$$

is called a fragment of local observation.

There are positive functional φ_{y}: $\mathrm{I} \to R_h^+$, defined on the set $\mathrm{I} = \{\Delta^{\mathrm{k}}\mathrm{y}\}$ of the fragments (3), which translate anomalous fragments $\Delta^{\mathrm{k}}\mathrm{y}$ into "uplifted" parts of the corresponding curve $\varphi_{\mathrm{y}}(\mathrm{k})$ (see Fig. 4.38). Here we mean that a fragment of the curve is uplifted if the area under this fragment is significantly larger than the corresponding areas under other fragments of the curve having a similar length.

We call such functionals φ_{y}, rectifying functionals of the time series. Correspondingly, we call the function $\varphi_{\mathrm{y}}(\mathrm{k})$ the rectification of y. Different rectifying functionals are tuned for different types of anomalies. The choice of a rectifying functional is one of the free parameters of the algorithm.

In this study we applied DRAS with the following rectifying functionals:

(1) Length of the fragment of local observation: $\mathrm{L}\left(\Delta^{\mathrm{k}}\mathrm{y}\right) = \sum_{j=k-\frac{\Delta}{h}}^{k+\frac{\Delta}{h}-1} |y_{j+1} - y_j|$.

(2) Energy of the fragment of local observation: $\mathrm{E}\left(\Delta^{\mathrm{k}}\mathrm{y}\right) = \sum_{j=k-\frac{\Delta}{h}}^{k+\frac{\Delta}{h}} \left(y_j - \bar{y}_k\right)^2$.

(3) Departure of $\Delta^{\mathrm{k}}\mathrm{y}$ from its regression of order n:

$$\mathrm{R_n}\left(\Delta^{\mathrm{k}}\mathrm{y}\right) = \sum_{j=k-\frac{\Delta}{h}}^{k+\frac{\Delta}{h}} \left(y_j - Regr^n_{\Delta^k y}(jh)\right)^2. \tag{4.48}$$

where $Regr^n_{\Delta^k y}$ is the optimal mean squares approximation of order n of y on the fragment $\Delta^k y$. It is straightforward that $\mathrm{R_0}\left(\Delta^k y\right) = \mathrm{E}\left(\Delta^k y\right)$.

4.8.2.3 Potential Anomaly Domains Construction

We introduce $\Lambda \gg \Delta$, $\Lambda \in \Re_h^+$—another free parameter of DRAS. It will be called the "parameter of global observation". In the same way as for $\Delta^k y$ (Eq. 4.3), the set $\Lambda^{\mathrm{k}}\varphi_{\mathrm{y}} = \{\varphi_{\mathrm{y}}(\mathrm{kh} - \Lambda), \ldots, \varphi_{\mathrm{y}}(\mathrm{kh}), \ldots, \varphi_{\mathrm{y}}(\mathrm{kh} + \Lambda)\}$ will be called a fragment of global observation.

We consider the set S′ of the calm (regular) points, defined by formula (1), as a fuzzy set (S′, μ) (e.g. Jensen and Toll 1983). (By definition, the fuzzy set (S′, μ) is the set of objects {s} that belong to S′ a certain degree that is measured by the

membership functions μ: $0 \leq \mu(s) \leq 1$. The goal of this phase of the algorithm is to construct a sufficiently good approximation of the membership function Y: → [0, 1] and to define the weighting function. Set S′ of potentially calm (regular) points on this basis. We introduce the weighting function:

$$\delta_{kh}(\bar{k}h) = \frac{\Lambda + h - h|\bar{k} - k|}{\Lambda + h} (\bar{k}h \in [kh - \Lambda, kh + \Lambda]) \tag{4.49}$$

This weighting function has the form of an equal sided triangle with the summit at point kh, a height equal to 1, and basis length [kh – Λ − h, kh + Λ + h]. Another free parameter of the algorithm is $\alpha \in \Re$; $\min_{k \in Y\varphi y}(k) < \alpha < \max_{h \in Y\varphi y}(k)$, which is called the vertical level of noise. For a given α, we introduce the left side measure $L_\alpha \varphi_y$ on the rectifications by the formula:

$$L_\alpha \varphi_y(k) = \frac{\sum_k \delta(kh) : \phi y(\bar{k}) \leq \alpha}{\sum_k \delta(kh)} \tag{4.50}$$

In the sum of formula (4.50) $k \in [k - \Lambda/h, k]$.

In the same way we introduce the right sided measure where $\bar{k} \in [k, k + \Lambda/h]$. These measures are the DRAS major tools which are used to find the effects of anomalies on the recording rectifications. The following definitions are used:

1.1 Point k is a α-calm point from the left (from the right), if:

$$\Phi y\ (\bar{k}) \leq \alpha \forall \bar{k} \in [k - \frac{\Lambda}{h}, k] \quad \left(\forall \bar{k} \in \left[k, k + \frac{\Lambda}{h}\right]\right). \tag{4.51}$$

1.2 Point k is an a-not calm point from the left (from the right), if there is at least one point

$$\begin{gathered} \bar{k} \in [k - \tfrac{\Lambda}{h}, k] \\ (\bar{k} \in [k, k + \tfrac{\Lambda}{h}]) \text{ such as } \varphi_y(\bar{k}) > \alpha \end{gathered} \tag{4.52}$$

1.3 Point k is a-anomalous from the left (from the right), if $\varphi_y(\bar{k}) > \alpha$ Na for any point

$$\bar{k} \in [k - \frac{\Lambda}{h}, k] \ \ (\bar{k} \in \left[k, k + \frac{\Lambda}{h}\right]) \tag{4.53}$$

As follows from Eq. (4.53), the following statement is true.

Statement:

1.1 A point k is a-calm from the left (from the right), if and only if $(L_\alpha \varphi_y)(k) = 1$

$$\left(R_\alpha \varphi_y\right)(k) = 1 \tag{4.54}$$

1.2 A point k is a-not calm from the left (from the right), if and only if $0 < (L_\alpha \varphi_y)(k) < 1$.

$$0 < \left(R_\alpha \varphi_y\right)(k) < 1 \tag{4.55}$$

1.3 A point k is a-anomalous from the left (from the right), if and only if $(L_\alpha \varphi_y)(k) = 0$

$$\left(L_\alpha \varphi_y\right)(k) = 0 \tag{4.56}$$

In other words, the statement means that for a given level of noise, $L_\alpha \varphi_y$ and $R_\alpha \varphi_y$ can be considered as membership functions for the fuzzy set of α-calm points from the left and from the right, respectively. It is natural to consider that bigger values of the membership functions correspond to calmer points k of the time series y under consideration. On the contrary, smaller values of the membership functions correspond to more anomalous regions (i.e. anomalous from the left and/or from the right) of the record y. Mathematically speaking it means that $L_\alpha \varphi_y$ and $R_\alpha \varphi_y$ functions introduce relations of order in the set of a-not calm points.

Another free parameter is β, $0 \leq \beta \leq 1$, the horizontal level of noise. For a given β, we construct the decomposition of the definition domain Y into the union of the domain of calm (regular) points S′ and union of the domain of calm (regular) points S′ and the domain A′ in which has potential anomalies. The sets S′ and A′ are defined by the formulas:

$$S' = \{kh \in \Re_h^+ : \min\left(\left(L_\alpha \varphi_y\right)(k), \left(R_\alpha \varphi_y\right)(k)\right) \geq \beta \tag{4.57}$$

$$A' = \{\ kh \in \Re_h^+ : \min\left(\left(L_\alpha \varphi_y\right)(k),\ \left(R_\alpha \varphi_y\right)(k)\right) < \beta \tag{4.58}$$

In other words we classify as calm those points ($k \in S'$) which have sufficiently big values of both 'left' and 'right' membership functions $(L_\alpha \varphi_y)(k) \geq \beta$ and $(R_\alpha \varphi_y)(k) \geq \beta$. The simplest DRAS version corresponds to $\beta = 1$, point $k \in S'$ only if it is a-calm both from the left and from the right. The searched subsets A'_n of the potential anomalies in formula (1) are the coherent components of the set A′ defined by Eq. (4.6):

$$A' = \coprod_{n=1}^{N} A'_n \tag{4.59}$$

4.8.2.4 Genuine Anomaly Domain Construction

In this last phase, inside the sets of potential anomalies $A_n^{'}$, we establish the genuine anomaly domains $A_n \subset A_n^{'}$.

For that we consider the membership functions (4) and introduce the difference between the "left" and "right" ones:

$$\left(D_\alpha \varphi_y\right)(k) = \left(L_\alpha \varphi_y\right)(k) - \left(R_\alpha \varphi_y\right)(k). \tag{4.60}$$

Let a point $k \in A_n'$ be a point of maximum for the function, and suppose $(D_\alpha \varphi_y)(k)$, and suppose $(D_\alpha \varphi_y)(k) > 0$ [here, by the maximum we just mean that $(D_\alpha \varphi_y)(k) > (D_\alpha \varphi_y)(k+1)$] and $(D_\alpha \varphi_y)(k) > (D_\alpha \varphi_y)(k-1)$. In such a situation the difference between "α-calmness from the left and α-calmness from the right" is bigger than at the neighboring points. Therefore it is natural to assume that point k is the beginning of a real (genuine) anomalous domain $A_n \subset A_n'$ (see Eq. 4.2). Let us denote $M_y(\alpha,n)$ the set of such points $k \in A_n'$. In the same way, if $(D_\alpha \varphi_y)(k)$ has a minimum at the point $k \in A_n'$, and $(D_\alpha \varphi_y)(k) < 0$, then we have a minimum difference between "α-calm level from the left" and "α-calm level from the right". It is natural to assume that such a point is the end of the searched domain of the genuine anomalies interval, $A_n \subset A_n'$ We denote $m_y(\alpha,n)$ the set of such points $k \in A_n'$.

The following situations are possible:

1. At least one of the sets $M_y(\alpha,n)$, $m_y(\alpha,n)$, is empty. Then, by definition, there is no real anomaly in the set of potential anomalies. In other words $A_n = \phi$.
2. $M_y(\alpha,n) \neq \phi, m_y(\alpha,n) \neq \phi$ and $\forall k \in My(\alpha,n)$, $\forall k' \in m_y(\alpha,n)$, the condition $k < k'$ is true. In this situation we have a contradiction. Indeed, the potential end of A_n takes place later than its potential beginning. That cannot be, and there is no genuine anomaly domain inside our potential anomaly set A_n'. In this case $A_n = \phi$.
3. $M_y(\alpha,n) \neq \phi$, $m_y(\alpha,n) \neq \phi$, and there is at least one pair (k, k′), such as $k \in M_y(\alpha,n)$ and $k' \in m_y(\alpha,n)$ $k < k'$. In this case we define the real potential domain A_n as the segment $A_n = [a_n, b_n]$, where:

$$\begin{aligned} a_n &= \min\{k : k \in M_y(\alpha, n)\} \\ b_n &= \max\{k' : k' \in m_y(\alpha, n)\} \end{aligned} \tag{4.61}$$

The latter step completes DRAS construction of the searched decomposition.

Figure 4.38 illustrates the three phases of the algorithm by applying it to synthetic data. The first graph shows a synthetic time series y(t) containing three genuine anomalies of different morphology. The first one is a high-frequency oscillating anomaly, the second is a spike, and the third and latest one is a high-frequency oscillation superimposed on the background of a high amplitude signal. We apply DRAS for recognition of these obvious anomalies to demonstrate how the algorithm identifies them at its different stages.

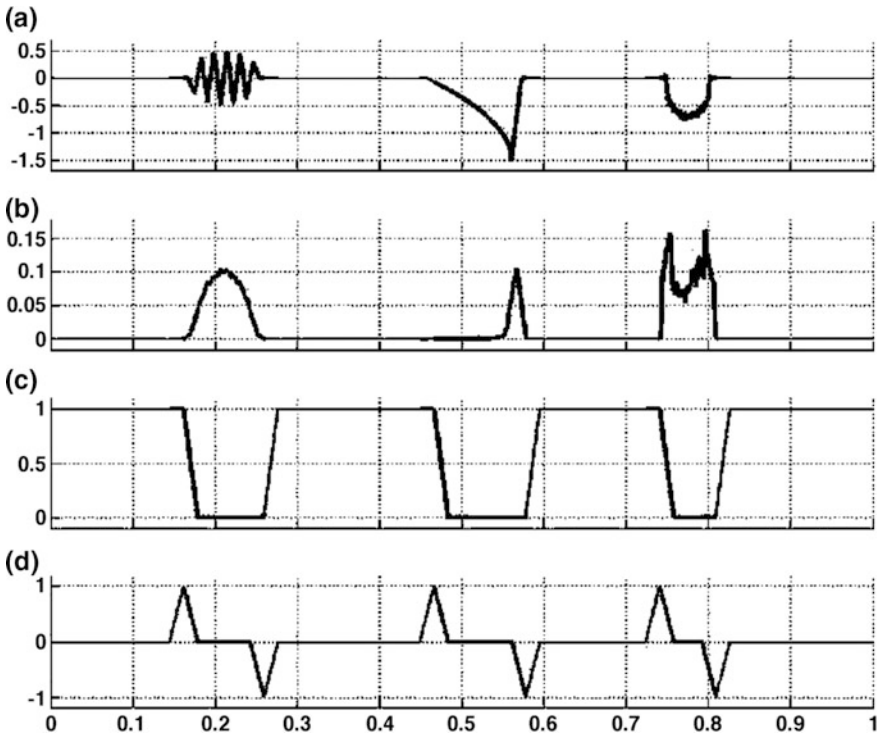


Fig. 4.38 Synthetic example. **a** Synthetic signal containing three events of different morphology; **b** rectification functional (length); **c** left membership function (see Eq. 4 of the appendix); **d** difference $(D\alpha\varphi\gamma)(k)$ (Zlotnicki et al. 2005)

The three anomalies are already clearly recognized in the preparation phase, through the rectification $L(\Delta^k y)$, represented in the second graph. In the second step, the fuzzy membership function (third graph) recognizes the anomalies in the same way regardless their morphology.

The last graph $(D_\alpha\varphi_y)(k)$—the membership function difference) clearly shows that in the final stage DRAS is able to recognize the anomalies in a truly homogeneous way. The form of the "anomaly signals" in the $(D_\alpha\varphi_y)(k)$ graph is indeed practically identical for the three types of anomalies under consideration. That illustrates the fact that DRAS does recognize anomalies in the general behavior of the time series and not just from a specific part of the curve.

4.8.3 Application of the DRAS Algorithm to Observational Data

The DRAS algorithm was used to study SP variations recorded at five stations: DON, DOS, CSV, BAV and MAV shown in Fig. 4.37.

As described in the previous section the DRAS algorithm uses fuzzy sets to mathematically model the natural logic used by the interpreter to search for the anomalies in the time series, recorded, by a robust unbiased estimation. The main reason to use the fuzzy approach in this case is that the general notion of an anomaly is a rather vague concept. The interpreters' logic is modeled via DRAS. At first, the interpreter inspects relatively short fragments of the record whose length is comparable to the length of activity periods in the recorded parameter y(t).

Using this procedure he/she somehow attributes an estimate of activity level to these fragments (or the centers of fragments). This procedure consists in a substitution of the initial record by a positive function of time defined on the same time span as the initial time series: $\varphi_y(t)$ namely the "rectification" function of the initial time series. The $\varphi_y(t)$ characterizes the level of activity which at particular time intervals where y(t) presents a more intense activity which will be given larger values of $\varphi_y(t)$. The rectification transform can be defined in many various ways, depending on what the interpreter, looking for a specific type of anomaly, wishes to emphasize, in DRAS the choice of "rectification functional" is a free feature of the algorithm.

As an example when the anomaly searched for is characterized by a high level of oscillation, the rectification functional can be defined as zero crossings rate (Zlotnicki et al. 2005).

In the next stage the interpreter looks to find 'hills' in the rectification function, which correspond to time sets {t} where the values of $\varphi_y(t)$ are large enough to make the sets distinct from the rest of the series. In this way, the concept 'large enough' is a linguistic variable and the interpreter works at two levels:

- A local level, where the rectification of the record substitutes for its initial from
- A global level, where the "hills" in the rectification function $\varphi_y(t)$ are searched.

Consequently, DRAS similarly operates at two levels. At the local level it constructs the "rectification functional" I and "rectification functions" $\varphi_y(t)$.

Different rectifications, based on different parameters of the time series can be used:

- Energy in a fragment.
- Length of the period's activity in a fragment.
- Level of the recorded oscillations.
- etc.

Zlotnicki et al. (2005) showed that the choice of the rectification function is not crucial for the effectiveness of the signal.

DRAS then scrutinizes the signal at the global level. As the graph of the rectification function can be complicated, the analysis of the ordinates $\varphi_y(t)$ is insufficient by itself to recognize the "hills", anomalies may have high ordinate but short durations, active segments may be separate by short calm intervals. In the light of these facts, DRAS explores the problem at the global level in two stages. In the first stage, the record y(t) is represented as a union of background (calm) and potentially anomalous parts. In the second stage, a procedure is applied which searches for real anomalous fragments inside the potentially anomalous part of the record.

The left and right measures, $L\varphi_y(t)$ and $R\varphi_y(t)$ plays a basic role in this process. These measurements are interpreted as membership functions of the fuzzy set of sufficiently calm (non-anomalous) points of the initial time series y(t). The membership functions $L\varphi_y(t)$ and $R\varphi_y(t)$ characterize how calm a recording is on the left side and on the right side of a point 't'. The difference $D\varphi_y(t) = L\varphi_y(t) - R\varphi_y(t)$ appears to be a crucial parameter to discriminate a genuine anomalous point 't' from background levels.

The beginnings of the anomalous segments are located at the positive maximum of the function $D\varphi_y(t)$. Indeed, at such points {t], the difference between the calmness right of t reaches a maximum value. And for the same reason, the ends of anomalous segments are located at points of negative minima of $D\varphi_y(t)$.

The result of DRAS method, the domain of y(t) is represented as the union of three fuzzy sets of point t (shown in Fig. 4.39):

- Calm (non-anomalous) points (shown in black in the Fig. 4.39).
- Potentially anomalous points (shown in blue in the Fig. 4.39).
- Really anomalous points (shown in red in the Fig. 4.39).

It is necessary to mention that the algorithm is tuned by a learning process before its application to data. Briefly, this tuning is applied by adjusting the free parameters of the algorithm so that all the anomalies chosen manually are recognized automatically by DRAS.

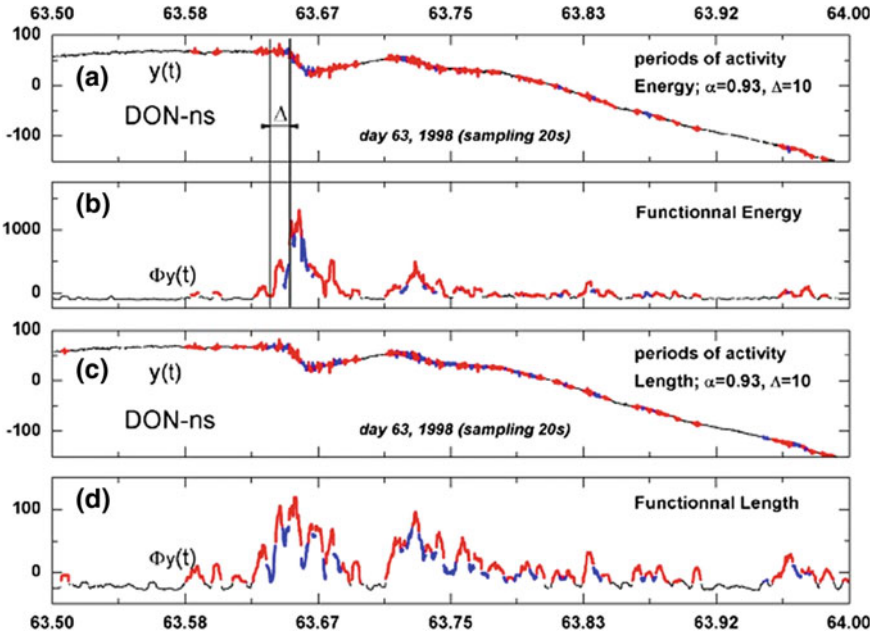


Fig. 4.39 Comparison of two different functional U: the length **a**, **b** and the energy **c**, **d** and corresponding periods of activity for DON NS channel, between days 63.5 and 64.0, 1998. **d** Is the time window on which DRAS is applied. Non active periods are shown in black; probable periods of activity in blue; periods of activity in red (Zlotnicki et al. 2005)

After completing the learning process, the number of points of the local observation parameter Δ was taken equal to be to 30 (600s) and the value of the free parameter α equal to 0.97. Using these parameters more than 95% of the learning signals were recognized by DRAS which was then applied to the whole duration of the different time series of the stations. As an example of the results that (Zlotnicki et al. 2005) obtained we show the automatic recognition of SP periods of activity for recording of DON station for the time interval March 15, 1997–December 31, 1998. It is seen from Fig. 4.40 that the anomalous periods detected by DRAS algorithm can be subdivided into two groups:

(1) Periods containing intensive high-frequency oscillations of relatively low amplitude (up to 100 mV/km within a few minutes).
(2) Periods including high-amplitude spikes and steps (up to 1000 mV/km or more within time intervals of tens of minutes and more).

It is necessary to note that heavy rains disturb the recording of electric fields. Thus; it is legitimate to assume that the groups of anomalous periods of activity mentioned two have different origins and that the anomalies of the second group are generated by rain (Fig. 4.41).

So, Zlotnicki et al. (2005) performed the DRAS analysis again only on days with rainfall less than 10 mm.

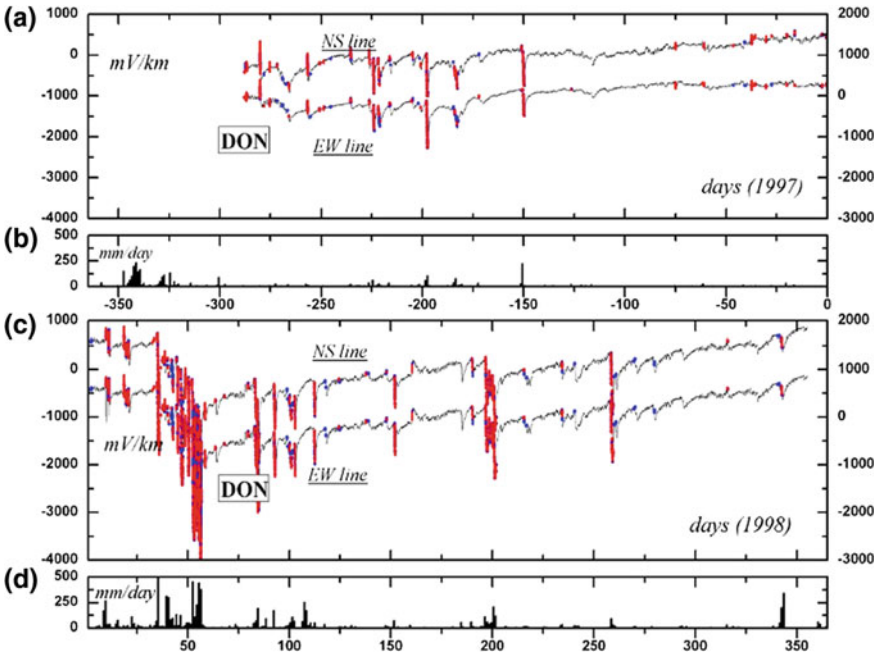


Fig. 4.40 SP variations recorded by the station DON on NS and EW channels between March 15, 1997 and December 31, 1998 (graphs **a** and **c**). January 1, 1998 is referenced to day 1. Periods of activity found by DRAS are shown by blue and red colors. Parameters of algorithm are a = 0.97 and D = 30 (see text for details). Daily rainfall is presented in cartoons (**b** and **d**) (Zlotnicki et al. 2005)

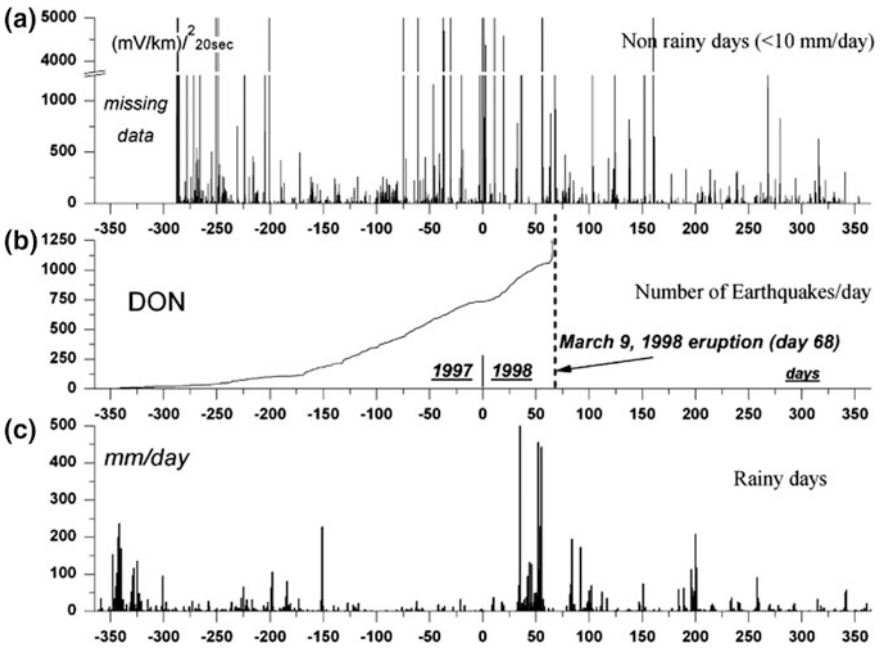


Fig. 4.41 **a** Average of the energies on both NS and EW channels at DON station, for the non-rainy (V10 mm/day) days computed at only points recognized by DRAS as belonging to a period of activity from January 1, 1997 to December 31, 1998. **b** Cumulative number of earthquakes per day according to Staudacher (1998) and Aki and Ferrazini (2000). **c** Daily rainfall (Zlotnicki et al. 2005)

Comparison of Figs. 4.40 and 4.43 reveals that almost all high amplitude spikes and steps have disappeared which means that they indeed essentially occur on rainy days. On the contrary, low-amplitude high-frequency oscillations are identified as signals on both figures, which prove that these signals do not originate from rain.

Such an analysis would not be practicable without the application of an automated formal fuzzy algorithm such as DRAS. The 2 year data sets considered by Zlotnicki et al. (2005) cannot be done manually (by eye) because the data were too long (16×10^6 data points per year) and the results would risk being biased by the interpreter's "preferences" and prior idea.

4.9 Operational Earthquake Forecasting Using Linguistic Fuzzy Rule-Based Models from Imprecise Data

Earthquake prediction is one of the most important unresolved problems in geosciences. Many researchers globally have been actively involved in earthquake precursor and prediction studies using multi-disciplinary tools and methods without a common consensus as to any reliable and consistent methodologies which can

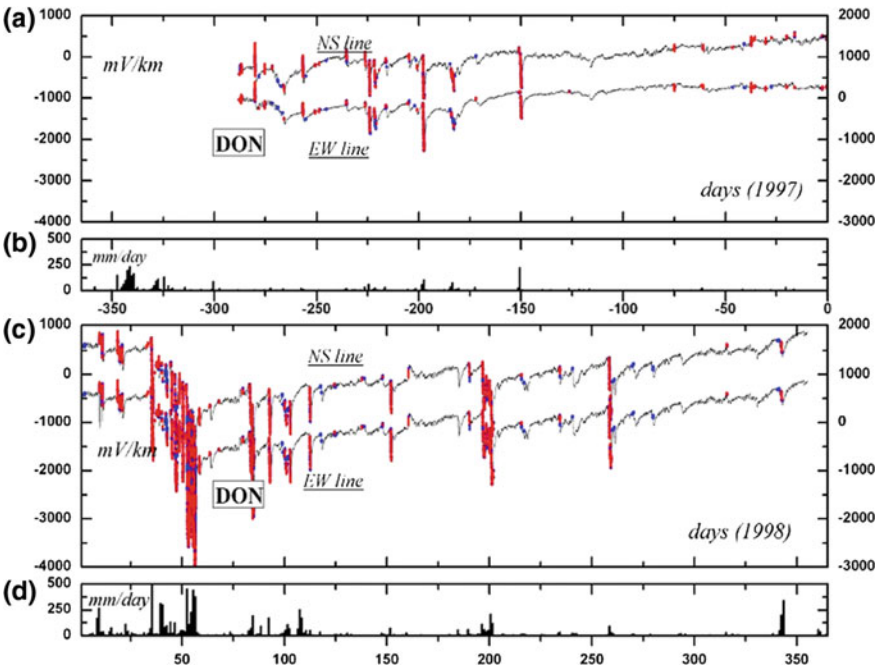


Fig. 4.42 SP variations recorded by the station DON on NS and EW channels between March 15, 1997 and December 31, 1998 (graphs **a** and **c**). January 1, 1998 is referenced to day 1. Periods of activity found by DRAS are shown by blue and red colors. Parameters of algorithm are a = 0.97 and D = 30 (see text for details). Daily rainfall is presented in cartoons (**b** and **d**) (Zlotnicki et al. 2005)

predict an earthquake (Mishra 2012). The main techniques applied were probabilistic models, precursor models, neural networks, active fault models, Bayesian belief network and decision trees which provide primary solutions to the problems inherent in the prediction of earthquakes.

In investigations of earthquake occurrence there is a need to recognize and analyze multiple variable processes having mutual origins and mutual attributes. Based on this fact, Dutta et al. (2013) devised a procedure for finding quantitative relationships estimated by missing values and coarsely discrete data values and the total error of the sample data between these variables through weighted regression. The objective of theirs study was interpreting the spatial-temporal properties of geographical objects with the help of regression equations and fuzzy rules for finding interconnectedness among the attributes for the underlying physical phenomena of seismic behavior.

On one hand, some researchers have strongly suggested that earthquake occurrence is completely unpredictable by nature (Geller et al. 1997) with the deterministic localization of a future event in a narrow time window as highly improbable but on the other hand many researchers across the world, especially in the U.S.A. (Shimazaki and Stuart 1985; Dmowska 1997), Japan (Asada 1982), Italy

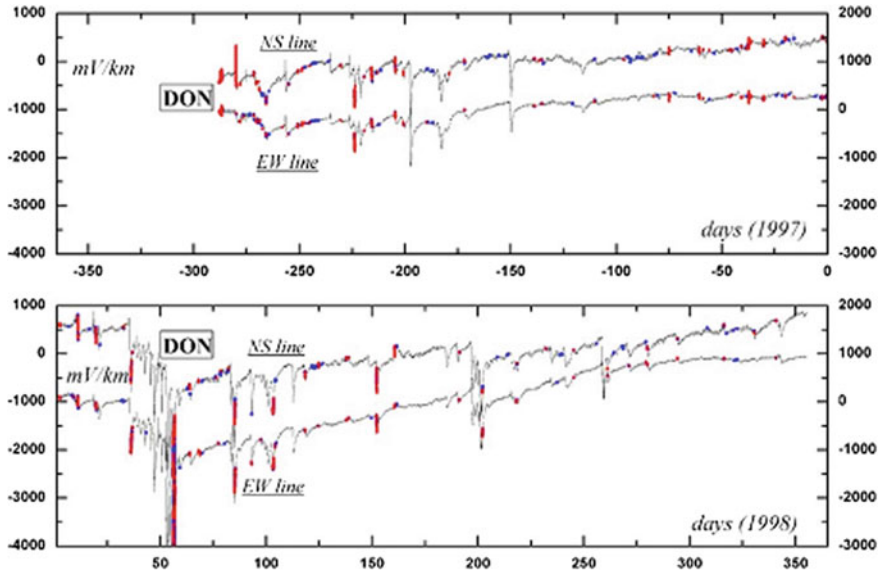


Fig. 4.43 The same analysis as shown Fig. 4.42 (a = 0.97, D = 30) but only for non-rainy days (V 10 mm/day) (Zlotnicki et al. 2005)

(Dragoni and Boschi 1992), Turkey (Vogel and Itsikara 1980), China (Shin-jung 1993), Netherlands (Kisslinger 1986), India (Guha and Patwardhan 1985) have been monitoring earthquake pattern and clusters in order to enhance the process of operational earthquake forecasting for a very long time. Operational earthquake prediction techniques have been rejuvenated with the advent of new seismic monitoring resources and instrumentation. The recent decade has been buoyed by optimistic outcomes and marred by total disappointments (Geller 1997). Dutta et al. (2013) presented a review of operational earthquake forecasting methodologies using linguistic fuzzy rule-based models from imprecise data with weighted regression approach. Earthquake disaster analysis always yields some amount of "impreciseness" or "vagueness" or "fuzziness" due to heterogeneity in the underlying phenomena, and/or explanatory variables, and/or response variables. For a model with more realistic features, there is a need to incorporate this aspect in traditional models, for example weighted linear regression. In this way, Dutta et al. (2013) critically examined some of the modern seismological earthquake algorithms used for analyzing seismo-electro-telluric-geodetic data across the globe. A general schematic of prediction by extraction of the dynamics of observable behavior of earthquake system are depicted in Fig. 4.44.

The dynamics of the lithosphere are stochastic and very non-linear and can't be easily modeled or forecast.

One statistical method, which has been attempted, is probabilistic forecasting. If the probability of a target event during a fixed forecasting interval is P(t), the decision rule might be to flag a regional alarm for the subsequent interval whenever

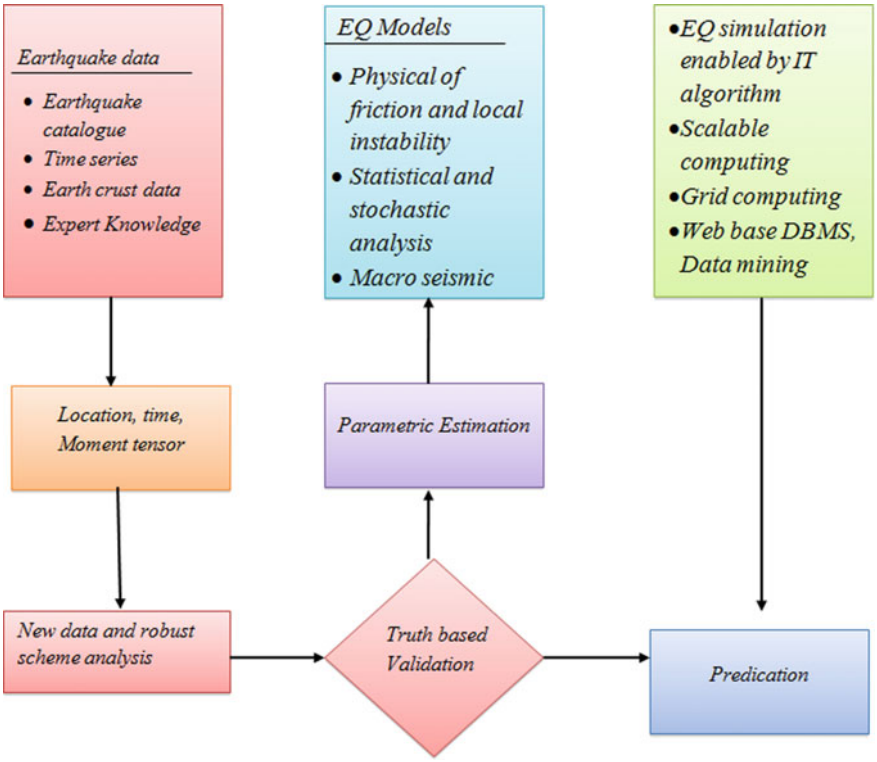


Fig. 4.44 General scheme of prediction by extraction of dynamic of observable behavior of earthquake. Redrawn after Dutta et al. (2013)

this time-dependent probability exceeds some threshold value P_0. If P(t) is low at all times, which is typical in forecasting large earthquakes over short periods, at least one of the prediction error rates will always be high, regardless of the decision rule. Such predictions always contain less information than the forecast from which they were derived. Consequently, probabilistic forecasting provides a more complete description of prospective earthquake information than deterministic prediction. In this method of analysis, fuzzy membership functions are very important. The components of soft computing include neural networks, fuzzy system, and evolutionary computation and swarm intelligence. Many of these soft computing methods are used for earthquake prediction.

It is found that premonitory increase of the earthquake correlation range was a useful parameter; these chains are the dense, long, and rapidly formed sequences of small and medium earthquakes.

Historical data are collected which follow the time series methodology, the mined data are combined for preprocessing and the fuzzy logic rules to predict the likelihood of an earthquake are finally applied. Time series values are transformed to phase space using a nonlinear method and then fuzzy logic is applied to predict the optimum value as shown in Fig. 4.45.

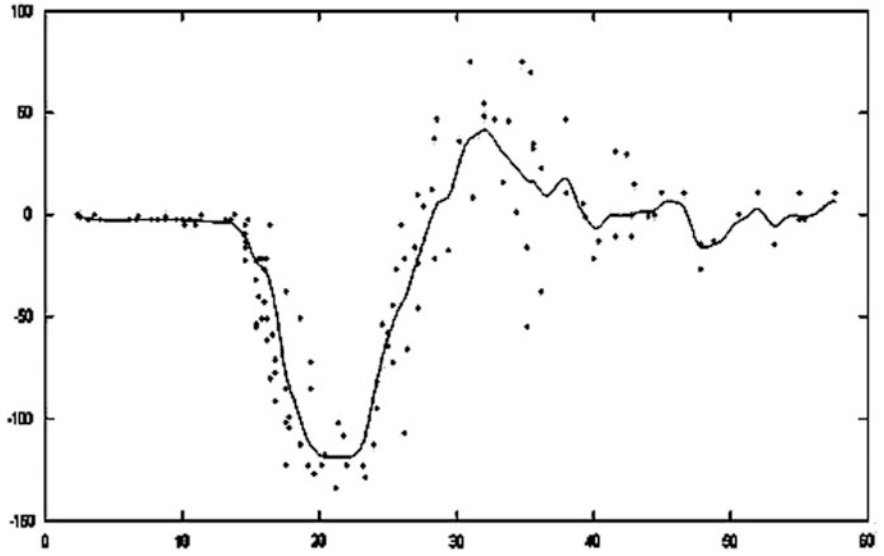


Fig. 4.45 Data points and predictions (Dutta et al. 2013)

Fuzzy linear regression (FLR) model can be broadly classified into two approaches:

- Linear programming (LP) based methods.
- Fuzzy Least Squares (FLS) methods.

The FLR model was proposed by Tanaka et al. (1982) through the parametric models below:

$$Y = A_0 + A_1X_1 + \cdots + A_pX_p \tag{4.62}$$

where $A_i = (a_{ic}, a_{iw})$, $Y = (y_c, y_w)$.

The parameters are estimated by minimizing the vagueness of model-data combinations subject to constraints that each data point must lie within an estimated value of the response variable. This can be visualized as a LP problem and solved by using the "simplex procedure" as shown in Fig. 4.46.

It was shown that the duration of prediction intervals in respect of fuzzy linear regression model were much less than those for multiple linear regression models. As the number of a data points increase the number of constraints in LP increases proportionally thereby resulting in computational difficulties.

A second method based on the Fuzzy Least Squares (FLS) method, was pioneered by Diamond (1988). This method is a fuzzy extension of the least squares method based on a new defined distance over the space of fuzzy numbers. The graphical output of the prediction is illustrated in Fig. 4.47 as the fuzzy surface diagram. As it can be seen in this figure, the earthquakes are described in this diagram where the level rules are as below:

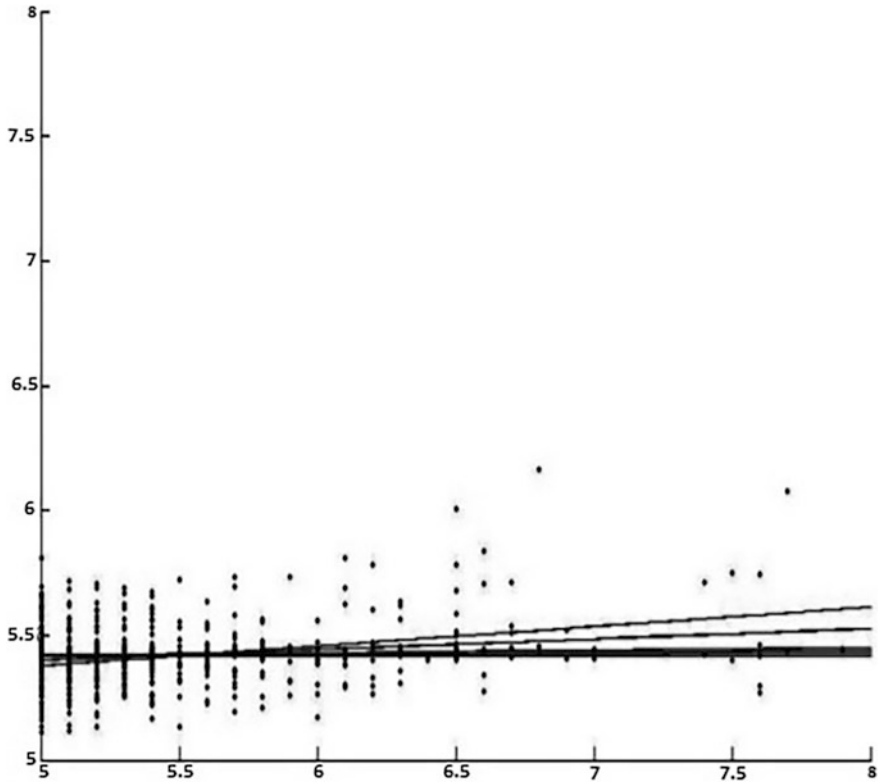


Fig. 4.46 Computation of predictions for test examples of quartiles for output of a pignistic distribution [In decision theory, a pignistic probability is a probability that a rational person will assign to an option when required to make a decision (Wikipedia). A pignistic probability transform will calculate these pignistic probabilities from a structure that describes belief structures (https://en.wikipedia.org/wiki/Pignistic_probability)] (Dutta et al. 2013)

- If M (Magnitude) is low and D_t (Depth) is shallow then the impact of earthquake is medium.
- If M is high and D_t is shallow then the impact of earthquake is high.
- If M is low and D_t is shallow, then the impact of the earthquake is medium.

The maximum impact of the earthquake is shown in the surface diagram. The fuzzy logic approach helps us to predict the maximum number of occurrences. The fuzzy if-then rules are generated with the assumption that magnitude (M), Depth (Dt) and impact (I) dependent on latitude and longitude are linguistic variables. The possible values for these linguistic variables are listed in Table 4.11.

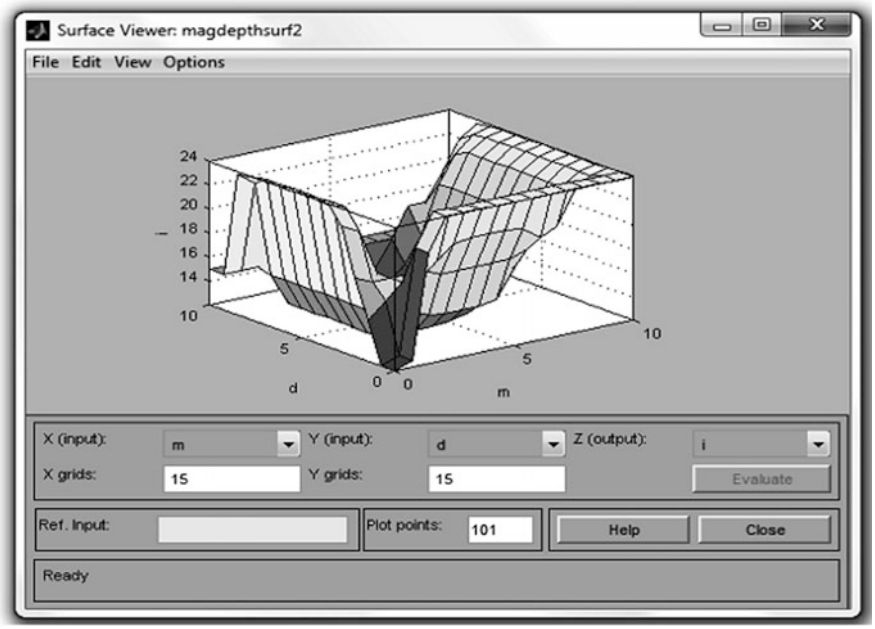


Fig. 4.47 Fuzzy rule surface for Impact of an earthquake

Table 4.11 Possible values for linguistic variable in (FLR) method

Earthquake specification	Related linguistic variable
Depth	Shallow (S), Deep (D)
Magnitude	Low (L), Medium (M), High (H)
Impact	Low (L), Medium (M), High (H)

The fuzzy knowledge base in FLR method is based on M fuzzy if-then rules of the following type:

Rule m: If M is A_{m1} and D_t is A_{mn} then I is B_m the rule weight is W_m; where $X = [X_1; X_2;\ldots;X_n]$ and y are, respectively, the input and the output values. A_{m1}, …,A_{mn} are linguistic labels or "or" combination of labels, Whose corresponding fuzzy sets are arranged in a press pacified fuzzy grid which is not changed during the learning process. The consequents B_m can be either singletons or fuzzy numbers. For example:
Rule 1: If M is low or medium or high and D_t is shallow or deep then I is low or medium with weight 0.8.

The rule base is interpreted according to the First Interference Then Aggregation (FITA) principle (Cornelis and Kerre 2003) as best way of preprocessing a dataset with a high degree of imprecision in the input.

Analyzing earthquake data with a regression analysis method based on fuzzy extension of belief function theory was investigated (Dutta et al. 2011). An example of the inferred relationship between the maximal magnitude of an expected earthquake M_{max} and its position was given using Weka and SPSS (Dutta et al. 2011). In this study, magnitude was taken as the dependent attribute, and the statistical and logical inference of the relationship between the latitude, longitude and focal depth was found to be an independent function.

Dutta et al. (2011) found that magnitude:

$$\text{Mr} = -0.0349^{*}\text{latitude} - 0.0145^{*}\text{longitude} + \text{focal depth} + 7.5245. \tag{4.63}$$

Trend and deviation analysis using regression techniques of the spatial characteristics of the earthquake parameters, for all of the seismicity occurring both before and after the largest events accumulates to a global structure consisting of a few separate clusters in feature space.

In Dutta et al. (2011) the earthquake size estimates in South Asia coming from various sources and consisting of different magnitude types and intensity data were plotted against unified magnitude Data association and their relationships between attributes yields a significant order based on weka and spss tools.

In Figs. 4.48 and 4.49 the line extraction using the weighted least square method is depicted.

Weighted least squares regression actually increases the influence of an outlier and so the results may be far inferior to an un-weighted least squares method. The earthquake cycle is not periodic and the time between successive earthquakes can

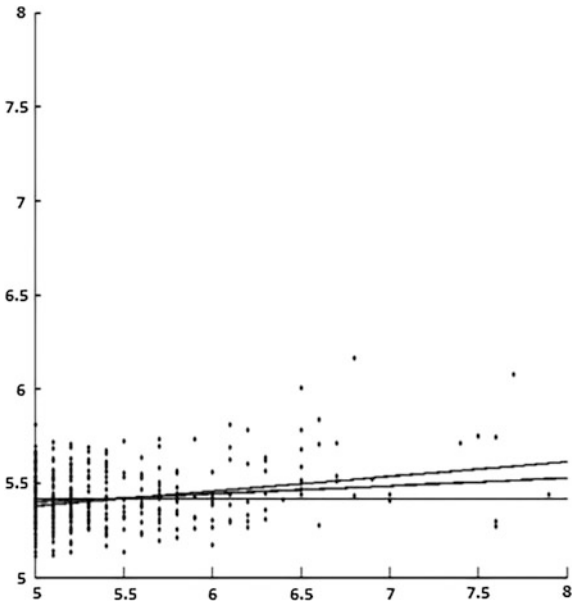


Fig. 4.48 Weighted line fit using Weka (Dutta et al. 2013)

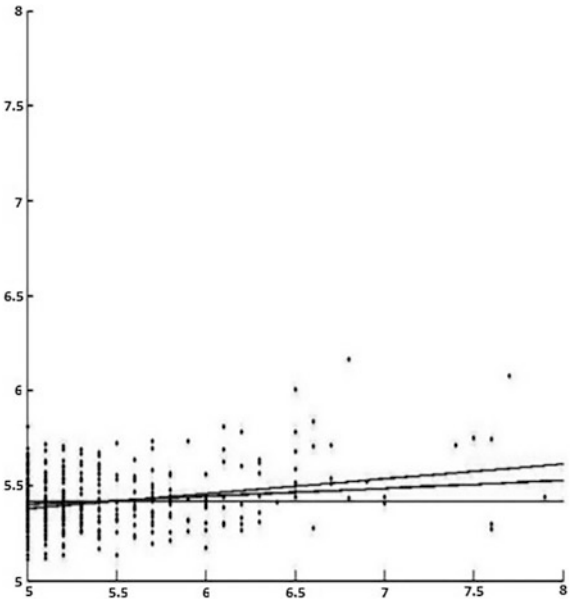


Fig. 4.49 Weighted line fit using SPSS (Dutta et al. 2013)

be highly irregular whereby fuzzy rule based definitions and weighted regression method play a significant role for time stamp definition of earthquake occurrence based on the relationship between magnitude, focal depth and impact based on location of the earthquake origin (Dutta et al. 2013).

Buckley and Feuring (2000) proposed "Evolutionary algorithm Solutions" for fitting some particular parametric Fuzzy Nonlinear models: FNMs. The FNM approach, for a given fuzzy dataset, searches from the "library" of fuzzy functions including linear, polynomial, exponential, logarithmic, etc. that function which best fits the data. A National Strong Motion Instrumentation Network (Chopra 2008) is being established in order to develop an early warning infrastructure for the Indian sub-continent. A great deal of research work is being put in by Indian scientists to develop indigenous solutions for earthquake forecasting and response in India.

References

Abdelrahman E.M., El-Araby T.M. and Abo-Ezz E. R., 2001. "Three least-squares minimization approach to depth, shape, and amplitude coefficient determination from gravity data", Geophysics 66, 1105–1109.

Aki K., V. Ferrazzini, Seismic monitoring and modelling of an active volcano for prediction, J. Geophys. Res. 105 (2000)16617–16640.

An P. , Moon W. M., Rencz A., 1991, Application of fuzzy set theory to integrated mineral exploration ,Canadian Journal of exploration geophysics, 27(1), 1-11.

Anderberg, M. R., Cluster Analysis for Applications, Academic Press, New York, 1973.

Asada, T. (1982). Earthquake Prediction Techniques. – Their application. Japan: University of Tokyo Press. 317.

Boulanger O. and Chouteau M., 2001, "Constraints in 3D gravity inversion", Geophysical Prospecting 49, 265–280.

Buckley, J. J. & Feuring, T. (2000). Linear and Nonlinear Fuzzy Regression: Evolutionary Algorithm Solutions. Fuz. Sets Syst., 112:381–94.

Cagnoli B., 1998, Fuzzy logic in volcanology, Episodes, Journal of international geosciences, 21 (2), 94–96.

Chen, X., Vierling, L. and Deering, D., 2005. A simple and effective radiometric correction method to improve landscape change detection across sensors and across time. Remote Sensing of Environment, 98 (1), 63–79.

Chopra, S., Yadav, R. B. S., Patel, H., Kumar, S., Rao K. M., Rastogi, B. K., Hameed A., & Srivastava, S. (2008). The Gujarat (India) Seismic Network," Seismological Research Letters, 79(6): 806–815.

Cornelis, C., & Kerre, E. (2003). A Fuzzy Inference Methodology Based on the fuzzification of Set Inclusion PhysicaVerlag. In: Recent Advances in Intelligent Paradigms and applications, 71–89.

Diamond, P. (1988). Fuzzy Least Squares. Inform. Sci., 46: 141-57.

Dekkers, M. J., C. G. Langereis, S. P. Vriend, P. J. M. van Santvoort, and J. de Lange, 1994 Fuzzy c-means cluster analysis of early diagenetic effects on natural remanent magnetization acquisition in a 1.1 Myr piston core from the Central Mediterranean, Phys. Earth Planet. Inter., 85, 155–171.

Dmowska, R. (1997). Earthquake Prediction–State of the Art, Pageoph Topical Volumes Max Wyss Ed., Birkhäuser Verlag, Basel: Boston, USA.

Dragoni, M., and E. Boschi., 1992, Earthquake Prediction: Proceedings of the International School of Solid Earth Geophysics, 5th Course, Il Cigno Galileo Galilei Edizioni Di Arte E Scienza, Rome.

Dombi, J., 1990. Membership function as an evaluation, Fuzzy Sets and Systems. 35 (1), 1–21.

Dutta P. K., Mishra O. P. & Naskar M. K., 2013, Review of operational earthquake forecasting methodologies using linguistic fuzzy rule-based models from imprecise data with weighted regression approach, Journal of Sustainability Science and Management, 8(2), 220–235.

Dutta, Pushan, Naskar, M. K. & Mishra, O. P. (2011). South Asia Earthquake Catalog Magnitude Data Regression Analysis. International Journal of Statistics and Analysis, 1(2): 161–170.

Elawadi E., Salem A. and Ushijima K., 2001, "Detection of cavities and tunnels from gravity data using a neural network", Exploration Geophysics 32, 75–79.

Fisher, P.F., 2010. Remote sensing of land cover classes as type 2 fuzzy sets. Remote Sensing of Environment, 114 (2), 309–321.

Fisher, R.V., and Schmincke, H.-U., 1984, Pyroclastic rocks: Springer-Verlag, Berlin, 472.

Fung, T., 1990. An assessment of TM imagery for land cover change detection. IEEE Transactions on Geoscience and Remote Sensing, 28 (4), 681–684.

Fung, T., and Ledrew, E., 1988. The determination of optimal threshold levels for change detection using various accuracy indices. Photogrammetric Engineering and Remote Sensing, 54 (10), 1449–1454.

Geller, R. J. (1997). Earthquake Prediction: A Critical Review. Geophysics J. Int., 131: 425–450.

Geller, R. J., Jackson, D. D., Kagan, Y. Y. & Mulagria, F. (1997). Earthquake Cannot Be Predicted. Science, 275(5306): 1616–1617.

Grandjean G., Bitri A, Pennetier C., Meric O. & Malet J.-P. (2006a). Caractérisation de la structure interne et de l'état hydrique de glissements argilo-marneux par tomographie géophysique : l'exemple du glissement-coulée de Super-Sauze. – C. R. Acad. Sci., 338, 587–595.

Grandjean G., Malet J.-P., Bitri A., & Meric O., (2006b), Geophysical data fusion by fuzzy logic for imaging the mechanical behavior of mudslides, Bull. Soc. géol. Fr., 2006, t. 177, no 2, 127–136.

Grêt A.A., E.E. Klingelé, H.-G. Kahle, 2000, Application of artificial neural networks for gravity interpretation in two dimensions: a test study, Bollettino Digeofisicateorica Edapplicata, 41(1), 1–20.

Guha, S. K. & Patwardhan, A. M. (1985). Earthquake Prediction: Present Status: Proceedings of Symposium Held at the Department of Geology, University of Poona. Pune: University of Poona Pub.

Gupta O.P., 1983, "A least-squares approach to depth determination from gravity data", Geophysics 48, 357–360.

Gvishiani A.D., V.O. Mikhailov, S.M. Agayan, Sh.R. Bogoutdinov, S.A. Tikhotski, M. Diament, A. Galdiano, Artificial Intelligence Techniques in Potential Fields and Other Geophysical Studies, vol. 20, Cahiers de Centre Europeen de Geodynamique et de Seismologie, Luxembourg, 2003, 63–69.

Hajian A. and Styles P., 2012, "Shape factor and depth estimation of microgravity anomalies via combination of artificial neural networks and fuzzy rule based system-", 6th International Conference on Fuzzy Information and Engineering, Mazandaran University, IRAN, Oct. 25–26.

Hajian A., R. Kimiaeefar, H.R. Siahkoohi, 2016, (Random Noise Attenuation in Reflection Seismic Data Using Adaptive Neuro-fuzzy Interference System (ANFIS), 78th European Association of Geoscientists and Engineers, Vienna, Austria, EAGE 2016.

Hame, T.H., 1986. Satellite image aided change detection. In: Remote sensing-aided forest inventory. Department of Forest Mensuration and Management, University of Helsinki, Helsinki, Finland. https://en.wikipedia.org/wiki/Pignistic_probability

Jamshidi, M., Vadiee, N., and Ross, T.J., 1993, Fuzzy logic and control, software and hardware applications: Prentice Hall, Englewood Cliffs, 397.

Jantzen J., 2004, A tutorial on adaptive fuzzy control, Journal of Technical University of Denmark, Dinamarca, in: http://www.eunite.org/

Jensen, J.R., and Toll, D.L., 1982, Detecting residential land use development at the urban fringe. Photogramm. Eng. Remote Sens., 48 (4), 629-643.

Jensen, J.R. and Toll, D.L., 1983. Detecting residential land use development at the urban fringe. Photogrammetric Engineering and Remote Sensing, 48 (4), 629–643.

Kisslinger, C. (1986). Practical Approaches to Earthquake Prediction and Warning.

Klir, G. J. and T. A. Folger, Fuzzy Sets, Uncertainty and Information, Prentice hall of India Private Ltd., New Delhi, 2000.

Kruiver, P. P., Y. S. Kok, M. J. Dekkers, C. G. Langereis, and C. Laj, A pseudo-Thellier relative palaeo intensity record, and rockmagnetic and geochemical parameters in relationship to climate during the last 276 kys in the Azores region, Geophys. J. Int., 136(3), 757–770, 1999.

Li X. and Chouteau M., 1998, "Three-dimensional gravity modeling in all space", Surveys in Geophysics 19, 339–368.

Li Y. and Oldenburg D.W., 1998, "3-D inversion of gravity data", Geophysics 63, 109–119.

Lu, D., Mausel, P., Brondizio, E. and Moran, E., 2004. Change detection techniques. International Journal of Remote Sensing, 25 (12), 2365–2401.

Lu, D., Mausel, Batistella, M., and Moran, E., 2005, Land-cover binary change detection methods for use in the moist tropical region of the Amazon: a comparative study. Int. J. Remote Sens. 26 (1), 101-114. https://doi.org/10.1080/01431160410001720748.

Madanian M., Soffianian A., Hajian A., 2014, "Change detection through four techniques using multi-temporal Landsat Thematic Mapper data: a case study on Falavarjan area, Isfahan, Iran", Journal of Environmental Informatics, 23(2), 58–66.

Metternicht, G., 1999. Change detection assessment using fuzzy sets and remotely sensed data: an application of topographic map revision. ISPRS Journal of Photogrammetry and Remote Sensing, 54, 221–233.

MINERAL EXPLORATION, Canadian journal of exploration geophysics, 27(1), p 1–11.

Mishra, O. P. (2012). Seismological Research in India. Proceeds. Ind. Nat. Sci. Acad. Publication (PINSA), 76(3): 361–375.

Mohan N.L., Anandadabu L. and Roa S., 1986, "Gravity interpretation using Mellin transform", Geophysics 52, 114–122.

Moon M.W., An P., 1991, Integration of Geophysical, Geological and Remote Sensing Data Using Fuzzy Set Theory. Geoinformatics, 2(2),p 171–176.
Nelson, S.A.C., Sorrano, P.A. and Qi, J., 2003. Land cover change in the Upper Barataria Basin Estuary, Louisiana, from 1972–1992: increases in wetland area. Environmental Management, 29 (5), 716–727.
Netherlands: Kluwer Academic Pub., Amsterdam.
Newhall, C.G., and Self, S., 1982, The volcanic explosivity index (VEI): an estimate of explosive magnitude for historical volcanism: Journal of Geophysical Research, 87, 1231–1238.
NIFLE A. & REYNAUDR., 2000, Double representation of information and hybrid combination for identification systems. –Proc. 3rd Int. Conf. on Information Fusion, Paris France, Vol. 1, 23–30.
Nyquist J.E., C.E. Corry, Self-potential: the ugly duckling of environmental geophysics, Lead. Edge (2002) 446–451.
Odegard M. E. and Berg J. W., 1965, "Gravity interpretation using the Fourier integral", Geophysics 30, 424–438.
Osman O., Albora A.M. and Ucan O. N., 2007, "Forward Modeling with Forced Neural Networks for Gravity Anomaly Profile", Mathematical Geology 39, 593–605. https://doi.org/10.1007/s11004-007-9114-8.
Pilon, P.G., Howarth, P.J., Bullock, R.A. and Adeniyi, P.O., 1988. An enhanced classification approach to change detection in semi-arid environments. Photogrammetric Engineering and Remote Sensing, 54 (12), 1709–1716.
Reid A.B., Allsop J.M., Granser H., Millet A.J. and Somerton I.W., 1990, "Magnetic interpretation in three dimensions using Euler Deconvolution", Geophysics 55, 80–91.
Ridd, M.K. and Liu, J., 1998. A comparison of four algorithms for change detection in an urban environment. Remote Sensing of Environment, 63, 95–100.
Salem A., Elawadi E., Abdelaziz A. and Ushijima K., 2003, "Imaging subsurface cavities from microgravity data using Hopfield neural network", Proceeding of the 7th International Symposium on Recent Advances in Exploration Geophysics in Kyoto, RAEG2003, 199–205.
Serra, P., Pons, X. and Sauri, D., 2008. Land-cover and land-use change in a Mediterranean landscape: a spatial analysis of driving forces integrating biophysical and human factors. Applied Geography, 28 (3), 189–209.
Sharma B. and Geldrat L.P., 1968, "Analysis of gravity anomalies of two-dimensional faults using Fourier transforms", Geophysical Prospecting 16, 77–93.
Shaw R.K. and Agarwal P., 1990, "The application of Walsh transforms to interpret gravity anomalies due to some simple geometrical shaped causative sources: A feasibility study", Geophysics 55, 843–850.
Shih-jung, M. (1993). Introduction to Earthquake Prediction in China. she Pub. Pei-ching, China: Ti chen chiu pan.
Shimazaki, K. & Stuart, W. (1985). Earthquake Prediction. Basel: Boston, USA: Birkhäuser Pub.
Singh, A., 1989. Digital change detection techniques using remotely sensed data. International Journal of Remote Sensing, 10 (6), 989–1003.
Singh, v., Moon. WM. and Fedikow. M., 1989. Investigation of Airborne MEIS-II and MSS data for biogeochemical exploration of mineralized zones, Farley Lake, Manitoba: Canadian Journal of Remote sensing 15, 122.133.
Smith R.A., 1959, Some depth formulate for local magnetic and gravity anomalies. Geophysical Prospecting 7, 55–63.
Sridharan M., 2009, Fuzzy mathematical model for the analysis of geomagnetic field data, Earth Planets Space, 61, 1169–1177.
Staudacher, La Fournaise. Smithonian institution, Bull. Glob. Volcanism Netw. 23-3 (1998) 2–4.
Tanaka, H., Uejima, S., and Asia, K., 1982, Linear Regression Analysis with Fuzzy Model,IEEE Trans. Systems Man. Cybernet., 12, 903-907.
Thompson D.T., 1982, EULDPH-A new technique for making computer-assisted depth estimations from magnetic data, Geophysics 47, 31–37.

Tso, B. and Mather, P.M., 2009. Classification methods for remotely sensed data. London: Taylor & Fransis.

Uyeda S., Introduction to the VAN method of earthquake prediction, in: J. Lighthill (Ed.), A Critical Review of VAN, World Scientific, Singapore, 1996, 3–28.

Vogel, A., and Itsikara, A. M., 1980, Multidisciplinary Approach to Earthquake Prediction: International Symposium on Earthquake Prediction in the North Anatolian Fault Zone (1980: Istanbul, Turkey), proceedings of the International Symposium on Earthquake Prediction in the North Anatolian Fault Zone, held in Istanbul, March 31-April 5, 1980. F. Vieweg Pub., Braunschweig, Germany, 1982.

Wulder, M.A., White, J.C., Goward, S.N., Masek, J.G., Irons, J.R. and Herold, M., et al. 2008. Landsat continuity: Issues and opportunities for land cover monitoring. Remote Sensing of Environment, 112 (3), 955–969.

Zadeh, L. A., 1965, Fuzzy sets, Inform. Control, 8, 338–353.

Zimmermann H. and Zysno P., 1980, Latent connectives, human decision making, Fuzzy Sets and Systems, 4, 37-51.

Zlotnicki J., MouJl J L L., Gvishiani A., Agayan S., Mikhailov V., Bogoutdinov S., Kanwar R., Yvetot P., 2005, Automatic fuzzy-logic recognition of anomalous activity on long geophysical records: Application to electric signals associated with the volcanic activity of La Fournaise volcano (Réunion Island)Earth and Planetary Science Letters 234, 261–278.

Zlotnicki J., Y. Nishida, Review on morphological insights of self-potential anomalies on volcanoes, Surv. Geophys. 24(2003) 291–338.

Part III
Combination of Neural Networks and Fuzzy Logic

Chapter 5
Neuro-fuzzy Systems

– Principles of ANFIS
– Design and Test of ANFIS using Matlab

5.1 Hybrid Systems

5.1.1 Introduction

A **hybrid intelligent system** involves combining two intelligent technologies; e.g., a combination of a neural network with a fuzzy system to produce a hybrid neuro-fuzzy system. Generally combining probabilistic reasoning, fuzzy logic, evolutionary computation together with neural networks produces hybrid systems which form the core of **soft computing.**

Soft Computing is an emerging approach to building hybrid intelligent systems, which can implement reasoning and learning processes within uncertain and imprecise environments (Hajian et al. 2012). As we described in Chap. 2 one of the imprecise environments that our brains works with in everyday life, is verbal reasoning with words or linguistic variables. Although words are intrinsically less precise than numbers, this inherent precision comes with a high computational price tag. When we can tolerate a degree of imprecision we can use words which can also have nuance which is difficult to deliver using mathematics.

We also use words when the data which are available cannot be defined precisely enough to be described numerically. Soft computing trades off this tolerance for uncertainty and imprecision against greater tractability and robustness while simultaneously lowering the cost of computation.

A hybrid intelligent system is neither intrinsically good or bad; this will depend on the selection of the components which constitute the hybrid. So the main challenge is in the selection of the optimal components for building a good hybrid

A. Hajian and P. Styles, *Application of Soft Computing and Intelligent Methods in Geophysics*, Springer Geophysics,
https://doi.org/10.1007/978-3-319-66532-0_5

system to achieve the desired outcomes, i.e. intelligent interpretations in geophysics (Hajian 2010).

Artificial intelligence (based on fuzzy logic) and neural networks are frequently combined in order to minimise the inherent difficulties and limitations of each paradigm when used individually and when they are used synergistically, they are called Neuro-fuzzy Systems.

These systems are characterised by fuzzy sets and fuzzy rules which are adjusted using input/output patterns. This is still a juvenile discipline and there are many different implementations of neuro-fuzzy systems, espoused by individual authors and their models. This section, based on the review article by Vieira et al. (2004), describes the most commonly used hybrid neuro-fuzzy techniques, and considers their advantages and disadvantages.

While the techniques of artificial intelligence have applications in almost all fields of human knowledge, perhaps the biggest success of these techniques is found in the engineering field. Neural networks and fuzzy logic are often applied in combination to solve engineering problems where classic techniques cannot deliver simple and accurate solutions, for example in noise attention of seismic data (Hajian et al. 2016).

Neuro-fuzzy reasoning fuses these two techniques, but as each researcher combines these in a different way, there is confusion and no consensus about its exact meaning. However, generally the neuro-fuzzy term indicates a system characterized as having a fuzzy controller where the fuzzy sets and rules are tuned using neural network techniques iteratively utilising input and output system data vectors.

Systems like this show two distinct modes of behaviour.

There are two main stages in how a neuro-fuzzy system works:

- ***A Learning phase****: where the system behaves like a neural network which learns its internal parameters off-line.*
- ***Execution phase****: it behaves like a fuzzy logic system.*

Each of these techniques has advantages and disadvantages which, when combined, provide better results than that achieved with the use of each technique individually.

Since fuzzy systems have become popular in industrial applications, it has become apparent that developing membership functions and appropriate rules is often a lengthy process of trial and error. In order to streamline this task, neural networks, that have efficient learning algorithms, have been suggested as alternative routes to the automation of the tuning of fuzzy systems.

Studies of neuro-fuzzy systems began in the 90s, with Lin and Lee (1991), Jang (1992), Berenji (1992) and Nauck (1994). The first new applications were in process control but these applications gradually spread into data analysis, classification, imperfection detection and support to decision-making (Vieira et al. 2004).

Neural networks introduce the computational characteristics of learning into fuzzy systems while benefitting from the interpretation and clarity of system representation. The disadvantages of the fuzzy systems are ameliorated by the

capabilities of neural models and these have been implemented successfully in geophysical data interpretation, for example in Ground Penetrating Radar (GPR) interpretation (Hajian and Fazelian 2016). The different combinations of neural networks and fuzzy logic techniques can be divided into the following classes:

5.1.2 Cooperative Neuro-fuzzy Systems

Cooperative systems have a pre-processing phase where the neural network learning mechanisms define some sub-blocks of the fuzzy system (Fig. 5.1). For instance, we can utilise fuzzy sets and/or fuzzy rules or clustering algorithms to define the rules for the fuzzy sets. Once these fuzzy sub-blocks are calculated, the neural network learning methods are removed and we only execute the fuzzy system.

In a cooperative system, neural networks are only used to determine the sub-blocks of the fuzzy system using the training data and are then removed and only the fuzzy system is executed but the fact that the structure is not totally integrated may be disadvantageous.

5.1.3 Concurrent Neuro-fuzzy Systems

Concurrent systems differ in that the neural network and the fuzzy system continuously work together (Fig. 5.2) with the neural networks pre-processing the inputs (or post-processing the outputs) of the fuzzy system. A concurrent system is not a neuro-fuzzy system sensu stricto, as the inputs of the fuzzy system, are pre-processed while the neural network processes the outputs and vice versa, and therefore, the results are not completely interpretable.

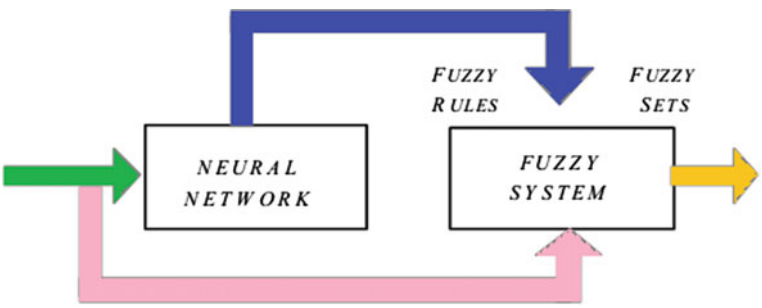


Fig. 5.1 Schematic diagram of a cooperative neuro-fuzzy system

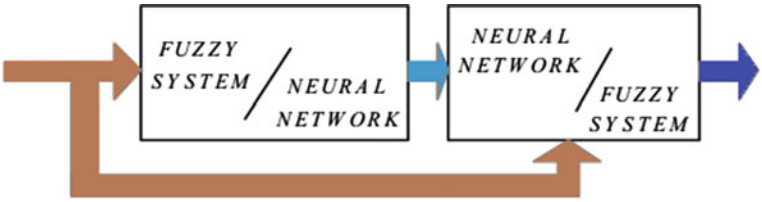


Fig. 5.2 Schematic diagram of a concurrent system

5.1.4 Hybrid Neuro-fuzzy Systems

Nauck (1994) defined these thus:

> A hybrid neuro-fuzzy system is a fuzzy system that uses a learning algorithm based on gradients or inspired by the neural networks theory (heuristical learning strategies) to determine its parameters (fuzzy sets and fuzzy rules) through pattern processing (input and output).

In this category, a neural network learns some of the parameters of the fuzzy system (fuzzy rules and weights of the rules) in an iterative way and most researchers confine the use of the term neuro-fuzzy for hybrid neuro-fuzzy systems. A neuro-fuzzy system can be interpreted as a set of fuzzy rules created solely from input/output data or initialised with prior knowledge. The fusing of fuzzy systems and neural networks has the advantage of learning through patterns and the ease of interpretation of its functionality. Again, in this juvenile research subject, each researcher has defined their own models, similar in essence, but with certain differences.

Many neuro-fuzzy systems constitute neural networks that implement logical functions but this is not a firm rule for integrating learning algorithms into fuzzy systems. However, the representation through a neural network does facilitate the visualisation of the flow of data through the system and the error signals which are used to update the parameters while allowing the comparison of different models and the visualisation of their structural differences (Vieira et al. 2004).

Several neuro-fuzzy architectures are listed in Table 5.1.

In this chapter we have mainly focused on neuro-fuzzy systems, and a brief summary of Expert Systems, Fuzzy Systems, Neural Networks and Genetic Algorithms is listed in Table 5.2. The genetic algorithm (GA) will be discussed in detail in Chap. 7 but for a comprehensive comparison the GA is also compared here to other intelligent methods. This is a good guidance table, which should enable someone to select a method from various intelligent alternatives, and furthermore to know which intelligent methods can be combined in a complementary manner with other methods. For example, the explanatory capability of neural networks is bad while it is good for fuzzy systems; this means that the weakness of explanation ability in neural-networks model can be solved through a combination with fuzzy models. The reason for this weakness of neural networks is their black-box structure

Table 5.1 Various types of hybrid-neuro-fuzzy systems

Neuro-fuzzy architect	Abbreviation	Developer(s)	Year
Fuzzy adaptive learning control network	FALCON	C. T. Lin and C. S. Lee	(1991)
Adaptive network based fuzzy inference system	ANFIS	R. R. Jang	(1992)
Generalized approximate reasoning based intelligence control	GARIC	H. Berenji	(1992)
Neuronal fuzzy controller	NEFCON	D. Nauck & Kruse	(1994)
Fuzzy inference and neural network in fuzzy inference software	FINEST	Tano, Oyama and Arnould	(1996)
Fuzzy net	FUN	S. Sulzberger, N. Tschichold and S. Vestli	(1993)
Self-constructing neural fuzzy inference network	SONFIN	Juang and Lin	(1998)
Fuzzy neural network	NFN	Figueiredo and Gomide	(1999)
Dynamic/evolving fuzzy neural network	EFNN and DEFNN	Kasabov and Song	(1999)

Table 5.2 Comparison of expert systems (ES), Fuzzy System (FS), Genetic Algorithm (GA) and Neural Networks (NN)

	ES	*FS*	*NN*	*GA*
Knowledge representation	☑	☑	☐	☐
Uncertainty tolerance	☑	☑	☑	☑
Imprecision tolerance	☐	☑	☑	☑
Adaptability	☐	☐	☑	☑
Learning ability	☐	☐	☑	☑
Explanation ability	☑	☑	☐	☐
Knowledge discovery and data mining	☐	☐	☑	☑
Maintainability	☐	☑	☑	☑

* The terms used for grading are:

☐- bad, ☐- rather bad, ☑- rather good and ☑- good

Source http://www.computing.surrey.ac.uk/teaching/2006-07/cs364/lecturenotes/week10_Hybrid_Intelligent_Systems.ppt

in which the user is not able to know how the inputs are affecting the outputs or on the other hand the rules of the model that represents the relation between input(/s) and outputs(/s) are not transparent. Combining a neural network system with a fuzzy system overcomes this weakness as the fuzzy system's learning ability is bad while the learning ability for the neural networks model is very good. A neuro-fuzzy model (fuzzy system with a neural network), helps **the fuzzy if-then rule** based model to learn from the training data.

5.2 Neural Expert Systems

- Expert systems use logical inferences and decision trees to model human reasoning and treat the brain as a black-box.
- Neural networks utilise parallel data processing to focus on modelling a human brain through looking at its structure and functions, particularly at its ability to learn.
- Expert systems Neural networks.
- A rule-based expert system represents knowledge by IF-THEN rules while knowledge in neural networks is simply stored as a set of synaptic weights applied between neurons.

In expert systems, knowledge is defined by individual rules and a user can see and understand each quantum of knowledge while a neural network appears as a **black-box** to its user. We can combine the advantages of expert systems and neural networks in order to develop powerful and efficient expert systems. A hybrid system combining a neural network with a rule-based expert system is called a neural expert system (or a connectionist expert system) and the basic structure of a **neural expert system** is illustrated in Fig. 5.3.

5.2.1 The Inference Engine

The core of a neural expert system is the *inference engine* which controls the information flow in the system and initiates inferences applied over the neural knowledge base and delivers approximate *reasoning*.

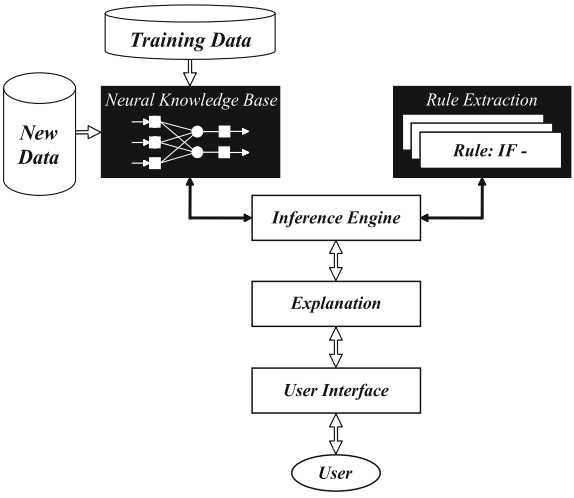


Fig. 5.3 Basic structure of a neural expert system (http://www.computing.surrey.ac.uk/teaching/2006-07/cs364/lecturenotes/week10_Hybrid_Intelligent_Systems.ppt)

5.2.2 Approximate Reasoning

In rule-based expert systems, the inference engine compares the conditional part of each rule with the data in the database. When the "IF" part of the rule matches the data, the rule is satisfied and the "THEN" part executed. An exact match is required at this stage as inference engines cannot cope with noisy or incomplete data.

Neural expert systems use a trained neural network instead of the knowledge base and the input data need not exactly match the data used in network training and so this capability is called *approximate reasoning*.

5.2.3 Rule Extraction

Neurons in a network connect through links, which each have a numerical weight which determines the strength or importance of the associated neuron inputs.

5.2.4 The Neural Knowledge Base

We start this part with a famous example of trying to design a model, which can distinguish a flying object: is it a Bird, a Plane or a Glider?

The input parameters are the scores for having wings; tail, Beak, feathers and engine. The outputs are the object type estimation among bird, plane and glider. The main problem is how the system distinguishes a bird from an airplane?

If each input of the input layer is set the value of either +1 (true), −1 (false), or 0 (unknown), we can apply the rules and deduce the end result for any output neuron (Fig. 5.4).

Here three rules are used; rule one for distinguishing a bird, rule 2 for distinguishing a plane and rule 3 for distinguishing a glider.

For example, if the object has *Wings* (+1), *Beak* (+1) and *Feathers* (+1), but does not have *Engine* (−1), then by applying the rules we are forced to the conclusion that this is a *Bird* (+1):

$$X_{Rule1} = 1 \cdot (-0.8) + 0 \cdot (-0.2) + 1 \cdot 2.2 + 1 \cdot 2.8 + (-1) \cdot (-1.1) = 5.3 > 0 \quad (5.1)$$

$$Y_{Rule1} = Y_{Bird} = +1$$

Similarly we can conclude that this object is not a *Plane*:

$$\begin{aligned} X_{Rule2} &= 1 \cdot (-0.7) + 0 \cdot (-0.1) + 1 \cdot 0.0 + 1 \cdot (-1.6) + (-1) \cdot 1.9 = -4.2 < 0 \\ Y_{Rule2} &= Y_{Plane} = -1 \end{aligned} \quad (5.2)$$

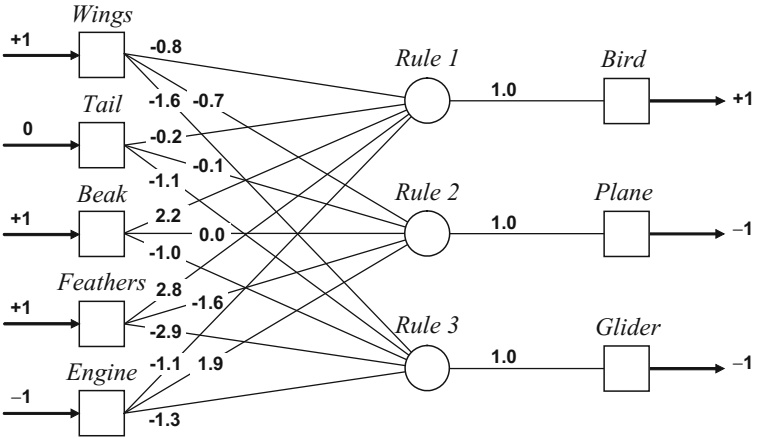


Fig. 5.4 The structure of a neural-knowledge base to distinguish flying objects among birds, planes and gliders (http://www.computing.surrey.ac.uk/teaching/2006-07/cs364/lecturenotes/week10_Hybrid_Intelligent_Systems.ppt)

And not glider:

$$X_{Rule\,3} = 1 \cdot (-0.6) + 0 \cdot (-1.1) + 1 \cdot (-1.0) + 1 \cdot (-2.9) + (-1) \cdot (-1.3) = -4.2 < 0$$
$$Y_{Rule\,3} = Y_{Glider} = -1 \tag{5.3}$$

By attaching a corresponding question to each input neuron, we can enable the system to prompt the user for initial values of the input variables:

Neuron#1: ***Wings***

Question: Does the object have wings?

Neuron#2: ***Tail***

Question: Does the object have a tail?

Neuron#3: ***Beak***

Question: Does the object have a beak?

Neuron#4: ***Feathers***

Question: Does the object have feathers?

Neuron#5: ***Engine***

Question: Does the object have an engine?

An inference can be drawn if the known net weighted input to a neuron exceeds the sum of the absolute values of the weights of the unknown inputs.

$$\sum_{i=1}^{n} x_i w_i > \sum_{j=1}^{n} |w_j| \tag{5.4}$$

where i ∈ known, $j \notin$ known and n is the number of neuron inputs.

For instance if *the initial value for the input Feathers is* +1 then:

KNOWN = 1.2.8 = 2.8
UNKNOWN = |−0.8| + |−0.2| + |2.2| + |−1.1| = 4.3
KNOWN < UNKNOWN

And if also the initial value for the input Beak is +1 then:

KNOWN = 1.2.8 + 1.2.2 = 5.0
UNKNOWN = |−0.8| + |−0.2| + |−1.1| = 2.1
KNOWN > UNKNOWN

Then we can categorically conclude: Bird is TRUE for this object.

5.2.5 *Multi-layer Knowledge Base*

A multi-layer knowledge base consists of at least 5 layers:

Layer 1: Inputs
Layer 2: Conjunction layer
Layer 3: Disjunction Layer
Layer 4: Conjunction Layer
Layer 5: Disjunction Layer

A multi-layer knowledge base with 5 inputs, two outputs and 8 rules is depicted in Fig. 5.5.

5.3 Neuro-fuzzy Systems

Fuzzy logic and neural networks are complementary tools, which can be used to build intelligent systems. Neural networks are low-level computational structures that deal well with raw data, while fuzzy logic reasons on a higher level, and uses linguistic information supplied by domain experts.

While fuzzy systems cannot learn or adjust to a new environment, neural networks are able to learn, but are opaque to the user.

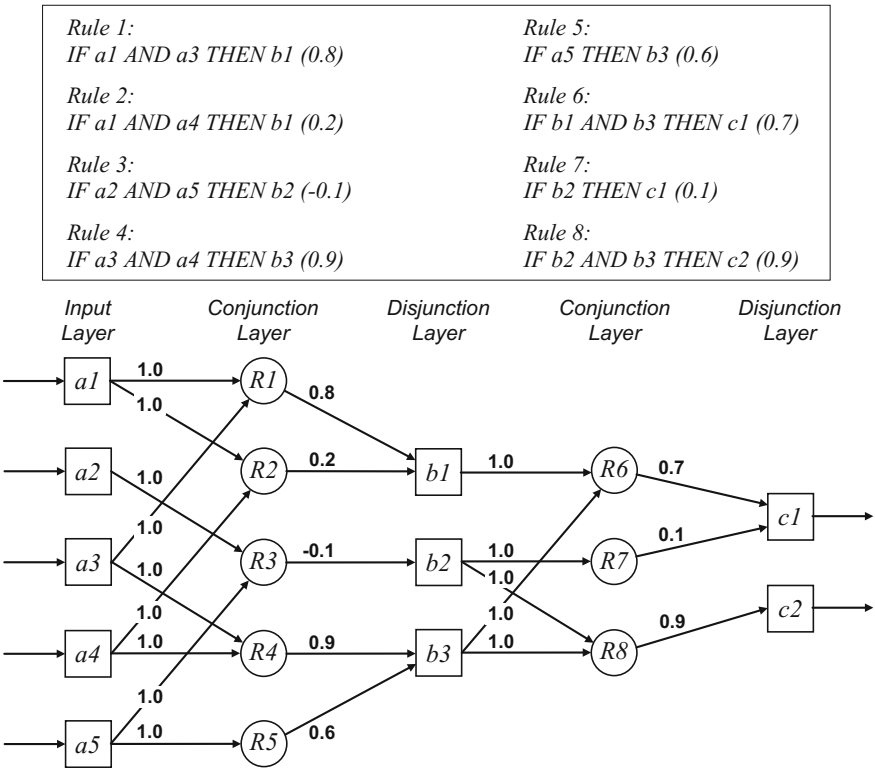


Fig. 5.5 A 5-layer knowledge base model with 5 inputs, 8 rules and 2 outputs (http://www.computing.surrey.ac.uk/teaching/2006-07/cs364/lecturenotes/week10_Hybrid_Intelligent_Systems.ppt)

5.3.1 Synergy of Neural and Fuzzy Systems

Integrated neuro-fuzzy systems merge the parallel computational and learning abilities of neural networks with the human-like knowledge representation and comprehension capabilities of fuzzy systems with the result that neural networks may become more transparent, while fuzzy systems develop a measure of learning ability. If a representative set of examples is available, a neuro-fuzzy system is able to automatically develop a robust set of fuzzy IF-THEN rules, reducing the dependence of the process on expert knowledge when building intelligent systems.

A neuro-fuzzy system is functionally equivalent to a fuzzy inference model and can be trained to develop IF-THEN fuzzy rules and determine the membership functions for input and output variables of the system and in addition, expert knowledge can be incorporated its structure., The connectedness of the structure avoids fuzzy inference, significantly reducing the computational burden.

A neuro-fuzzy system has a similar structure to a multi-layer neural network with five layers: input and output layers, and three hidden layers representing membership functions and fuzzy rules. A schematic diagram of a sample of a neuro-fuzzy system, its layers and connections is shown in Fig. 5.6.

Each layer in the neuro-fuzzy system corresponds to a particular step in the fuzzy inference process.

***Layer* 1** is the **input layer** where each neuron transmits external crisp (precise) signals directly to the next layer. That is:

$$y_i^{(1)} = x_i^{(1)} \tag{5.5}$$

***Layer* 2** is the ***fuzzification* layer**.

- Neurons in this layer represent fuzzy sets used in the antecedents of fuzzy rules.
- A fuzzification neuron receives a crisp input and evaluates the degree to which this input belongs to that neuron's fuzzy set.
- The activation function of a membership neuron is then set to the function that specifies the neuron's fuzzy set.
- We use triangular sets, and the activation functions for the neurons in *Layer* 2 are correspondingly set to the **triangular membership functions**.

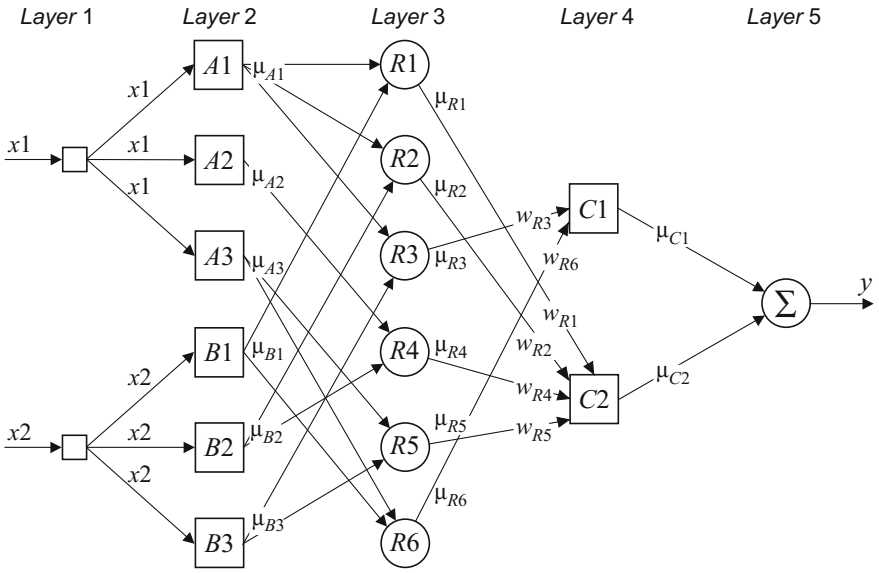


Fig. 5.6 Schematic of the structure of layers in a neuro-fuzzy system. *Source* http://www.computing.surrey.ac.uk/teaching/2006-07/cs364/lecturenotes/week10_Hybrid_Intelligent_Systems.ppt

- A triangular membership function is specified by two parameters $\{a, b\}$ as follows:

$$y_i^{(2)} = \begin{cases} 0, & \text{if } x_i^{(2)} \le a - \frac{b}{2} \\ 1 - \frac{2\left|x_i^{(2)} - a\right|}{b}, & \text{if } a - \frac{b}{2} < x_i^{(2)} < a + \frac{b}{2} \\ 0, & \text{if } x_i^{(2)} \ge a + \frac{b}{2} \end{cases} \tag{5.6}$$

The parameters a and b plays role on the shifting and band width of the triangular membership function as shown in Fig. 5.7a, b.

***Layer* 3** is the **fuzzy rule layer**.

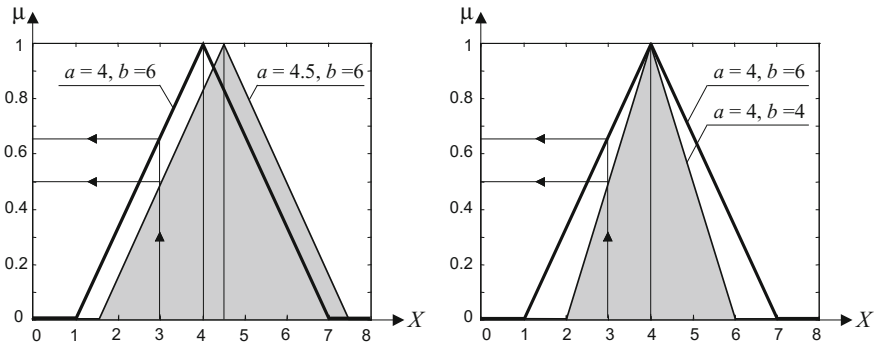
Each neuron in this layer corresponds to a single fuzzy rule and receives inputs from the fuzzification neurons corresponding to fuzzy sets in the rule antecedents. E.g., neuron $R1$, which corresponds to *Rule* 1, receives inputs from neurons $A1$ and $B1$ (see Fig. 5.6).

In neuro-fuzzy systems, intersection is represented by the **product operator and** therefore the output of neuron i in *Layer* 3 is obtained as:

$$y_i^{(3)} = x_{1i}^{(3)} \times x_{2i}^{(3)} \times \cdots \times x_{ki}^{(3)} \tag{5.7}$$

$$y_{R1}^{(3)} = \mu_{A1} \times \mu_{B1} = \mu_{R1} \tag{5.8}$$

***Layer* 4** is the **output membership layer**, which combines the consequences of all of the inputs using the fuzzy operation **union**. This operation can be implemented by the **probabilistic OR** i.e.,



(*a*) Effect of parameter *a*. (*b*) Effect of parameter *b*.

Fig. 5.7 Effetcts of parameters "a" and "b" on triangular MF (http://web.cecs.pdx.edu/~mperkows/CLASS_479/2013%20lectures/2012-1161.%20Neuro-Fuzzy%20Systems.ppt)

$$y_i^{(4)} = x_{1i}^{(4)} \oplus x_{2i}^{(4)} \oplus \cdots \oplus x_{li}^{(4)} \tag{5.9}$$

$$y_{C1}^{(4)} = \mu_{R3} \oplus \mu_{R6} = \mu_{C1} \tag{5.10}$$

where μ_{C1} is the integrated firing strength of fuzzy rule neurons $R3$ and $R6$.

***Layer* 5** is the **defuzzification layer** where neurons in this layer produces a single output of the neuro-fuzzy system by taking the output fuzzy sets, clipped by the respective integrated firing strengths, and combining them into a single fuzzy set.

Neuro-fuzzy systems can apply standard defuzzification methods, which include the centroid technique. For our purposes we use the **sum-product composition** method.

The sum-product composition calculates the crisp output to be the weighted average of the centroids of all of the output membership functions. For example, the weighted average of the centroids of the clipped fuzzy sets, $C1$ and $C2$ is:

$$y = \frac{\mu_{C1} \times a_{C1} \times b_{C1} + \mu_{C2} \times a_{C2} \times b_{C2}}{\mu_{C1} \times b_{C1} + \mu_{C2} \times b_{C2}} \tag{5.11}$$

5.3.2 *Training of a Neuro-fuzzy System*

The method used for its training lies at the heart of a neuro-fuzzy system and the important question is: how does a neuro-fuzzy system learn?

A neuro-fuzzy system is in essence a multi-layer neural network, and can therefore apply any of standard learning algorithms which have been developed for neural networks, including the back-propagation algorithm (Bohlooli et al. 2011).

When a training input–output example is presented to the neuro-fuzzy system, the back-propagation algorithm computes the system output and compares this with the desired output provide by the training example. The error between these is propagated backwards through the network from the output layer to the input layer and the neuron activation functions are modified as the error is communicated backwards. In order to determine the necessary perturbations, the back-propagation algorithm differentiates the activation functions of the neurons. We can show how a neuro-fuzzy system works with a simple example. Let us assume that the training patterns are (X1, X2, Y) as shown in Fig. 5.8, where Y is a function of X1, X2.

The data set used for training the five-rule neuro-fuzzy system is shown in Fig. 5.9a. The updating of the weighting rules during the training process versus for each of the first 50 epochs is shown in Fig. 5.9b.

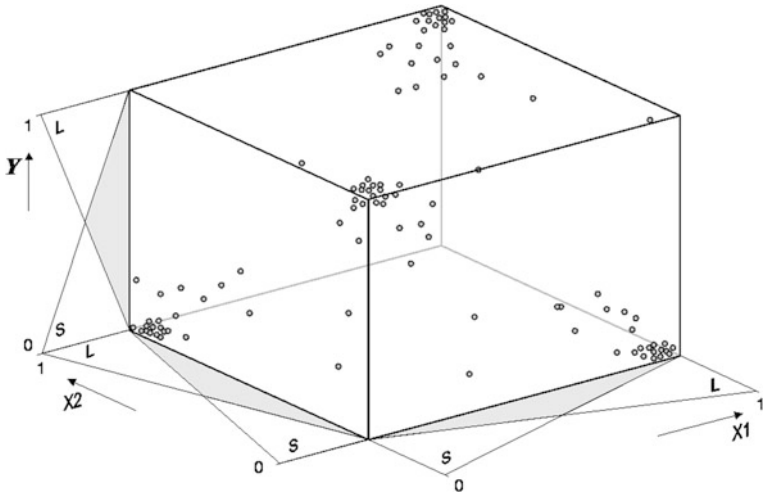


Fig. 5.8 Training space of data (http://web.cecs.pdx.edu/~mperkows/CLASS_479/2013%20lectures/2012-1161.%20Neuro-Fuzzy%20Systems.ppt)

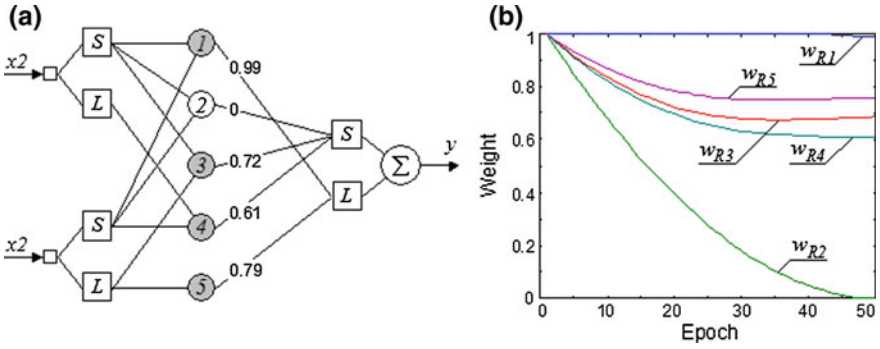
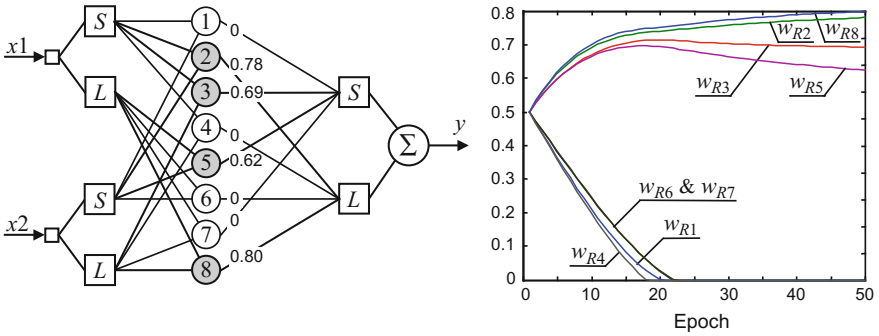


Fig. 5.9 **a** A five—rule system, **b** training for 50 epochs. *Source* http://web.cecs.pdx.edu/~mperkows/CLASS_479/2013%20lectures/2012-1161.%20Neuro-Fuzzy%20Systems.ppt

5.3.3 *Good and Bad Rules from Expert Systems*

Prior or existing knowledge can dramatically improve system training and if the quality of training data is poor, expert knowledge may be the only way a conclusion can be achieved. However, experts are not infallible, and some rules used in a neuro-fuzzy system may be false or redundant. Therefore, a neuro-fuzzy system should also be capable of identifying bad rules. Let us suppose that the fuzzy IF-THEN rules incorporated into the system structure are supplied by a domain expert.



(a) Eight-rule system. (b) Training for 50 epochs.

Fig. 5.10 Structure and training for an eight-rule neuro-fuzzy system. *Source* http://web.cecs.pdx.edu/~mperkows/CLASS_479/2013%20lectures/2012-1161.%20Neuro-Fuzzy%20Systems.ppt

Example: A neuro-fuzzy system for XOR

If it is given input and output linguistic values, a neuro-fuzzy system can automatically generate a complete set of fuzzy IF-THEN rules. As an example we will create the system for the XOR example.

This system consists of $2^2 \times 2 = 8$ rules. As no expert knowledge is embodied in the system initially, we set the initial weights between *Layer* 3 and *Layer* 4 to 0.5 but after training we can eliminate all rules whose certainty factors are less than some threshold, say 0.1. We obtain the same set of four fuzzy IF-THEN rules representing the XOR operation. This Eight-rule neuro-fuzzy system for XOR and its training for 50 epochs is illustrated in Fig. 5.10a, b respectively.

5.4 Adaptive Neuro-fuzzy Inference System: ANFIS

5.4.1 Structure of ANFIS

The Takagi-Sugeno (1985) fuzzy model generates fuzzy rules from a given input-output data set and a typical Sugeno fuzzy rule is expressed in the following form:

$$\begin{aligned} &\text{IF } x_1 \text{ is } A_1 \\ &\text{AND} \qquad\qquad\qquad\qquad x_2 \text{ is } A_2 \\ &\quad \ldots\ldots \\ &\text{AND} \qquad\qquad\qquad\qquad x_m \text{ is } A_m \\ &\text{THEN } y = f(x_1, x_2, \ldots, x_m) \end{aligned}$$

where x_1, x_2, …, x_m are input variables; A_1, A_2, …, A_m are fuzzy sets.

There are several orders of ANFIS:

- *Zero-Order ANFIS*
- *First-Order ANFIS*

When y is a constant, we obtain *a **zero-order Sugeno fuzzy*** model in which the consequent of a rule is specified by a singleton. When y is a first-order polynomial, i.e.

$$y = k_0 + k_1 x_1 + k_2 x_2 + \cdots + k_m x_m \tag{5.12}$$

We obtain a **first-order Sugeno fuzzy** model.

The general layer structure of an ANFIS is depicted in Fig. 5.11. Considering the inputs as a layer of ANFIS it has five layers:

***Layer* 1** is the **input layer** where Neurons pass external crisp signals to *Layer* 2.
***Layer* 2** is the **fuzzification laye**r where Neurons in perform fuzzification and have a **bell activation function** in Jang's model.
***Layer* 3** is the **rule layer** in which each neuron corresponds to a single Sugeno-type fuzzy rule. A rule neuron receives inputs from the respective fuzzification neurons and calculates the firing strength of the rule it represents. In an ANFIS, the end-result of the rule antecedents is given by the operator **product**.

In an ANFIS, the conjunction of the rule antecedents is represented by the operator **product** and, the output of neuron i in *Layer* 3 is obtained as:

$$y_i^{(3)} = \prod_{j=1}^{k} x_{ji}^{(3)} \tag{5.13}$$

$$y^{(3)} = \Pi_{\mu A1 \times \mu B1 = \mu_1} \tag{5.14}$$

where the value of μ_1 represents the firing strength, or the truth value, of *Rule* 1.

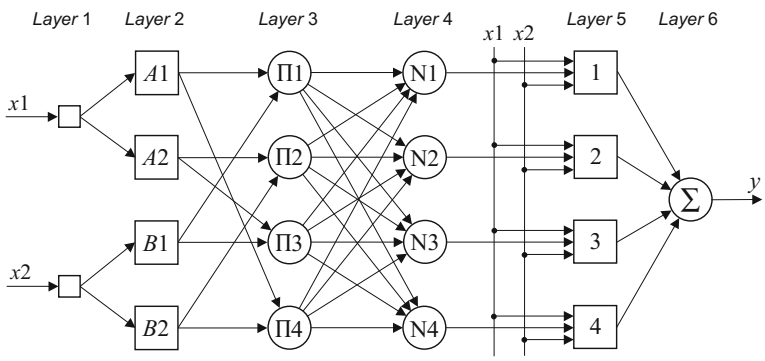


Fig. 5.11 Schematic diagram of the layers of an ANFIS (http://web.cecs.pdx.edu/~mperkows/CLASS_479/2013%20lectures/2012-1161.%20Neuro-Fuzzy%20Systems.ppt)

***Layer* 4** is the **normalisation layer** where each neuron receives inputs from all neurons in the rule layer, and calculates the **normalised firing strength** of a given rule.

The normalised firing strength is the ratio of the firing strength of any given rule to the sum of firing strengths of all rules and represents the contribution of a given rule to the final result. Thus, the output of neuron i in *Layer* 4 is determined as:

$$y_i^{(4)} = \frac{x_{ii}^{(4)}}{\sum_{j=1}^{n} x_{ji}^{(4)}} = \frac{\mu_i}{\sum_{j=1}^{n} \mu_j} = \bar{\mu}_i \tag{5.15}$$

$$y_{\mathrm{N1}}^{(4)} = \frac{\mu_1}{\mu_1 + \mu_2 + \mu_3 + \mu_4} = \bar{\mu}_1 \tag{5.16}$$

***Layer* 5** is the **defuzzification layer**. Where each neuron is connected to the respective normalisation neuron, and receives initial inputs, x_1 and x_2. A defuzzification neuron calculates the weighted consequent value of a given rule as:

$$y_i^{(5)} = x_i^{(5)} \ [k_{i0} + k_{i1} \ x1 + k_{i2} \ x2] = \bar{\mu}_i \ [k_{i0} + k_{i1} \ x1 + k_{i2} \ x2] \tag{5.17}$$

where $x_i^{(5)}$ is the input and $y_i^{(5)}$ is the output of defuzzification neuron i in *Layer* 5, and k_{i0}, k_{i1} and k_{i2} is a set of consequent parameters of rule i.

***Layer* 6** is represented by a single **summation neuron**.

This neuron calculates the sum of outputs of all defuzzification neurons and produces the overall ANFIS output, y as:

$$y = \sum_{i=1}^{n} x_i^{(6)} = \sum_{i=1}^{n} \bar{\mu}_i \ [k_{i0} + k_{i1} \ x1 + k_{i2} \ x2] \tag{5.18}$$

There is a frequent question that most students and/or researchers who are going to use ANFIS ask about such a question like this:

We might ask whether an ANFIS can deal with problems where we have no prior knowledge of the rule consequent parameters.

The answer is that:

- Any prior knowledge about parameters of rule consequent is not essentially needed.
- ANFIS itself can learn rule parameters and tune MFs.

So in the next section, we explain that how an ANFIS can learn from training data.

5.4.2 Learning in the ANFIS Model

An ANFIS is a hybrid-learning algorithm combining the least-squares estimator and the gradient descent method. In the ANFIS training algorithm, each epoch is composed of a forward pass in which, a training set of input patterns (an input vector) is presented to the ANFIS, neuron outputs are calculated on the layer-by-layer basis, and rule consequent parameters are identified and this is then followed by a backward pass.

The rule consequent parameters are identified by a least-squares estimator. In the Sugeno-style fuzzy inference, the output, y, is a linear function and given the values of the membership parameters and a training set of P input-output patterns, we form P linear equations in terms of the consequent parameters as:

$$\begin{cases} y_d(1) = \bar{\mu}_1(1)f_1(1) + \bar{\mu}_2(1)f_2(1) + \ldots + \bar{\mu}_n(1)f_n(1) \\ y_d(2) = \bar{\mu}_1(2)f_1(2) + \bar{\mu}_2(2)f_2(2) + \ldots + \bar{\mu}_n(2)f_n(2) \\ \qquad\qquad \vdots \\ y_d(p) = \bar{\mu}_1(p)f_1(p) + \bar{\mu}_2(p)f_2(p) + \ldots + \bar{\mu}_n(p)f_n(p) \\ \qquad\qquad \vdots \\ y_d(P) = \bar{\mu}_1(P)f_1(P) + \bar{\mu}_2(P)f_2(P) + \ldots + \bar{\mu}_n(P)f_n(P) \end{cases} \tag{5.19}$$

In matrix notation, we have:

$$\boldsymbol{y}_d = \mathbf{A}\boldsymbol{k}, \tag{5.20}$$

where $\boldsymbol{y}_d$ is a $P \times 1$ desired output vector:

$$\boldsymbol{A} = \begin{Bmatrix} \bar{\mu}_1(1) & \bar{\mu}_1(1)x_1(1) & \ldots & \bar{\mu}_1(1)x_m(1) & \ldots & \bar{\mu}_n(1) & \bar{\mu}_n(1)x_1(1) & \ldots & \bar{\mu}_n(1)x_m(1) \\ \bar{\mu}_1(2) & \bar{\mu}_1(1)x_1(2) & \ldots & \bar{\mu}_1(2)x_m(2) & \ldots & \bar{\mu}_n(2) & \bar{\mu}_n(2)x_1(2) & \ldots & \bar{\mu}_n(2)x_1(2) \\ \vdots & \vdots & \ldots & \vdots & \ldots & \vdots & \vdots & \ldots & \vdots \\ \bar{\mu}_1(p) & \bar{\mu}_1(p)x_1(p) & \cdots & \bar{\mu}_1(p)x_m(p) & \cdots & \bar{\mu}_n(p) & \bar{\mu}_n(p)x_1(p) & \cdots & \bar{\mu}_n(1)x_m(p) \\ \vdots & \vdots & \ldots & \vdots & \ldots & \vdots & \vdots & \ldots & \vdots \\ \mu_1(P) & \bar{\mu}_1(P)x_1(P) & & \bar{\mu}_1(P)x_m(P) & & \bar{\mu}_n(P) & \bar{\mu}_n(P)x_1(P) & & \bar{\mu}_n(P)x_m(P) \end{Bmatrix} \tag{5.21}$$

where $\boldsymbol{k}$ is an $n\ (1 + m) \times 1$ vector of unknown consequent parameters:

$$\boldsymbol{k} = [k_{10}k_{11}k_{12} \ldots k_{1m}k_{20}k_{21}k_{22} \ldots k_{2m} \ldots k_{n0}k_{n1}k_{n2} \ldots k_{nm}]^T \tag{5.22}$$

When the rule consequent parameters are established, we compute the actual network output vector, $\boldsymbol{y}$, and determine the error vector $\boldsymbol{e}$:

$$\boldsymbol{e} = \boldsymbol{y}_d - \boldsymbol{y} \tag{5.23}$$

The back-propagation algorithm is then applied and the error signals are propagated backwards, and the antecedent parameters updated according to the chain rule.

5.4.2.1 ANFIS Training Algorithm of Jang (1993)

The Jang (1993) ANFIS training algorithm optimises both antecedent parameters and consequent parameters are optimised. For the forward pass, the consequent parameters are adjusted while the antecedent parameters remain fixed while in the backward pass; the antecedent parameters are tuned while the consequent parameters are kept fixed (Table 3.3 show a compilation). The use of the squared error measure for given fixed values of the premise parameters then ensures that the consequent parameters are guaranteed to be the global optimum point in parameter space. This hybrid approach is much faster than the strict gradient descent (Jang 1993) (Table 5.3).

5.4.2.2 The Over-Fitting Problem in ANFIS

A very good convergence can be obtained for the training data set when the number of parameters in ANFIS is equal to or greater than the number of data entries, but the generalization may still be poor and established FIS may not be valid for the test or validation set. This is known a over-fitting. We usually try to compose an ANFIS network with as few parameters as possible, leading to the smallest possible number of fuzzy if-then rules (Hajian et al. 2011).

The number of fuzzy if-then rules is directly related to the number of membership functions in each variable space. If there are four variables in the input space each represented by two membership functions, then the number of fuzzy if-then rules is equal to $2^4 = 16$. Therefore, an extra membership function for each variable space leads to $3^4 = 81$ fuzzy rules (Akyilmaz and Kutterer 2004).

Table 5.3 Two passes in the hybrid learning procedure for ANFIS

Case	Forward pass	Backward pass
Premise parameters	Fixed	Gradient descent
Consequent parameters	Least-squares estimate	Fixed
Signals	Node outputs	Error rates

5.4.3 Function Approximation Using the ANFIS Model

In this example, an ANFIS tracks the trajectory of the non-linear function defined by the equation:

$$y = \frac{\cos(2\ x1)}{e^{x2}} \tag{5.24}$$

We initially choose an appropriate architecture for the ANFIS, which must have two inputs, $x1$ and $x2$, and one output, y.

The ANFIS is defined by four rules, with the training data including 101 training samples and has the structure shown in Fig. 5.12. This is represented by a 101 × 3 matrix $[\boldsymbol{x1x2y_d}]$, where $\boldsymbol{x1}$ and $\boldsymbol{x2}$ are input vectors, and $\boldsymbol{y_d}$ is a desired output vector. $\boldsymbol{x1}$, the first input vector, starts at 0, increments by 0.1 and ends at 10. $\boldsymbol{x2}$, the second input vector, is created by taking sin from each element of vector $\boldsymbol{x1}$, with the elements of the desired output vector, $\boldsymbol{y_d}$, determined by the function equation.

This ANFIS designed was tested in two states each for 1 epoch and 100 epochs of training. The learning process is depicted in Figs. 5.13 and 5.14 for State 1 and in Figs. 5.17 and 5.18 for State 2. State 1:2 Gaussian MFs assigned for each of inputs and linear MF for output. State 2:3 Gaussian MFs assigned for each of inputs linear MF for output (Fig. 5.15).

Comparing the results it is clear that we can achieve some improvement, but much better results are obtained when we assign three membership functions to each input variable. In this case, the ANFIS model will have nine rules, as shown in Fig. 5.15. Figure 5.16 shows how the two dimensional input space is partitioned into nine overlapping fuzzy regions, each of which is governed by a fuzzy if-then rule. In other words, the premise part of a rule defines a fuzzy region, while the consequent part specifies the output within the region (Figs. 5.17 and 5.18).

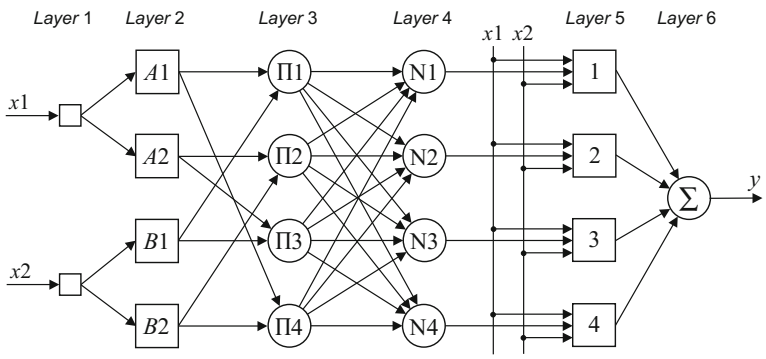


Fig. 5.12 Structure of a four rule ANFIS (http://web.cecs.pdx.edu/~mperkows/CLASS_479/2013%20lectures/2012-1161.%20Neuro-Fuzzy%20Systems.ppt)

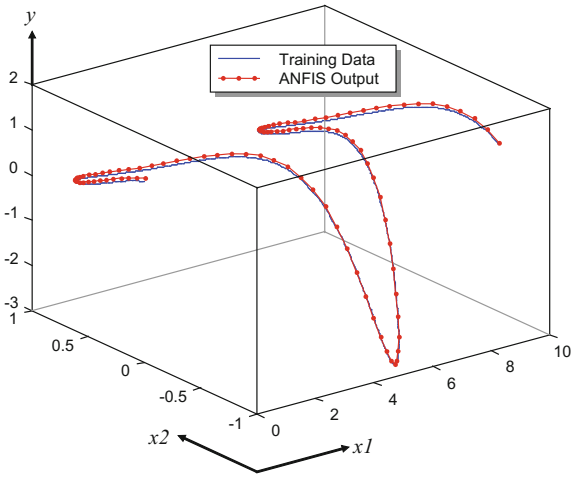


Fig. 5.13 Learning in an ANFIS with two membership functions assigned to each input, one epoch (http://web.cecs.pdx.edu/~mperkows/CLASS_479/2013%20lectures/2012-1161.%20Neuro-Fuzzy%20Systems.ppt)

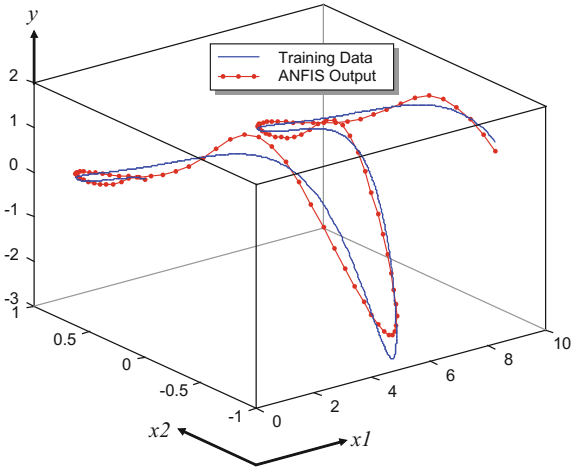


Fig. 5.14 Learning in an ANFIS with two membership functions assigned to each input, 100 epochs (http://web.cecs.pdx.edu/~mperkows/CLASS_479/2013%20lectures/2012-1161.%20Neuro-Fuzzy%20Systems.ppt)

5.5 ANFIS Design and Testing Using the Matlab Fuzzy Logic Toolbox

5.5.1 Introduction

Few books provide an easy tutorial, which enables readers to start implementing their own ANFIS either by bespoke code or by using MATLAB, and so we present here a short useful tutorial on these methods in MATLAB.

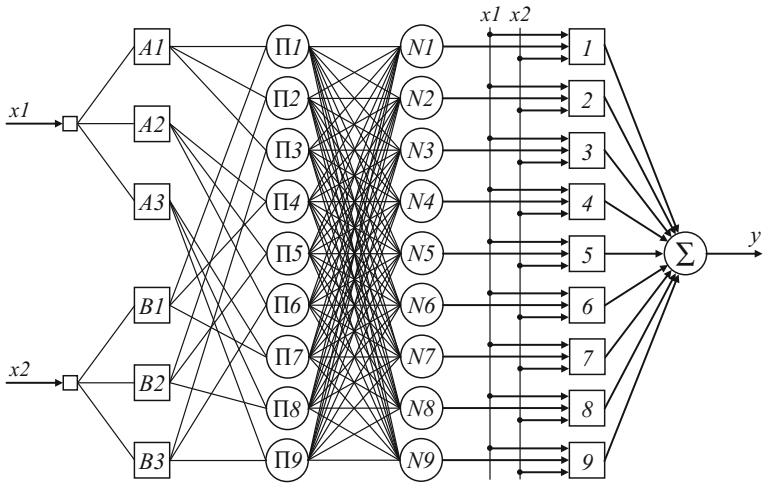


Fig. 5.15 Structure of an ANFIS model with nine rules (http://web.cecs.pdx.edu/~mperkows/CLASS_479/2013%20lectures/2012-1161.%20Neuro-Fuzzy%20Systems.ppt)

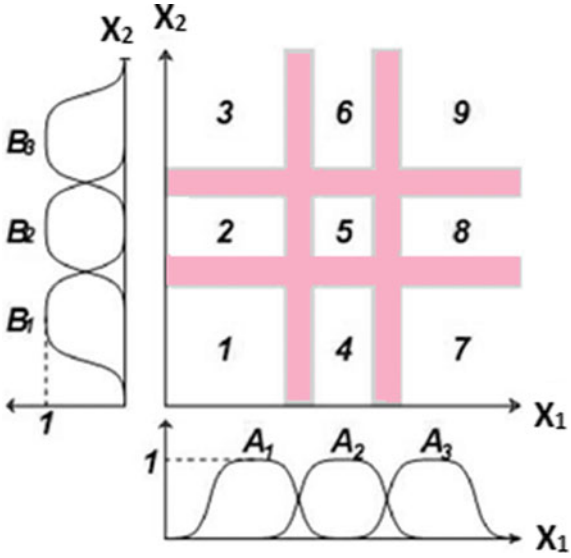


Fig. 5.16 Two dimensional input space partitioned into nine overlapping fuzzy regions

You can create and edit fuzzy inference systems with the Fuzzy Logic Toolbox software in MATLAB using graphical tools or command-line functions, or generate them automatically using either clustering or adaptive neuro-fuzzy techniques.

Using the Simulink software, you can test your fuzzy system in a block diagram simulation environment. This is made possible by a stand-alone Fuzzy Inference Engine which reads the fuzzy systems saved from a MATLAB session. This

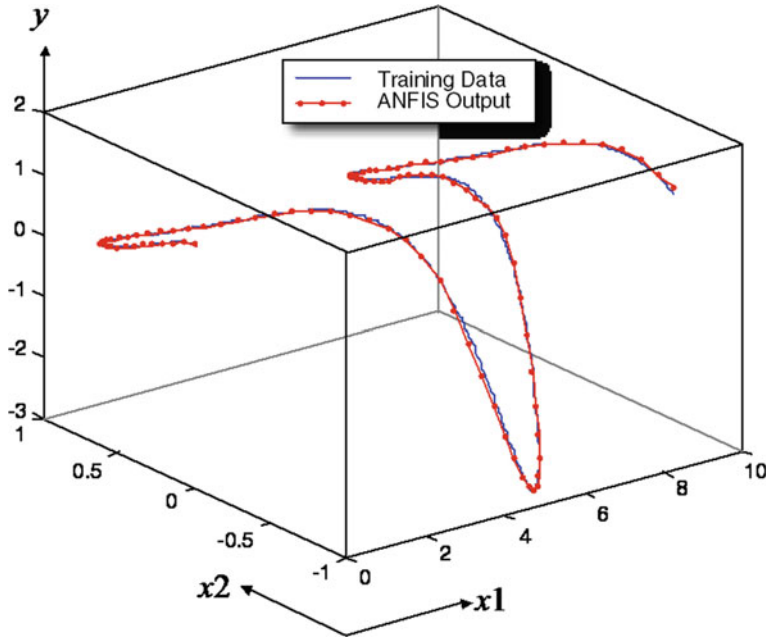


Fig. 5.17 Learning in an ANFIS with three membership functions assigned to each input, 1 epoch (http://web.cecs.pdx.edu/~mperkows/CLASS_479/2013%20lectures/2012-1161.%20Neuro-Fuzzy%20Systems.ppt)

stand-alone engine can be customized to build fuzzy inference into your own code (Fig. 5.19).

Because of the integrated nature of the MATLAB environment, you can customize the toolbox and link it with another toolbox, such as the Control System Neural Network, or Optimization Toolboxes.

5.5.2 *ANFIS Graphical User Interference*

Type "anfisedit" in command window:

$$\gg \textbf{anfisedit}$$

Then a window namely "Neuro-fuzzy Designer" is opened in which you can design your desired ANFIS (Fig. 5.20).

This window has four main parts:

ANFIS Info: here the number of inputs, the number of outputs and the number of membership functions is illustrated, once the user changes these parameters of the ANFIS they will be updated and the updated information will be illustrated.

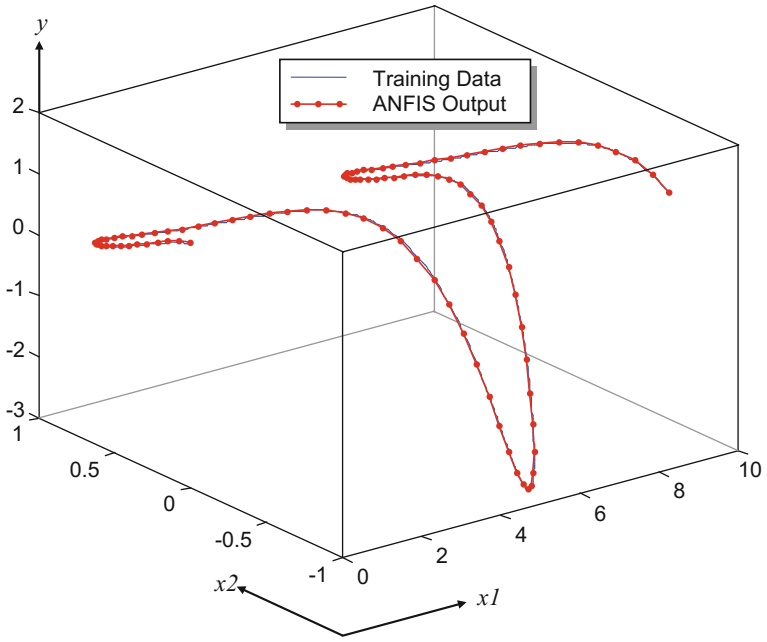


Fig. 5.18 Learning in an ANFIS with three membership functions assigned to each input, 100 epochs (http://web.cecs.pdx.edu/~mperkows/CLASS_479/2013%20lectures/2012-1161.%20Neuro-Fuzzy%20Systems.ppt)

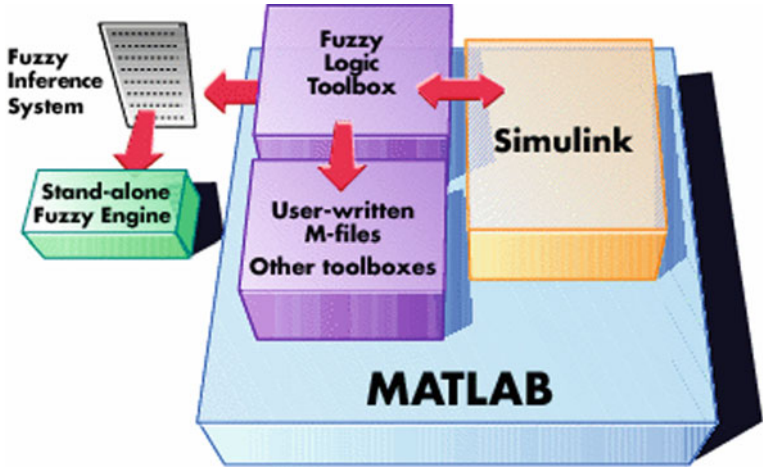


Fig. 5.19 Schematic of MATLAB toolbox abilities to design and test of fuzzy interference *Source* Matlab Fuzzy Toolbox Tutorial Help

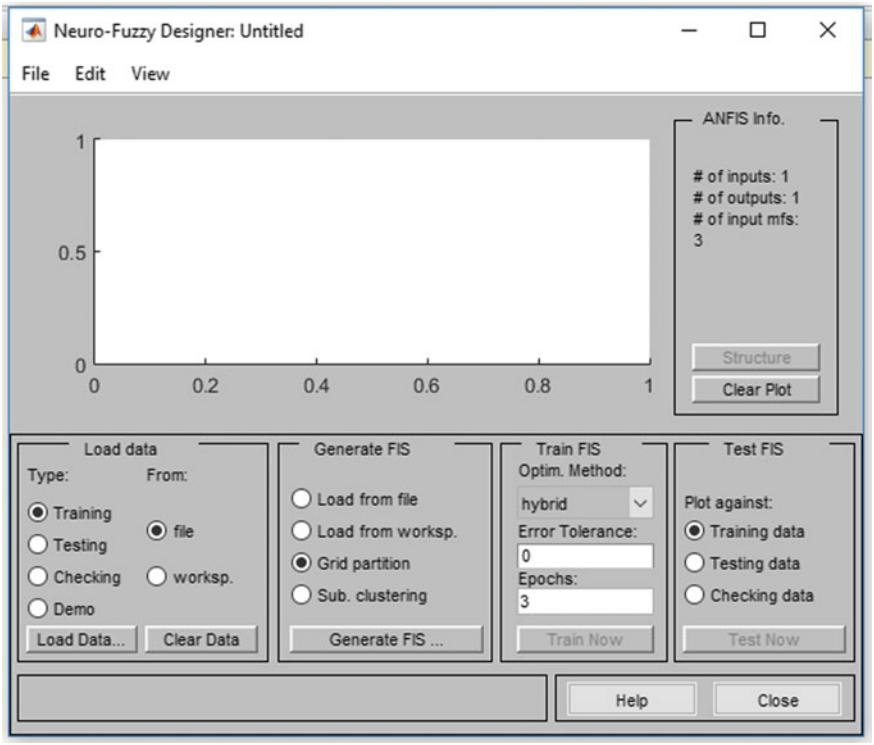


Fig. 5.20 The neuro-fuzzy designer window for ANFIS editor toolbox

In the "load data" section you can load training, testing and checking dataset for your ANFIS from file or from the workspace of MATLAB.

In this example we try to implement ANFIS modeling dataset of a function mentioned in a previous section, which was defined as:

$$\mathbf{Y} = \mathbf{cos}(\mathbf{2x_1})/\mathbf{exp}(\mathbf{x_2})$$

To produce the dataset, **all_data** the following codes are run:

```
≫x1 = 0:0.5:14.5;
≫x2 = −1:0.1:1.9;
≫y = cos(2*x1)./exp(x2);
≫d = [x1; x2; y];
```

Now we check the length of vectors x1 and x2 and dimensions of the matrix "d":

```
    ≫length(x1)
ans = 30
    ≫length(x2)
ans = 30
```

```
≫size(d)
ans = 3 30
```

It means we have 3 rows (as x1, x2, y) and 30 columns as 30 pairs of training. When the MATLAB ANFIS editor receives a matrix as a training data matrix it assumes inputs and output of ANFIS as its columns and also assumes all the columns before the last right column are inputs and the last column is assumed to be the output, but here matrix "d" contains the input and output in its rows (not columns), hence matrix "d" should be transposed:

$$\gg \mathbf{all\_data} = \mathbf{d}';$$

Now the 30 pairs of all_data are ready to use for training, testing and checking the desired ANFIS. In selecting training data from the work space we should use 70% of all data, and for testing and checking data, we should use 10 and 20% of all data, respectively.

To produce the training pairs, the above codes are run for 21 data; then:

```
≫x1 = 0:0.5:10;
≫x2 = −1:0.1:1;
≫y = cos(2*x1)./exp(x2);
≫d = [x1; x2; y];
≫training_data = d′;
```

Now we can repeat steps above for testing and checking data for 3 and 6 data, respectively. But it must be mentioned that we must select data for training, testing and checking, such that there isn't any overlap between the data sets.

For testing data:

```
≫x1 = 10.5:0.5:11.5;
≫x2 = 1.1:0.1:1.3;
≫y = cos(2*x1)./exp(x2);
≫d = [x1; x2; y];
≫testing_data = d′;
```

For checking data:

```
≫x1 = 12:0.5:14.5;
≫x2 = 1.4:0.1:1.9;
≫y = cos(2*x1)./exp(x2);
≫d = [x1; x2; y];
≫checking_data = d′;
```

Now the training, testing and checking data is ready to use for training and testing the desired ANFIS from the workspace.

To do this select "Training" as the type of data you want to read and "worksp." in the select field "Load data" as the location from which you want to receive data, and then a menu edit field box is opened and requests you to enter the variable name. Here we enter "training_data" as the input variable data as shown in Fig. 5.21.

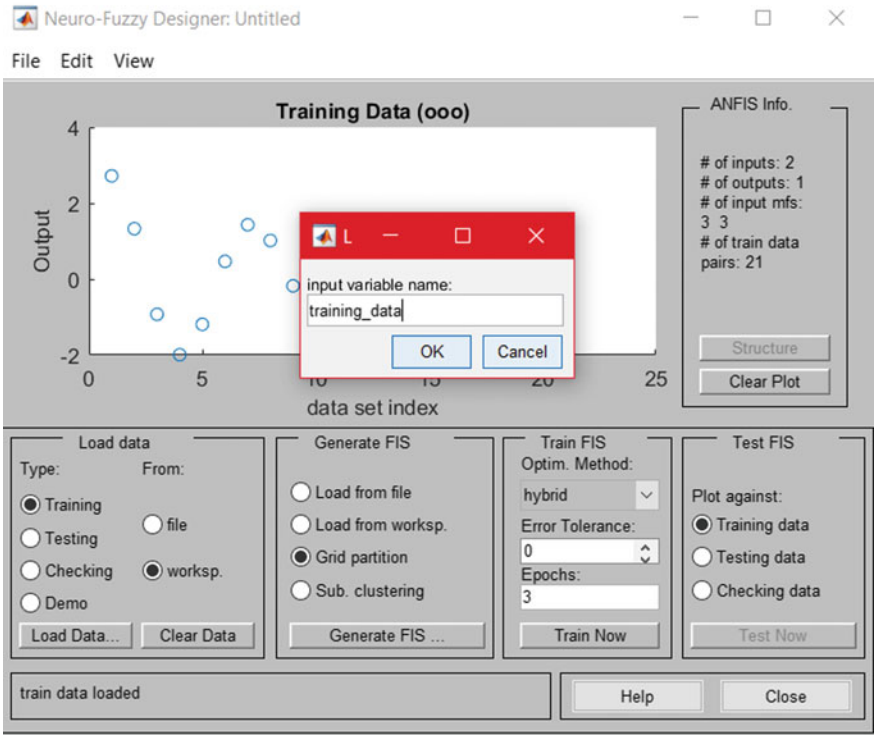


Fig. 5.21 The variable name is entered to be loaded from workspace of MATLAB

If you want to load data from a file with excel format, you can use command "xlsread":

≫y = xlsread("exact file location in directory i.e. c:/geophysics/seismic/data_group_1.xlsx")

We also load the testing and checking data from the workspace. After loading the training, testing and checking data from the workspace, the last column of the entered matrix is plotted as the desired output (Fig. 5.22); the data shown with "O" is for training data, the data shown with "." is for testing data, and the data shown with "+" is for checking data.

In the next step we generate the desired FIS, there are four items to select:

- Load from file: this item allows the user to load the designed FIS from file
- Load from worksp.: this item allows the user to load the designed FIS from work space
- Grid partition, and
- Sub. Clustering: Clustering is the process of organizing objects into groups whose members are similar in some way and can determine the intrinsic

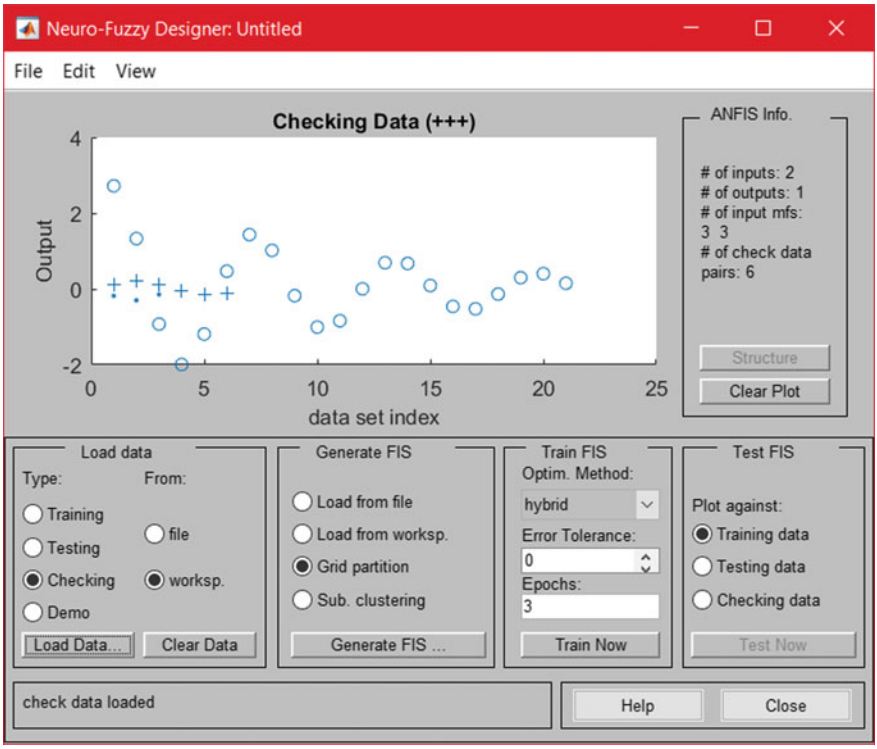


Fig. 5.22 Plot of the training data (output)

grouping for a set of unlabeled data Grid partitioning and Subtractive Clustering Based Approaches are the most commonly used.

Here, we select "Grid partition" to generate the FIS,

Note that ANFIS editor toolbox automatically determines the number of inputs from the training matrix we entered, for this example as the training data matrix size is 21*3 it will make a FIS with 2 inputs and one input (2 inputs + 1 output = 3). Generally, if the training data matrix is m*n, it will automatically make a FIS with n − 1 inputs and one output, and we will have an m-length training data set with the form{{n − 1 inputs}, {1 output}}.

In the next step, click on "Generate FIS…" in order to generate the FIS structure and after clicking on this icon a box-window is opened as shown in Fig. 5.23. Which has two parts:

INPUT in this part you can edit the number of membership functions (MFs) for each of inputs, to assign a different number of MFs to each input use spaces to separate the numbers. Also you can select inputs MF type which is selectable in "MF Type" pop-up menu (Fig. 5.23).

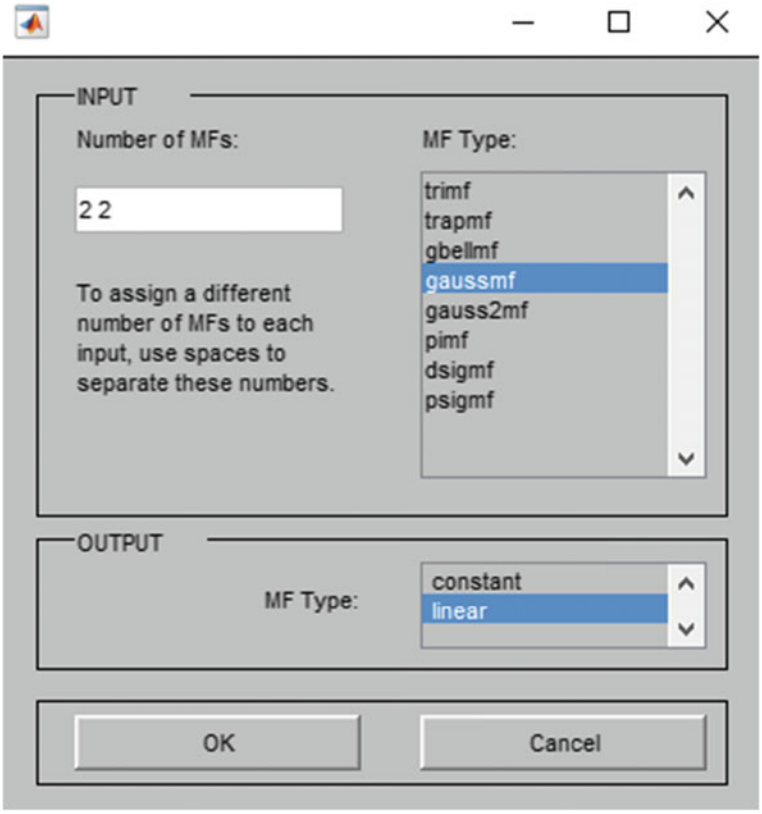


Fig. 5.23 This window opens after selecting "FIS generation", in which you can enter number of MFs for input and their types and also output MF type

OUTPUT in this part you can edit the membership functions (MFs) for output (constant or linear).

Here we select 2 MFs for input x1 and 2 MFs for input x2 and the type of MFs is selected as gauss (Gaussian) and linear MF for output.

To see the FIS designed structure click on "Structure" in "ANFIS info." and then the structure of the designed FIS will be displayed in a window named "Anfis Model Structure" (Fig. 5.24). In this window you can click on each nodes to see detailed information.

Now the ANFIS we have designed is ready to train; to start the training:

1. Select the optimization method from the pop-up menu, there are two choices: "backprop." which means the back propagation method to optimize the rules, and "hybrid" which is fusion of back propagation and gradient descent algorithm. Here, we select hybrid as it is better in most cases needing a lower number of epochs to reach the desired error tolerance.

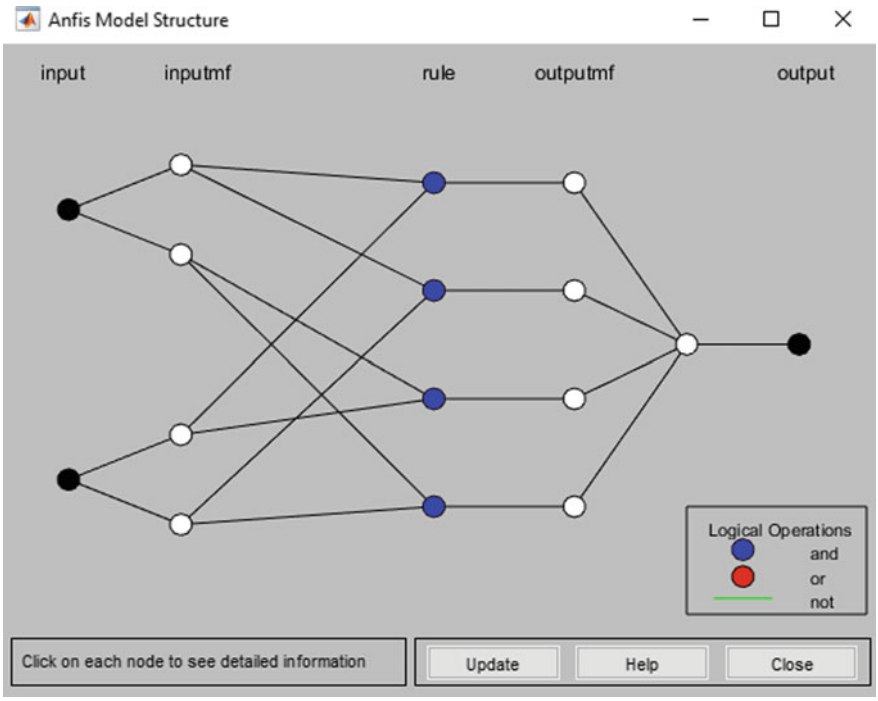


Fig. 5.24 ANFIS model structure with five layers which is presented after selecting "structure" from ANFIS info in the main ANFIS editor window

2. The number of training epochs is set to the desired value, under the **Epochs** listing on the GUI, note that the default value is 3.
3. On selecting "**Train Now**", and a window appears (Fig. 5.25 for trimf and Fig. 5.26 for gaussmf MF). Comparing the training error for the two conditions shows that the Gaussian membership function for inputs leads to less error.

Testing data against the trained FIS

To test the designed FIS against the checking data we select Checking data (Figs. 5.27, 5.28 and 5.29).

The next steps are testing of the FIS on testing and checking data as mentioned above for the test of training data.

Rules viewer

The Rule viewer is used to view the entire implication process of the fuzzy inference diagram from beginning to end, dynamically. As the line indices that correspond to the inputs change the system readjusts and computes the new output (Fig. 5.30).

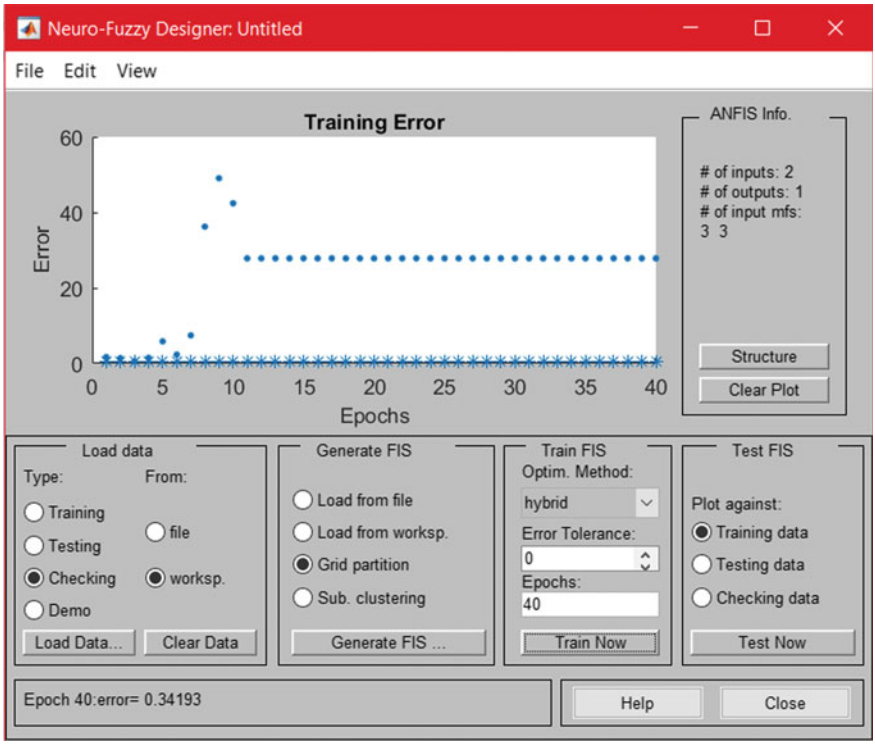


Fig. 5.25 ANFIS with 3 MF for input (x1) and 3 MFs for input (x2), type of MFs is triangular

Surface viewer

The Surface Viewer allows you to examine the output surface of an FIS, for any one or two inputs (Fig. 5.31) but does not alter the fuzzy system or its associated FIS at all. Pop-up menus, allow you to select the input variables you want assigned to the two input axes (X and Y), and the output variable to be assigned to the output (or Z) axis. Selecting the **Evaluate** button performs the calculation and plots the output surface which can be viewed from different angles and rotated using graphical handles. However, if there are more than two inputs to your system the constant values associated with unspecified inputs must be set. The **Ref. Input** field is used when there are more inputs required by the system than the surface is mapping. If you have a four-input one-output system and would like to see the output surface you can generate a three-dimensional output surface where any two of the inputs vary, but the other inputs must be held constant. In such a case the input would be a four-dimensional vector with NaNs assigned to indicate those input values that remain fixed.

To design a new FIS (Fuzz Interference System) with different properties compared to the ANFIS default, select “New FIS” in the “File” menu item (Fig. 5.32). Then you can select either the Mamdani or Sugeno models, which are

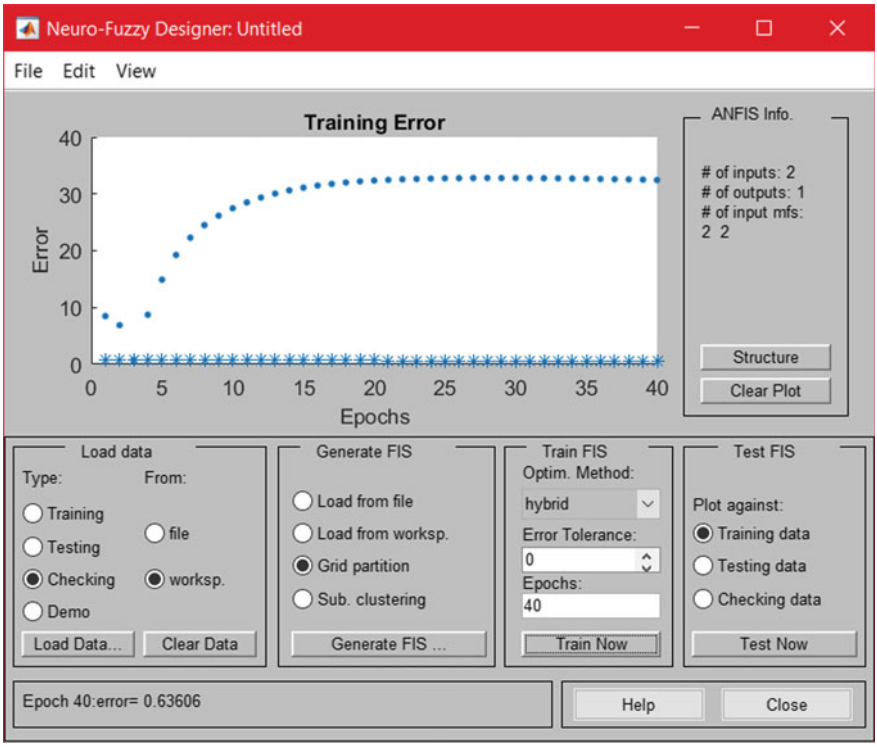


Fig. 5.26 Training process in 40 epochs for ANFIS with 2 inputs, 1 output, with Gaussian MFs type for inputs and outputs

the most common ANFIS's, with zero and first order ANFIS properties respectively.

If you select Sugeno then the window shown in Fig. 5.33 appears; its default is a Sugeno system with ONE INPUT and ONE OUTPUT. To add further input/s to this system you can click on "Edit" item and select "add input variable".

Also, if you wish to add new variables to the input/output select "add variable" in Edit menu bar (Fig. 5.34).

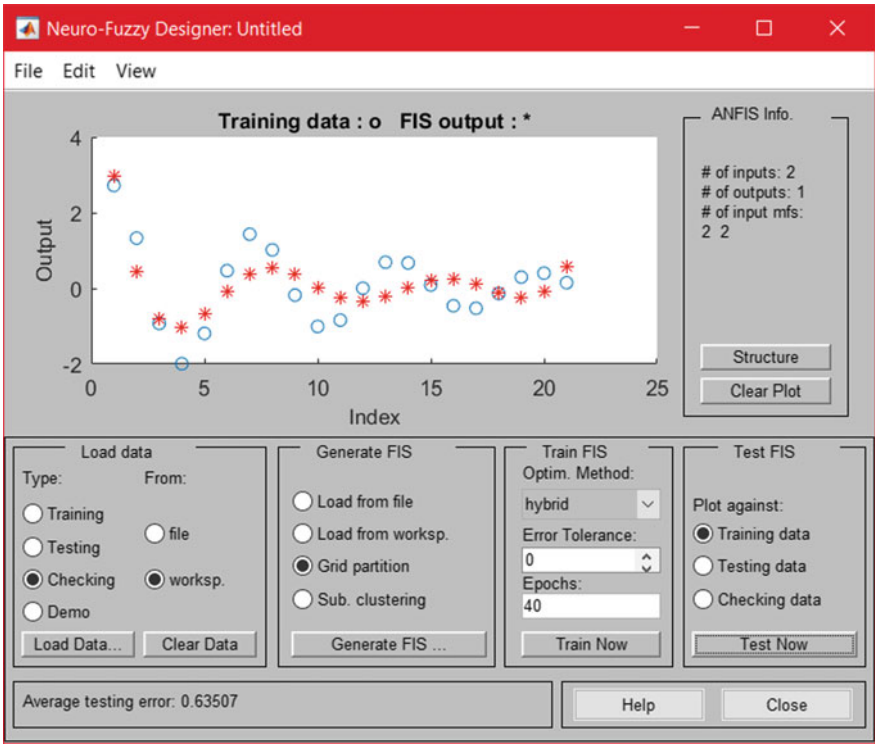


Fig. 5.27 Membership functions for inputs, type of MF Gaussian, output MF linear, grid partitioning, no. of epochs 40

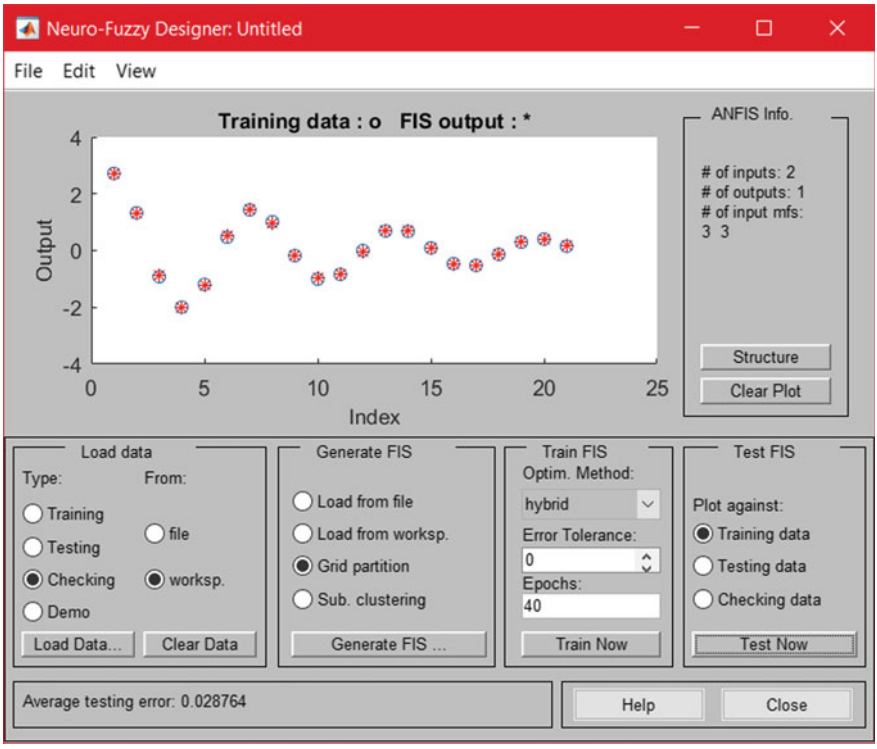


Fig. 5.28 Three membership functions for inputs, type of MF Gaussian, output MF linear, grid partitioning, no. of epochs 40

Fig. 5.29 ANFIS with 3 MFs for input (x1) and 3 MFs for input (x2) and MFs type triangle and linear MF output, after only one epoch training

Fig. 5.30 Rules for type of MF Gaussian, output MF linear, grid partitioning, no. of epochs 40

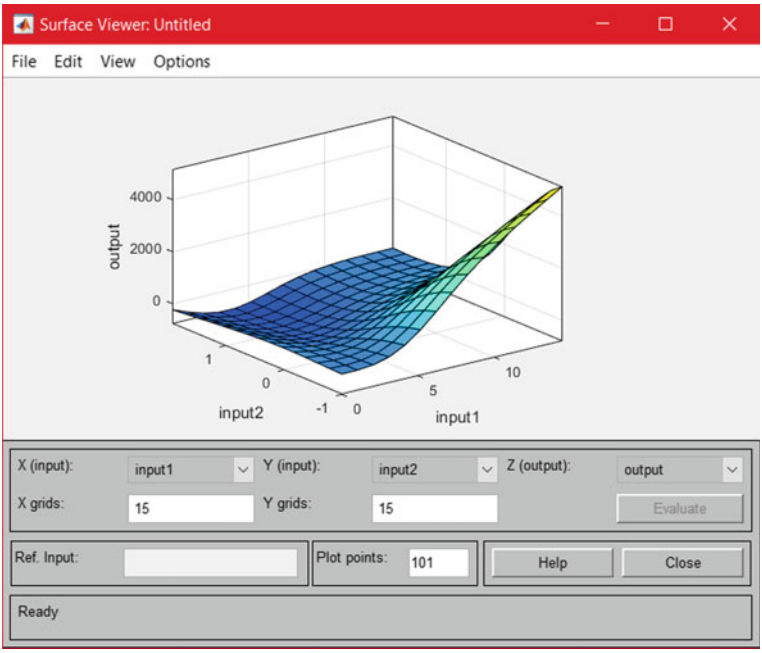


Fig. 5.31 Surface for type of MF Gaussian, output MF linear, grid partitioning, no. of epochs 40

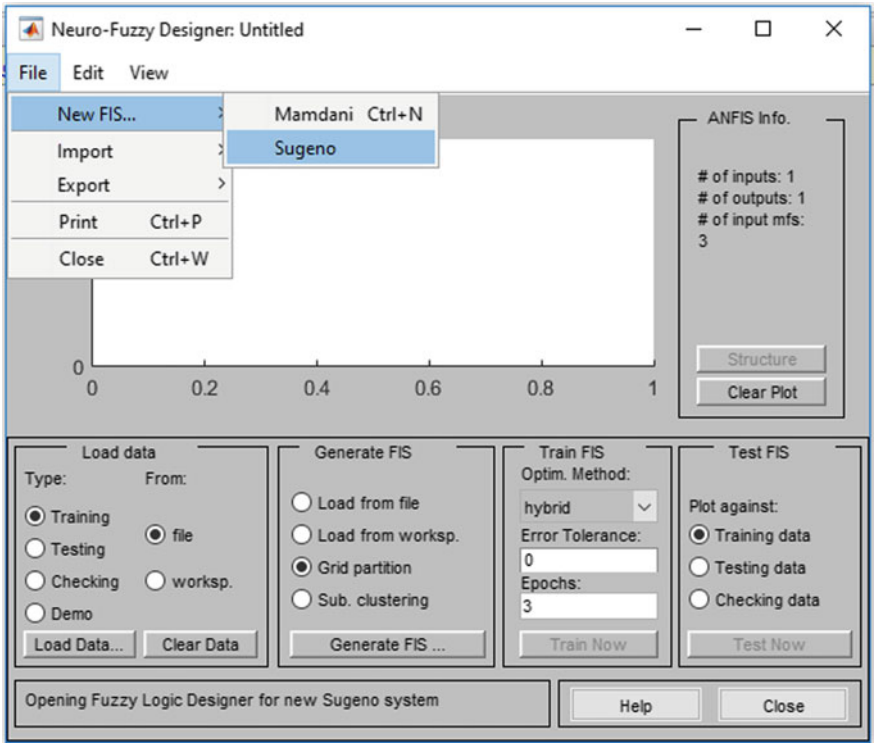


Fig. 5.32 Selecting type of FIS (Mamdani or Sugeno) using new FIS menu

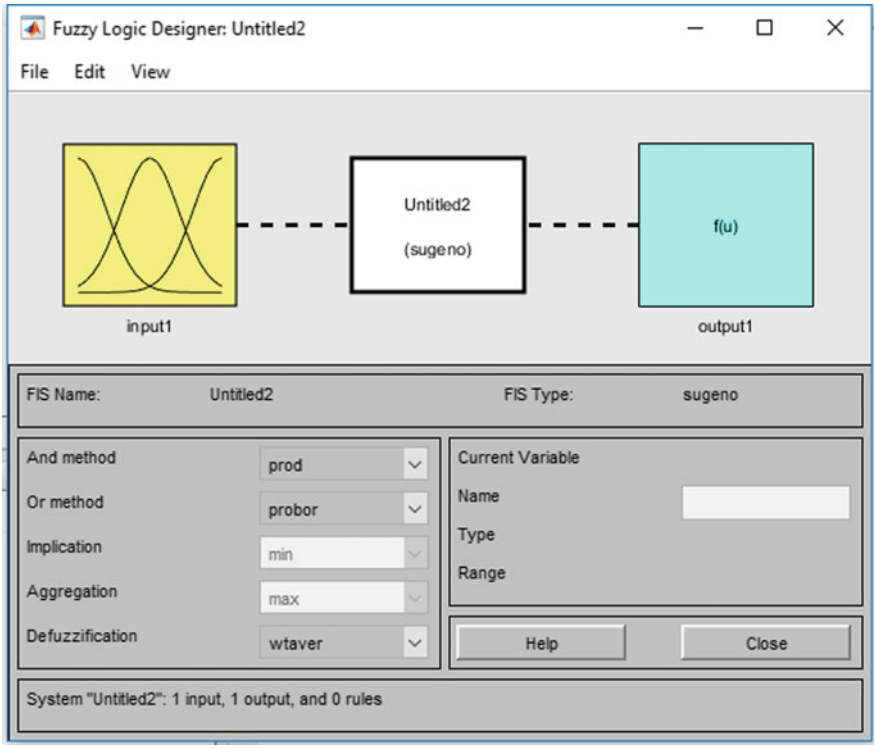


Fig. 5.33 Window for fuzzy logic designer, here you can design a Mamdani or a Sugeno fuzzy interference system

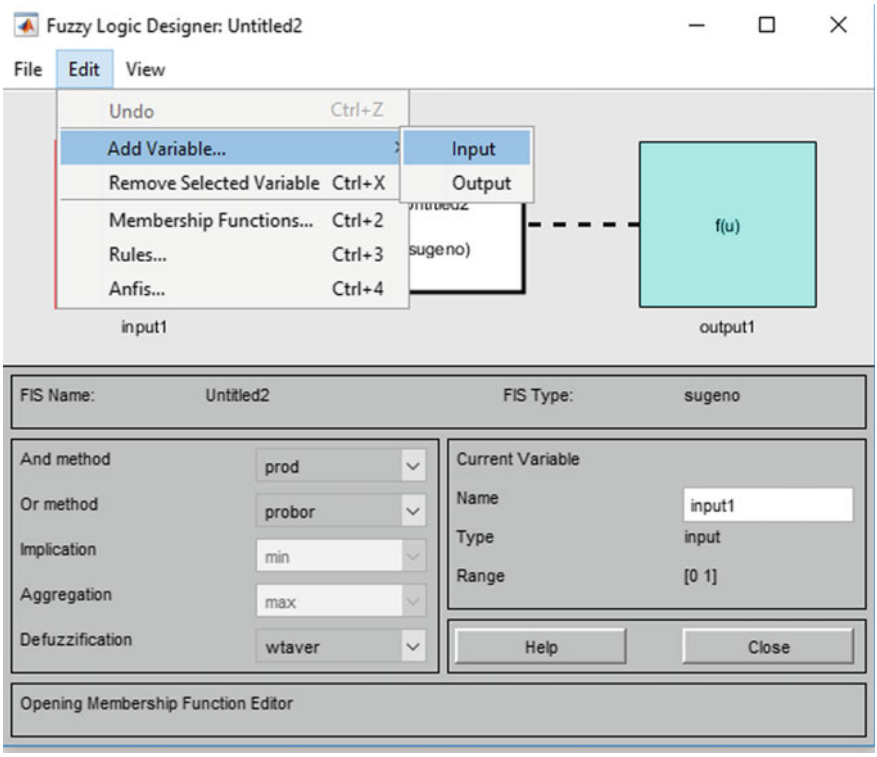


Fig. 5.34 Editing number of input/output variables for a Sugeno (or Mmdani) FIS

References

Akyilmaz O. and Kutterer H., 2004, Prediction of Earth rotation parameters by fuzzy inference systems, Journal of Geodesy, 78, 82–93, https://doi.org/10.1007/s00190-004-0374-5.

Berenji H.R., 1992, Fuzzy and Neural Control, in: An Introduction to Intelligent an Autonomous Control, Antsaklis, P.J. and K. M. Passino (eds.).

Bohlooli A. and Jamshidi K., 2011, A GPS-Free Method for Vehicle Future Movement Directions Prediction using SOM for VANET, Applied Intelligence, 36(3), 685–697.

Figueiredo M. and F. Gomide; Design of Fuzzy Systems Using Neuro-Fuzzy Networks, IEEE Transactions on Neural Networks, 1999, 10(4), 815–827.

Hajian A., 2010, Intelligent Interpretation of Gravity Data via a Fuzzy approach for detecting subsurface cavities, Proceeding of Asia Oceania Geosciences Society, AOGS2010, 5–9 July, Hyderabad, India, Article ID: SE 18-17-15 A29.

Hajian A. and Fazelian A., 2016, Interpretation of Ground Penetrating Radar data in order to detect buried pipes using fuzzy approach, 17th National Iranian Conference on Geophysics, Tehran, Geological and mineral exploration organization, Ministry of industries and mines, 922–925(in Persian).

Hajian A., Kimiaefar R. and Siahkoohi H.R., 2016, Random Noise Attenuation in Reflection Seismic Data Using Adaptive Neuro-fuzzy Interference System (ANFIS", 78th European Association of Geoscientists and Engineers, Vienna, Austria.

Hajian A., Styles P. and Zomorrodian H., 2011a, Depth Estimation of Cavities from Microgravity Data through Multi Adaptive Neuro Fuzzy Interference Systems, 17th European Meeting of Environmental and Engineering Geophysics, Leicester, UK, 2011.

Hajian A., Zomorrodian H., Styles P., Greco F. and Lucas C., 2012, Depth estimation of cavities from microgravity data using a new approach: Local Linear Model Tree(LOLIMOT), Near Surface Geophysics, 10, 21–234.

http://web.cecs.pdx.edu/∼mperkows/CLASS_479/2013%20lectures/2012-1161.%20Neuro-Fuzzy%20Systems.ppt

http://web.cecs.pdx.edu/∼mperkows/CLASS_479/2013%20lectures/2012-1161.%20Neuro-Fuzzy%20Systems.ppt

Jang J.-S.R. and Sun. C.-T., 1993, Functional equivalence between radial basis function networks and fuzzy inference systems. IEEE Transactions on Neural Networks, 4(1), 156–159.

Jang R., 1992, Neuro -Fuzzy Modeling: Architectures, Analysis and Applications, PhD Thesis, University of California, Berkley.

Juang C.F. and Lin C.T., 1998, An On-Line Self-Constructing Neural Fuzzy Inference Network and Its Applications IEEE Transactions on Fuzzy Systems, 6, 12–32.

Kasabov e N., Qun Song, 1999, Dynamic Evolving Fuzzy Neural Networks with 'm -out-of-n' Activation Nodes for On-Line Adaptive Systems, Technical Report TR99/04, Department of Information Science, University of Otago.

Lin T. C. and Lee C. S., Neural Network Based Fuzzy Logic Control and Decision System, IEEE Transactions on Computers, 1991, 40(12), 1320–1336.

Nauck D., 1994, Building Neural Fuzzy Controllers with NEFCON-I in: Kruse/Gebhardt/Klawonn: Fuzzy Systems in Computer Science, Vieweg, Wiesbaden, (ISBN: 3-528-05456-5), 141–151.

Sulzberger S.M., Tschichold-Gurman N. and Vestli S.J., 1993, FUN: optimization of fuzzy rule based systems using neural networks,, 1993, IEEE International Conference on Neural Networks, 312–316, DOI: https://doi.org/10.1109/icnn.1993.298575.

Takagi, T. and M. Sugeno, 1985, Fuzzy identification of systems and its applications to modeling and control. IEEE Trans. on Systems, Man, and Cybernetics 15, 116–132.

Tano S., Oyama T. and Arnould T., 1996, Deep Combination of Fuzzy Inference and Neural Network in Fuzzy Inference, Fuzzy Sets and Systems, 1996, 82(2), 151–160.

Vieira J., Morgado Dias F. and Mota A., 2004, Neuro-Fuzzy Systems: A Survey WSEAS Transactions on systems 3(2), 414–419, in: www.cee.uma.pt/morgado/down/483-343.pdf.

Chapter 6
Application of Neuro-Fuzzy Systems in Geophysics

– Applications of Neuro-Fuzzy systems in Geophysics with various examples

6.1 Depth Estimation of Cavities from Microgravity Data Using Multi Adaptive Neuro Fuzzy Interference Systems

Hajian et al. (2011) used neuro-fuzzy systems to estimate the depth of subsurface cavities from gravity data based on a Multiple Adaptive Neuro-Fuzzy Interference System (MANFIS). This is an intelligent way to interpret microgravity data to estimate the depth and the shape of the unknown cavities. The MANFIS model was trained to develop interpretations for two main cavity types: sphere and cylinder producing estimates of and depth and radius. MANFIS's with different number of rules were tested and the optimum value determined and this model was tested with the addition of 20% Gaussian and it displayed robust behavior. The method was also tested on real gravity data collected from Freeport Bahamas and the results were in good agreement with observed depth values of subsurface cavities confirmed and characterized by drilling.

6.1.1 Why Use Neuro-Fuzzy Methods for Microgravity Interpretation?

Subsurface cavities have a negative contrast density, so they are manifest as negative gravity anomalies. The accuracy of the gravimeter in for this kind of work must be at least ±5 micro Gal (1Gal = 0.01 m/s^2) for the small variations to be observed in Bouguer gravity signals. Various interpretation methods: the Analytical

A. Hajian and P. Styles, *Application of Soft Computing and Intelligent Methods in Geophysics*, Springer Geophysics,
https://doi.org/10.1007/978-3-319-66532-0_6

signal, Euler's equation, least squares minimization, regression, Fourier transform, to name but a few have been used over the years to enhance gravity interpretation but each method comes with its own difficulties. For example, the analytical signal method can only determine the location of the edges of an object, the Euler method, shows different responses for different window's sizes, Index structure and depends strongly on the experience of the interpreter. A new method, not dependent on the experience of the interpreter, has been developed using an Adaptive Neuro Fuzzy Interference System (MANFIS) model which is able to estimate the depth of cavities for any desired training domain. A significant advantage of this method is that there is no need to repeat complicated calculations for new gravity data sets as interpretation takes place in the same trained space domain, after training the model. Also the flexibility and robustness of the network in the presence of noise and for different depth domains is exemplary. Hajian et al. (2011) presented ANFIS abilities for the interpretation of noisy synthetic gravity data and also for real gravity data and in this section we explain their work in detail.

6.1.2 *Multiple Adaptive Neuro Fuzzy Interference SYSTEM (MANFIS)*

The acronym MANFIS denotes a Multi-adaptive neuro-fuzzy inference system. MANFIS is an extension of the ANFIS neuro-fuzzy system, in order to produce multiple outputs. A neuro-fuzzy system is a nonparametric regression tool for modeling regression relationships without reference to any pre-specified functional form. In its original form, ANFIS could only give a single output but the novel aspect of MANFIS is that it aggregates many independent ANFISs to obtain multiple outputs and this architecture is shown in Fig. 6.1.

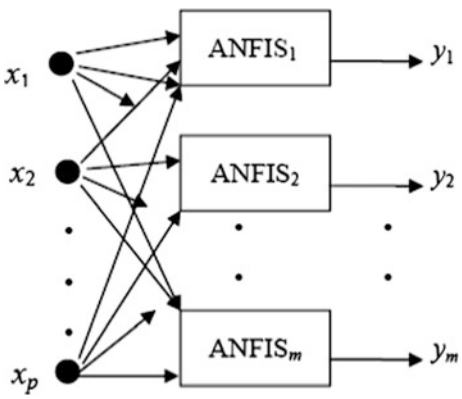


Fig. 6.1 Architecture of a general MANFIS (Hajian et al. 2011)

ANFIS works by approximating the functional relations between responses and input variables, by gradually fine-tuning the parameters at the adaptive nodes of ANFIS for the process under study.

The ANFIS architecture is shown in Fig. 6.2. Circular nodes represent fixed nodes whereas the square nodes are nodes at which the parameters must be learnt. A Two Rule Sugeno ANFIS has rules of the form:

$$If\ x\ is\ A_1\ and\ y\ is\ B_1 \quad THEN\ f_1 = p_1x + q_1y + r_1 \tag{6.1}$$

$$If\ x\ is\ A_2\ and\ y\ is\ B_2 \quad THEN\ f_2 = p_2x + q_2y + r_2 \tag{6.2}$$

In order to train the network we have a forward pass which propagates the input vector through the network, layer by layer and a backward pass where the error is sent back through the network (back propagation).

Layer 1: The output of each node is:

$$O_{1,i} = \mu_{A_i}(x) \quad for\ i = 1, 2, \tag{6.3}$$

$$O_{1,i} = \mu_{B_{i-2}}(y) \quad for\ i = 3, 4 \tag{6.4}$$

So, the $O_{1,i}(x)$ is essentially the membership grade for x and y.

The membership functions can have broad definitions but here we have used the bell shaped function:

$$\mu_A(x) = \frac{1}{1 + \left|\frac{x - c_i}{a_i}\right|^{2b_i}} \tag{6.5}$$

Where a_i, b_i, c_i are parameters to be learnt and are known as the premise parameters.

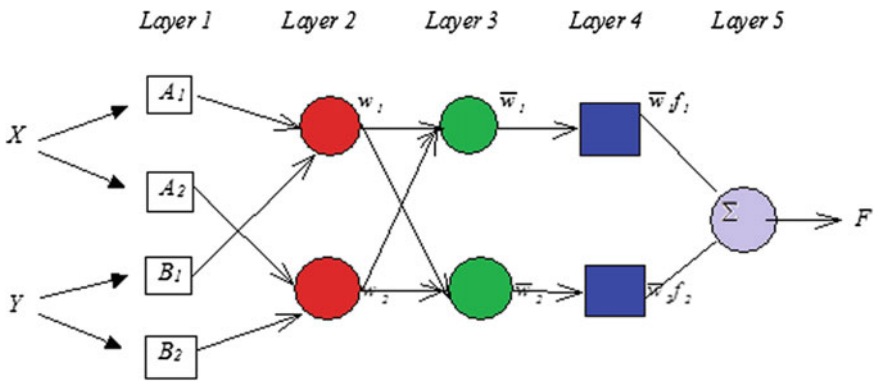


Fig. 6.2 An ANFIS architecture for a two rule Sugeno system (Hajian et al. 2011)

Layer 2: Each node in this layer is fixed and the t-norm is used to perform the operator 'AND' on the membership grades(for example the product):

$$O_{2,i} = w_i = \mu_{A_i}(x)\mu_{B_i}(y), \quad i = 1, 2 \tag{6.6}$$

Layer 3: Layer 3 contains fixed nodes which calculate the ratio of the firing strengths of the rules:

$$O_{3,i} = \bar{w}_i = \frac{w_i}{w_1 + w_2} \tag{6.7}$$

Layer 4: The nodes in this layer are adaptive and output the consequence of applying the rules:

$$O_{4,i} = \bar{w}_i f_i = \bar{w}_i(p_i x + q_i y + r_i) \tag{6.8}$$

The parameters in this layer (p_i, q_i, r_i) to be determined are known as the consequent parameters.

Layer 5: There is a single node here that computes the overall output:

$$O_{5,i} = \sum_i \bar{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{6.9}$$

This is typically how the input vector is fed through the network, layer by layer. There are a number of possible training approaches but here the hybrid learning algorithm which uses a combination of Steepest Descent and Least Squares Estimation (LSE) is used.

6.1.3 *Procedure of Gravity Interpretation Using MANFIS*

For a MANFIS system which estimates depth and shape of subsurface cavities, two ANFIS are paralleled with the same inputs (Features F_1–F_5 as defined in Eqs. 6.10:14); one with depth (z) as output and another with shape factor (q) as output. The MANFIS structure is depicted in Fig. 6.3. The general procedure to estimate shape factor and depth of cavities from microgravity data is shown in Fig. 6.4.

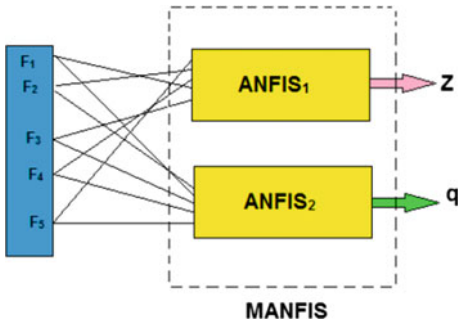


Fig. 6.3 The MANFIS model structure for depth and shape factor estimation

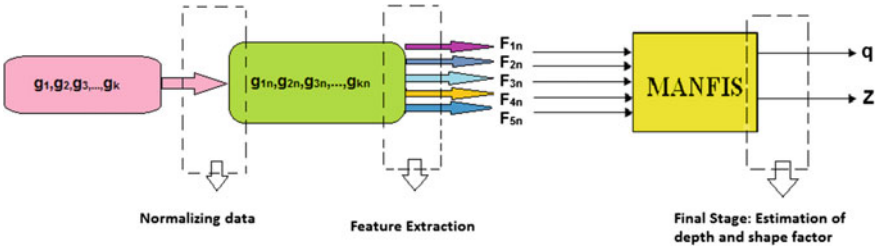


Fig. 6.4 The procedure of interpretation of microgravity data using MANFIS (Hajian 2012)

6.1.4 *Training Strategies and MANFIS Network Architecture*

To design the MANFIS model, the first stage was to select suitable inputs. If we apply all measured gravity points of a profile as inputs of ANFISs then its structure will be have a complex topology and be very time consuming to train with a lot of rules. To prevent this problem some features: F_1, F_2, F_3, F_4 and F_5 were extracted from the residual gravity (see Eqs. 6.10–6.14). These features are normalized and then applied as inputs of MANFIS (see Fig. 6.4).

We then tested ten different configurations of these features as inputs to MANFIS and calculated the statistical errors for each of them to find the best array as input for MANFIS. We prepared training data for two models, a sphere and a cylinder because the shape of many if not all subsurface cavities is quite well approximated by a sphere or a cylinder. The principal features are calculated for these bodies using Eqs. 6.10–6.14 (GRÊT et al. 2000).

$$F_1 = Xg_{50}/Xg_{75} \tag{6.10}$$

$$F_2 = (Xg_{25} - Xg_{66})/(Xg_{66} - Xg_{75}) \tag{6.11}$$

$$F_3 = \int_{xs}^{xe} g(x)dx \tag{6.12}$$

$$F_4 = Xg_{50} \tag{6.13}$$

$$F_5 = Xg_{75} \tag{6.14}$$

F_1, F_2, F_3, F_4 and F_5	Features calculated from gravity data;
g(x)	gravity value (in micro-Gal) at the point at the horizontal distance x (in meter);
Xg_{50}	x where the gravity value is 50% of the maximum amplitude of the gravity data;

Xg_{75} x where the gravity value is 75% of the maximum amplitude of the gravity data, and the same as for Xg_{25}, Xg_{66}.

Note that the domain of the integral in Eq. 6.10 is from the starting gravity point of the profile (xs) to the end point of the profile (xe).

As mentioned previously, in order to find the best selection for inputs among these features we tested the MANFIS models with different configuration of features as inputs and calculated the error indexes (see Tables 6.1 and 6.2): MSE (Mean square error), NMSE (Normalized Square error), MAPE (mean absolute percentage error), R^2 (Pierson Coefficient).The results showed that Model M10 is optimal, meaning that MANFIS with inputs F_1–F_5 has the least values of error indexes.

As shown in Fig. 6.3 the inputs of the Model (MANFIS) are F_1, F_2, F_3, F_4, F_5 and the outputs are R, Z where R is the radius of the cavity and Z is the depth of the

Table 6.1 The models with their selected features as inputs to MANFIS (Hajian and Zomorrodian 2016)

Inputs					Model
F_5	F_4	F_3	F_2	F_1	
		*	*	*	M1
*	*	*		*	M2
	*		*	*	M3
*	*	*	*		M4
	*	*	*	*	M5
*	*		*	*	M6
*		*	*		M7
*	*		*		M8
*	*	*			M9
*	*	*	*	*	M10

Table 6.2 The error index value for the models M_1–M_{10} (Hajian and Zomorrodian 2016)

MAPE	R^2	NMSE	MSE	Model
9.57	0.481	0.519	0.055	M1
3.25	0.921	0.079	0.008	M2
3.82	0.869	0.132	0.014	M3
4.35	0.837	0.164	0.015	M4
3.29	0.916	0.084	0.009	M5
3.89	0.876	0.124	0.013	M6
4.62	0.813	0.173	0.016	M7
3.73	0.824	0.183	0.017	M8
3.09	0.914	0.086	0.008	M9
3.01	0.928	0.073	0.008	M10

cavity. So the training set is defined as [F_1, F_2, F_3, F_4, F_5], [R, Z] and in order to prepare gravity training data we used the following Eq. 6.15, (Abdelrahman 2001):

$$g(x,z) = \frac{AZ}{(x^2+z^2)^q} \tag{6.15}$$

$$A = \begin{cases} \frac{4}{3}\pi GPR^3 : sphere \\ 2\pi GPR^2 : Cylinder \end{cases}$$

R Radius of the sphere or cylinder,
Z Depth of sphere or cylinder,
X Horizontal distance (see Fig. 6.5),
G Universal gravity constant,
P Contrast density,
q Shape factor (for sphere q = 1.5, for cylinder = 1).

Features F_1, F_2, F_3, F_4 and F_5 are calculated for various values of depth and radius throughout a domain of {[R_{min}, R_{max}], [Z_{min}, Z_{max}]} and so the MANFIS network should be able to detect all with a radius between R_{min}, R_{max} and with a depth between Z_{min}, Z_{max}.

For modeling purposes the gravity data are normalized in order to show the variation as a value between zero and one. The data are divided into 2 parts and the first part used for simulation/training of the network model, while the second part is for validation and testing the model (Fig. 6.6). In considering the data for modeling, the data are randomly sampled along with additive noise created by adding 5% Gaussian noise to the synthetic data, to make the model more realistic.

The MANFIS used in this study, was a system with 5 inputs and two outputs; the inputs are F_1, F_2, F_3, F_4 and F_5 and the outputs are R, Z, radius and depth of the cavity respectively (Fig. 6.7). This MANFIS is a combination of two ANFIS's each of them with 5 inputs and one output. The inputs are the same but the outputs differ. We used the Sugeno ANFIS architecture with the fixed bell membership function as input MF and a linear function as the output MF with the hybrid method for training. To find the optimized MF types for these inputs various MF types were tested and statistical error indexes were calculated for each MF types. The comparison of results shows (after 10 epochs) that Gaussian membership function is optimal with a much superior performance (Fig. 6.8 and Table 6.3).

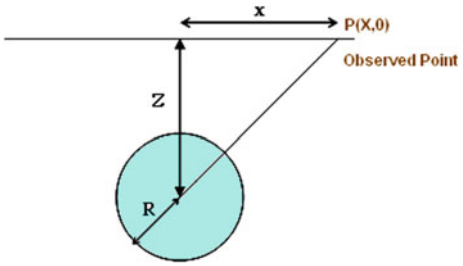


Fig. 6.5 Values of R, Z, x for a cavity under a gravity profile

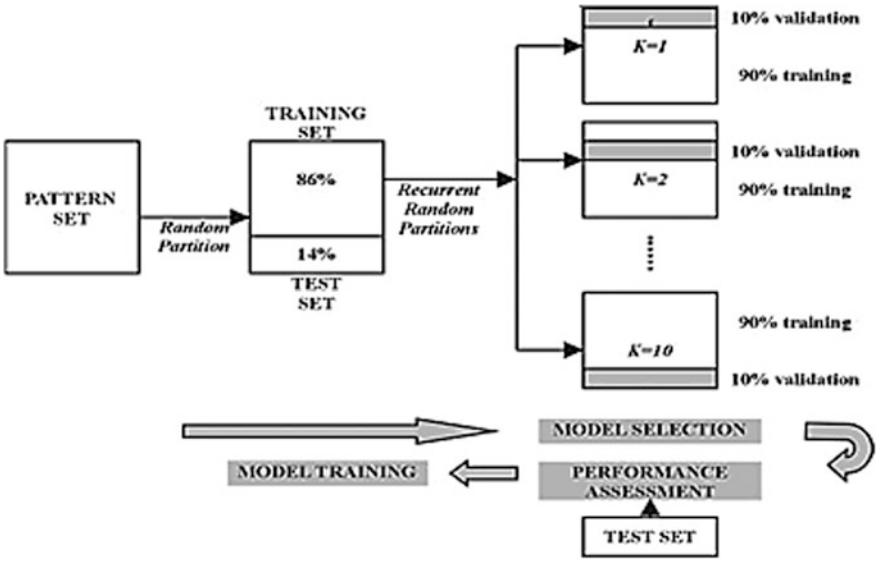


Fig. 6.6 Schematic diagram of preparing training, testing and validation sets for MANFIS (Hajian 2012)

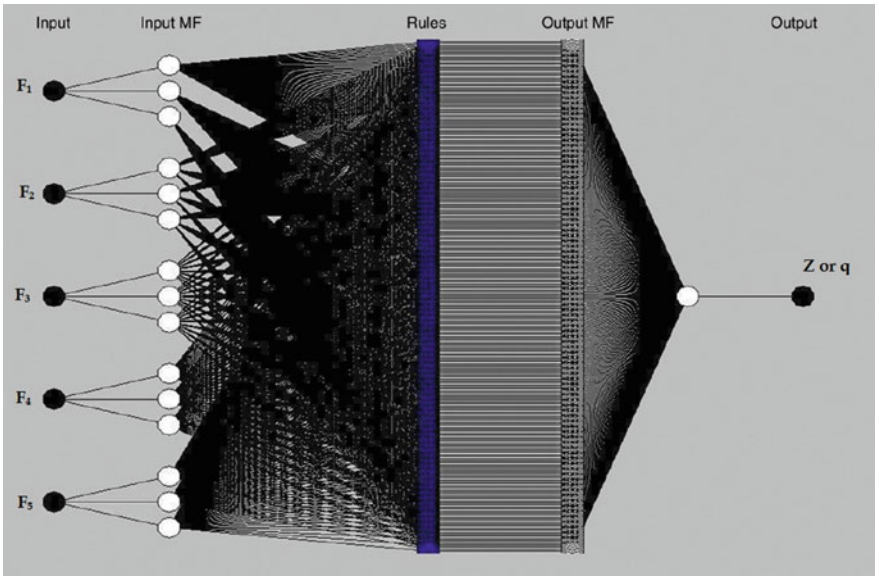


Fig. 6.7 Schematic structure of each of ANFIS models, with input MF, rules and output Mf with the links of each layer to next layer, used for depth (z) and shape factor (q) estimation (Hajian et al. 2012)

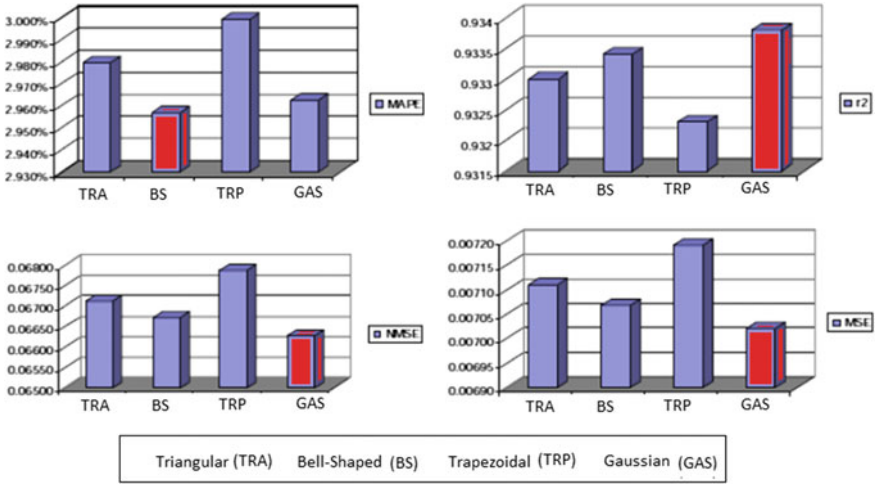


Fig. 6.8 Bar-gram of the statistical indexes for different MF (left-top: MAPE, right-top, R^2, bottom-left: NMSE, bottom-right: MSE, Hajian 2012)

Table 6.3 The statistical errors of model M_{10} for different membership functions (MF) (Hajian and Zomorrodian 2016)

MAPE (%)	R^2	NMSE	MSE	Model	MF
2.979	0.9330	0.06711	0.00711	M_{10}	Triangular
2.957	0.9334	0.06672	0.00707		Bell-shaped
2.999	0.9323	0.06787	0.00719		Trapezoidal
2.963	0.9338	0.06625	0.00702		Gaussian

ANFIS adjusts the membership function parameters specifying the shapes and partition of the membership function and the hybrid approach was used in the present study for training. The network model was well trained requiring only 22 epochs to reach an RMSE error (Root Mean Square Error) below 0.00004 (Fig. 6.9).

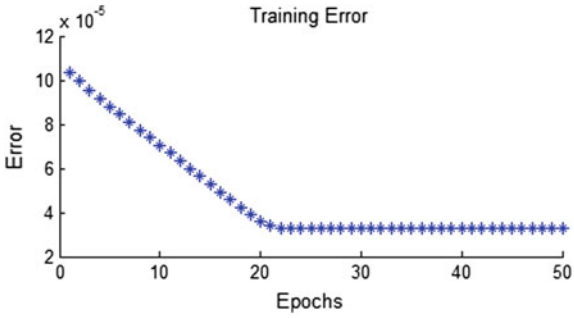


Fig. 6.9 ANFIS network training for 50 epochs, best training happened in epoch 22 (Hajian 2012)

The model validation was also tested with the input vectors from other input/output data sets on which it had not been trained, to see how well the FIS model performs. Another type of data set, referred to as the checking data set can be used for model validation in MANFIS and is used to limit the potential for the model over-fitting the data. When this checking data was used as well as training data, the FIS model is selected to have its parameters corresponding to the minimum checking data model error.

6.1.5 *Test of MANFIS in Present of Noise and for Real Data*

After the training process of MANFIS model, the designed MANFIS model was tested with noisy data with 20% of random noise. The R^2 of outputs results are represented in Figs. 6.10 and 6.11, for depth and shape factor estimation, respectively, which shows the robustness of the method in presence of noise.

The MANFIS model was also tested for real data. The gravity data was from a data set measured on a site selected in Freeport Grand Bahama underlain by coral interrupted by significant inter-connected cave systems and the residual gravity anomaly is depicted in Fig. 6.12. We compared the depth estimation of MANFIS

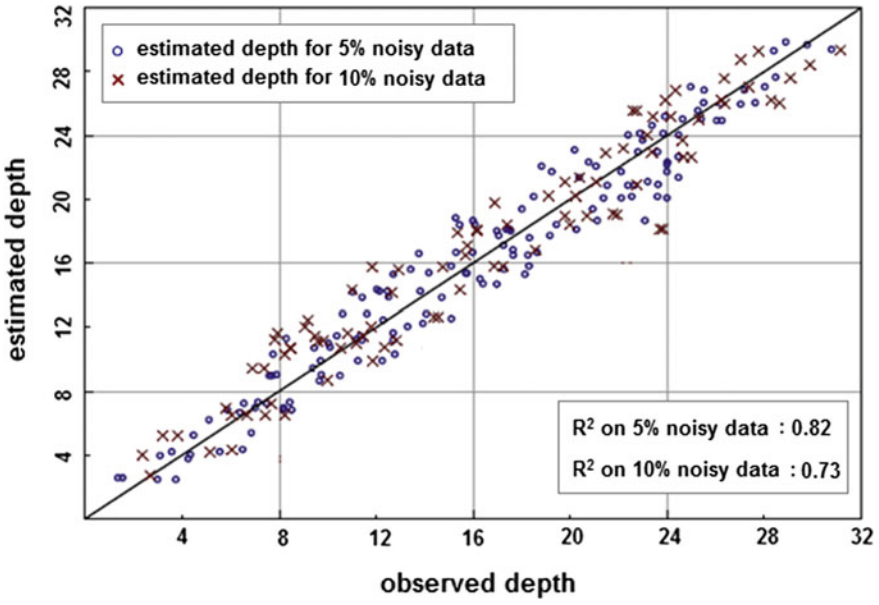


Fig. 6.10 Estimated depth versus observed depth for 5 and 10% noisy data (Hajian 2012)

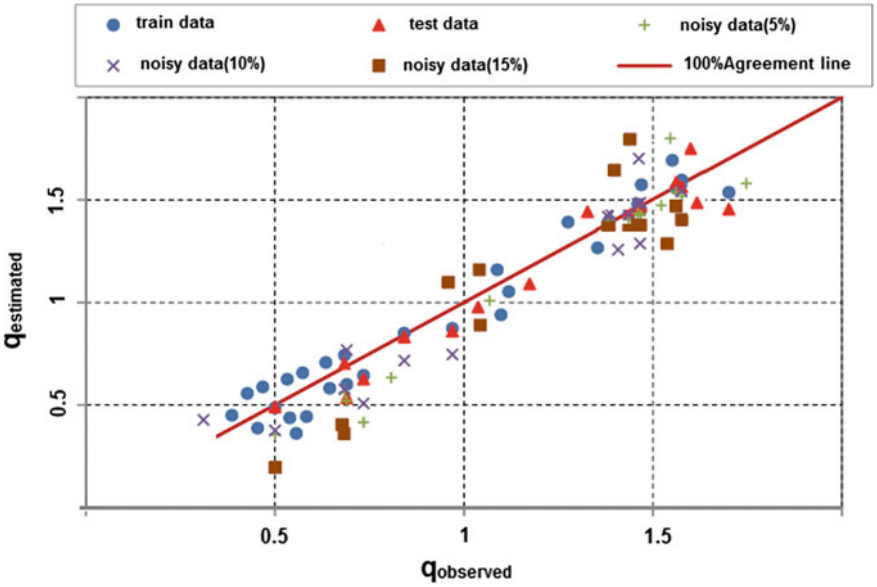


Fig. 6.11 Estimated shape factor versus observed shape factor for training, test and 5, 10% noisy data (Hajian 2012)

with the MLP (Multi-Layer Perceptron) neural network method. Five principal profiles were chosen, which are presented with black lines in Fig. 6.12, and we calculated the features F_1, F_2, F_3, F_4 and F_5 which were applied to the trained MANFIS as inputs. The estimated parameters are compared with MLP results (Table 6.4). From the available excavations, MANFIS outputs are very near to the real depth and radius of the available cavities. The model works well with any data sets which fall within the limits of the training range, used here. If it becomes unsuccessful it can be retrained for different possible environments to make it more general.

6.2 Surface Settlement Prediction Using ANFIS for a Metro Tunnel

6.2.1 ANFIS Structure

As explained in Chap. 2 Sect. 2.7, Hajian et al. (2014) used Neural Networks to predict the surface settlement for a metro tunnel using geological, geophysical and geotechnical data. They also used a fuzzy—neural network in their study which was designed using ANFIS implemented in the MATLAB software. The database includes 346 data related to cross-sections of the studied route, and out of these

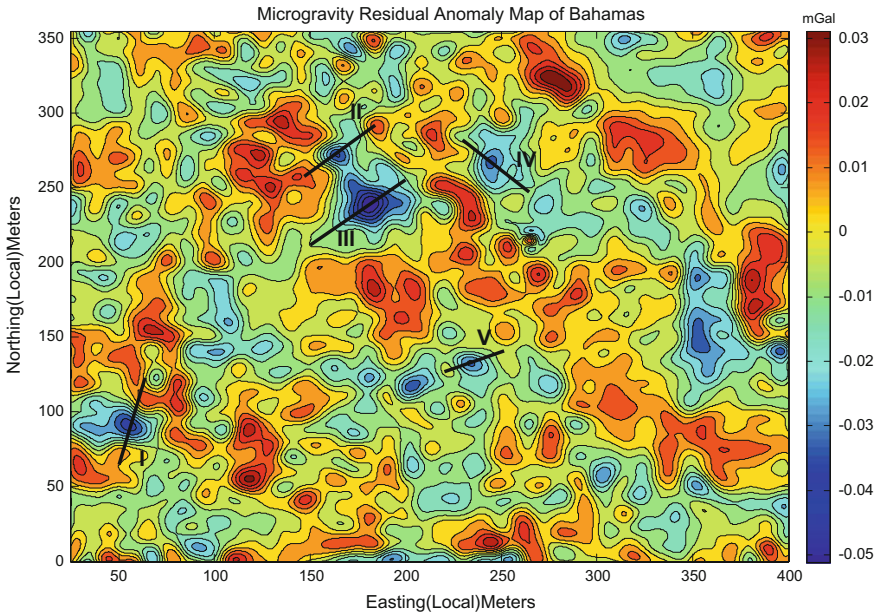


Fig. 6.12 Residual gravity anomaly of Bahamas free PORT SITE with the selected principal profiles (Hajian et al. 2011)

Table 6.4 Interpreted cavity depth and radius through MANFIS method for principal selected gravity profiles from the Grand Bahamas site, compared to the borehole results and MLP network method (Hajian et al. 2011)

Selected principal profile	Borehole results depth (m) to		Results of MLP depth (m) to		Results of MANFIS depth (m) to	
	Top	Bottom	Top	Bottom	Top	Bottom
Profile I	2.74	5.79	3.36	6.52	2.40	5.50
Profile II	13.72	6.76	12.54	15.66	13.48	16.61
Profile III	12.50	6.64	11.96	5.48	12.14	15.94
Profile IV	13.25	15.75	12.63	5.21	12.73	15.47
Profile V	13	16	13.75	6.59	13.25	16.31

data, 225 data were used for training, 52 were used for testing and 69 data were used for checking and all were randomly selected. The number and description of the inputs and outputs of system were presented in Table 1 (to know the input arrays see Chap. 2 Sect. 2.7).

6.2.2 ANFIS Training and Testing

After data normalization, the data were loaded as the inputs and output of the system. In order to achieve lower errors for the state of 2 Multiplying factors (MFs) for inputs in this case study, the primary structure of the fuzzy inference system was generated with 2 MFs for all inputs as well as different types of MF for inputs and a constant MF for output, and then these models of the system were trained for 40 epochs using a hybrid optimizing method. The values of errors resulting from training and testing of the system are presented in Table 6.5.

As is evident, from the results obtained from the training and testing of the system, the application of the gauss2mf as the MF for inputs gives fewer errors in testing compared with the others, especially in testing; and therefore, this model has been used as the optimal model in the training of system. Figure 6.13 shows the MF plots for input #1 as a sample of the MF of inputs.

The predicted amounts of settlement obtained using inputs related to each cross-section produced with ANFIS are shown in Fig. 6.14.

Figure 6.15 shows the surface plot between the inputs #1 and #4 and the output. The amounts of settlement predicted using ANFIS versus the amounts of settlement

Table 6.5 The results of the training and testing of ANFIS (Hajian et al. 2014)

Number of MFs for inputs	Type of MF for inputs	Type of MF for output	Error at the end of training	Average testing error in training	Average testing error in testing	Average testing error in checking
2	Gbellmf	Constant	0.0037270	0.0037271	0.0064418	0.0031027
2	gaussmf	Constant	0.0032390	0.0040371	0.0050826	0.0032760
2	gauss2mf	Constant	0.0040860	0.0040857	0.0050442	0.0028206
2	Primf	Constant	0.0089500	0.0089502	0.0061178	0.0024701
2	Dsigmf	Constant	0.0042863	0.0042863	0.0060775	0.0027071
2	Psigmf	Constant	0.0042863	0.0042863	0.0060775	0.0027071

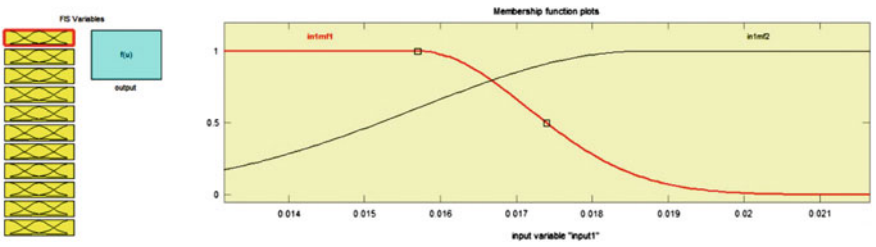


Fig. 6.13 The MF plots for input #1 (Rezazadeh Anbarani 2014)

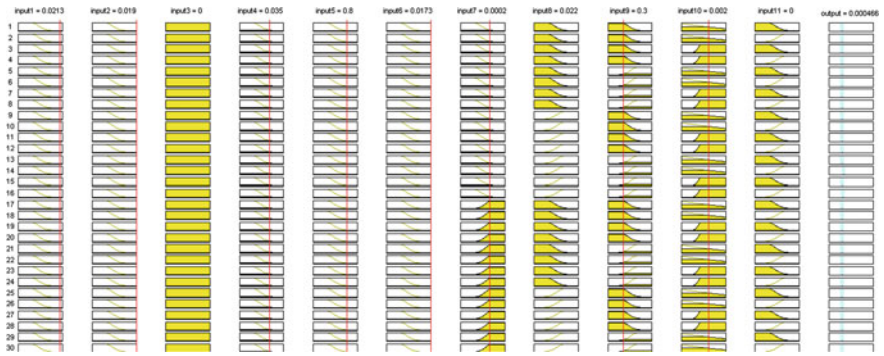


Fig. 6.14 Diagram of the rules created by ANFIS for settlement (output) prediction (Rezazadeh Anbarani 2014)

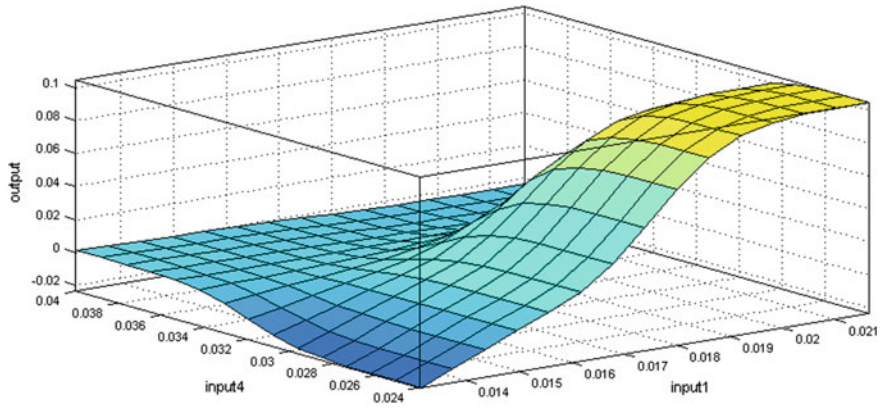


Fig. 6.15 A sample of the surface plot between inputs and output; the unit of inputs and output is in mm (Hajian et al. 2014)

measured using precise surveying instrumentation are plotted and show a good correlation between the predicted and actual amounts. Figure 6.16 illustrates the correlation coefficient between the actual and predicted data. The ANFIS results were also compared to that of ANN and FEM (Finite Element Method) and the measured settlement (Table 6.6). This showed ANFIS was significantly better especially than the FEM modeling.

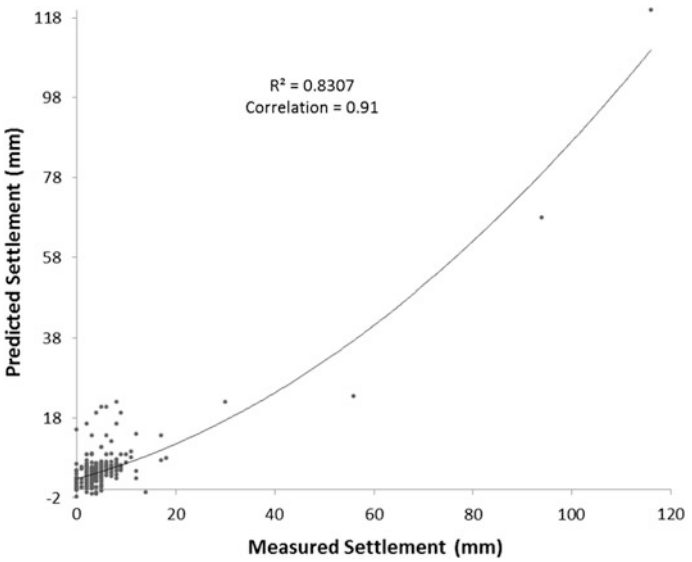


Fig. 6.16 The actual amounts of settlement versus the amounts of settlement predicted using ANFIS (Hajian et al. 2014)

Table 6.6 The actual amounts of surface settlement and the ones estimated using ANN, ANFIS and FEM (Hajian et al. 2014)

Chainage	Settlement measured by instrumentation (mm)	Settlement predicted using ANN (mm)	Settlement predicted using ANFIS (mm)	Settlement calculated using FEM (mm)
1 + 050	3.00	3.40	3.47	8.05
2 + 090	5.00	4.60	4.71	5.64
13 + 160	0.00	0.40	0.58	0.017

6.2.3 Conclusion

Measurement of surface settlement during tunneling is essential to mitigate hazard and/or financial losses. Powerful tools are available for the estimation of this surface settlement. Applying ANNs, which use the numerical data to predict the results, is a powerful tool in this field. FEM can be an exact numerical method for surface settlement calculation but the combination of fuzzy inference systems and ANNs, has the potential to predict the results very accurately, and rapidly and provide a more powerful tool known as ANFIS. By checking the observed data with the

results of ANN and ANFIS, it was seen that the correlation coefficient between the actual amounts of settlement and the predicted ones is 0.86 for ANN and 0.91 for the ANFIS prediction respectively. These analyses showed the higher correlation resulting from ANFIS in comparison with ANN, because of the greater adaptability between the inputs and output in ANFIS due to the combination of simultaneous fuzzy logic and AN. The results obtained from the analysis of three models related to some cross-sections of the route are acceptable. The comparison of the results obtained from the three methods used in this study indicated that the amounts of surface settlement estimated using all methods are close to the actual ones measured by instrumentation in most cross-sections of the route; and three cross-sections have been presented here as confirmation.

6.3 The Use of the ANFIS Method for the Characterization of North Sea Reservoirs

6.3.1 Introduction

Mojeddifar et al. (2014) compared three versions of adaptive neuro-fuzzy inference system (ANFIS) algorithms and a pseudo-forward equation (PFE) to characterize a North Sea reservoir (F3 block) using seismic reflection data. Three different ANFIS models were constructed using clustering methods:

- *Grid partitioning (GP)*
- *Subtractive clustering method (SCM)*
- *Fuzzy c-means clustering (FCM).*

They used an experimental equation, called PFE based on similarity attributes, to estimate porosity values of the reservoir to compare with the neuro-fuzzy technique results.

In this section we briefly explain their work and the results as a good example of where ANFIS is not, in this case, the best solution for this particular geophysical problem, because they showed that ANFIS is unable to estimate the porosity adequately. In contrast, the ability of PFE to detect geological structures such as faults (gas chimney), folds (over a salt dome), and bright spots, together with porosity estimates of sandstone reservoirs, might be very useful in planning the drilling target locations and so they proposed that the PFE they had developed could be an applicable technique for characterizing the reservoirs of the F3 block.

6.3.2 Literature Review

The characterization of the spatial variations in petrophysical parameters between exploratory wells in a hydrocarbon reservoir has an important role to play in petroleum engineering. Hence, laboratory measurements of core properties, the interpretation of geophysical well logs and the inversion of seismic attributes can provide valuable estimate of the physical properties of any reservoir (Bhatt and Helle 2002).

The inversion of seismic data to give acoustic impedance (AI, the product of density and velocity) is widely used in hydrocarbon exploration for the estimation of petro-physical properties and is often used as a proxy for porosity based on empirical relationships between acoustic impedance and porosity.

However because the compaction model varies both laterally and vertically, the relationship differs from area to area and in many cases, porosity cannot be estimated directly from acoustic impedance using a single transform function (Anderson 1996). Schults et al. (1994) suggested using multiple seismic attributes to estimate log properties aside from well control. Several data integration techniques (kriging or neural networks) have been used to derive petrophysical properties directly from seismic attributes. A short review of the recent research on intelligent and soft computing methods used for reservoir characterization is listed in Table 6.7.

Modern artificial intelligent methods such as neuro-fuzzy systems may be useful for the prediction of petro-physical properties in addition to conventional techniques with the following advantages for reservoir characterization:

- Fast, reliable and low-cost solutions.
- Dynamic, non-linear and noisy data can be handled especially when the underlying physical relations are very complex and not fully understood.

PFE (The Pseudo-forward Equation) transforms the similarity attributes of a sandstone reservoir to porosity values and Mojeddifar et al. (2014) developed the structure of a PFE using a data set from the F3 gas reservoir block in the North Sea.

6.3.3 Geological Setting

Chalky sediments were deposited in the F3 block in Dutch sector of the North Sea at the end of the early Paleocene, but the Laramide tectonic phase produced a sudden increase in the supply of silica-clastics, which terminated the deposition of the Chalk (Ziegler 1990). The Neogene fluvio-deltaic system in the southern North Sea is shown in Fig. 6.17.

Table 6.7 Some of the recent research on intelligent and soft computing methods used for reservoir characterization

Method	Application	Location	Year	Researcher(s)
Fuzzy logic and artificial neural networks (ANN)	Predict core properties from well logs	Offshore Korea	2004	Lim, J. S.
Neural networks	Predict lithology from seismic properties	–	2007	Singh et al.
Multi-attribute transforms	Porosity and lithologic estimation using rock physics	Balcon Field, Colombia	2007	Calderon, J. E., Castagna, J.
Neural network	Shale stringers in a heavy oil reservoir	PanCanadian's Christina Lake	2002	Tonn R.
Neural networks	From 3D seismic attributes to pseudo-well-log volumes	Eastern Venezuela	2002	Banchs, R. E., Michelena, R. J. Rafael, E. B., Reinaldo, J. M.
Neural networks	Porosity and permeability prediction from wireline logs	North Sea	2001	Helle, H. B., Bhatt, A., Ursin, B.
Multiple regression analysis	Spatial prediction of petro-physical properties through integration of petro-physical measurements and 3D seismic observations	'XLD' Field, Niger Delta	2013	Adekanle, A., Enikanselu, P. A.
Dynamic radial basis function network	Predict the reservoir's properties from seismic attributes	–	2011	Lei, L., Wei, X., Shifan, Z., Zhonghong, W.
3D petrophysical modeling	Interpretation of log properties from complex seismic attributes and limited well log data	–	2011	Eftekharifar, M., Han, D. H.
Three types of neural network	Prediction of porosity from seismic attributes	Persian Gulf	2011	Hosseini, A., Ziaii, M., KamkarRouhani, A., Roshandel, A., Gholami, R., Hanachi, J.
Multi-attribute transform and probabilistic neural network	Generate effective porosity volume	Lower Brushy Canyon channeled sandstones Southeast New Mexico	2001	Leiphart, D. J., Hart, B. S.

(continued)

Table 6.7 (continued)

Method	Application	Location	Year	Researcher(s)
Multi-attribute transformations	Porosity prediction from seismic data	Auger field, Gulf of Mexico	2009	Valenti, J. C. A. F.
Genetic algorithms	Porosity and permeability prediction of oil reservoirs using seismic data	An oil reservoir in the Norne field, Norway	2013	Joonaki, E., Ghanaatian, S. H., Zargar, G. H.

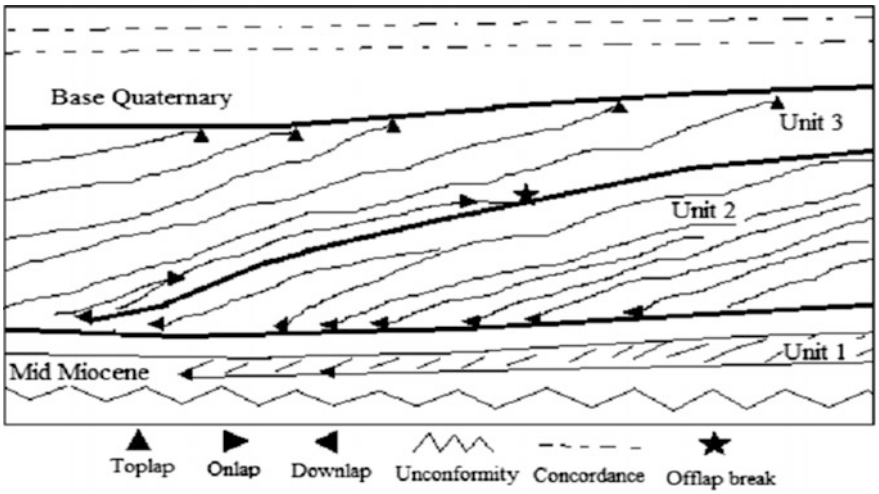


Fig. 6.17 Sketch of the Neogenefluvio-deltaic system in the southern North Sea (after Steeghs et al. 2000)

6.3.4 Data Set

A seismic survey of approximately 16 × 23 km^2 has been made public in F3 block consisting of 646 in-lines and 947 cross-lines and is described by Aminzadeh and Groot (2004). The line spacing was 25 m for both in-lines and cross-lines at a sample rate of 1 ms. Standard seismic data processing procedures were applied to the data. Sonic and gamma ray logs data were also available from four wells in the area and density logs were reconstructed from the sonic logs using neural network techniques by dGB Earth and these were also used to calculate porosity logs for the four wells. A seismic line connecting wells F03-04, F03-02, F02-01 and F06-01 is shown in Fig. 6.18, showing the main horizon correlations.

Wells F02-01 and F06-01 lie in the south-western part of the F3 block, at the bottom of the clinoform sequence. Well F03-02 is located to the north at the top set of the sequence. Well F03-04 lies in the eastern part of the block (Tetyukhina et al. 2008).

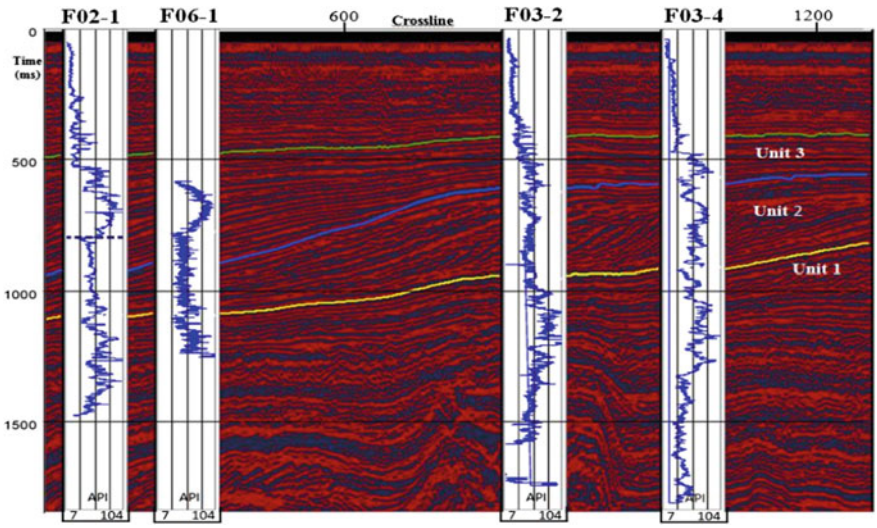


Fig. 6.18 The seismic section derived from original seismic data (in-line 425), showing the location of wells and the gamma ray logs for every well (Mojeddifar et al. 2014)

Figure 6.19a shows the seismic cross-section for in in-line 441, and well F03-4 which was selected to interpret the various layers in the target zone.

'Truncation 1' and 'MFS4' (white dashed lines) mark the position of the target zone interval in the seismic data. Figure 6.19b shows the cross-plot of the gamma ray values against acoustic impedance values, computed from the velocity and density logs (Mojeddifar et al. 2014).

6.3.5 Preprocessing to Select the Most Suitable Attributes

Good correlations between seismic attributes and porosity are seen often enough to demonstrate seismic attributes can be applied as a proxy for porosity in reservoir characterization. Good correlations between seismic attributes and porosity, derived from the density logs of available wells have been demonstrated from statistical studies of more than 15 attributes, which are presented in Table 6.8.

According to Table 6.8, the highlighted attributes are considered to be the optimal ones to predict porosity as the output in linearity and non-linearity mode.

ANFIS (GP), ANFIS (SCM) and ANFIS (FCM), Genfis1, Genfis2 and Genfis3 commands were used to generate initial structures of the Sugeno fuzzy inference system using grid partition, subtractive clustering and fuzzy c-mean clustering algorithms.

Figure 6.19 shows the fuzzy rule architecture for various ANFIS algorithms and Table 6.9 shows the ANFIS algorithms used in the Mojeddifar et al. (2014) study.

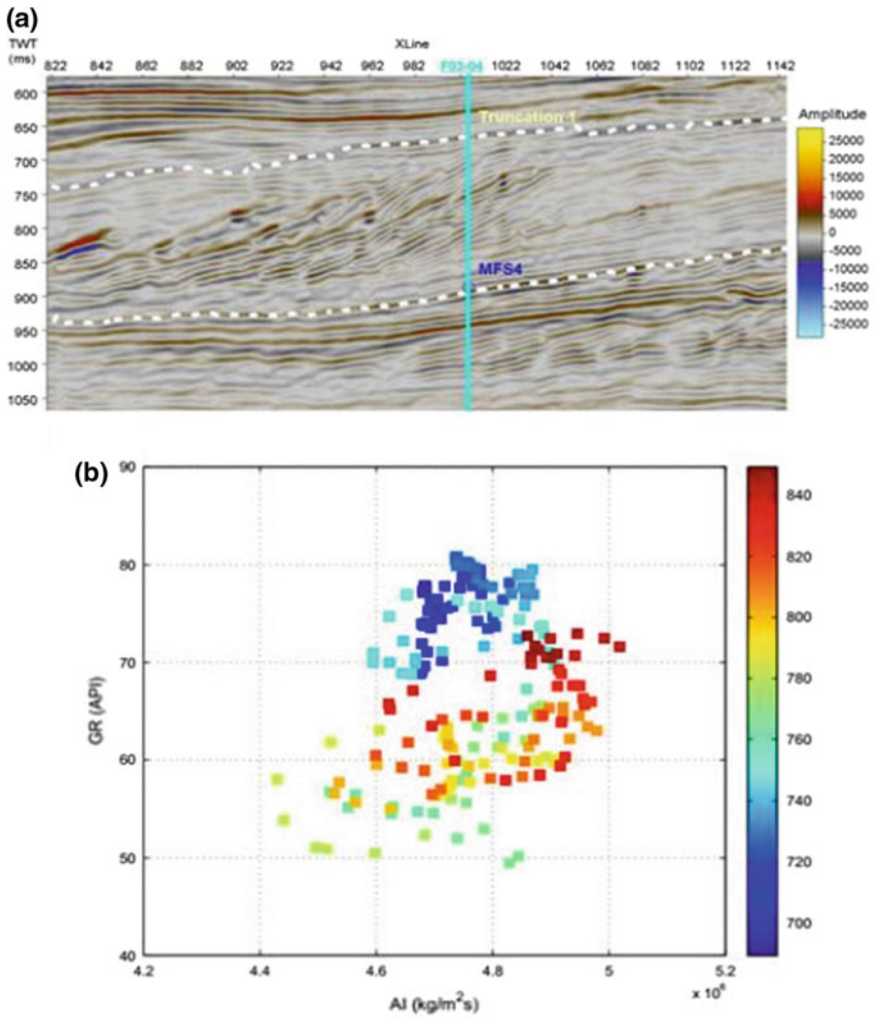


Fig. 6.19 **a** The seismic section (in-line 441) used for the cross-plot analysis (Tetyukhina et al. 2008); **b** a cross-plot of the gamma ray values against acoustic impedances values within the target zone (Tetyukhina et al. 2008)

The ANFIS algorithms were trained with 1427 values and the changes in the final (post-training) Gaussian shaped membership functions for the input parameters are shown in Fig. 6.20. The training data set was extracted from the sand-rich sediments of the three wells (F02-1, F06-1, F03-2) where gamma ray values were less than 70 API.

Figures 6.21 and 6.22 illustrate the results of the three ANFIS algorithms applied to the validation and testing sets showing the slight superiority of ANFIS (SCM) over ANFIS (GP and FCM). The R-squared coefficient, (strength of the

Table 6.8 List of attributes used and their correlation coefficients with porosity values (Mojeddifar et al. 2014)

Descriptive statistics							
Studied attributes	Number of points	Minimum	Maximum	Average	Std. deviation	Correlation coefficient	Sig.[a]
Porosity	927	0.221	0.344	0.286638	0.0236882	1	
Energy	927	425250	16411000	4163775	3456700.9	**−0.23**	**0.000**
Envelope	927	139.5344	8602.5	2411.165	1748.5479	**−0.26**	**0.000**
Instantaneous phase	927	−3.3283	3.1552	−0.191617	1.7112253	0.017	0.457
Instantaneous frequency	927	−178.51	689.4833	209.9355	80.7153086	−0.101	0.000
Hilbert transformation	927	−7365.5	6536.3	−36.39097	1992.0581	0.066	0.004
Amplitude 1st derivative	927	−664760	548200	−3885.769	198827.97	−0.036	0.115
Amplitude 2nd derivative	927	−111040000	126270000	−4254.58	34531237	−0.023	0.305
Cosine phase	927	−1.0178	1.0477	−0.056109	0.6814381	−0.107	0.000
Envelope weighted phase	927	−3.0598	2.4849	−0.101663	1.0937449	−0.034	0.137
Envelope weighted frequency	927	−14.5334	305.4846	203.4539	51.1291469	−0.157	0.000
Phase acceleration	927	−73638	64199	49.61784	12914.86	0.064	0.005
Thin bed indicator	927	−363.73	415.4329	6.48161	48.7964219	−0.002	0.918
Bandwidth	927	−0.5155	108.298	12.81416	11.9160097	−0.159	0.000
Q factor	927	−5232.1	5676	−41.45138	398.0633953	−0.022	0.333
Instantaneous rotate phase	927	−6536.3	7365.5	36.39097	1992.0581	−0.066	0.004
Spectral decomposition	927	304.8516	13142	4835.486	2861.6049	**−0.307**	**0.000**
Similarity	927	0.6861	0.9488	0.871362	0.0608897	**−0.45**	**0.000**

[a]The sig. is the statistical significance of the result. The result is significant if it is smaller than 5%

Table 6.9 The specification of ANFIS models developed by MATLAB (after Mojeddifar et al. 2014)

ANFIS details	ANFIS (GP)	ANFIS (SCM)	ANFIS (FCM)
Number of nodes	55	117	67
Number of linear parameters	80	55	30
Number of nonlinear parameters	16	88	48
Total number of parameters	96	143	78
Number of training data pairs	1427	1427	1427
Number of fuzzy rules	16	11	6

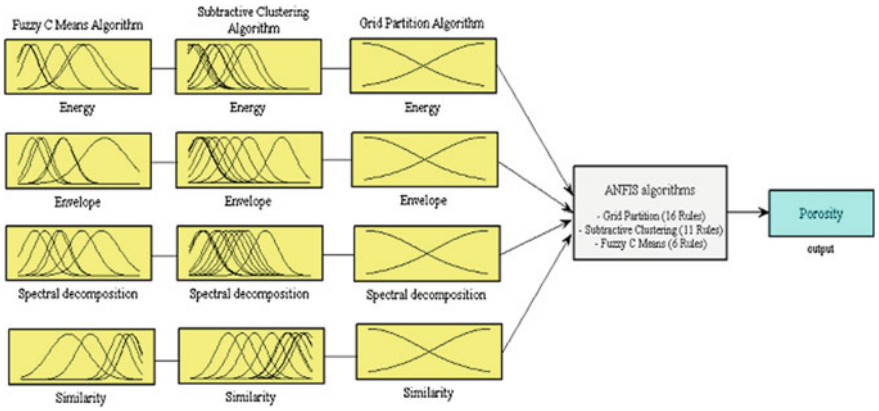


Fig. 6.20 Architecture of ANFIS, based on GP, SCM and FCM (Mojeddifar et al. 2014)

linear relationship between the two (actual and predicted) R = 0.7935 which might seem to indicate that ANFIS might be a good method to estimate porosity distribution. However when ANFIS is applied to the testing set the correlation coefficient falls to 0.252 indicating that the results do not match the experimental values.

Mojeddifar et al. (2014) compared the PFE method ANFIS (for more details about PFE method see Mojeddifar et al. 2014). Figure 6.23 plots the observed values versus the predicted outputs for the PFE model. It seems that both developed models perform with acceptable precision near to the training wells as might be predicted. As the testing data used are exactly the same, it seems that the PFE experimental method predicts the porosity values further away from the training wells, better than the ANFIS algorithms. Therefore, there is a considerable performance gap between training and testing locations for ANFIS algorithms, while the proposed experimental model is superior, while not itself perfect.

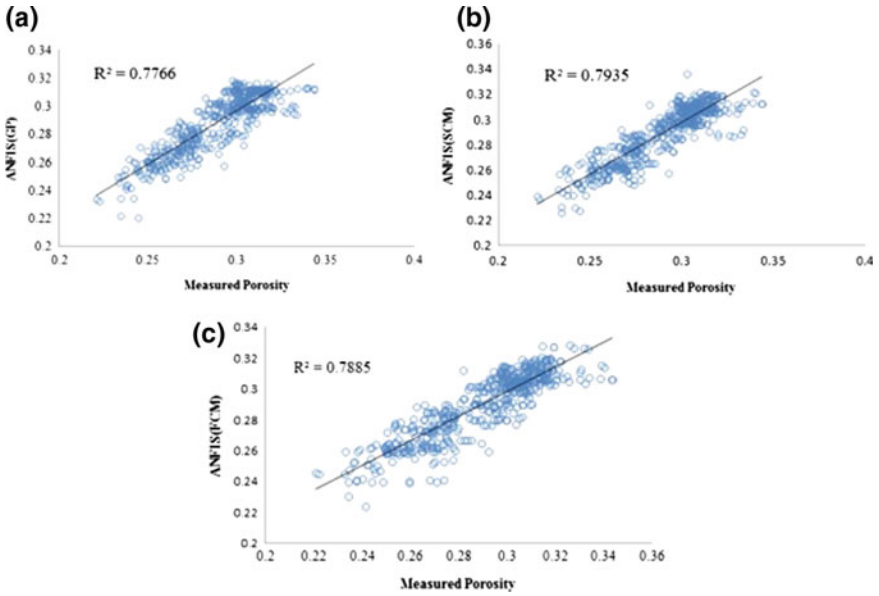


Fig. 6.21 Comparison of the measured and predicted porosity of the validation set: **a** grid partition; **b** subtractive clustering; **c** fuzzy c-means clustering (Mojeddifar et al. 2014)

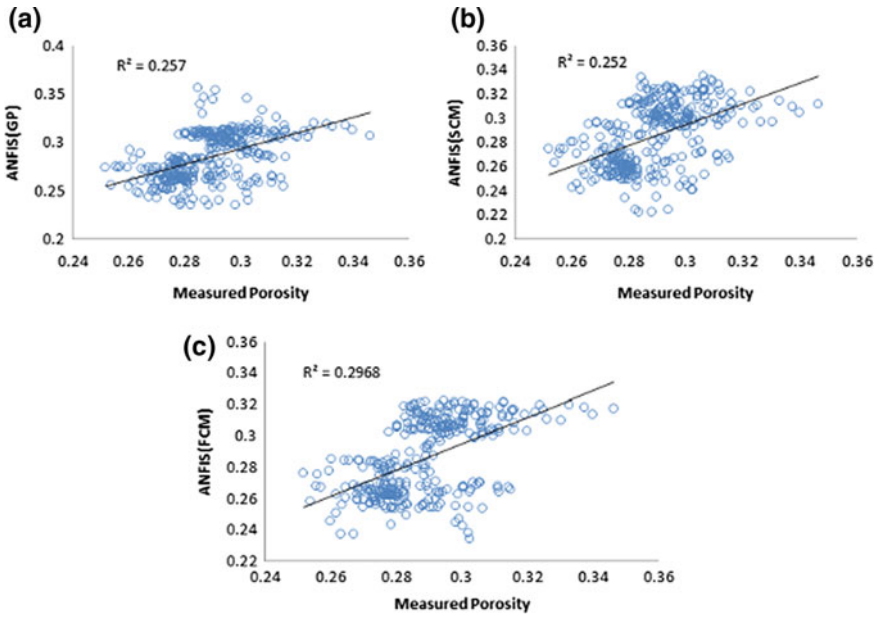


Fig. 6.22 Comparison of the measured and the predicted porosity of testing set: **a** grid partition; **b** subtractive clustering; **c** fuzzy c-means clustering (Mojeddifar et al. 2014)

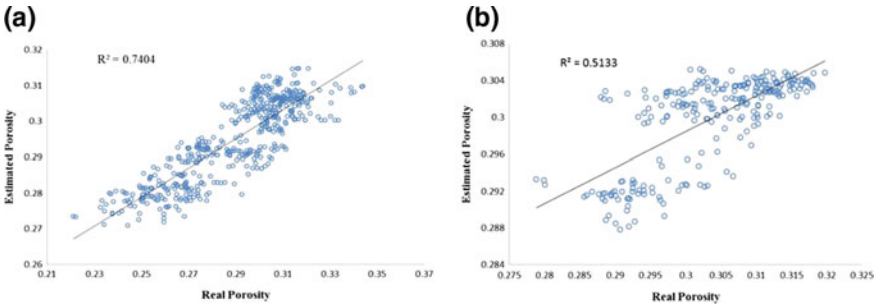


Fig. 6.23 Comparison of the measured and the predicted porosity by PFE: **a** validation set; **b** testing set (Mojeddifar et al. 2014)

6.3.6 Reservoir Characterization Using ANFIS and PFE

The interpretation of the target zone depends critically on the predicted spatial distribution of porosity from the two models. Figures 6.24 and 6.25 show the outputs of (in-lines 244 and 442) and Figs. 6.26a, b show the results of the PFE model for the same sections.

Mojeddifar et al. (2014) suggest that a possible cause of the inaccuracy in the ANFIS algorithms is the lack of sufficient available well data to provide adequate generality to the trained models in the learning process. However, inaccuracies can also be seen in the PFE model, albeit with less intensity. The PFE model was only developed with the data set from sandy sediments, and so is likely to perform better for the sand units than the shale layers. In Fig. 6.26a, this seems to be true at spot 'A' where there are deposits of shale sediments but at spot 'B' the model seems to estimate the estimate porosity distribution correctly. However, despite these individual problematic areas the PFE model has performed reasonable well within the gas-bearing sand reservoir of the F3 block, but we note that ANFIS algorithms seem to require a higher number of wells in order to develop models with credible accuracy (Mojeddifar et al. 2014).

6.4 Neuro-Fuzzy Approach for the Prediction of Longitudinal Wave Velocity

6.4.1 Introduction

P-wave amplitude is a proxy for peak particle velocity during blasting in a mine and is an important control parameter for minimizing damage caused by ground vibrations. Fracture propagation is influenced both by the physico-mechanical parameters of the rock and also on the compressional wave velocity. Verma and Singh (2012) applied an ANFIS to predict P-wave wave velocity for different rocks.

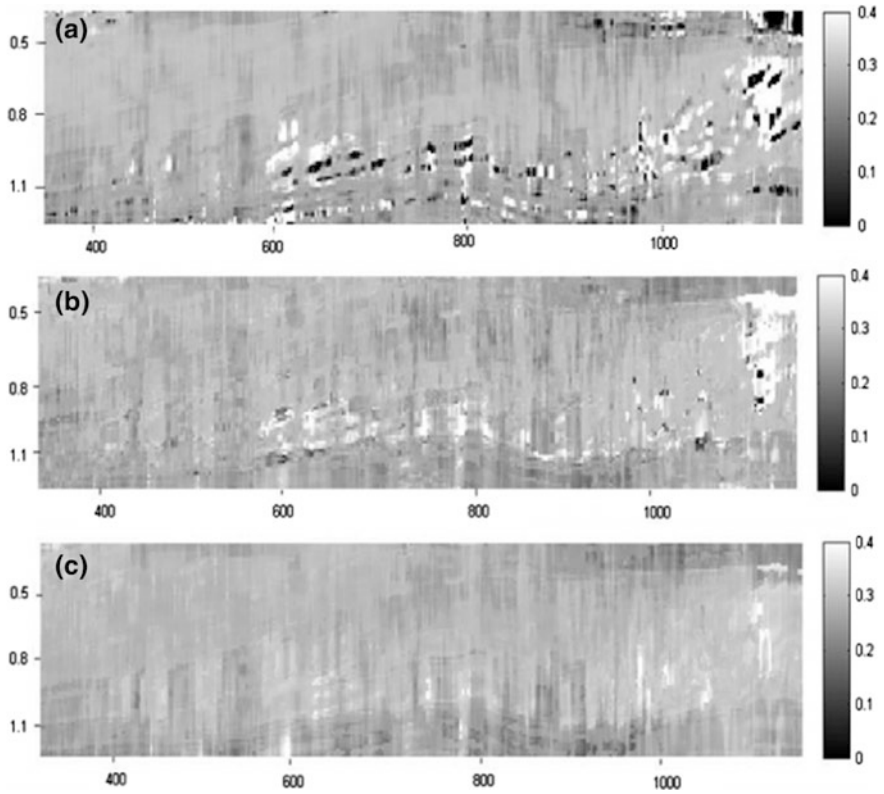


Fig. 6.24 The seismic sections of porosity (line 244) estimated by ANFIS: **a** grid partition; **b** subtractive clustering; **c** fuzzy c-means clustering (Mojeddifar et al. 2014)

Input 1 is hardness,
Input 2 is porosity,
Input 3 is absorption ratio,
Input 4 is compressive strength of rock,
Input 5 is density and
Input 6 is the fracture roughness coefficient,

and the output is the predicted p-wave velocity calculated using a neuro-fuzzy model. The structure of the ANFIS model is shown in Fig. 6.27, with four input parameters, one output parameter and five rules.

6.4.2 Training of the Neuro-Fuzzy Model

Verma and Singh (2012) use subtractive clustering to identify the natural groupings of data within large numbers of dataset in order to produce a concise representation of a system's behavior.

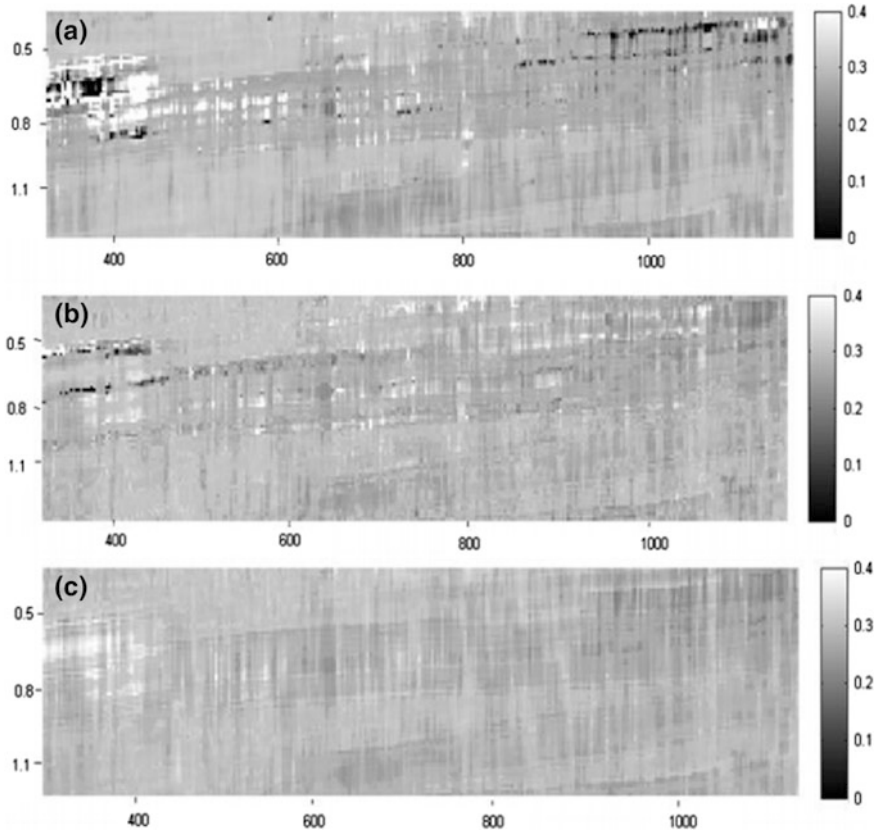


Fig. 6.25 The seismic sections of porosity (line 442) estimated by ANFIS: **a** grid partition; **b** subtractive clustering; **c** fuzzy c-means clustering (Mojeddifar et al. 2014)

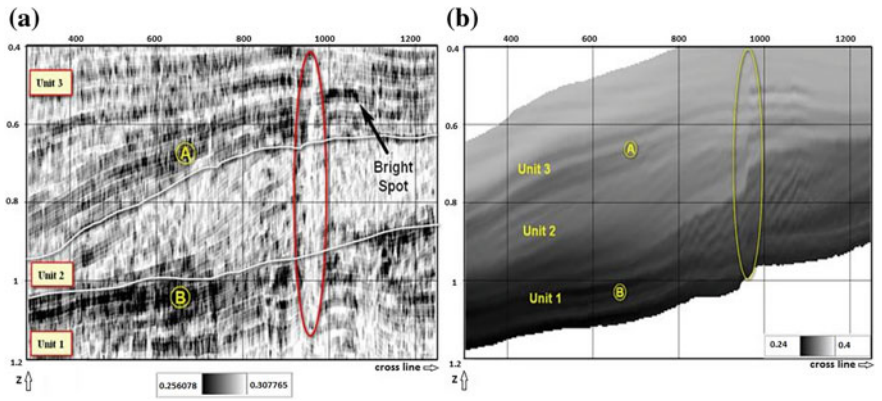


Fig. 6.26 **a** The seismic section of porosity (in-line 244) estimated by PFE; **b** the seismic section of porosity (in-line 244) provided by dGB Earth Sciences Company (Mojeddifar et al. 2014)

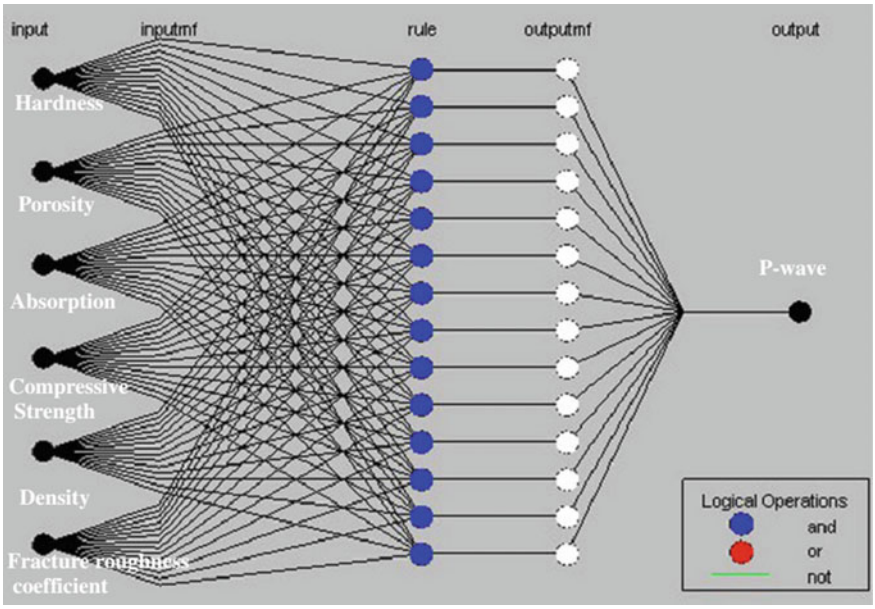


Fig. 6.27 ANFIS structure for the PR model, neuro-fuzzy, with 6 input parameters and 14 rules. Layer-1 represents inputs, layer-2 input membership function, later-3 rules, layer-4 output membership function, layer-5 weighted sum output and layer-6 output (Verma and Singh 2012)

They divided the 136 available datasets into three parts:

- 100 training sets
- 26 testing sets
- 10 datasets were selected for checking the model.

The membership function for each input was tuned using a hybrid method:

- input membership function parameters were tuned using back propagation
- output membership function parameters were tuned using least square (Fig. 6.28a–f).

A measure of how well the FIS (fuzzy inference system) system is modeling the input/output data is computed using a gradient vector of the rate of convergence.

- the number of nodes in the training data was 205
- the number of linear parameters was 98
- the number nonlinear parameters 168.

The hypothesized initial number of membership functions and the type used for each input were 10 and Gaussian, respectively. The hypothesized FIS model is then trained and the training goal reached after 30 epochs.

The final configuration for the FIS is listed in Tables 6.10 and 6.11.

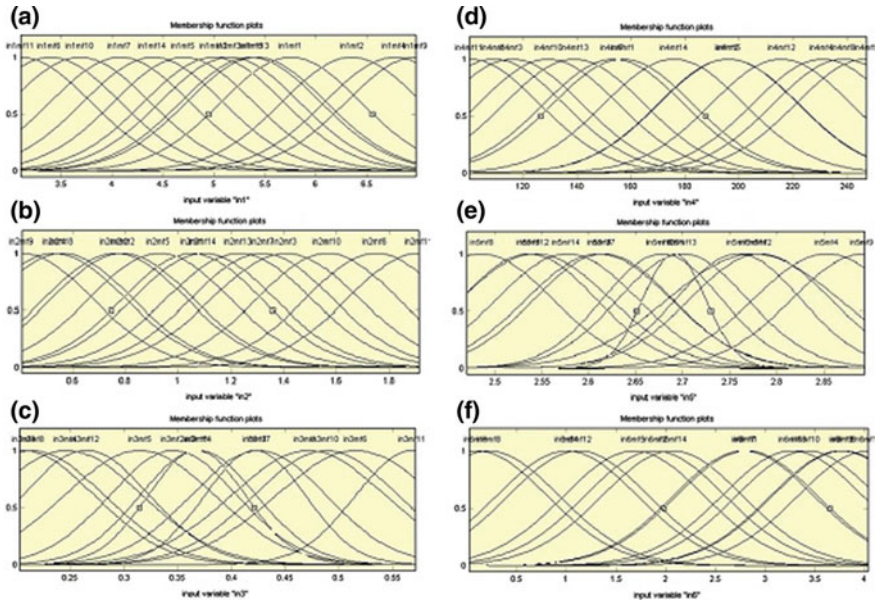


Fig. 6.28 **a** Membership function plot for input 1 (hardness). **b** Membership function plot for input 2 (porosity). **c** Membership function plot for input 3 (absorption). **d** Membership function plot for input 4 (compressive strength). **e** Membership function plot for input 5 (density). **f** Membership function plot for input 6 fracture roughness coefficient (Verma and Singh 2012)

Table 6.10 Specifications of the designed ANFIS (Verma and Singh 2012)

Number of inputs	6
Number membership functions for each input	14
Type of membership functions for each input	Gaussian
Type of membership functions for each output	Linear
Number of rules	14
Number of output membership function	14
Number of training epochs	30
Number of training datasets	26
Number of checking datasets	10
Error goals	0
Error achieved	0.1659

Table 6.11 MAPE and coefficient of correlation for the neuro-fuzzy model that Verma and Singh used (2012)

Case	MAPE (%)	Coefficient of correlation
Testing dataset	0.51	0.9995
Checking dataset	2.251	0.9253

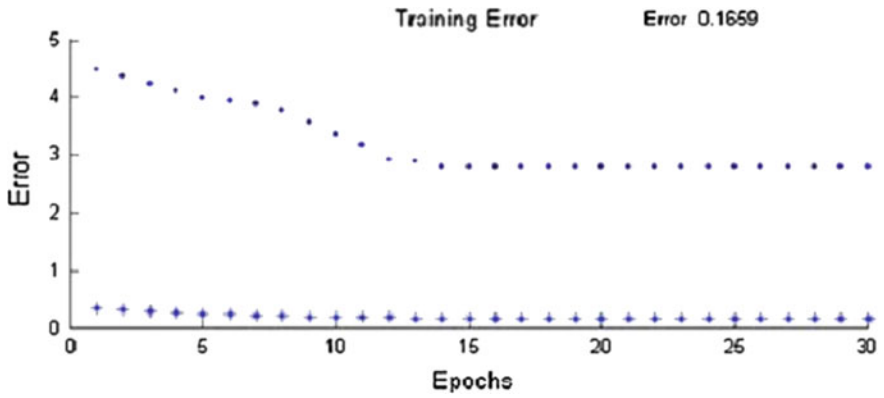


Fig. 6.29 Performance graph of the neuro-fuzzy model (Verma and Singh 2012)

Figure 6.29 shows that the checking error reduces continuously as the training of the model is progressing indicating that the model is not over-fitting the training dataset. ANFIS automatically sets the FIS parameters to correspond to the minimum checking error, (end of 30 epochs). Predicted and observed values of p-wave velocity for training and checking dataset, and their percentage errors are given in Tables 6.12 and 6.13.

Figure 6.30a, b show the correlations between observed and predicted values of p-wave velocity for the training and checking datasets from (Verma and Singh 2012).

The lower Mean Average Performance Error (MAPE) of 0.51% for the testing datasets and 2.251% for training datasets obtained by the ANFIS method seem acceptable (Table 6.11).

6.4.3 ANFIS Testing

Surface plots of the inputs and the P-wave velocity are shown in Fig. 6.31a–c. showing that the variation of predicted value (p-wave velocity) with the input parameters agrees well with the field results indicating the very good identification capability of the ANFIS algorithm and its robustness.

6.4.4 Conclusion

Verma and Singh (2012) conclude that ANFIS is an emerging computational tool successfully combining fuzzy logic and artificial neural network methods which can cope in a logical manner with subjectivity and uncertainty in the engineering process and allowing the use of vague and imprecise (fuzzy) information about the subsurface. It also allows the use of data for which the physical meaning is not immediately obvious and handles uncertainty, nonlinearity and expert knowledge in a much more satisfactory manner.

Table 6.12 Observed and predicted value of p-wave from the neuro-fuzzy model for testing datasets (Verma and Singh 2012)

Sl. No.	Hardness	Porosity (%)	Absorption (%)	Compressive strength (Mpa)	Density (g/cc)	Fracture roughness coefficient	Observed p-wave (cm/s)	Predicted p-wave (cm/s)	Error (%)
1	6.700	0.630	0.245	222.00	2.842	1.438	652.344	654.369	−0.310
2	6.680	0.640	0.255	219.00	2.835	1.507	650	651.942	−0.299
3	6.630	0.670	0.268	218.00	2.828	1.556	647.656	650.971	−0.512
4	6.580	0.680	0.279	211.00	2.821	1.585	644.531	644.660	−0.020
5	6.060	0.890	0.36	178.00	2.756	2.222	599.219	603.883	−0.778
6	6.040	0.910	0.367	177.00	2.751	2.3	595.313	601.456	−1.032
7	6.010	0.940	0.376	174.00	2.743	2.34	591.406	594.175	−0.468
8	5.320	1.250	0.444	135.00	2.655	3.455	467.188	471.845	−0.997
9	5.270	1.270	0.448	131.00	2.649	3.553	456.25	458.252	−0.439
10	5.230	1.350	0.458	125.00	2.641	3.651	442.969	443.204	−0.053
11	6.930	0.430	0.215	244.00	2.889	0.236	671.094	675.728	−0.691
12	6.880	0.460	0.219	240.00	2.882	0.452	668.75	671.845	−0.463
13	6.830	0.490	0.226	238.00	2.874	0.804	665.625	667.961	−0.351
14	6.810	0.520	0.229	236.00	2.862	0.911	663.281	665.534	−0.340
15	6.580	0.680	0.279	211.00	2.821	I.585	644.531	646.117	−0.246
16	6.540	0.710	0.289	208.00	2.814	1.605	642.188	643.204	−0.158
17	6.500	0.730	0.315	206.00	2.804	1.646	639.844	640.777	−0.146
18	6.120	0.870	0.354	185.00	2.766	2.144	607.813	608.738	−0.152
19	6.060	0.890	0.36	178.00	2.756	2.222	599.219	602.427	−0.535
20	6.040	0.910	0.367	177.00	2.751	2.3	595.313	600.485	−0.869
21	5.450	1.190	0.429	141.00	2.679	3.182	503.906	503.883	0.004
22	5.410	1.220	0.439	139.00	2.672	3.268	493.75	495.146	−0.283
23	5.380	1.240	0.441	138.00	2.669	3.357	430.469	481.553	−0.226

(continued)

Table 6.12 (continued)

Sl. No.	Hardness	Porosity (%)	Absorption (%)	Compressive strength (Mpa)	Density (g/cc)	Fracture roughness coefficient	Observed p-wave (cm/s)	Predicted p-wave (cm/s)	Error (%)
24	5.320	1.250	0.444	135.00	2.655	3.455	467.188	471.845	−0.997
25	5.143	0.683	0.256	225.86	2.51	0.775	453.299	459.223	−1.307
26	5.089	0.716	0.261	223.61	2.53	0.913	451.269	458.252	−1.548

Table 6.13 Observed and predicted value of p-wave from neuro-fuzzy model for checking datasets (Verma and Singh 2012)

Sl. No.	Hardness	Porosity (%)	Absorption (%)	Compressive strength (Mpa)	Density (g/cc)	Fracture roughness coefficient	Observed p-wave (cm/s)	Predicted p-wave (cm/s)	Error (%)
1	5.046	0.734	0.267	219.580	2.530	1.020	448.223	451.923	−0.825
2	4.368	1.129	0.374	171.590	2.580	2.139	401.523	401.923	−0.100
3	4.320	1.159	0.382	167.900	2.590	2.266	394.416	398.077	−0.928
4	4.261	1.187	0.390	164.650	2.600	2.324	387.817	336.538	13.222
5	4.223	1.228	0.398	163.150	2.610	2.462	382.741	378.846	1.018
6	3.876	1.430	0.449	141.560	2.660	3.130	320.305	325.962	−1.766
7	3.839	1.465	0.458	138.590	2.660	3.237	315.228	317.303	−0.660
S	3.781	1.497	0.464	136.580	2.670	3.286	303.629	306.250	0.771
9	3.734	1.541	0.472	132.300	2.680	3.364	300.000	303.346	−1.282
10	3.289	1.812	0.542	104.980	2.760	3.905	239.594	244.231	−1.935

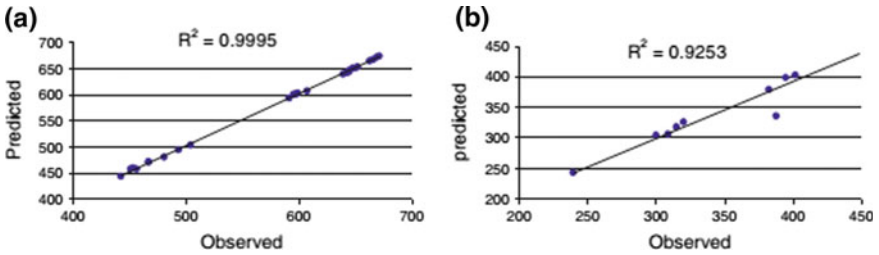


Fig. 6.30 **a** Correlation between predicted and observed values of p-wave velocity for testing datasets. **b** Correlation between predicted and observed values of p-wave velocity for checking dataset (Verma and Singh 2012)

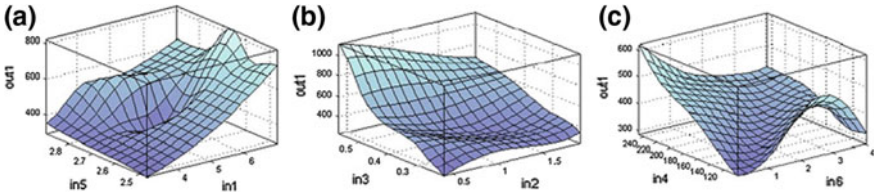


Fig. 6.31 **a** Surface plot of the relationship between hardness and density with P-wave velocity. **b** Surface plot of the relationship between porosity and absorption ratio with P-wave velocity. **c** Surface plot of the relationship between compressive strength and fracture roughness with P-wave velocity

6.5 Estimation of Electrical Earth Structure Using an Adaptive Neuro-Fuzzy Inference System (Anfis)

6.5.1 Introduction

Various methods (graphical, modeling, inversion) for interpretation of geophysical electrical resistivity sounding data for a three layered Earth have been used. Adaptive Neuro Fuzzy inference (ANFIS) can provide a novel way to do this based on a soft computing approach (Srinivas et al. 2012). They used Electrical resistivity data collected with the VES (Vertical Electrical Sounding) geometry as the training data set for ANFIS. The data collected from the field are known as the apparent resistivity but interpretation provides the resistivity and thickness of the individual layers. They used this trained data set to train ANFIS by optimizing the membership function parameters to provide the best fit to this data. In this section we explain the Srinivas et al. (2012) results of using ANFIS for three-layer interpretation of electrical resistivity.

6.5.2 Data Collection

Srinivas et al. (2012) used the Schlumberger electrode array with a maximum half-separation of the current electrodes (AB/2) of 100 m (Fig. 6.32) to study the electrical resistivity distribution of the subsurface (resistivity, thickness and depth) to determine the hydrogeological characteristics of the sub-surface.

The generic, characteristic sounding curves illustrated in Fig. 6.33 show the types of sounding curves, which might be obtained for various configurations of layer resistivity.

6.5.3 ANFIS Training

Srinivas et al. (2012) described the detailed MATLAB procedures for their work which is very useful for readers to follow step-by-step, and in this section we will outline this.

The following are the step-by-step procedures of the program and the MATLAB 2008b software methodology, which is used to simulate the program (which is also applicable in newer versions of MATLAB i.e. 2016b).

The command "**genfis**" (generate FIS) of the MATLAB software specifies the structure and initial parameters of the FIS (Fuzzy Inference System) and generates initial membership functions that are equally spaced covering the whole input space. The FIS architecture is constructed as given below:

==========Matlab codes==========

FISMAT = GENFIS1 (DATA, NUMMFs, MFTYPE)

==========

This code explicitly specifies:

– FISMAT: is the output FIS matrix for minimal training error which should be used for training routines in Sugeno-type fuzzy inference systems.

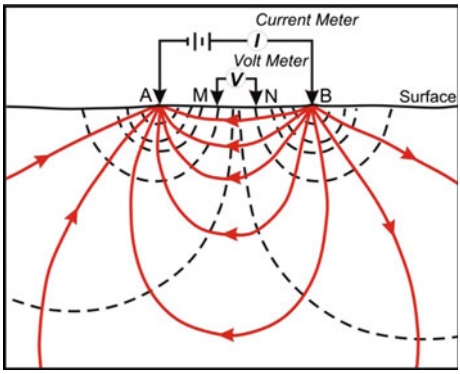


Fig. 6.32 The electrical resistivity sounding method. Solid red lines represent current flow. Dashed black lines are contours of electrical potential (voltage) (Clark and Page 2011)

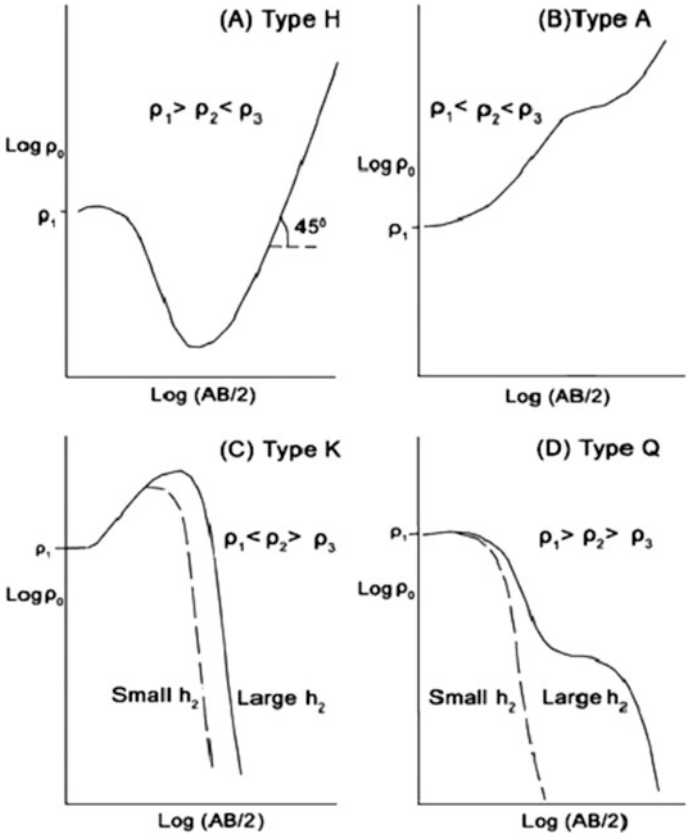


Fig. 6.33 Characteristic sounding curves for three layer VES data (after Telford et al. 1990)

- The syntax "DATA" represents the data to be trained in the present case it is electrical resistivity data.
- NUMMFS: represents the number of membership functions.
- MFTYPE: specifies the membership function type.

Srinivas (2012) used "gbellmf" command to produce a generalized bell-shaped built-in membership function. To identify the membership function parameters of single-output, Sugeno type fuzzy inference systems (FIS) the codes listed below were run in MATLAB:

```
==================================================
  [OUT−FIS, ERROR]= ANFIS (DATA, FISMAT, EPOCHS)
==================================================
```

- ERROR output is based on Root Mean Square Error (RMSE) occurs while the program running through the particular number of Epochs.
- IN−FIS: represents the FIS output.

– EPOCHS: represents the number of iterations used for training.
– The syntax:

```
==================================================
OUT = EVALFIS (FISMAT, OUT−FIS)
==================================================
```

Simulates the Fuzzy Inference System.

After framing the algorithm suitable for training, Synthetic data of Vertical Electrical Sounding data is used for training the network. Training data AB/2 with Apparent Resistivity (ρ_a) values has been trained using the algorithm.

Training error with respect to number of epochs is shown in Fig. 6.34. Training is checked with the number of synthetic data sets and one of them as example is shown in Fig. 6.35.

The following is the MATLAB sample source code used for training one set of synthetic data.

```
==============Matlab codes==================
trnData = [x y],numMFs = 5;mfType = gbellmf;
epoch−n = 20;
fismat = genfis1(trnData,numMFs,mfType);
out−fis = anfis(trnData,fismat,200);
plot(x,y,'*k',x,evalfis(x,out−fis),'-r');
==========================================
```

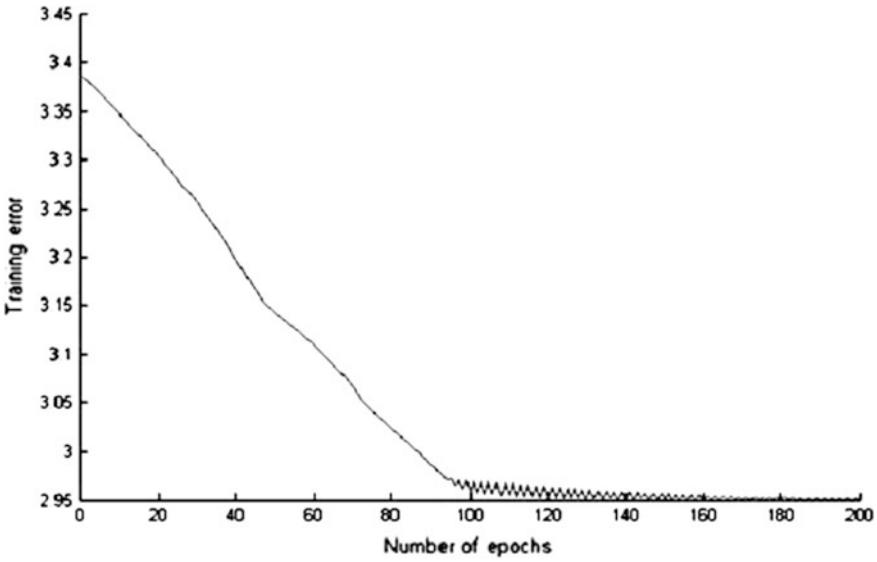


Fig. 6.34 ANFIS training error response between input and output (Srinivas et al. 2012)

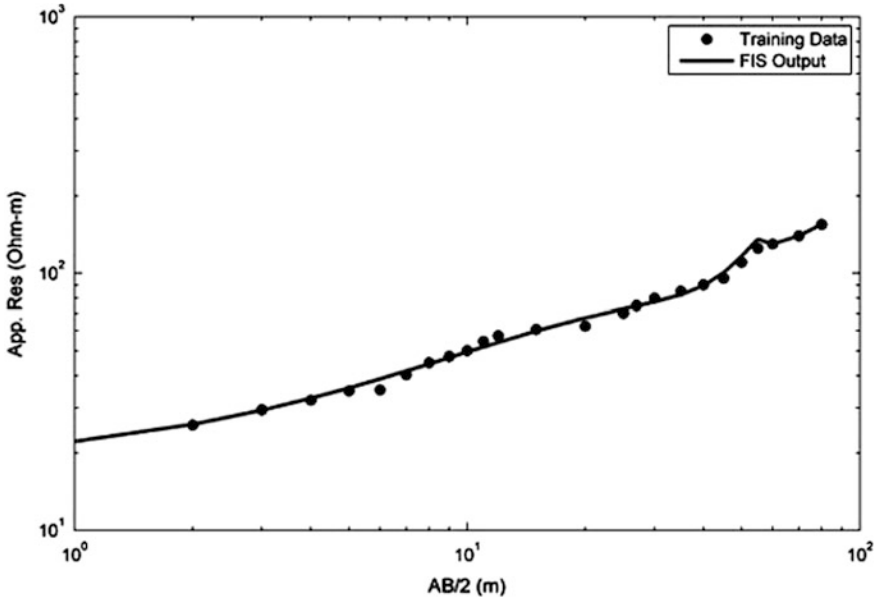


Fig. 6.35 The apparent resistivity synthetic data and its ANFIS output (Srinivas et al. 2012)

The sample code mentioned here is repeated for all the synthetic data sets for training. For all the data sets the training error has been minimized using the Root Mean Square Error (RMSE). The corresponding layer model for the trained synthetic dataset is stored in the memory of the network. After successful training of the synthetic data sets, the program is executed for testing the field data for validation (Srinivas et al. 2012).

6.5.4 ANFIS Performance Validation Using Real Data

After the training process was complete they tested the network with field data collected from the Abishekapatti study area located near latitude 8.44° and longitude 77.44°, Tirunelveli, India. The data tested; together with the FIS output pattern is shown in Fig. 6.36.

Here VES 13 has been used as the field data (as a sample) and the interpreted layer model in terms of resistivity and thickness is shown in Fig. 6.37 with a lower percentage error (0.0851) than the conventional method which had a percentage error of 2.63, (which itself is a high accuracy). As this is a case study and to ensure the ANFIS performance for various different three-layer cases, Srinivas et al. (2012) tested ANFIS for some other samples of three-layer resistivity interpretations and compared the ANFIS results to the conventional method based on curve fitting. The results showed that when ANFIS is trained well enough with a properly validated data set, its accuracy is much better than that of conventional methods (Table 6.14).

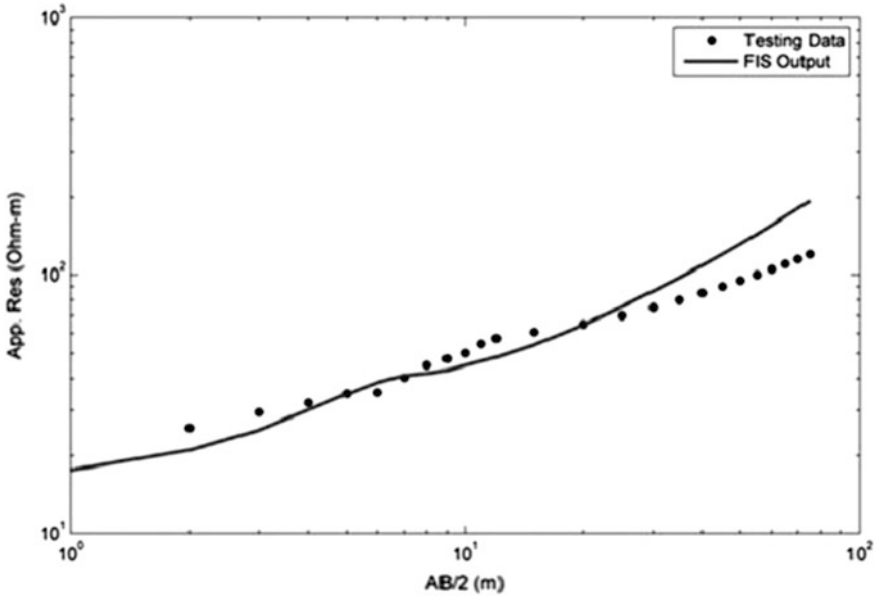


Fig. 6.36 The apparent resistivity field data and its ANFIS network response (Srinivas et al. 2012)

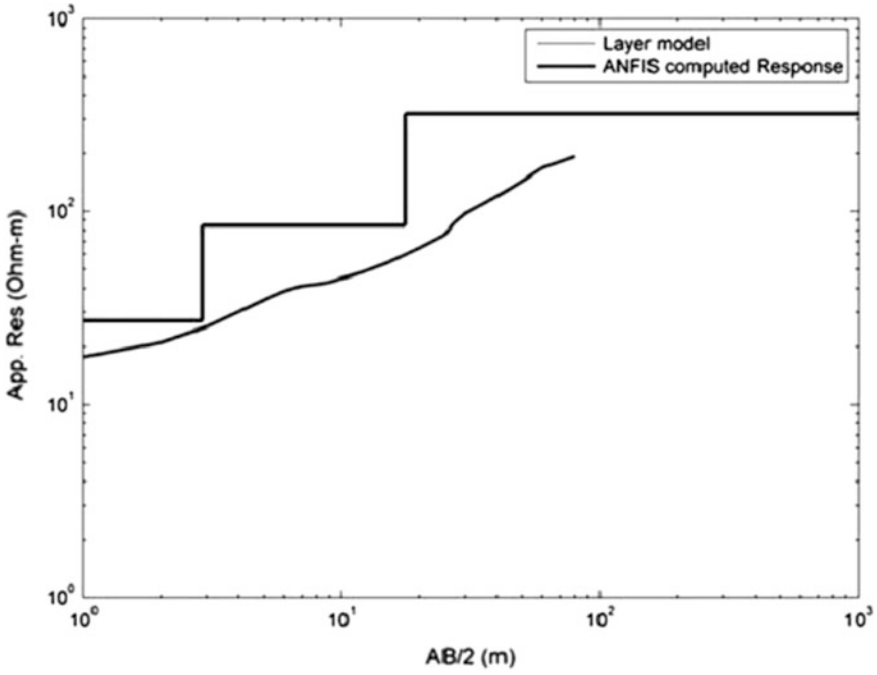


Fig. 6.37 The interpreted layer model and the ANFIS computed response curve of the field data (Srinivas et al. 2012)

Table 6.14 Comparison of ANFIS interpreted layer model and conventional method (Srinivas et al. 2012)

Sounding data	Curve types	Conventional method interpretation	Error percent	ANFIS Interpretation	Error percent
VES 1	H type	$\varrho 1 = 0.649$, $\varrho 2 = 0.383$, $\varrho 3 = 10.3$ $T1 = 4.04$, $T2 = 18.4$	19.3	$\varrho 1 = 0.5108$, $\varrho 2 = 0.4558$, $\varrho 3 = 4.6318$ $T1 = 3.5$, $T2 = 17.1$	0.0818
VES 2	H type	$\varrho 1 = 0.446$, $\varrho 2 = 0.0889$, $\varrho 3 = 8.71$ $T1 = 5.09$, $T2 = 5.13$	26.8	$\varrho 1 = 0.7416$, $\varrho 2 = 0.3606$, $\varrho 3 = 3.7626$ $T1 = 2.8$, $T2 = 17,9$	0.0926
VES 3	H type	$\varrho 1 = 0.533$, $\varrho 2 = 0.264$, $\varrho 3 = 93.5$ $T1 = 4.27$, $T2 = 14.2$	23.3	$\varrho 1 = 0.8043$, $\varrho 2 = 0.3773$, $\varrho 3 = 4.6383$ $T1 = 3.3$, $T2 = 14.7$	0.0883
VES 4	H type	$\varrho 1 = 0.473$, $\varrho 2 = 0.046$, $\varrho 3 = 7.74$ $T1 = 3.59$, $T2 = 4$	25.8	$\varrho 1 = 0.8084$, $\varrho 2 = 0.1953$, $\varrho 3 = 2.58$ $T1 = 1.4$, $T2 = 6.5$	0.1194
VES 5	H type	$\varrho 1 = 164$, $\varrho 2 = 29.5$, $\varrho 3 = 943$ $T1 = 1.7$, $T2 = 5.46$	6.08	$\varrho 1 = 131$, $\varrho 2 = 24$, $\varrho 3 = 417$ $T1 = 1.5$, $T2 = 7.3$	$4.2 \times 10e-7$
VES 6	H type	$\varrho 1 = 191$, $\varrho 2 = 32.6$, $\varrho 3 = 483$ $T1 = 1.22$, $T2 = 4.22$	8.54	$\varrho 1 = 189$, $\varrho 2 = 37$, $\varrho 3 = 417.7$ $T1 = 8$, $T2 = 1.03$	0.0356
VES 7	H type	$\varrho 1 = 215$, $\varrho 2 = 43.1$, $\varrho 3 = 476$ $T1 = 1.13$, $T2 = 4.99$	7.16	$\varrho 1 = 131$, $\varrho 2 = 48$, $\varrho 3 = 162$ $T1 = 1.6$, $T2 = 12.5$	0.0534
VES 8	A type	$\varrho 1 = 19.1$, $\varrho 2 = 5.12$, $\varrho 3 = 37.4$ $T1 = 2.07$, $T2 = 8.33$	7.65	$\varrho 1 = 34.70$, $\varrho 2 = 5.12$, $\varrho 3 = 32$ $T1 = 1.3$, $T2 = 8.0$	0.1094
VES 9	H type	$\varrho 1 = 59.1$, $\varrho 2 = 0.29$, $\varrho 3 = 981$ $T1 = 1.27$, $T2 = 1.73$	24.7	$\varrho 1 = 34.60$, $\varrho 2 = 0.17757$, $\varrho 3 = 492$ $T1 = 1.2$, $T2 = 1.5$	0.0057
VES 10	H type	$\varrho 1 = 177$, $\varrho 2 = 16.7$, $\varrho 3 = 389$ $T1 = 0.924$, $T2 = 4.73$	10.3	$\varrho 1 = 115.097$, $\varrho 2 = 12.897$, $\varrho 3 = 952$ $T1 = 1.9$, $T2 = 6.3$	0.0979
VES 11	A type	$\varrho 1 = 60.22$, $\varrho 2 = 105.7$, $\varrho 3 = 286.5$ $T1 = 2.297$, $T2 = 21.12$	1.72	$\varrho 1 = 54.48$, $\varrho 2 = 97.22$, $\varrho 3 = 271.926$ $T1 = 2.6$, $T2 = 17.8$	0.0261
VES 12	A type	$\varrho 1 = 31.2$, $\varrho 2 = 84.3$, $\varrho 3 = 313$ $T1 = 2.05$, $T2 = 21.7$	2.63	$\varrho 1 = 27.08$, $\varrho 2 = 84.68$, $\varrho 3 = 297.08$ $T1 = 3.0$, $T2 = 17.7$	0.0851
VES 13	A type	$\varrho 1 = 37.2$, $\varrho 2 = 87.9$, $\varrho 3 = 301$ $T1 = 2.13$, $T2 = 21.3$	2.37	$\varrho 1 = 32.76$, $\varrho 2 = 88.66$, $\varrho 3 = 297.06$ $T1 = 2.4$, $T2 = 17.8$	0.0690
VES 14	A type	$\varrho 1 = 43.1$, $\varrho 2 = 92$, $\varrho 3 = 293$ $T1 = 2.19$, $T2 = 21.2$	2.16	$\varrho 1 = 38.24$, $\varrho 2 = 88.64$, $\varrho 3 = 278$ $T1 = 2.5$, $T2 = 17.7$	0.0481
VES 15	A type	$\varrho 1 = 54.6$, $\varrho 2 = 101$, $\varrho 3 = 287$ $T1 = 2.267$, $T2 = 21.13$	1.84	$\varrho 1 = 49.11$, $\varrho 2 = 97.214$, $\varrho 3 = 271.91$ $T1 = 2.5$, $T2 = 17.8$	0.0141
VES 16	A type	$\varrho 1 = 48.91$, $\varrho 2 = 96.44$, $\varrho 3 = 290$ $T1 = 2.232$, $T2 = 21.16$	1.98	$\varrho 1 = 43.70$, $\varrho 2 = 92.8$, $\varrho 3 = 273$ $T1 = 2.5$, $T2 = 17.8$	$2.6755 \times 10e-7$
VES 17	Q type	$\varrho 1 = 57.6$, $\varrho 2 = 21.9$, $\varrho 3 = 2.36$ $T1 = 3.74$, $T2 = 7.22$	2.1	$\varrho 1 = 91.52$, $\varrho 2 = 33.12$, $\varrho 3 = 2.61$ $T1 = 4.46$, $T2 = 6.18$	0.0745
VES 19	Q type	$\varrho 1 = 96$, $\varrho 2 = 54.2$, $\varrho 3 = 23.2$ $T1 = 3.22$, $T2 = 4.24$	2.13	$\varrho 1 = 111.002$, $\varrho 2 = 42.002$, $\varrho 3 = 22.10$ $T1 = 4.7$, $T2 = 5.3$	0.0027
VES 20	Q type	$\varrho 1 = 105.9$, $\varrho 2 = 65.76$, $\varrho 3 = 31.51$ $T1 = 3.223$, $T2 = 4.237$	1.79	$\varrho 1 = 111.018$, $\varrho 2 = 49.01$, $\varrho 3 = 32.01$ $T1 = 4.7$, $T2 = 4.9$	0.0189
VES 21	Q type	$\varrho 1 = 116.5$, $\varrho 2 = 69.44$, $\varrho 3 = 42.37$ $T1 = 3.786$, $T2 = 3.105$	1.59	$\varrho 1 = 121.01$, $\varrho 2 = 56.01$, $\varrho 3 = 41.91$ $T1 = 4.9$, $T2 = 4.6$	0.0136

ϱ – true resistivity of subsurface layers in Ohm-meters, T – thickness of layers in meters

6.5.5 Conclusion

The results of Srinivas et al. (2012) for testing of ANFIS for both synthetic and real resistivity data showed the capability of this approach to estimate the subsurface strata with high accuracy.

6.6 Discrimination Between Quarry Blasts and Micro-earthquakes Using Adaptive Neuro-Fuzzy Inference Systems

6.6.1 Literature Review

The CTBT (Comprehensive Nuclear Test Ban Treaty) plays the role of monitoring for and discriminating nuclear weapons test from other seismic events and in order to do this, a global verification system including a network of 321 monitoring stations is distributed worldwide. One of their major problems lies in the identification of and discrimination of the widespread and regular low-yield mining events from possible nuclear detonations. Walter and Hartse considered that discrimination between small magnitude banned nuclear tests, background earthquakes and mining-induced seismic events is a challenging research problem.

Neural networks have been used successfully by various workers for determination of regional seismic events (Dysart and Pulli 1990; Dowla et al. 1990; Joswig 1995; Wang and Teng 1995; Musil and Plesinger 1996; Tiira 1996; Lee and Oh 1996; Gitterman et al. 1998; Tiira 1999; Ursino et al. 2001; Jenkins and Sereno 2001; Del Pezzo et al. 2003; Scarpetta et al. 2005; Yildirim and Horasan 2008).

Muller et al. (1999), Yildirimet al. (2011), Vasheghani-Farahani et al. (2012) have used fuzzy methods and Ait Laasri et al. (2015) recently used a fuzzy expert system for automatic seismic signal classification using the ANFIS method with feature selection to distinguish between quarry blasts and micro-earthquakes in the south and southeast of Tehran.

In this region a large number of quarry blasts “contaminate” the earthquake catalog (Fig. 6.38) and In order to identify the real seismicity (tectonic earthquakes) from induced events she used ANFIS as a classifier for identifying and categorizing these two kinds of seismic events.

6.6.2 Feature Selection

Feature selection is critical to minimize classification error, because inappropriate features make classification difficult and identification errors increase markedly. In the Tehran region, Vasheghani Farahani (2015) used forward feature selection for

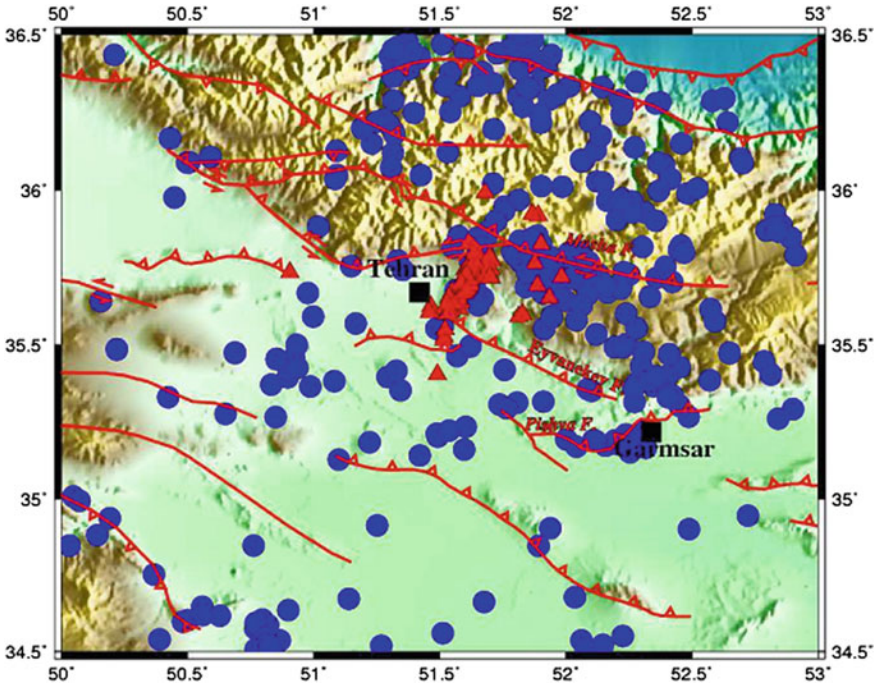


Fig. 6.38 Distribution of seismic events (2004–2010), for micro-earthquakes and for quarry blasts (Vasheghani Farahani 2015), with appreciation to IUGS for permission to use this Figure

categorization of seismic events. But initially there were no identified features, which were added one by one. In fact some features decreased correct classification rate (CCR) and were deleted from the input feature catalogue. At each step the feature was retrained, which helped minimize the classification error. Otherwise, it was redundant for classification purposes and was abandoned. The first feature chosen was time of occurrence and then latitude was added; at the next step magnitude was added. The CCR result for "time" was very high because quarry blasts frequently occur at specific times; noon for instance, during the day and this input feature was very useful in the classification process (Vasheghani-Farahani et al. 2012).

6.6.3 Spectral Characteristics

There are significant differences between earthquake and quarry blast spectra in the Tehran region. Vasheghani Farahani et al. (2015) evaluated spectral features based on the Corner frequency (fc is defined as the frequency where the high- and low-frequency spectral trends intersect) for the displacement of Pg waves (vertical component) for all signals in the regional network.

They used the mean fc for Pg waves and amplitude spectral velocity from the BIN network. Figures 6.39a and 6.40a show seismograms of all three components recorded at station DAMV (Damavand) for micro-earthquakes and quarry blasts, respectively, and Figs. 6.39b and 6.40b show the amplitude spectra of the vertical components. It is apparent that micro-earthquakes contain higher energy in the velocity spectra in the frequency range of 0.5–10.0 Hz velocity (Vasheghani Farahani and Zare 2014).

Figure 6.41a, b shows spectra from events recorded by the BIN network in the Tehran region and it is seen that the mean fc of Pg waves, (vertical component) for micro-earthquakes was 5.9 Hz and quarry blasts was 3.8 Hz; lower corner frequency for small explosions is related to higher attenuation in the shallow crust.

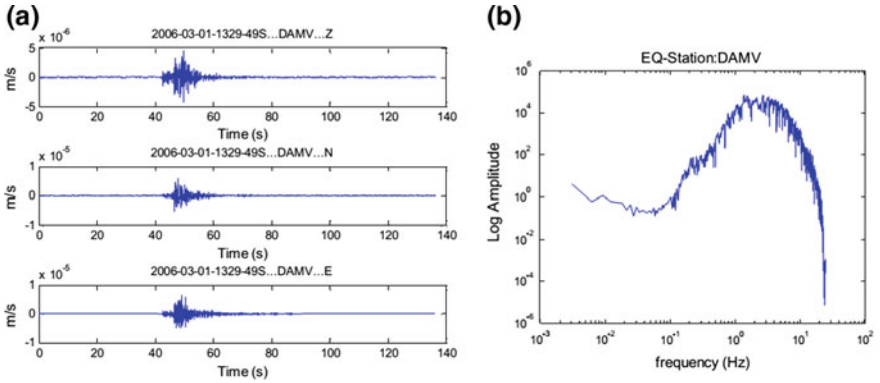


Fig. 6.39 **a** Typical example of a three component micro-earthquake at station DAM with ML = 2.1 at a distance of 26.2 km, **b** the spectrum amplitude for the vertical component of micro-earthquake (Vasheghani Farahani 2015, with thanks to IUGS for permission)

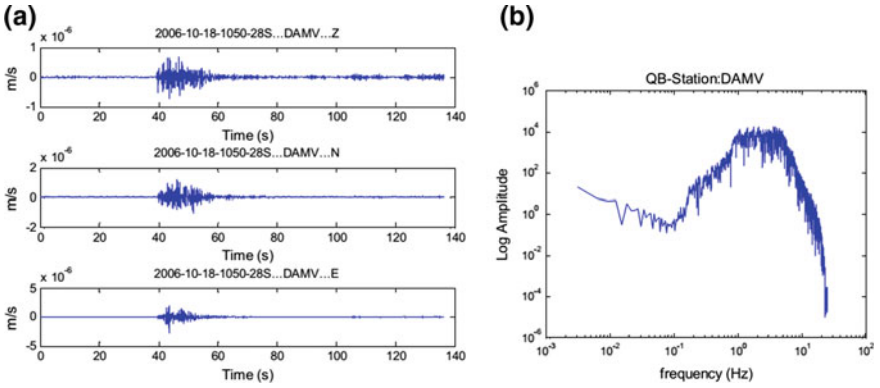


Fig. 6.40 **a** Typical example of three component quarry blast at station DAMV with ML = 1.3 at a distance of 17.1 km, **b** the spectrum amplitude for vertical component of quarry blast (Vasheghani Farahani 2015, with thanks to IUGS for permission)

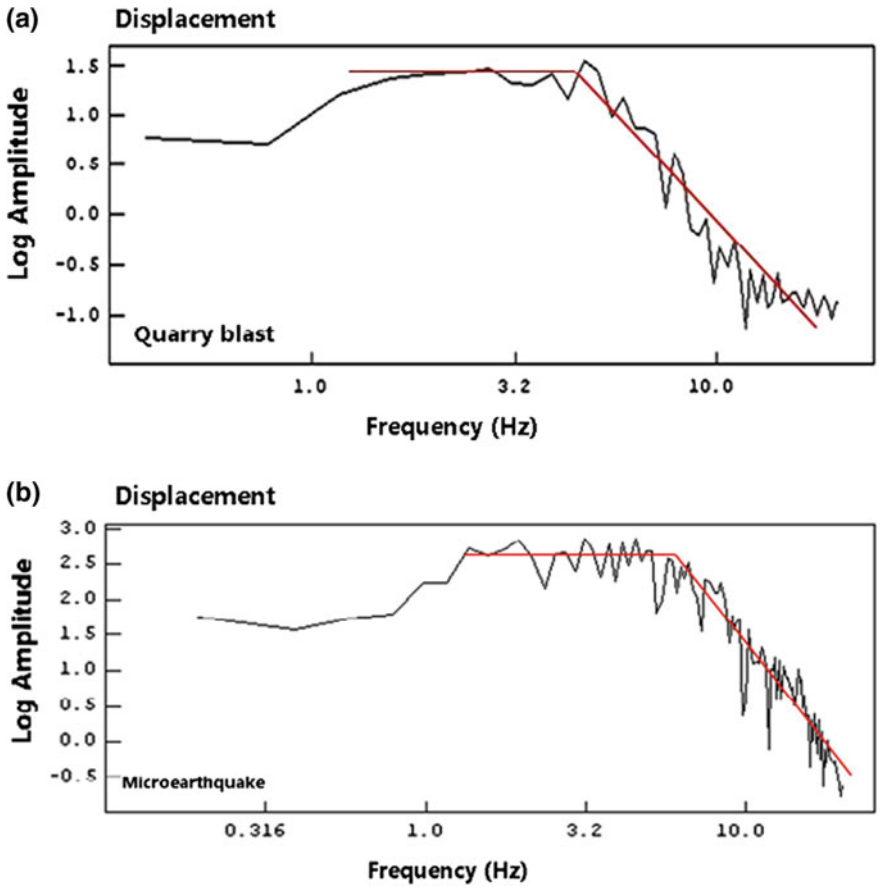


Fig. 6.41 Displacement spectral characteristics for **a** quarry blast and **b** micro-earthquake (Vasheghani Farahani 2015, with thanks to IUGS for permission)

Consequently, Vasheghani Farahani (2015) used displacement Pg-wave and velocity spectra as inputs and the output is "0" if an event is a micro-earthquake and "1" if it is a quarry blast.

6.6.4 Training and Test of ANFIS

320 out of 2530 signals (506 events) were selected for the training and testing database to secure the desired output. The ANFIS network was developed with 65% of the data for training (208 signals) and 35% of the remaining data (112 signals) were used for testing the model. The remaining 2210 data were then clustered based on 320 data clusters. This data which had unknown sources (whether they were

Table 6.15 Statistical parameter values for designed and trained ANFIS for 50 training periods

Method	Sensitivity (%)	Specificity (%)	Accuracy (%)
ANFIS	98.3	98.11	98.21

Table 6.16 Statistical parameter values for designed and trained ANFIS for whole database

Method	Sensitivity (%)	Specificity (%)	Accuracy (%)
ANFIS	99.3	98.95	99.09

natural earthquakes or blasts), was classified into several small categories as a validation using the same structure in the ANFIS determined from the training data (208) signals. Three membership functions were tested and the Gaussian membership functions selected with a maximum equal to one and a minimum of zero. The outputs of ANFIS were then examined to determine whether they were near the desired output (1 for earthquakes and 0 for quarry blasts). 50 training periods were selected, and the final test error value obtained. The performance of the classifiers was evaluated in terms of sensitivity, specificity and accuracy. The definition criteria were:

Specificity: Percentage of correctly classified quarry blast records
Sensitivity: Percentage of correctly classified micro-earthquake records
Accuracy: Percentage of correctly classified records out of the total number of records.

The experiment was repeated ten times, and then the average calculated and the statistical parameters of the classifier were calculated (Table 6.15).

Simulation results for the total number of events detected by the network are shown. There were 506 events in the seismic database and the 2210 signals (442 events) for which the target source type was not available were tested against the ANFIS model. The statistical parameter values for the database are shown in Table 6.16.

This appears to be a very effective method for the prediction of seismic event type and the spectral feature in the ANFIS achieved excellent recognition percentages and significantly reduced the errors in classifying the seismic events. It was concluded that spectral analysis (displacement Pg-wave and velocity spectrum for the complete seismogram signal) delivered high reliability for the discrimination of seismic events in the Tehran region.

6.7 Application of Neuro-Fuzzy Pattern Recognition Methods in Borehole Geophysics

6.7.1 Literature

Diverse techniques including regression methods (Hawkins et al. 1992), fuzzy recognition (Bezdek 1980; Bezdek and Pal 1992; Zadeh 1965, 1971) and neural

Networks (Van der Baan and Jutten 2000) have been used to evaluate hydrocarbon reservoir formations using well logging data. In addition, Baldwin et al. (1990) have applied neural network methods to well log data and tried to solve the problem of mineral identification; Nikravesh and Aminzadeh (2001) worked on mining and fusion of petroleum data with fuzzy logic and neural network agents.

There are many difficulties in the use of conventional evaluation methods. Fuzzy logic and neural network are widely used (Van der Baan and Jutten 2000) and Singh et al. (2010) used a novel approach known as Adaptive Neurofuzzy Inference System technique (ANFIS) to identify the stratigraphy of Prydz Bay basin, east Antarctica. They developed a 1-D geological model using datasets obtained from this area. The 1-model resulting from an ANFIS realization is able to make geological sense of even additional thin sand horizons sandwiched between clayey silt layers, which were unable to be resolved by other conventional methods. They showed that the analysis of ANFIS results can map horizons for hydrocarbon prospecting verifiable against known coring datasets and deduced that the results of AFNIS are encouraging and provide stable and consistent solutions. In this section we explain their work.

6.7.2 *Inputs-Output Structure of the Designed ANFIS*

Singh et al. (2010) provided an automatic strata interpreter for the study area. The gamma ray log, neutron porosity logs, density logs, sonic transit time, and separation between deep and shallow resistivity logs were used as input for the training of the ANFIS process (Fig. 6.42). An ANFIS method was constructed to identify the strata of the study area on the basis of the mean grain sizes of sedimentary rocks using a borehole dataset from PrydzBay, East Antarctica, which are prospective for hydrocarbon-bearing zones. The above geophysical logs were taken as input variables with layer formation as output for the ANFIS process. The core analysis of the study area principally identified the formations as diamictite, silt/clay and sand. In the ANFIS lithology system, lithology types must be converted to a crisp set in order for the mathematical estimation process to work. The output sets are assigned a code number from 1 to 3 signifying diamictite, silt/clay and sand for each lithology.

Fig. 6.42 Input/output of ANFIS lithology interpreter

6.7.3 Training of ANFIS

Initially the ANFIS based algorithm was trained on 80% of the total dataset and then tested on the remaining 20% of the dataset obtained from bore holes sites 1165C and 1166A of ODP leg 188 (Singh et al. 2010) in the Prydz Bay, East Antarctica (Stagg et al. 1985). The training and testing performance of ANFIS method they used had accuracies of 98.06 and 83.08%, respectively. The results obtained are more stable and are well correlated with available core samples. Singh et al. (2010) deduced that this method can easily define a permeable sand formation by distinguishing between silts and sands and by determining the mean grain size variation in these sands. In addition, this also resolved some thin strata of hydrocarbon bearing sedimentary rock (Singh et al. 2010).

6.7.4 Training of ANFIS Performance

The well log data from sites 1166A and 1165C (Leg 188) have been taken as the case study for this work. These log datasets are averaged for 5–6 values/meter in the depth range of 198.1204–983.8948 m (1165C) and 36.0264–358.962 m (1166A) and assigned to a specific depth. A total number of 5194 data of borehole 1165C and 2436 data of borehole 1166A were converted into 795 and 323 data samples and were divided into two parts: training and test datasets. The training datasets were used to construct the ANFIS lithology system by carefully adjusting the ANFIS input variables, by reducing the ANFIS lithology rules to construct a rule-based database. The test datasets were used to validate the system's prediction capability and based on the 80%/20% rule, the training and test datasets contained 735 and 60 samples respectively.

The ANFIS results were compared with the lithology of core analysis (the true lithology). The first output results are a division into as diamictite, silt, clay and sand while the second output identifies the major rock component in each layer—sand, silt or gravel. Figure 6.43 shows the network training error versus epoch

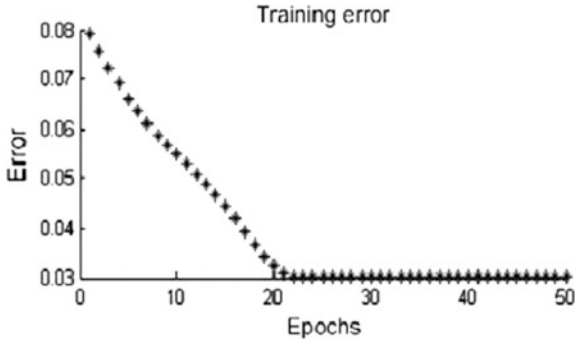


Fig. 6.43 The response of the errors through the ANFIS training process (Singh et al. 2010)

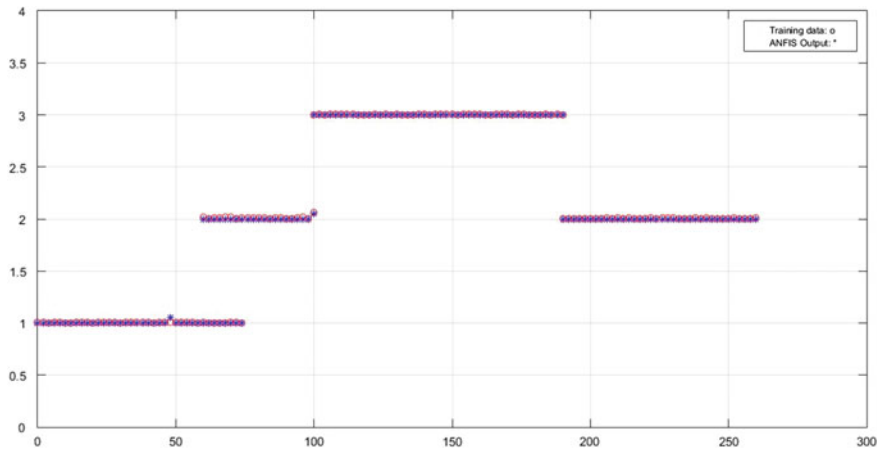


Fig. 6.44 The trained network shows the trained well logging data (blue star) by ANFIS matches with field data (red circle) with an accuracy of 98.03 (reproduced after Singh et al. 2010)

(iteration) progressing to completion at a global minimum and converging at 21 epochs (iteration) with an error 0.0305.

The output crisp values i.e. sand and gravel were then compared with the Moncrieff (1989) classification system of poorly-sorted sediments with a gravel component. The classification between the major rock formations and sand/silt/clay in various layers is done using MSCS (major rock formation size) which aids differentiation between silty clay and clayey silt as the first output gives does not distinguish silt and clay. After training the ANFIS lithology system its performance and prediction ability are validated. In Fig. 6.44 the red circles represent the true lithology, the star marks with blue color represent the ANFIS lithology, the vertical axis shows the output and the horizontal axis represents the different well-logging normalized data. 234 training data sets were identified correctly from the total 258 training data sets (Fig. 6.44) with a success rate of 98.08%.

6.7.5 Validation of ANFIS Performance

The validation (testing) of the network system was completed with 65 test datasets from borehole 1166A. 57 test datasets out of 65 datasets were predicted correctly (Fig. 6.45) with an 83.08% success rate. The error might be due to heterogeneous and/or anisotropic conditions at the wells.

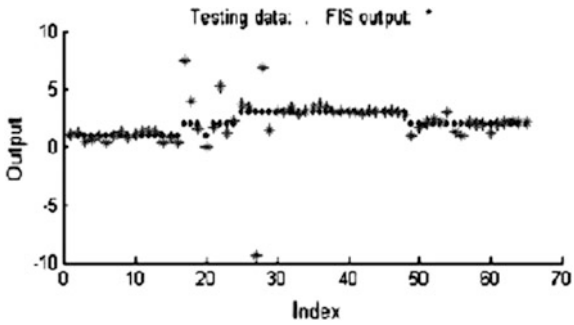


Fig. 6.45 The network shows the testing well logging data (.) by ANFIS matches with field data (*) with an accuracy of 83.083% (Singh et al. 2010)

6.7.6 Application of ANFIS Methods to Real Borehole Geophysics Data

In this example, the five parameters considered to be the actual field data were already known but the principle laid down is quite general in nature and need not be confined solely to a specific set of parameters.

This ANFIS method was constructed to identify the strata of the study area on the basis of the mean grain sizes of sedimentary rocks using the borehole dataset from PrydzBay of East Antarctica, where hydrocarbon bearing zones might be found. The above geophysical logs were taken as input variables and layer formation identification provided the outputs from the ANFIS process.

Figure 6.46 shows the Prydzbay main borehole log data selected for the ANFIS training process and its responses, (a) layer identification, (b) gamma-ray (API units), (c) neutron porosity (%), (d) density (g/cc), (e) sonic transit time (msec), (f) resistivity difference (ohm m) from depth 36.5 to 358.5 m, and (g) identified (predicted) layer boundary using the above log data. Figure 6.47 shows ANFIS testing and its responses, (a) gamma-ray (API units), (b) neutron porosity (%), (c) density (g/cc), (d) sonic transit time (msec), (e) resistivity difference (ohm m) from depth 36.5 to 358.5 m, and (f) identified layer boundary using above borehole log data by ANFIS method. The results obtained by ANFIS, which agree well with available coring data show that the model can be extended to any new environment and that this approach provides an efficient and robust method for identifying true lithology from well log data and that the ANFIS method is capable of generating a more accurate result within a fraction of the time needed for conventional analysis methods (Singh et al. 2010).

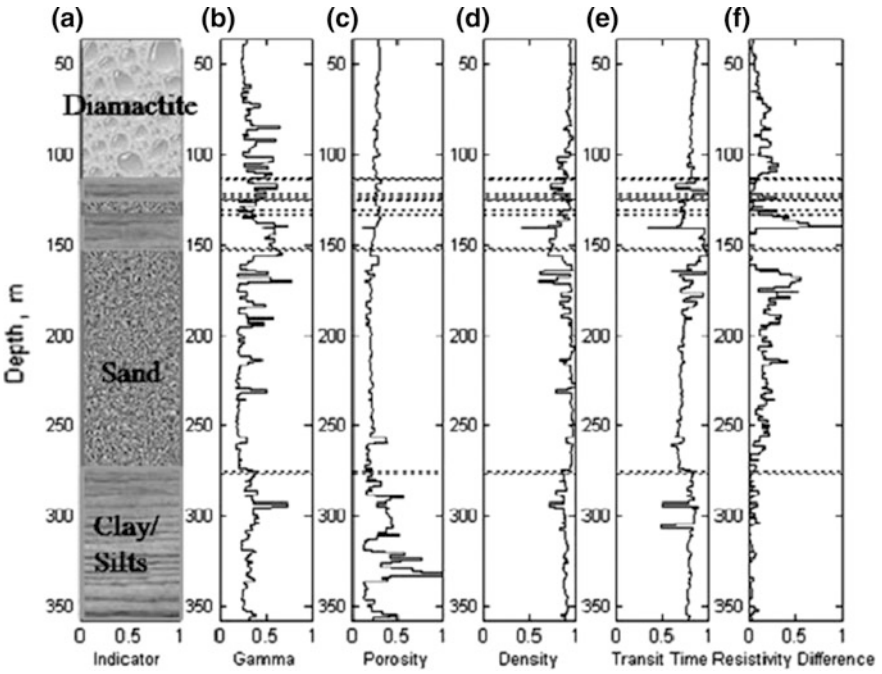


Fig. 6.46 Pryzbay main borehole log data taken for ANFIS training process and its response, **b** gamma-ray (API units), **c** neutron, porosity (msec), **f** resistivity difference (ohm m) from depth 36.5 to 358.5 m are normalized between [0 1], and **a** identified layer boundary using above borehole log data by ANFIS method (Singh et al. 2010)

6.8 A Fuzzy Interference System for the Prediction of Earth Rotation Parameters

6.8.1 Introduction

The prediction of ERP time series is commonly done using linked celestial and terrestrial reference frames methods such as GPS (global positioning system), very long baseline interferometry (VLBI), satellite laser ranging (SLR), amongst others, but these conventional methods cannot meet the requirements for real-time applications such as high-precision terrestrial navigation, for navigation of Earth satellites and interplanetary spacecraft, and for laser ranging to the Moon and artificial satellites (Schuh et al. 2002). These space-geodetic techniques require preprocessing in order to calculate ERP, and so it is not possible to have knowledge of ERP in real time For GPS, this processing takes 3 h, and for VLBI and SLR it takes a few days or more and so we must have the capability to predict the ERP over at least a few days (Akyilmaz and Kutterer 2004).

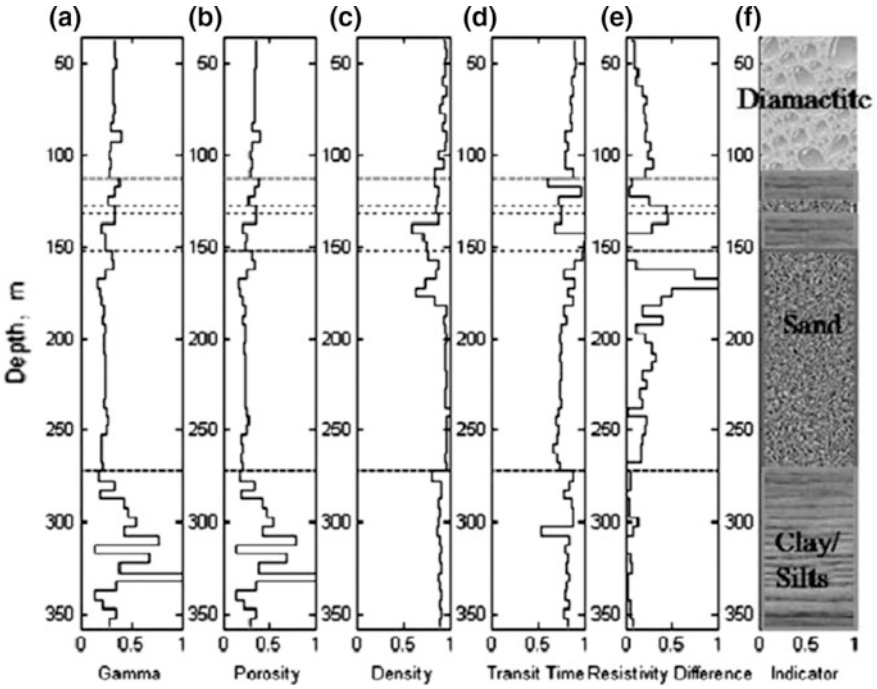


Fig. 6.47 Pryzbay main borehole log data taken for ANFIS testing and its response, **a** gamma-ray (API units), **b** neutron, porosity **c** density (g/cc), **d** sonic transit time (msec) **e** resistivity, **f** identified layer boundary using above borehole log data by the ANFIS method (Singh et al. 2010)

Prediction methods have been developed, e.g. by Zhu (1981, 1982), McCarthy and Luzum (1991), Freedman et al. (1994), Malkin and Skurikhina (1996), McCarthy (1996) which either estimate and extrapolate the parameters of harmonic into the future or use stochastical methods. e.g. Auto-regressive integrated moving average processes. Most use a combined model consisting of the deterministic part, which is either known or estimated by means of the least-squares (LS) method, and a predicted part, which can be stochastic or non-stochastic (Akyilmaz and Kutterer 2004).

An alternative method of real-time prediction is provided by soft computing methods, artificial neural networks and neuro-fuzzy systems. Ulrich (2000) and Schuh et al. (2002) used an a-priori deterministic model in conjunction with artificial neural networks (ANN) to predict short- and long-term ERP. Akyilmaz and Kutterer (2004) developed a new method to predict the earth rotation parameters (ERP) based on ANFIS. They predicted the Earth rotation parameters (ERP) (length-of-day and polar motion) up to 10 days ahead by means of ANFIS and then extended the prediction to 40 days further using the formerly predicted values as input data. The ERP C04 time series with daily values from the International Earth Rotation Service (IERS) served as the database. The well-known

effects such as the tides of the solid Earth and the oceans or atmospheric seasonal variations, were removed a priori from the C04 series. They used the residual series for training and validation of the network trying different network architectures compared their predictions with those of other methods. Not only is the ANFIS method they presented easier to implement than those conventional methods but they also showed that the Short-term ERP values predicted by ANFIS had errors which were lower than the alternative methods. In this section we explain further how Akyilmaz and Kutterer (2004) used ANFIS to predict ERP time series and show how the results compare to conventional methods.

6.8.2 *Prediction of Earth Rotation Parameters by ANFIS*

The output of each ANFIS model for the prediction of individual days in the future is in vector form. Almost every soft computing assessment requires the partitioning of the data into two parts, a training set and a validation set. The training set is used to optimize the model parameters whereas the validation set is used to confirm the parameters defined by the training set (Akyilmaz and Kutterer 2004).

To avoid the errors introduced by extrapolation a linear trend was removed from the polar motion series; a linear best-fitting function [model (t) = $a_0 + a_1.t$] was estimated by the LS method individually for the x and y components of polar motion from the time series between 1980 and 1998 and then extrapolated until 2001 and the trend then subtracted from the actual time series for the interval between 1980 and 2001. The residual data were used as training patterns. In the case of the length of day (LOD), tidal effects were first removed according to the IERS Conventions (McCarthy 1996) and a linear trend was calculated by means of the LS method and extrapolated until 2001 and the residual series used for training and validation of the network. ERPs before and after revisualization are given in Fig. 6.48. Note that reducing the linear trend function does change the amplitude; the change of amplitudes in the LOD comes from the reduction of tidal effects (Akyilmaz and Kutterer 2004).

6.8.3 *Patterns for Polar Motion Components x and y*

6.8.3.1 Patterns for Polar Motions

While composing the training patterns for ANFIS prediction of polar motion, a different method was used for predicting the first two days and for the latter days in the future. The input pattern for the prediction of the *first future day* is given by the values of the four previous days in the time series:

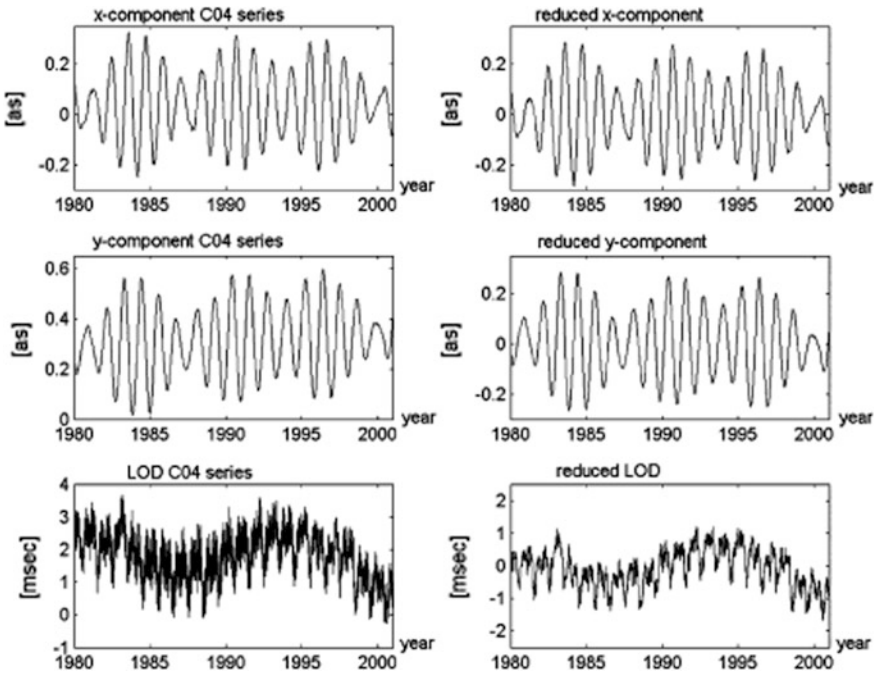


Fig. 6.48 ERP of the series C04 of the IERS before and after reduction. A linear trend was removed from the original polar motion and LOD data. For LOD the tidal influence was additionally reduced according to the IERS Conventions 1996 (Akyilmaz and Kutterer 2004)

$$\text{INPUT} : \{X(t-4),\ X(t-3),\ X(t-2),\ X(t-1)\} \rightarrow \text{OUTPUT} : \{X(t)\}$$

Similarly, the pattern for the prediction of *the second day* in the future is defined as:

$$\text{INPUT} : \{X(t-5),\ X(t-4),\ X(t-3),\ X(t-5)\} \rightarrow \text{OUTPUT} : \{X(t)\}$$

If k indicates the day in the future to be predicted, starting from 3, (i.e. k = 3, 4, 5…). Then the pattern becomes:

$$\text{INPUT} : \{X(t-8k),\ X(t-4k),\ X(t-2k),\ X(t-k)\} \rightarrow \text{OUTPUT} : \{X(t)\}$$

These patterns are then shifted across the whole time series of both reduced polar motion components (Akyilmaz and Kutterer 2004).

6.8.3.2 Patterns for LOD

The values of the reduced LOD time series for the last five days are the inputs and the day to be predicted is the output. Patterns for the prediction of the first 10 days

into the future are given below. These pattern matrices are then shifted along the whole time series of the reduced LOD, forming a multitude of pattern pairs (Akyilmaz and Kutterer 2004).

$$\begin{aligned}
&\{\mathrm{LOD}(t-5),\ \mathrm{LOD}(t-4),\ \mathrm{LOD}(t-3),\ \mathrm{LOD}(t-5),\ \mathrm{LOD}(t-1)\} \quad \rightarrow \\
&\{\mathrm{LOD}(t-6),\ \mathrm{LOD}(t-5),\ \mathrm{LOD}(t-4),\ \mathrm{LOD}(t-3),\ \mathrm{LOD}(t-2)\} \quad \rightarrow \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\
&\{\mathrm{LOD}(t-14),\ \mathrm{LOD}(t-13),\ \mathrm{LOD}(t-12),\ \mathrm{LOD}(t-11),\ \mathrm{LOD}(t-10)\} \quad \rightarrow
\end{aligned}$$

6.8.4 *Design of ANFIS Structure*

The specifications of both ANFIS that Akyilmaz and Kutterer (2004) used are listed in Table 6.17.

A typical training run is plotted in Fig. 6.49 showing that the error in the training set decreases with increasing number of training cycles. One interesting point is that, after reaching the global minimum in the training cycles, the RMS error no longer oscillates; it remains constant for further iterations. Experiences show that the global minima for validation sets is often very close to that for the training sets (Akyilmaz and Kutterer 2004).

Table 6.17 Description of specifications of ANFIS used for LOD and polar motion parameters prediction (Akyilmaz and Kutterer 2004)

Case	Description
Number of MFs for each of inputs	2
Type of MFs for inputs	Gaussian
Number of fuzzy if–then rules established for polar motion predictions	$2^4 = 16$
Number of fuzzy if–then rules established for LOD predictions	$2^5 = 32$
Learning algorithm	Hybrid (for both ANFIS)
Output function	Constant variable (singleton)
Number of iterations needed for convergence to the global minima	40 for each of two ANFIS
Calculation time	5 min

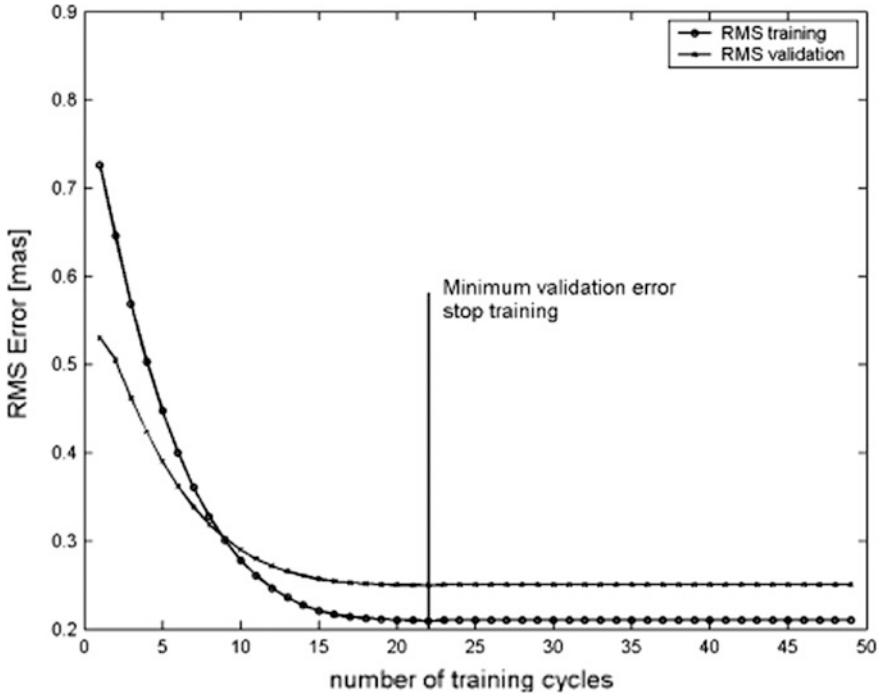


Fig. 6.49 RMS error characteristics during the training procedure. Note that the RMS errors for both the training and the validation sets are monotonously decreasing (Akyilmaz and Kutterer 2004)

6.8.5 Test of ANFIS for Real Data

Akyilmaz and Kutterer (2004) computed each FIS for prediction up to the 10th day and after the 10th day, they used predicted values as inputs to already existing models composed for the previous days' predictions to predict the values for days 11, 12, …, 40. The results of ANFIS were found to be equal to or even better than those from ANN (Table 6.18) (Schuh et al. (2002).

The RMS error measure for the ANFIS results as compared to other methods for the prediction of polar motion is shown in Fig. 6.50. Figure 6.51 shows that the proposed ANFIS provides predictions which match or exceed other methods until the twentieth day. After the twentieth day, this deteriorates but this is due to using predicted values which carry errors as inputs. Figure 6.51 shows the results of ANFIS predictions of LOD (top) for 10 days into the future (black) and the C04 series with the tidal effects and the linear trend function (gray) removed, and the corresponding prediction errors (bottom).

Comparing ANFIS-derived results with those from other methods clearly indicates that ANFISs can usefully predict ERPs, especially for the short-term range, up

Table 6.18 Comparison of ANFIS and ANN RMS prediction errors. In addition, the maximum absolute errors of ANFIS prediction are given (Akyilmaz and Kutterer 2004)

Polar motion			LOD			
Prediction day	ANFIS prediction RMS^{pm} (mas)	ANN prediction RMS^{pm} (mas)	ANFIS prediction RMS^{LOD} (ms)	ANN prediction RMS^{LOD} (ms)	ANF1S polar motion max absolute error (mas)	ANFIS LOD max absolute error (ms)
1	0.24	0.29	0.017	0.019	0.69	0.054
2	0.55	0.57	0.045	0.049	1.52	0.102
3	0.84	0.95	0.067	0.074	1.97	0.157
4	1.25	1.30	0.088	0.097	2.74	0.196
5	1.64	1.79	0.115	0.121	3.89	0.247
6	1.85	2.10	0.139	0.142	4.03	0.264
7	2.06	2.39	0.153	0.159	4.54	0.282
8	2.41	2.67	0.170	0.174	5.21	0.305
9	2.78	2.95	0.182	0.184	5.72	0.344
10	3.17	3.25	0.188	0.193	6.78	0.373
15	4.75	4.70	0.251	0.246	9.93	0.678
20	6.37	6.28	0.259	0.251	13.65	0.694
25	8.02	7.78	0.267	0.249	16.86	0.749
30	9.12	8.89	0.275	0.245	19.05	0.771
35	10.28	10.14	0.281	0.263	21.84	0.813
40	11.32	10.96	0.290	0.258	24.17	0.894

to 10 days, a very useful period for real-time operations (Akyilmaz and Kutterer 2004).

The following conclusions can be drawn:

- ANN produces good predictions but is difficult to handle and time consuming and requires an a priori model before training the network.
- Although there is also some prior process in ANFIS prediction, it is superior to ANN.
- Training of ANNs takes longer than that of ANFIS networks.
- Since the RMS error for the training and the validation sets oscillates significantly determining the global minimum is problematic.
- In the case of ANFIS training, the RMS error of both training and validation sets shows a continuously decreasing trend and the global minimum is guaranteed by the step-size factor.
- ANFIS requires a significantly fewer iterations.
- It is not necessary to modify or replace many parameters to obtain an optimal ANFIS prediction, whereas this is required when using ANNs.
- Smaller number of input variables is used in ANFIS prediction models than in ANN prediction models.

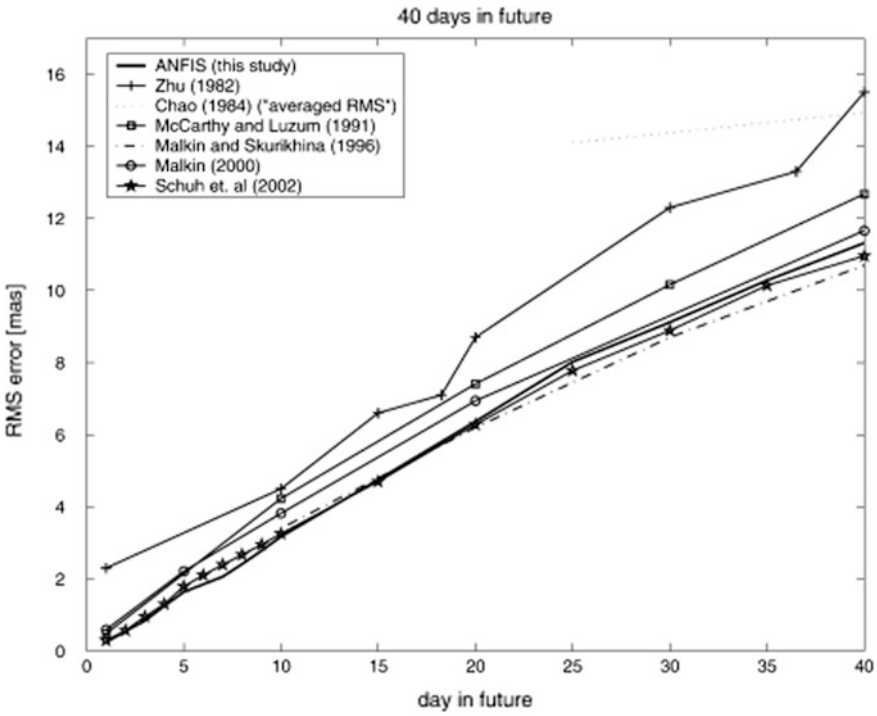


Fig. 6.50 RMS error of short-term prediction (up to 40 days) of polar motion; ANFIS results and other results according to Schuh et al.(2002) (Akyilmaz and Kutterer 2004)

- However, despite ANFIS being simpler than ANN modeling, it is still more complicated than other methods such as, for example, the one used in the IERS EOP service.

6.9 Coherent-Event-Preserving Random Noise Attenuation Using Wiener-Anfis Filtering in Seismic Data Processing

6.9.1 Literature Review

The resolving power of seismic reflection sections is inversely proportional to the amount of background random noise (Sheriff 1997) present in seismic data acquisition. By incorporating certain prior information based on the characteristics of the seismic events and the existing random noise, a supervising expert can attenuate random noise using conventional methods such as band pass filtering

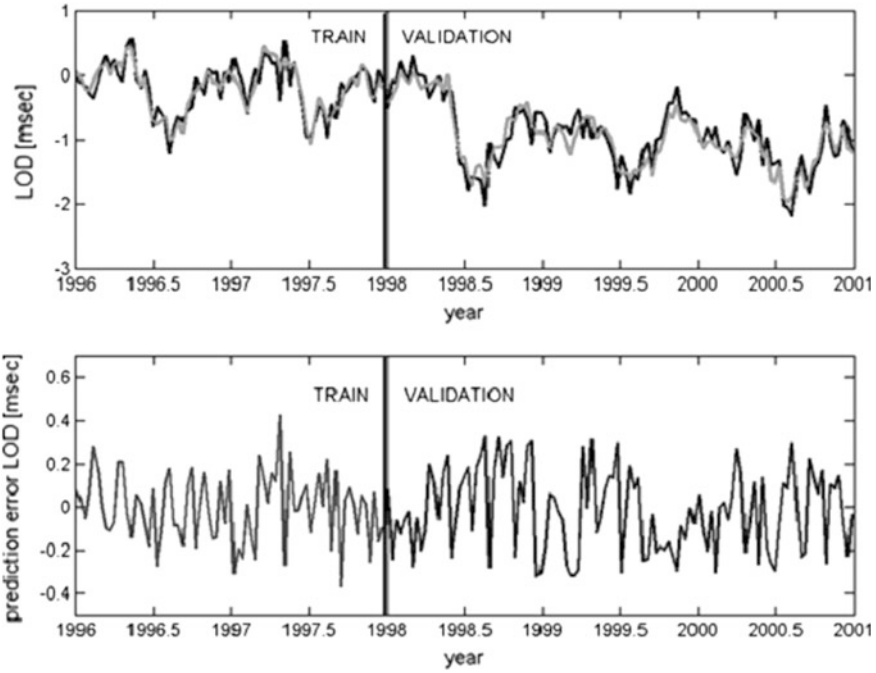


Fig. 6.51 LOD prediction (black) compared with the reduced C04 series (gray) and the corresponding prediction errors for the prediction of the tenth day in the future (Akyilmaz and Kutterer 2004)

(Stein and Bartley 1983), Karhunen-Loeve filtering (Al-Yahya 1991), f-x deconvolution (Bekara and Baan 2009), Fourier and Radon transforms, time-frequency analysis including multi resolution (Neelamani et al. 2008), median filtering (Liu 2013), and peak filtering methods (Lin et al. 2014).

Zhang et al. (2010) used a back propagation neural network architecture in which the error function had been tuned so as to attenuate the random noise in a shot record. Djarfour et al. (2008) also attenuated the noise in synthetic and real shot records using neural networks. Lin et al. (2014) applied fuzzy clustering based on time-frequency peak filtering, to attenuate random noise. Recently, Kimiaefar et al. (2015) proposing a semi-automatic algorithm and deployed an Artificial Neural network (ANN) with wavelet packet analysis for attenuating random noise in real stacked sections and common offset sorted gathers. Hajian et al. (2016) used an ANFIS system in order to enhance the S/N ratio of the CMP seismic reflection data by attenuating background random noise, through model discrimination and by utilizing local neighborhood information of each sample from adaptive Wiener analysis.

6.9.2 Wiener-ANFIS Filtering

In the method proposed by Hajian et al. (2016), the coordinates of some data sample points were first selected randomly in the section (for instance, coordinates of one percent of the dataset). At the next stage, the statistical features selected for the analysis of these points were calculated in a given neighborhood. Such statistical features may include mean, variance, median, extremum etc. The philosophy of this analysis assumes that the statistical behavior of noise is different from that of the coherent events (Fani and Hashemi 2011). For example, for each point of the random data, the local extremum of the amplitude in a window with fixed or variable length is calculated and the value is compared to the extremum of the amplitude of the adjacent points in a three by three neighborhood. Figure 6.52 illustrates this by zooming in on a part of a real seismic section. The comparison and separation of the extracted statistical features of input data is carried out through Fuzzy C-Mean Clustering method (FCM) whose output is the membership functions of a given point within all clusters which indicate how much the point belongs to a certain cluster. To determine the cluster corresponding to the coherent events, a modified version of Fani and Hashemi's (2011) method was applied. Two criteria decide whether or not a cluster is noise:

- *The first criterion is the sum of similarity values of one cluster with others in the pair wise mode. The cluster with lower summation value is considered to be noise.*
- *The second criterion is to count the number of acceptable (greater than a preselected threshold) similarity values of each cluster with the others. Any cluster that has low similarity with a bigger number of clusters is flagged as a possible noise cluster.*

After the determination of the cluster corresponding to the random events, the statistical features of each data sample and its neighborhoods are used as the input

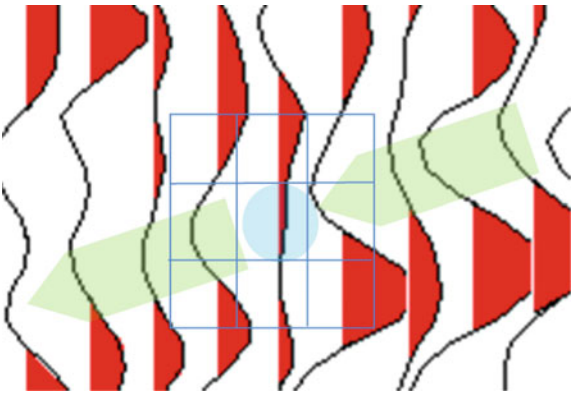


Fig. 6.52 Three by three neighborhood window of a seismic event (zoomed)

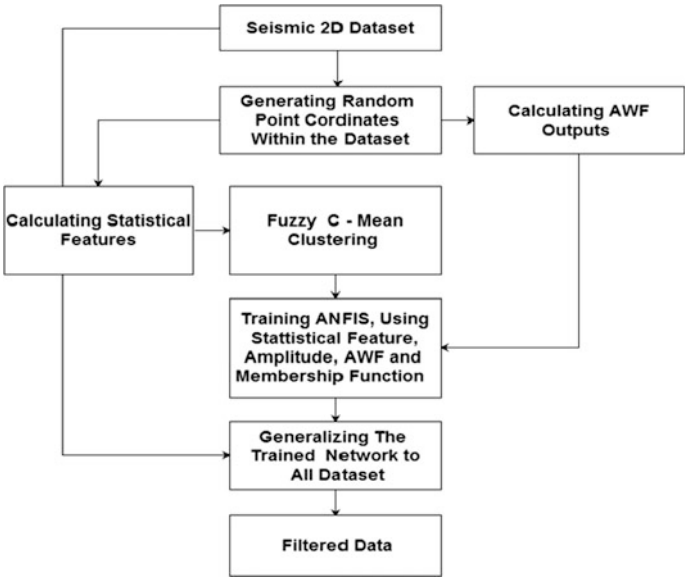


Fig. 6.53 General flowchart of the Wiener-Anfis Filtering (Hajian et al. 2016)

for ANFIS, and the value of the AWF and the amplitude of the trace for each data sample point, determine the output of the network in a weighted manner based on the membership function. The flowchart of the method is shown in Fig. 6.53.

6.9.3 Application to a Real Stacked Seismic Section

To examine the efficiency of the method, a synthetic CMP gather was used initially (Figs. 6.54a, b). This method was compared with four methods, Adaptive Median Filtering (AMF) used by Liu et al. (2009), Bayesian Estimation Filtering (BEF) (Candy 2009), Stationary Wavelet Transform Filtering (SWFT) used by Rawat and Dyal (2010), and AWF (Jeng et al. 2009) and the results are illustrated in Figs. 6.54c–g. As a validation test, the Structural Similarity Index Measure (SSIM) was used, and this was also applied by Lari and Gholami (2014) to determine the similarity of the noisy and filtered images to the original section (Fig. 6.55). Figure 6.55 shows the improved efficiency of the proposed method. The method also was applied to a real seismic stacked gather recorded in Alaska (Miller 2000) and the results shown in Fig. 6.56.

As seen in Fig. 6.56a many weak reflections in all parts of the seismic section are masked due to the presence of background random noise and this has made it difficult to trace the reflectors. The comparison of outputs resulted from implementing different filtering methods indicates the higher efficiency of the

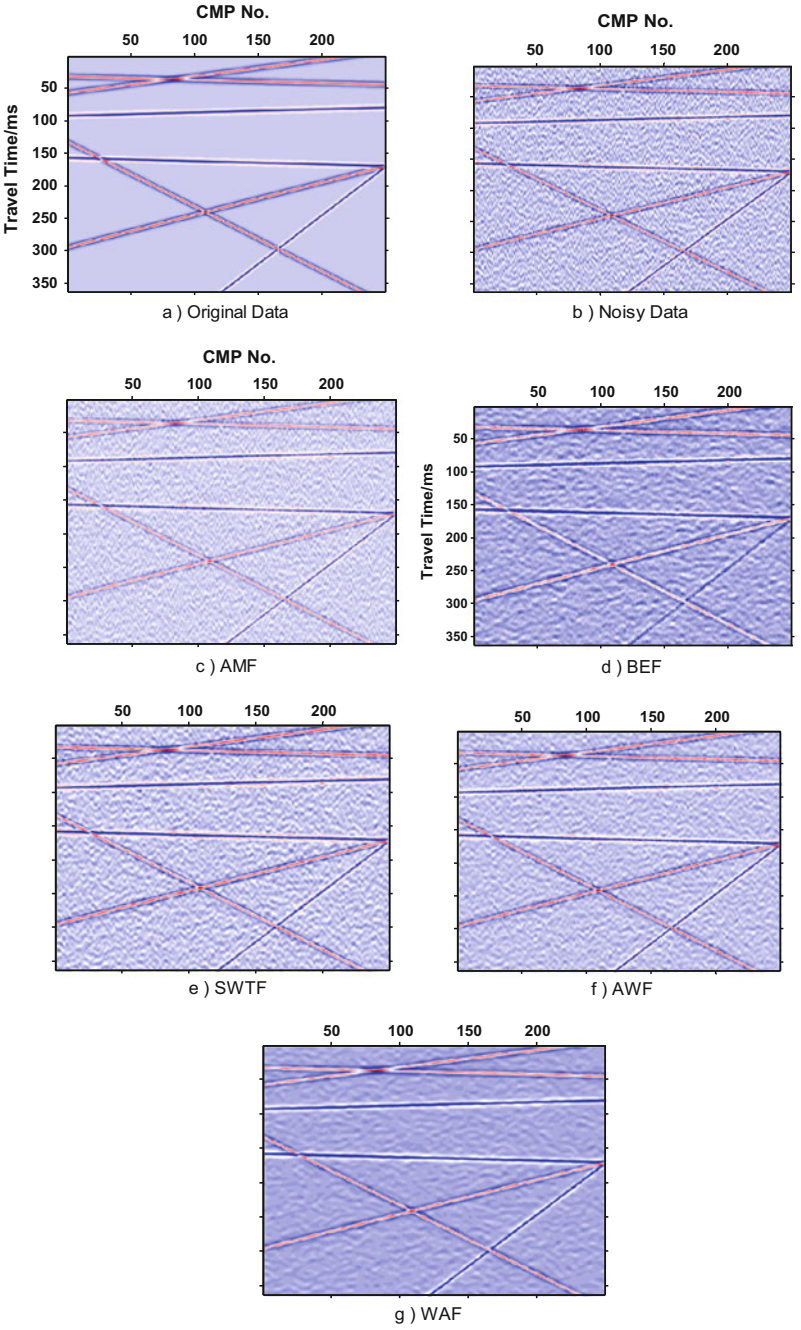


Fig. 6.54 **a** A synthetic CMP gather and **b** noisy gathers and filtered gathers: **c** AMF, **d** BEF, **e** SWTF, **f** AWF and **g** WAF (Wiener-ANFIS Filtering) (Hajian et al. 2016)

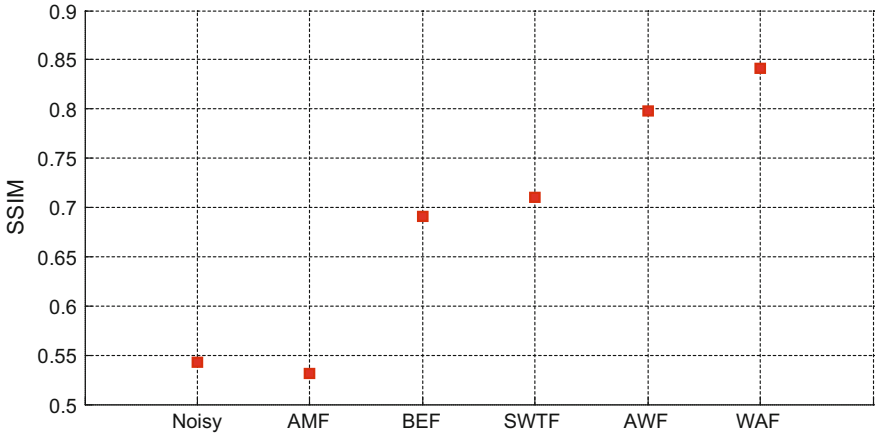


Fig. 6.55 SSIM calculated between original, noisy and filtered gathers in Fig. 6.54 (Hajian et al. 2016)

Wiener-ANFIS Filtering method in attenuating while preserving the amplitude of the coherent events. Figure 6.57 shows the amplitude spectra of the sections in Fig. 6.56 (Hajian et al. 2016).

Two indicators in normalized amplitude spectrums were sought:

- *Lower amplitudes at higher frequencies (based on the assumption of white Gaussian random noise).*
- *Higher amplitudes around the frequency band related to maximum amplitude. It can be seen that, the amplitude spectrum of the WAF performs better than other methods.*

6.9.4 Conclusions

A high performance automatic filter was designed based on the training of ANFIS with clustered data derived from the FCM clustering method. This has demonstrated the high capability of ANFIS in model discrimination and utilizing the local neighborhood information of each data sample from AWF yields satisfactory results in attenuating random noise and simultaneously preserving coherent events. The validation procedure was carried out through standard methods and the results were compared with most common conventional methods. The analysis of the outputs showed that the Wiener-ANFIS Filtering method has a considerably higher capability for realizing the research objectives (Hajian et al. 2016).

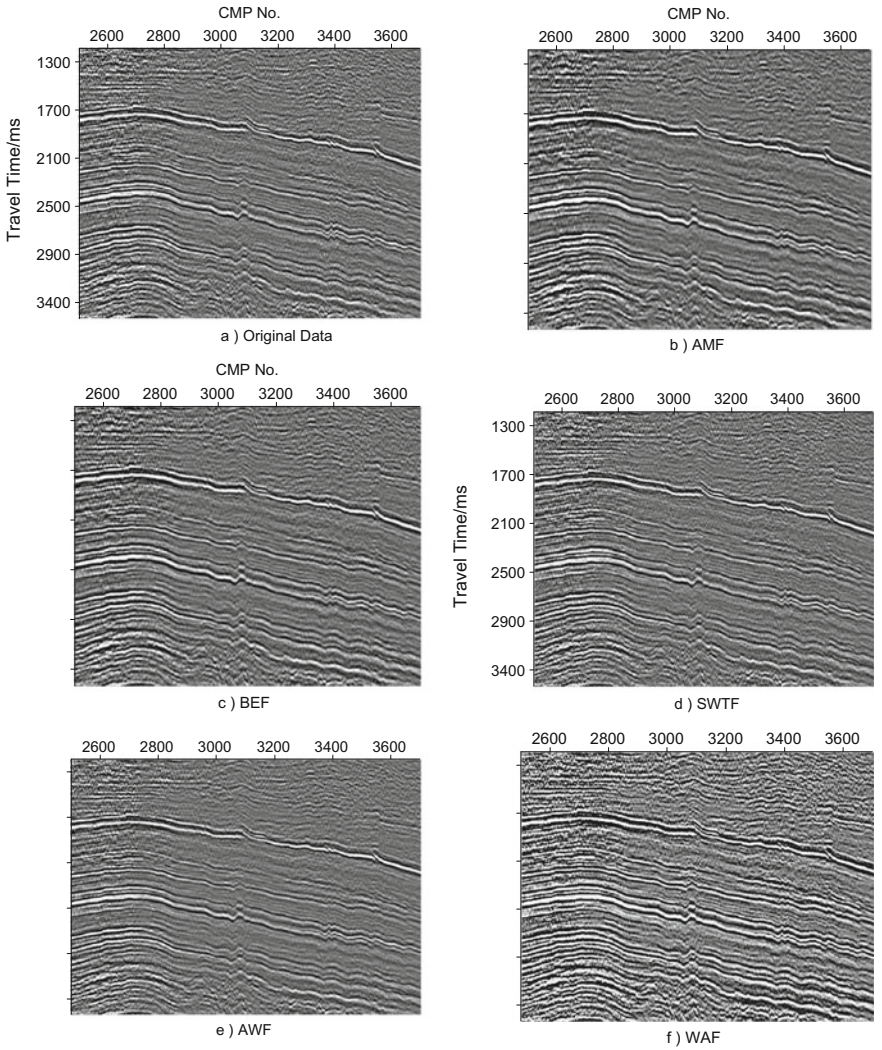


Fig. 6.56 Part of a real CMP section recorded in Alaska and filtered sections using the mentioned methods (Hajian et al. 2016)

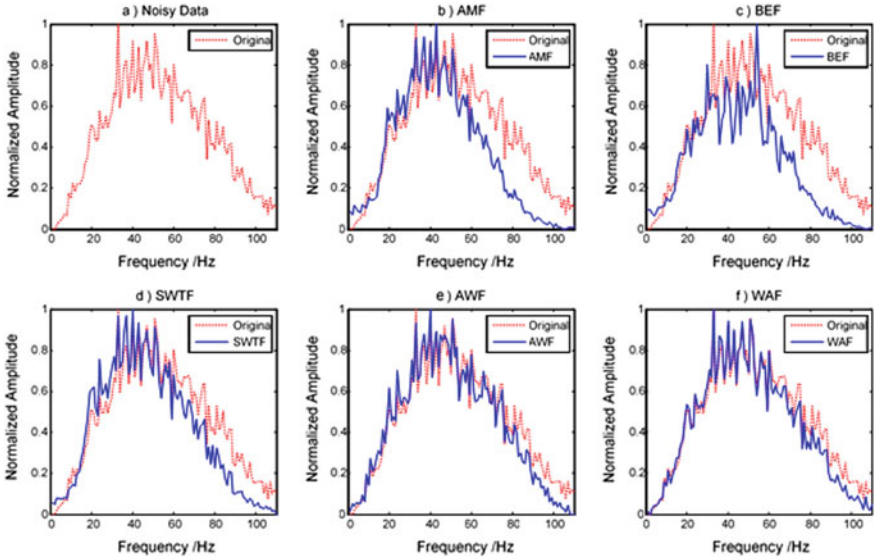


Fig. 6.57 Amplitude spectrum for the original and filtered sections in Fig. 6.56 (Hajian et al. 2016)

References

Abdelrahman E.M., El-Araby T.M., El-Araby H.M. and Abo-Ezz. E.R., 2001, A new method for shape and depth determinations from gravity data, Geophysics, 66: 1774–1780.

Adekanle A. and Enikanselu P. A., 2013, Porosity Prediction from Seismic Inversion Properties over 'XLD' Field, Niger Delta, American Journal of Scientific and Industrial Research, 4(1), 31–35.

Ait Laasri E.H., Akhouayri E.S., Agliza D., Zonta D. and Atmani A., 2015, A fuzzy expert system for automatic seismic signal classification, Expert Systems with Applications, 42(3), 1013-1027. DOI 10.1016/j.eswa.2014.08.023.

Akyilmaz O. and Kutterer H., 2004, Prediction of Earth rotation parameters by fuzzy inference systems, Journal of Geodesy, 78: 82–93.
DOI https://doi.org/10.1007/s00190-004-0374-5.

ALYahya K., 1991, Application of the partial Karhunen-Loeve transforms to suppress random noise in seismic sections. Geophysical Prospecting 39:77–93.

Aminzadeh F. and Groot P. D., 2004, Soft Computing for qualitative and quantitative seismic object and reservoir property prediction, Part 1. Neural Network Applications, 22:49–54.

Anderson, J.K., 1996, Limitations of seismic inversion for porosity and pore fluid: Lessons from chalk reservoir characterization exploration, Society of Exploration Geophysicists, SEG Annual Meeting, Denver, Colorado, 309–312.

Baldwin J.L., Bateman A. R. M. and Wheatley C.L., 1990, Application of neural networks to the problem of mineral identification from log wells, Log Analyst, 3, 279.

Banchs R. E., Michelena R. J., 2002, From 3D seismic attributes to pseudo-well-log volumes using neural networks: Practical considerations, The Leading Edge, 21(10), 996–1001.

Bekara M., Baan M. V. D., 2009, Random and coherent noise attenuation by empirical mode decomposition. Geophysics 74(5), 89–98.

Bezdek J. C., 1980, A convergence theorem for the fuzzy ISODATA clustering algorithms. IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume: PAMI-2, Issue: 1, Jan. 1980. https://doi.org/10.1109/tpami.1980.4766964.

Bezdek J. C., Pal S. K., 1992, Fuzzy Models for Pattern Recognition. Methods that Search for Patterns in Data. IEEE Press, New York.

Bhatt A. and Helle H.B., 2002, Committee neural networks for porosity and permeability prediction from well logs, Geophysical Prospecting. 50, 645–660.

Calderon J. E. and Castagna, J.,2007, Porosity and lithologicestimation using rock physics and multi-attribute transforms in Balcon Field, Colombia, The Leading Edge, 26(2), 142–150.

Candy J. V., 2009, Bayesian signal processing: Classical, modern and particle filtering methods. John Wiley & Sons.

Clark J.A. and Page R., 2011, Inexpensive Geophysical Instruments Supporting Groundwater Exploration in Developing Nations, Journal of Water Resource and Protection, 3(10), 768–780, https://doi.org/10.4236/jwarp.2011.310087.

Del Pezzo E., Esposito A., Giudicepietro F., Marinaro M., Martini M. and Scarpetta S., 2003, discrimination of earthquakes and underwater explosions using neural networks. Bulletin of the Seismological Society of America, 93, 215–223.

Djarfour N., Aifa T., Baddari k., Mihoubi, A. Ferahtia J., 2008, Application of feedback connection artificial neural network to seismic data filtering. C. R. Geoscience, 340, 335–344.

Dowla F.U., Taylor S.R. and Anderson R.W., 1990, Seismic discrimination with artificial neural networks: preliminary results with regional spectral data. Bulletin of the Seismological Society of America, 80, 1346–1373.

Dysart P.S. and Pulli J.J., 1990, Regional seismic event classification at the NORESS array: seismological measurement and the used of trained neural network. Bulletin of the Seismological Society of America, 80, 1910–1933.

Eftekharifar M. and Han D. H., 2011, 3D Petrophysical modeling using complex seismic attributes and limited well log data, Society of Exploration Geophysicists, SEG San Antonio 2011 Annual Meeting, 1887–1891.

Fani R. and Hashemi H., 2011, Random noise attenuation by application of GK clustering on relevant seismic attributes, 124th SEG Conference, Tokyo, Japan.

Freedman AP, Steppe JA, Dickey JO, Eubanks TM and Sung LY, 1994, The short-term prediction of universal time and length of day using atmospheric angular momentum. J Geophys Res 99 (B4), 6981–6996.

Gitterman Y., Pinsky V. and Shapira A., 1998, Spectral classification methods in monitoring small local events by the Israel seismic network. Journal of Seismology, 2, 237–256.

Grêt A.A., E.E.Klingelé, H.-G. Kahle, 2000, Application of artificial neural networks for gravity interpretation in two dimensions: a test study, Bollettino Digeofisicateorica Edapplicata, 41(1), 1–20.

Hajian A., Kimiaefar R., H.R. Siahkoohi, 2016, Random Noise Attenuation in Reflection Seismic Data Using Adaptive Neuro-fuzzy Interference System (ANFIS), 78th European Association of Geoscientists and Engineers, Vienna, Austria.

Hajian A., Rezazadeh Anbarani M., Mobarra Y., 2014, Comparison of neural networks, fuzzy - neural networks and finite element methods for the estimation of surface settlement due to tunneling in the Mashhad subway line 22nd International Congress of Recent Advances in Engineering, Islamic Azad University, Khomeinishahr Branch, ICRAE2014.

Hajian A., Styles P., Zomorrodian H., 2011, Depth Estimation of Cavities from Microgravity Data through Multi Adaptive Neuro Fuzzy Interference Systems, 17th European Meeting of Environmental and Engineering Geophysics, Leicester, UK.

Hajian A., 2012, Interpretation of microgravity anomalies using neural networks and fuzzy logic, Ph.D. Thesis, Science and Research Branch, Islamic Azad University (IAU), Tehran, Iran (in Persian, unpublished).

Hajian A. and Zomorrodian H., 2016, Estimation of depth and shape of subsurface cavities via Multi Adaptive NeuroFuzzy Interference System using Gravity data, Journal of the Earth and Space Physics, 42(3), 535-548.

Hawkins J.M., Schraufnagel R.A. and Olszeweski A. J., 1992, Estimating coal bed gas content and sorption sot herm using well log data. Paper SPE 24905 presented at the 1992. SPE Annual Technical Conference and Exhibition held in Washington.

Helle, H. B., Bhatt and A. and Ursin, B., 2001, Porosity and permeability prediction from wireline logs using artificial neural networks: a North Sea case study, Geophysical Prospecting, 49, 431–444.

Hosseini A., Ziaii M., Kamkar Rouhani A., Roshandel, A., Gholami R. and Hanachi J., 2011, Artificial Intelligence for prediction of porosity from Seismic Attributes: Case study in the Persian Gulf, Iranian Journal of Earth Sciences, 3, 168–174. IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-2, 1–8.

Jeng Y., Li Y. W., Chen C. S., Chien H. Y., 2009, Adaptive filtering of random noise in near-surface seismic and ground-penetrating radar data. Journal of Applied Geophysics 68, 36–46.

Jenkins R.D. and Sereno T. S., 2001, Calibration of regional S/P amplitude ratio discriminants. Pure and Applied Geophysics, 158, 1279–1300.

Joonaki E., Ghanaatian SH., Zargar GH., 2013, An Intelligence Approach for Porosity and Permeability Prediction of Oil Reservoirs using Seismic Data, International Journal of Computer Applications, 80(8), 19–26.

Joswig M., 1995, Automated classification of local earthquake data in the BUG small array. Geophysical Journal International, 120, 262–286.

Kimiaefar R., Siahkoohi H. R., Hajian A. and Kalhor A., 2015, Seismic Random Noise Attenuation Using Artificial Neural Network and Wavelet Packet Analysis. Arabian Journal of Geosciences.

Lari H. A. and Gholami A., 2014, TV regularized Bregman iteration for seismic random noise attenuation. Journal of Applied Geophysics, 109, 233–241.

Lee K. C. and Oh S. B., 1996, An intelligent approach to time series identification by a neural network-driven decision tree classifier, Decision Support Systems, 17, 183–197.

Lei L., Wei X., Shifan Z. and Zhonghong W., 2011, Reservoir property prediction using the dynamic radial basis function network, Society of Exploration Geophysicists, SEG San Antonio 2011 Annual Meeting, 1754–1758.

Leiphart D. J. and Hart B. S., 2001, Comparison of linear regression and a probabilistic neural network to predict porosity from 3-D seismic attributes in Lower Brushy Canyon channeled sandstones, southeast New Mexico, Geophysics, 66(5), 1349–1358.

Lin H., Li Y., Yang B, Ma H. and Zhang C., 2014, Seismic random noise elimination by adaptive time-frequency peak filtering. IEEE Geosciences and Remote Sensing Letters 11(1), 337–341.

Liu Y., 2013, Noise reduction by vector median filtering. Geophysics 78(3), 79–86.

Liu Y., Liu C. and Yang D., 2009, A 1D time-varying median filter for seismic random, spike-like noise elimination. Geophysics 74(1), 17–24.

Malkin Z, Skurikhina E,1996, On prediction of EOP. Commun IAA 93 (Technical Report).

McCarthy DD (ed), 1996, IERS technical note 21, IERS Conventions, 1996, Observatoire de Paris, Paris.

McCarthy DD, LuzumBJ, 1991, Prediction of Earth orientation, BullGe´od 65, 18–22.

Miller, J. J., 2000, Four regional seismic lines: National Petroleum Reserve- Alaska. USGS Geologic Division, Central Region Energy Team, USA.

Mojeddifar S., Kamali G., Ranjbar H. and SalehipourBavarsad B., 2014, A Comparative Study between a seudo-Forward Equation (PFE) and Intelligence Methods for the Characterization of the North Sea Reservoir, International Journal of Mining and Geo-Engineering. 48(2), 173–190.

Muller., S. Garda., P . Muller., J.D. and Cansi., Y., 1999, Seismic events discrimination by neuro-fuzzy merging of signal and catalogue features,Physics and Chemistry of the Earth, Part A: Solid Earth and Geodesy, 24A(3), 201-206.

Moncrieff A.C.M., 1989, Classification of poorly sorted sediments, Sedimentary Geology, 65, 191-194.

Musil M. and Plesinger A., 1996, Discrimination between local microearthquakes and quarry blast by multi-layer perceptrons and Kohonen maps. Bulletin of the Seismological Society of America, 80, 1077–1090.

Neelamani R., Baumstein A. I., Gillard D. G., Hadidi M. T. and Soroka W. I., 2008, Coherent and random noise attenuation using the curvelet transform. The Leading Edge 27, 240–248.

Nikravesh M. and Aminzadeh F., 2001, Mining and fusion of petroleum data with fuzzy logic and neural network agents, Journal of Petroleum Science and Engineering, 29, 221–238.

Rawat A. and Dyal S. S., 2010, Resolution enhancement of seismic data using stationary wavelet transform. 8th Biennial International Conference & Exposition on Petroleum Geophysics, Hyderabad.

Rezazadeh Anbarani M., 2014, Prediction of ground surface settlement due to tunneling using fuzzy neural networks (case study: Mashhad Subway), MS. Thesis, Najafabad Branch, Islamic Azad University, Isfahan, Iran (Unpublished).

Scarpetta S., Giudicepietro, F., Ezin, E. C., Petrosino, S., DelPezzo, E., Martini, M. and Marinaro, M., 2005, Automatic classification of seismic signals at Mt. Vesuvius Volcano, Italy, using neural networks. Bulletin of the Seismological Society of America, 95, 185–196. https://doi.org/10.1785/0120030075.

Schuh, H., Ulrich, M., Egger, D., Muller J and Schwegmann W., 2002, Prediction of Earth orientation parameters by artificial neural networks. J Geod 76, 247–258.

Schultz P. S., Ronen S., Hattori M. and Corbett C., 1994, Seismic-guided estimation of log properties (Part 1: A data-driven interpretation methodology). The Leading Edge, 13, 305–310. September 1999. Astron Soc Pac, San Francisco 208, 505–510.

Sheriff R. E., 1997, Seismic resolution a key element. AAPG Explorer 18(10): 44–51.

Singh U.K., Singh DK, Singh H., 2010, Application of Neurofuzzy pattern recognition method in borehole geophysics, Acta Geod. Geoph. Hung., 45(4), 417–425, https://doi.org/10.1556/ageod.45.2010.4.2.

Singh V., Painuly P. K., Srivastava A. K., Tiwary D. N., Chandra M., 2007, Neural networks and their applications in lithostratigraphic interpretation of seismic data for reservoir characterization, 19th World Petroleum Congress, Madrid, Spain, 1244–1260.

Srinivas Y., Stanley Raj A., Hudson Oliver D., Muthuraj D. and Chandrasekar N., 2012, Estimation of Subsurface Strata of earth using Adaptive-Neuro-Fuzzy Interference System (ANFIS), ActaGeod. Geoph. Hung., 7(1), 78–89, https://doi.org/10.1556/ageod.47.2012.1.7.

Stagg H.M.J., 1985, The structure and origin of Prydz Bay and Mac Robertson Shelf, East Antarctica, Tectonophysics, 114, 315–340.

Steeghs P., Overeem I., and Tigrek S., 2000, Seismic volume attribute analysis of the Cenozoic succession in the L08 block (Southern North Sea): Global and Planetary Change, 27, 245–262.

Stein R. and Bartley N., 1983, Continuously time-variable recursive digital band-pass filters for seismic signal processing: Geophysics, 48, 702–712.

Telford W.M., Geldart L.P. and Sheriff R.E., 1990, Applied Geophysics (second edition), Cambridge University Press, Cambridge.

Tetyukhina D., Luthi S. M., Van Vliet L. J., Wapenaar K., 2008, High-resolution reservoir characterization by 2-D model-driven seismic Bayesian inversion: an ex ample from a Tertiary deltaic clinoform system in the North Sea, Society of Exploration Geophysicists, SEG Las Vegas 2008 Annual Meeting, 1880–1884.

Tiira T., 1996, Discrimination of nuclear explosions and earthquakes from teleseismic distances with a local network of short period seismic stations using artificial neural networks. Physics of the Earth and Planetary Interiors, 97, 247–268.

Tiira T., 1999, Detecting teleseismic events using artificial neural networks, Computers and Geosciences, 25, 929–939.

Tonn, R., 2002, Neural network seismic reservoir characterization in a heavy oil reservoir, The Leading Edge, 21(3), 309–312.

Ulrich M., 2000, Vorhersage der Erdrotationsparameter mit Hilfe Neuronaler Netze. IAPG/FESG no. 9, Institut fu¨r Astronomische und Physikalische Geoda¨sie, Technische Universita¨t Mu¨nchen.

Ursino A., Langer H., Scarfì L., Di Grazia G and Gresta, S., 2001, Discrimination of quarry blasts from tectonic earthquakes in the Hyblean platform (southeastern Sicily). Annals of Geophysics, 44, 703–722.

Valenti J. C. A. F., 2009, Porosity prediction from seismic data using multiattribute transformations, N Sand, Auger field, Gulf of Mexico, M.Sc. Thesis, The Pennsylvania State University, The Graduate School.

Van der Baan M and Jutten C, 2000, Neural networks in geophysical applications Geophysics, 65, 1032–1047.

Vasheghani Farahani J., 2015, Discrimination of quarry blasts and micro-earthquakes using adaptive neuro-fuzzy inference systems in the Tehran region, International Journal f Episodes, IUGS.162–168.

Vasheghani-Farahani J. and Zaré M., 2014, Site characterizations for the Tehran network in Tehran Region Using Micro-Earthquake, Microtremor and Quarry Blast Data. Soil Dynamics and Earthquake Engineering Journal; 63, 235–247.

Vasheghani Farahani J., Zaré, M., and Lucas C., 2012, Adaptive neurofuzzy inference systems for semi-automatic discrimination between seismic events: a study in Tehran region. Journal of Seismology, 16(2), 291–303.

Verma A. K. and Singh T. N., 2012, A neuro-fuzzy approach for prediction of longitudinal wave velocity, Neural Comput&Applic, 22, 1685–1693, https://doi.org/10.1007/s00521-012-0817-5.

Wang J. and Teng T.L., 1995, Artificial neural network-based seismic detector, Bulletin of the Seismological Society of America, 85, 308–319.

Yildirim E. and Horasan G., 2008, Discrimination of earthquake and quarry blast data using artificial neural networks. In: Proceedings of the 18th International Geophysical Congress and Exhibition, Ankara, Turkey, 1–4.

Yildirim E., Gulbag A., Horasan G. and Dogan E., 2011, Discrimination of quarry blasts and earthquakes in the vicinity of Istanbul using soft computing techniques. Computers and Geosciences, 37, 1209-1217.

Zadeh L A, 1965, Fuzzy sets, Information and control, 8, 338–353.

Zadeh L A, 1971, Similarity relations and fuzzy orderings Information Science 3, 177–200.

Zhang Y., Tian X., Deng X. and Cao Y., 2010, Seismic denoising based on modified BP neural network. Sixth International Conference on 4: 1825–1829.

Zhu SY, 1981, Prediction of Earth rotation and polar motion. Rep320, Department of Geodetic Science and Surveying, The Ohio State University, Columbus.

Zhu SY, 1982, Prediction of polar motion. Bull Ge´od 56: 258–273.

Ziegler P. A., 1990, Geological atlas of western and central Europe. Amsterdam, Elsevierfor ShellInternationale Petroleum. Maatschappij, B.V.

Part IV
Genetic Algorithm

Chapter 7
Genetic Algorithm with Applications in Geophysics

Mrinal K. Sen and Subhashis Mallick

7.1 Introduction

Much of what we know about the Earth's subsurface has been derived from indirect measurements. Three-dimensional images of the Earth's interior have been derived from recordings of earthquakes caused by tectonic forces existing inside and on the surface of the earth. Variations in the gravity, magnetic, electric and electromagnetic fields recorded with advanced instruments placed on the surface, boreholes and airplanes, have also been used for this purpose. In seismic exploration, we make use of the principles of and lessons learnt from earthquakes to determine finer scale-subsurface features to identify zones that are favorable for the accumulation of hydrocarbons. Similarly monitoring of fluid movement during hydrocarbon production is also carried out by repeat seismic measurements (called time-lapse seismic experiments). The potential for application of geophysical data for estimating subsurface properties is simply stupendous. It keeps pace with the advancement in the computing and electronic technology, and we have witnessed a steady growth of large-scale applications of Geophysics.

The most essential element of geophysical analysis is estimation of subsurface rock properties (e.g., elastic constants, electrical conductivity, permeability, porosity etc.) from remotely sensed data—this is not a trivial task. Our approach to learning about the earth is no different from the basic premise to doing science. Prof. Richard Feynman, in 1964, eloquently elaborated the essence of the scientific method as follows:

M. K. Sen (✉)
Jackson School of Geosciences, The University of Texas, 2305, Speedway Stop, C1160, Austin, TX, USA

S. Mallick
Department of Geology and Geophysics, The University of Wyoming, P.O. Box 3006, Laramie 82071-3006, USA

A. Hajian and P. Styles, *Application of Soft Computing and Intelligent Methods in Geophysics*, Springer Geophysics,
https://doi.org/10.1007/978-3-319-66532-0_7

> In general, we look for a new law by the following process: First we guess it; then we compute the consequences of the guess to see what would be implied if this law that we guessed is right; then we compare the result of the computation to nature, with experiment or experience, compare it directly with observation, to see if it works. If it disagrees with experiment, it is wrong. In that simple statement is the key to science. It does not make any difference how beautiful your guess is, it does not make any difference how smart you are, who made the guess, or what his name is — if it disagrees with experiment, it is wrong.

Thus there are three essential elements to doing science: guess, compute, and compare. In essence, Feyman described an inverse problem (Berryman—unpublished lecture notes at Stanford University) http://sepwww.stanford.edu/sep/berryman/NOTES/chapter1.pdf). Parameter estimation from geophysical measurements follows these three steps only. In other words, we attempt to match our observations with theoretical computed data (this procedure is called *forward modeling*) until we have an acceptable fit. Note that here we introduce a feedback to update our guess. When we find an acceptable fit, we assume that we have discovered the underlying parameters (we will call these *model parameters*) that we are interested in. This procedure (Fig. 7.1) is referred to as *model based inversion* (e.g., Sen and Stoffa 2013).

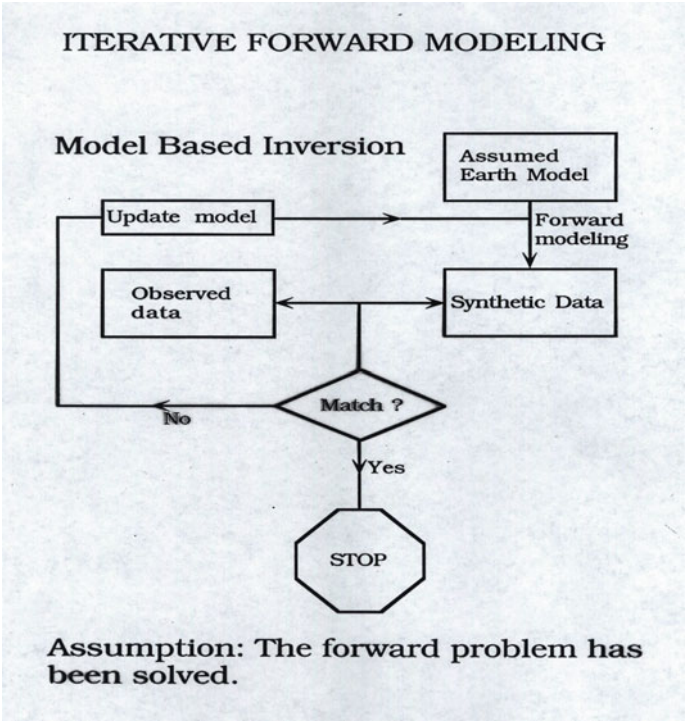


Fig. 7.1 Basic flowchart of 'model-based' inversion

Essential elements of an inverse problem include the following:

Data: this is represented as a column vector

$$\mathbf{d} = [d_1\, d_2\, d_3 \ldots d_N]^{\mathrm{T}}, \tag{7.1}$$

which comprises N discrete measurements, and T denotes matrix-transpose. We will use $\mathbf{d}_{obs}$ and $\mathbf{d}$ to represent observed and theoretical data respectively.

Model: this is represented as a column vector

$$\mathbf{m} = [m_1\, m_2\, m_3 \ldots m_M]^{T}, \tag{7.2}$$

which comprises M discrete parameters describing the surface model. Note that this parameter is often unknown and can also be considered a variable as used in a trans-dimensional inverse problem (e.g., Sen and Biswas 2017; Hong and Sen 2009). In general, $M \neq N$.

Forward modeling: This, in most applications, is given by a partial differential equation or some simpler form derived from it. Forward modeling relates the model parameters to the data. In other words, it computes the data for a given model, i.e.,

$$\mathbf{d} = f(\mathbf{m}), \tag{7.3}$$

where f is the nonlinear forward modeling operator. For most applications, such an explicit non-linear relationship does exist between data and model. When the relationship **is** linear, the above equation is given by

$$\mathbf{d} = \mathbf{G}\mathbf{m}, \tag{7.4}$$

where G is a matrix, often referred to as the data kernel (e.g., Menke 1984).

Objective function: Note that the model-based inversion approach involves measuring a misfit (or fitness) between the observed data and synthetic data, given by the following norm

$$F(\mathbf{m}) = \left[\sum_{i=1}^{N} |\mathbf{d} - \mathbf{d}_{obs}|^{p}\right]^{1/p}, \tag{7.5}$$

which represents a general L_p norm. The most commonly used norm is an L_2 norm (least squares measure) given by

$$F(\mathbf{m}) = \left[\sum_{i=1}^{N} |\mathbf{d} - \mathbf{d}_{obs}|^{2}\right]^{1/2}. \tag{7.6}$$

Or equivalently

$$F(\mathbf{m}) = (\mathbf{d}_{obs} - g(\mathbf{m}))^{\mathrm{T}}(\mathbf{d}_{obs} - g(\mathbf{m})). \tag{7.7}$$

Thus inversion involves searching for a model that minimizes the objective function—a procedure known as optimization. Note also that our definition of 'model-based inversion' as given in Fig. 7.1 is fairly simplistic in that our objective function can be minimized using multiple possible models. This complicates the search. One approach to addressing this is to impose constraints on data and model; this is typically done by modifying the objective function given above, to the following form:

$$F(\mathbf{m}) = (\mathbf{d}_{obs} - g(\mathbf{m}))^{\mathrm{T}}\mathbf{W}_d(\mathbf{d}_{obs} - g(\mathbf{m})) + \alpha^2\left(\mathbf{m} - \mathbf{m}_{pr}\right)^{\mathrm{T}}\mathbf{W}_m\left(\mathbf{m} - \mathbf{m}_{pr}\right), \tag{7.8}$$

where $\mathbf{W}_d$ and $\mathbf{W}_m$ are the data and model weighting matrices and $\mathbf{m}_{pr}$ is the a priori model vector. When described in a statistical framework (Tarantola 2000), data and model weighting matrices can be interpreted as data and model covariance matrices respectively.

7.2 Optimization

The primary task of geophysical inversion is to find an optimal set of model parameters by minimization of a suitably defined objective function—a task carried out by optimization. An additional and even more important task is to quantify uncertainty in the derived answer. We will not discuss the latter in detail here and focus on the optimization aspect in this chapter. Local optimization methods are generally most commonly used in geophysical inversion; these methods depend strongly on the choice of the starting and make use of local properties (gradient, Hessian etc.) to compute an update to the current solution. The objective function in a geophysical inverse problem is generally highly multi-modal containing multiple local and global minima. In Fig. 7.2, we display a plot of one such optimization problem.

Minimization or maximization of a multi-modal misfit or fitness function is generally carried out by meta-heuristics that belong to the category of global optimization. Several global optimization methods exist; among these the most popular methods include simulated annealing (SA), genetic algorithm (GA) and neighborhood algorithm (NA). Several geophysical applications of these algorithms are described in Sen and Stoffa (2013). In this chapter, we will only describe application of GA for a few geophysical inverse problems.

The Genetic algorithm (GA) was first applied to Geophysics by Stoffa and Sen (1991); this was followed closely by Sen and Stoffa (1992), Sambridge and Driekonigen (1992), and Scales et al. (1992). Since then there have been numerous applications of this approach to a plethora of geophysical problems. These include

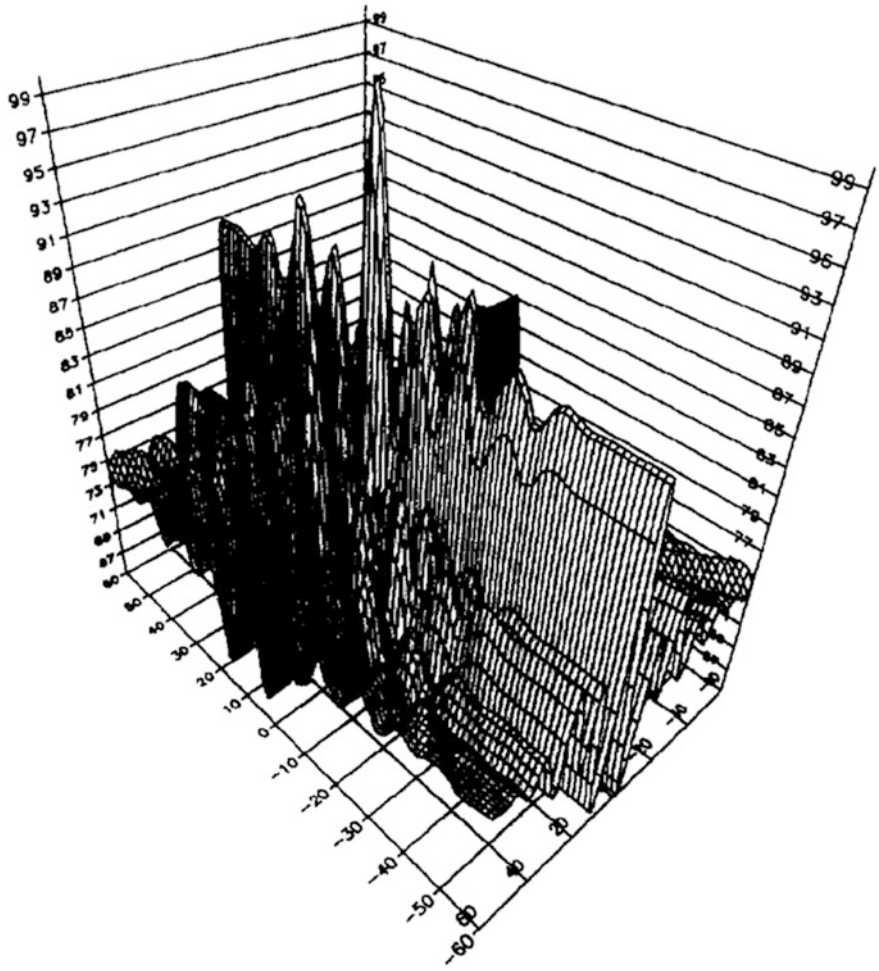


Fig. 7.2 Plot of a fitness function for a model surface-consistent statics problem. Reproduced from Smith et al. (1992)

seismic waveform inversion in 1D, earthquake location, tomography, inversion of gravity and magnetic data, well-placement problems etc. The applications continue to grow as computer power increases. Many of the applications are reported in Sen and Stoffa (2013). Here we will describe a few selected applications; it is not possible to report on all applications. The authors express their apologies for not being able to include many examples primarily because of restriction on the length of this chapter.

In the following, we provide a brief overview of genetic algorithms followed by examples of its application to two important geophysical inverse problems, namely,

a trans-dimensional approach to seismic waveform inversion, and joint inversion of disparate geophysical datasets.

7.3 Genetic Algorithm

Since the work of Kirkpatrick et al. (1983), Simulated Annealing (SA) has been applied to a wide variety of problems in geophysics. Most SA algorithms are statistically guaranteed to attain equilibrium distribution and possibly to reach the global optimum, and therefore they are suitable for best-fitting model estimation. In practice, however, these algorithms are sometimes problematic due to inappropriate parameter selection. For example, it is important to choose the starting temperature and cooling schedule properly.

Unlike SA, which is based on an analogy with a physical annealing process, the Genetic Algorithm (GA) (Holland 1975; Goldberg 1989) is based on an analogy with the process of natural biological evolution. The basic GA is quite robust and not particularly sensitive to the randomly selected starting models if enough models are employed. The primary advantage of a basic GA is that it always converges toward models with higher fitness values. The convergence of a GA can be premature if a small number of models are employed. Rapid convergence to minimize computational cost may be undesirable since the model space will not be sufficiently explored and the population may become homogeneous around a local fitness maximum that is not near the global maximum. In contrast, convergence toward the global maximum may be slow because some model parameters have only minor impact on the fitness, so that extensive sampling of model space often results in minor *improvements* at significant computational cost.

Genetic Algorithm (GA) is an intelligent maximization technique for functions defined on high-dimensional spaces, which simulates the biological evolutionary processes of *selection*, *crossover* and *mutation* to increase the fitness toward better solutions. The unique feature of a GA is that it works with a population of models. A typical GA involves the following steps:

- model representation,
- evaluation of fitness function for a population of models,
- probabilistic (biased) selection of models,
- mixing of models using processes of *crossover* and *mutation*.

7.3.1 Model Representation

GA starts with a population of randomly chosen models (called *chromosomes*) from the constrained model search space. Unlike other optimization methods, an essential element of GA is to represent a model **m** in some coded form. Several different

model-coding schemes have been proposed and implemented in GA applications. They are described below, for completeness.

Bit Coding: The minimum and maximum limits and the resolution of a model parameter are used to determine the total number of bits required to represent a model parameter. For example, let us assume that the compressional wave velocity (a model parameter in seismic inversion problems) is desired to be within the v_{min} = 1500 m/s and v_{max} = 1810 m/s with a resolution v = 10 m/s. Note that the entire range of possible velocities can now be represented by 5 bits only. An example of binary coding of possible values of compressional wave velocities is shown in Fig. 7.3.

One difficulty with binary coding is that several bits may need to be changed to change the value of one model parameter significantly. For example, 00111 = 1570 m/s; however, 01000 = 1580 m/s, indicating that we need change 4 bits to be able change the velocity value by 10 m/s.

Gray Coding: Gray coding (Forrest 1993), logarithmic scaling, and delta coding (Whitley et al. 1991) have been used to avoid some of the problems associated with simple binary coding. A Gray code is such that the binary representation of two consecutive Gray coded numbers differs by only one bit as shown in Fig. 7.4. Gray codes are generated by forming *bitwise exclusive or* of an integer i with the integer part of i/2 (Press et al. 1989).

Real Coding: Rather than coding the model parameter values, we may work directly with real numbers (floating point). In this case, genetic processes of crossover and mutation are slightly differently compared to those for bit-coded model parameters. The advantages (Spall 2000) of real number coding include relative ease of implementing constraints and that they have natural interpretation. Several

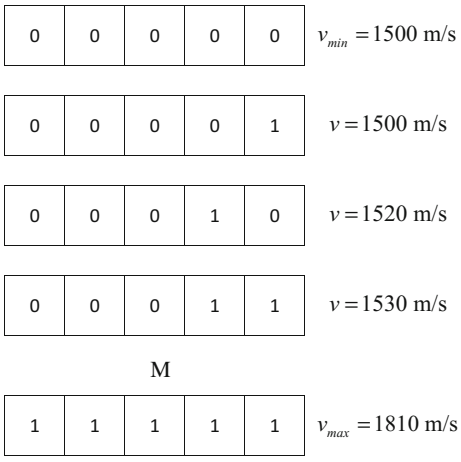


Fig. 7.3 Simple binary coding of model parameters—actual values of the compressional wave velocity are shown in the right panel

Fig. 7.4 Comparison between binary and gray coding

Integer	*Gray Code*	*Binary Code*
0	0000	0000
1	0001	0001
2	0011	0010
3	0010	0011
4	0110	0100
5	0111	0101
6	0101	0110
7	0100	0111
8	1100	1000
9	1101	1001
10	1111	1010
11	1110	1011
12	1010	1100
13	1011	1101
14	1001	1110
15	1000	1111

applications have reported superior performance of real coding over binary coding. Some additional details of real coding is provided below under multi-objective optimization.

Once a coding has been chosen, the model parameters are concatenated one after another to form a string or a vector—this is called a *chromosome*.

7.3.2 Model Selection

Once a population of models is selected, the fitness value (objective function) for each of the models in the population is evaluated. Following this, models are selected from this population in which some models may be repeated while some may be rejected. This is a biased selection process in which models with high fitness values have a higher probability of being selected. Details of this method can be found in Sen and Stoffa (2013).

7.3.3 Crossover and Mutation

First the models are paired and crossover sites are selected at random. The bits on the right of these two models are then swapped to generate new models. This is followed by mutation in which a mutation site is selected at random with low probability and the bit at that location is simply swapped. While the crossover introduces mixing of the model, the mutation imposes diversity in the population.

After selection, crossover and mutation, we repeat the process until the population becomes nearly homogeneous.

7.4 Applications

Since the work of Sen and Stoffa (1991), several applications of GA have been reported in Geophysical literature; many of those are described in Sen and Stoffa (2013). In the following, we describe two applications that are different from the mainstream examples in that the first example focuses on GA's ability to address the issue of model dimensionality in seismic inversion and the second example addresses the application of GA to joint inversion of disparate geophysical datasets.

7.4.1 Multi-scale GA for Trans-Dimensional Inversion

In most geophysical inverse problems, the number of model parameters is assumed known. In other words, the dimension of the model space is held fixed. In general, this parameter is unknown. In reality, model parameters are generally continuous functions of space coordinates. Deriving an infinite number of parameters from a finite set of measurements makes the optimization problem *ill posed*. Classic papers on inverse theory (e.g., Backus and Gilbert 1967) provide a nice exposition of the tradeoff between model resolution and variances. In practice, one chooses the dimension of the model space primarily based on *prior* information. It is often dictated by the availability of computer resources. Ideally, one would like to treat the number of model parameters as a variable and let that be estimated by the data. Such an inverse problem is termed trans-dimensional inverse problem. Most recently, a few papers have been published in geophysics literature on such applications (e.g., Malinverno 2002; Dosso et al. 2014; Sen and Biswas 2017 and references therein).

Hong and Sen (2009) developed a multi-scale GA and applied it to one-dimensional seismic waveform inversion. They addressed two important shortcomings of a 1-D waveform inversion problem (Sen and Stoffa 1991), namely, (1) multi-modality of the objective or fitness function caused primarily by cycle-skipping, and (2) trans-dimensionality caused by the lack of knowledge of the number of subsurface layers. The subsurface is parameterized by a stack of horizontal layers with each layer characterized by Vp (P-wave velocity), Vs (shear wave velocity), and ρ (density). For a given configuration of a model, seismograms are computed by the reflectivity method (Kennett 1983). In general, it is difficult to choose, a priori, the number of model parameters. Sen and Stoffa (1991) clearly demonstrated the effect of under and over-parameterization of the model space in seismic waveform inversion using SA and GA.

Hong and Sen (2009), in their GA application to 1D seismic waveform inversion suggested two major modifications:

1. A hybrid GA-SA method was used in which after the generation of a set of models using the genetic operators (selection, cross-over, and mutation), a Metropolis acceptance criterion (see step 4 in Fig. 4.5) was used to decide if the

generated models could be accepted. The hybrid approach offers tremendous flexibility in controlling the convergence rate. In particular, their algorithm made use of real-coded GA.

2. Several GA chains were run in parallel with each GA having a variable number of model parameters. After each generation, models were swapped between chains. At convergence, they obtained optimal models with optimal number of model parameters.

Hong and Sen (2009)'s multi-scale GA is described in Fig. 7.5. The inverted optimal V_p structures based on the observed seismic gathers are shown in Figs. 7.6 and 7.7, which are compared against the true structures represented by the well logs. In general, it is very difficult to choose an appropriate number of layers in the model before actually inverting the observed data. Therefore, instead of assuming a certain number of layers, Hong and Sen (2009) use multi-scaling to avoid layer definition.

For comparison, the conventional single-scale GA was also implemented four times for the four scales of 10, 20, 40 and 80 layers individually, and 1500 generations of updates were also run on a total of 28 single-scale models for each of the four scales. Every implementation of the single-scale GA also consisted of five different runs, which started with different random seeds. As an example, Fig. 7.6 summarizes the optimization results of V_p from the four individual implementations of the conventional single-scale GA. This is separately done on the four different parameterization cases. Similarly, all the parameterized models finally converged to the actual model, and the finer-scale models lead to better fits. This comparison of these with the results from multi-scale GA (Fig. 7.7) shows that, by using multi-scaling and exchanging information between different scales, convergence of the fine scales is accelerated to an excellent fitness value that a conventional single-scale GA can only obtains after a much longer time period. Thus, the new multi-scale hybrid GA is demonstrated to have better performance in terms of efficiency and accuracy compared to a single-scale stand-alone GA, in that the additionally incorporated coarse scales of faster mixing property facilitate better exploitation of the model space on the fine scale, and this leads to an accurate parameter estimation (Fig. 7.8).

7.4.2 Multi-objective Optimization

So far, we have discussed the optimization problems involving a single objective. Many problems of practical interest however, require interpreting multiple sets instead of a single set of data, which, in turn, involves optimizing or inverting these multiple data for a consistent model that can satisfactorily explain each data-type. These multiple data types are sometimes explained via the same physics but sometimes different physics is required for different data components. Additionally, each data-type may sample the model space at different scales of resolution. Over

Multi-scale hybrid GA

1) For generation 1 (t=1), randomly select an ensemble of N models for total R scales, so each scale has N/R models. For scale i
$\mathbf{m}_{t,j}^{(i)}$, j=1, … , N/R; i=1, … , R; t=1, … , NG.
2) Track temperature T_t along a pre-defined cooling schedule.
3) Evaluate the fitness function $f_{t,j}^{(i)}$ for each model of each scale:
if t=NG maximum generations → STOP
let $F_t^{(i)} = \max\left(f_{t,j}^{(i)}\right)$ and $\overline{F}^{(i)} = \sum_{k=1}^{N/R} f_{t,k}^{(i)} / (N/R)$ i=1, …, R; j=1, …, N/R;
if $F_t^{(i)}$ > some limit → STOP
if $\overline{F}^{(i)} / F_t^{(i)}$ > some limit → STOP
4) For t>1 (all but the first generation), compare current fitness function $f_{t,j}^{(i)}$ to the last generation $f_{t-1,j}^{(i)}$ for each model and each scale
if $f_{t,j}^{(i)} > f_{t-1,j}^{(i)}$ → Accept $\mathbf{m}_{t,j}^{(i)}$
if $f_{t,j}^{(i)} < f_{t-1,j}^{(i)}$, then evaluate $p_{up}^{(i)} = \exp\left(\dfrac{f_{t,j}^{(i)} - f_{t-1,j}^{(i)}}{T_t}\right)$
if $p_{up}^{(i)} > rand[0,1]$ then
$\mathbf{m}_{t,j}^{(i)} = \mathbf{m}_{t-1,j}^{(i)}$, and $f_{t,j}^{(i)} = f_{t-1,j}^{(i)}$
else
Accept $\mathbf{m}_{t,j}^{(i)}$ and $f_{t,j}^{(i)}$
end if
5) Select N/R models for each scale for reproduction based on probability:

$$p_{t,j}^{(i)} = \frac{\exp(f_{t,j}^{(i)} / T_t)}{\sum_{k=1}^{N/R} \exp(f_{t,k}^{(i)} / T_t)}$$

6) Mixing and pairing all the selected total N models.
7) Decompose the models of each pair into two parts:

$$\mathbf{m}^{(i)} = f^{(i)}(\boldsymbol{\varphi}^{(i)}, \boldsymbol{\lambda}^{(i)})$$

8) For each pair, perform crossover on the $\boldsymbol{\varphi}$ part based on a pre-specified probability.
9) Compose the $\boldsymbol{\varphi}$ and $\boldsymbol{\lambda}$ parts into $\mathbf{m}_{t,j}^{(i)}$.
10) Perform mutation on all the N models based on a pre-specified probability.
11) Go to step (2) and repeat the procedure.

Fig. 7.5 Multi-scale genetic algorithm for seismic waveform inversion. From Hong and Sen (2009)

the past, these multi-physics, multi-scale, and multi-objective problems have been treated like single objective optimization, meaning that a single objective is defined as a weighted sum of each objective, which is then followed by a single-objective optimization as we previously discussed. Multi-objective problems are, however,

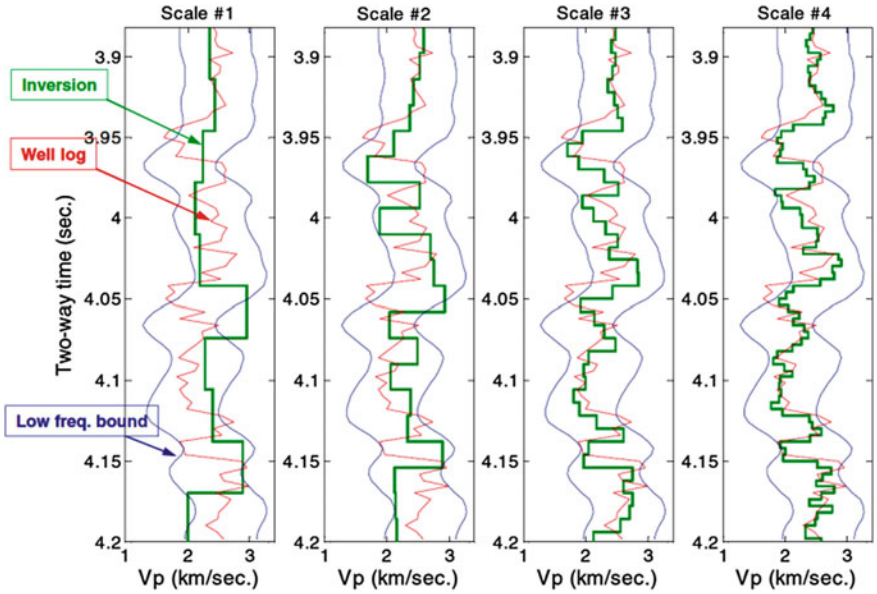


Fig. 7.6 Inverted optimal compressional velocity structure estimation (green—true red) obtained by running several independent GA runs with different resolutions from Hong and Sen (2009). Note that fine resolution results in convergence to the actual model while coarse resolution results in underfitting

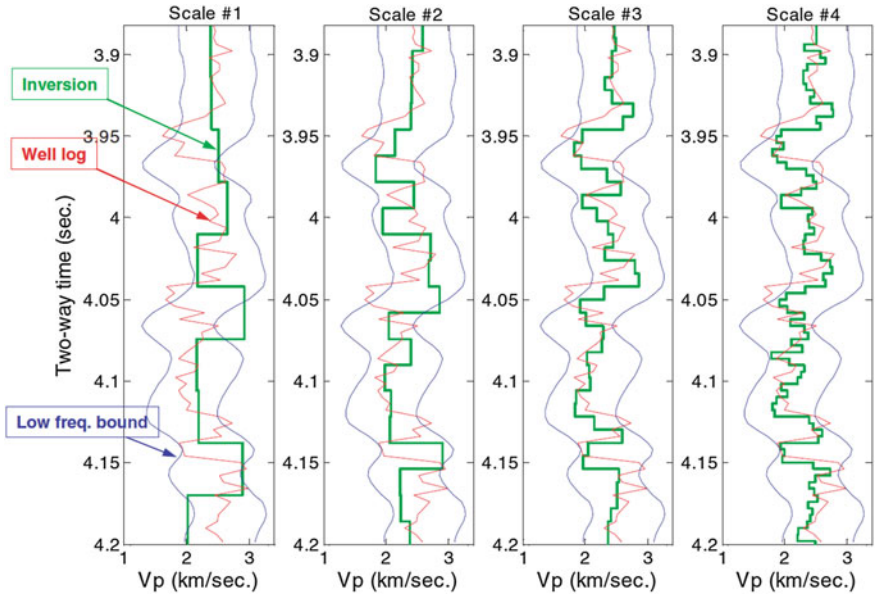


Fig. 7.7 Inverted velocity structures using multi-scale GA for four scales (green) and comparison with the true model (red) from Hong and Sen (2009). As shown, the over-parameterized models converge to the actual models

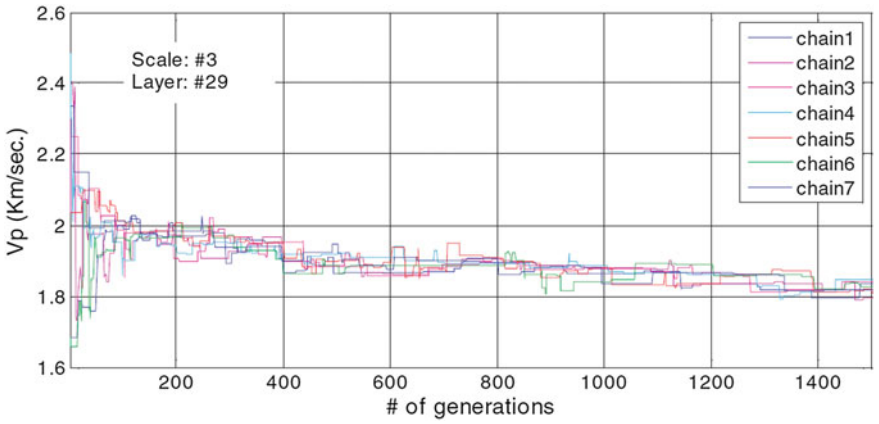


Fig. 7.8 Samples of the compressional wave velocity drawn by the seven MCMC chains at scale 3. As shown, all seven chains converge to the same value, even though they start at different scales. From Hong and Sen (2009)

non-unique with multiple sets of solutions known as the Pareto-optimal solutions, none of which could be considered better than the other in terms of satisfying all objectives (Deb et al. 2002). In multi-objective problems, it is important to estimate the Pareto-optimal set of solutions because it provides a way to calculate the uncertainty associated with estimating each model parameter. Note that if we cast a multi-objective into a single-objective problem, it gives only one out of the complete set of the Pareto-optimal solutions that may be biased by the weights we choose to define the objective. Of course, by changing weights and rerunning the inversion multiple times can in theory allow estimating the entire Pareto-optimal set, but such a procedure is too cumbersome to apply in practice. Although some techniques for automatically choosing and modifying the weights attached to each objective has been suggested (Hajela and Lin 1992; Ritzel et al. 1994; Miettinen 1999), they all give one solution or at most a small fraction of the Pareto-optimal set that may or may not be the most desirable. To handle multi-objective problems, it is therefore advisable to use an optimization method capable treating all objectives as the components of an "objective vector" and simultaneously minimize (or maximize) them all via an iterative process to estimate these Pareto-optimal set of solutions in a single run.

Before proceeding further, let us first define what we mean by the Pareto-optimal set of solutions in relation to a geophysical inverse problem where two-component (vertical and radial) seismic data are simultaneously inverted. Without any loss of generality, we can cast such a problem as a minimization problem where the suitably defined objectives for each data component are simultaneously minimized. Now consider the set of solutions from such a two-component optimization, shown in Fig. 7.9. In this figure the horizontal and vertical axes respectively denote the

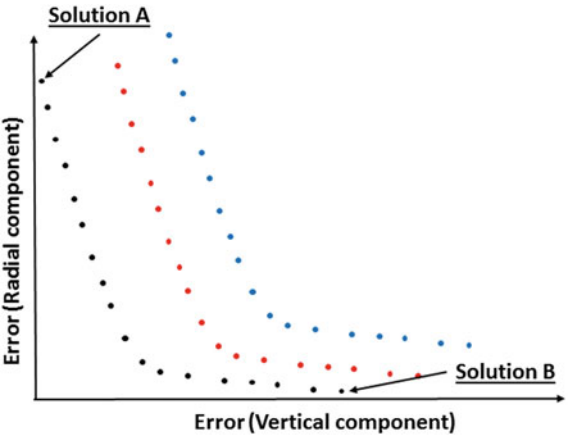


Fig. 7.9 Demonstration of the concept of Pareto-optimality in multi-objective optimization

error (or objective) of the vertical and radial data components and the dots represent sets of solutions. Now consider two solutions, marked as "Solution A" and "Solution B" in Fig. 7.9. Note that the solution A minimized the objective of the vertical data component better than the solution B. Solution B on the other hand minimized the objective of the radial data component better than the solution A. In fact, there are many more such solutions, represented as black dots in Fig. 7.9, none of which can be considered better than the other in terms of minimizing both objectives. This set of solutions is the Pareto-optimal set of solutions, which, in the objective space, form a front, called the Pareto optimal front, and is represented by the black dots in Fig. 7.9. This concept of Pareto-optimality can be extended further to define multiple sets of the Pareto-optimal solutions (Pareto-optimal fronts), as represented by red and blue dots in Fig. 7.9. Note that the solution set represented by the black dots can be considered better than the set represented by the red dots. The set represented by red dots are better than the set represented by the blue dots, and so on. All multi-objective methods aim at finding these Pareto-optimal solutions during the optimization.

7.4.2.1 Multi-objective Optimization Methods in Geophysics

For many years, seismic data have been the most effective tool for delineating and characterizing the subsurface. In exploration geophysics, these seismic data have traditionally been single-component. However, primary focus of the oil and gas industry is now shifting from the conventional to unconventional and naturally fractured reservoirs. To mitigate global warming from fossil-fuel burning, there is also a growing need for carbon dioxide (CO_2) sequestration in deep saline reservoirs. Additionally, fractured geothermal reservoirs are also a focus of attention as an alternate source of energy in recent years. All these reservoirs are anisotropic and to correctly handle the anisotropic effects of wave propagation, acquisition, processing,

and inversion of the full wave-field, seismic data is required. Such full wave-field data are multi-component (generally three- and sometimes nine-component). Inverting multi-component seismic data is a multi-objective optimization problem because each data component has its own objective. Such a multi-objective problem is single-physics because all data components are explained through the physics of the propagation of elastic waves through an anisotropic medium. However, because the vertical component seismic traces are dominated by the fast P-wave mode while the horizontal data components are dominated by the two slow S-wave modes, the problem is multi-scale. An accurate reservoir description and monitoring requires seismic data to be combined with other geophysical data (electromagnetic, gravity, etc.) and engineering (reservoir flow simulation and geomechanical analysis etc.) data. For example, while seismic data are effective in delineating different subsurface rock formations (lithology), they are not very sensitive to the different fluids contained within their porosity (oil, gas, water, CO_2, etc.). Electromagnetic (EM) data on the other hand, are not very sensitive to the subsurface lithology, but are very sensitive to the formation fluids. Consequently, combining seismic with EM is useful for accurately delineating the subsurface rock and fluid properties (Lang and Grana 2015). Additionally, reservoir description is not only delineating the properties once, but also monitoring them over time. As hydrocarbon is produced from conventional/unconventional/naturally-fractured reservoirs or energy is produced from the hydrothermal reservoirs for example, geophysical data must be reacquired and combined with the production data to monitor the subsurface fluid movements and optimize subsequent production. All CO_2 sequestration experiments require post-injection monitoring to ensure that the injected fluid is in place and does not leak out into the atmosphere. Such monitoring requires combining geophysical data with the engineering data such as the reservoir fluid-flow simulation and geomechanical modeling, etc., into a multi-objective optimization that is not only multi-scale, but also multi-physics.

7.4.2.2 General Overview of Multi-objective Optimization

Multi-objective inverse problems deal with the estimation of the Pareto-optimal set of solutions and is relatively a new field of research. In theory, an exhaustive search in the model space would allow such estimation. However, in most cases of practical importance it is computationally expensive, and due to the complexity of the underlying physics, it is often unfeasible to search for the exact set. Therefore, several stochastic strategies such as evolutionary algorithms, particle swarm optimization, scatter search, simulated annealing, etc. have been proposed and developed. All these approaches attempt to make a reasonable approximation to the Pareto set. A very good overview of the current state-of-the-art for multi-objective optimization can be found in Konak et al. (2006), Tang and Wang (2013), and Giagkiozis et al. (2015).

Although different Pareto-based optimizations differ from one another in terms of their specific implementation details, they all are based on evaluating multiple

objectives and then sorting the individual members in terms of their levels of non-dominance. Without any loss of generality, from here on the multi-objective optimization will be treated as a minimization problem, i.e., each objective to be optimized will be evaluated using equation 7.8 or its equivalent form, and then simultaneously minimized.

Multi-objective inverse problems first start with defining the model vector $\mathbf{x}$ in the decision space $\boldsymbol{X}$ given as

$$\mathbf{x} = (x_1, x_2, x_3, \ldots, x_n). \tag{7.9}$$

In Eq. 7.9, n defines the number of model parameters, and x_i represents the model or solution, associated with the model parameter i. We then define a function $\boldsymbol{f}$: $\boldsymbol{X} \rightarrow \boldsymbol{Y}$ which evaluates the quality of the specific solution by assigning it an objective vector $\mathbf{y}$ in the objective space $\boldsymbol{Y}$, given as

$$\mathbf{y} = (y_1, y_2, y_3, \ldots, y_k). \tag{7.10}$$

Clearly, k in Eq. (7.10) is the number of objectives. By defining a total number of k objectives as above, we then define a set of solutions $\mathbf{x^1}$ to dominate over another set, say $\mathbf{x^2}$ (denoted as $\mathbf{x}^1 \succ \mathbf{x}^2$) when the following conditions are satisfied:

- If $\mathbf{x}^1 = \left(x_1^1, x_2^1, \ldots, x_n^1\right)$ and $\mathbf{x}^2 = \left(x_1^2, x_2^2, \ldots, x_n^2\right)$ in the decision space $\boldsymbol{X}$ maps respectively onto $\mathbf{y}^1 = \left(y_1^1, y_2^1, \ldots, y_k^1\right)$ and $\mathbf{y}^2 = \left(y_1^2, y_2^2, \ldots, y_k^2\right)$ in the objective space $\boldsymbol{Y}$ then
  - $y_i^1 \leq y_i^2$, for all i = 1, 2,…, k.
  - $y_j^1 < y_j^2$, at least for one value of j where $1 \leq j \leq k$.

Having introduced the dominance $(\mathbf{x}^1 \succ \mathbf{x}^2)$ as above, we can now readily see that in Fig. 7.9, the Pareto-optimal set defined as black dots dominate the sets represented by the red and blue dots, and the set defined by the red dots is dominated by the set of black dots, but dominates the set defined by the blue dots, and so on.

In all multi-objective optimizations, a set of solutions in the decision space are first randomly generated based on their respective user-defined constraints. In multi-objective evolutionary algorithms (MOEA), these solutions are called the "members" or the "individuals" and the entire set is called the "population". These terminologies do however vary depending upon the method used. For example, in particle swarm optimizations (PSO), individual members are called the "particles" and the entire set of the members are called the "swarms", but from here on, the MOEA-based terminologies will be used.

Once the random population is generated, they are sorted, based on their levels of non-dominance. Two most popular approaches to such non-dominated sorting are (1) rank based sorting, and (2) domination-count or domination-strength based sorting (Konak et al. 2006). In rank based sorting, the members are grouped into different ranks based on their level of non-dominance. The members belonging to rank 1 are those which do not dominate one another, but they dominate the rest that

are not yet ranked. Similarly, the members belonging to rank 2 are dominated by the rank 1 members, but they dominate the rest. This process of such ranking is continued for all members of the population. Thus, in Fig. 7.9, the black dots would represent rank 1, the red dots would represent rank 2, and the blue dots would represent rank 3. On the other hand, in domination-count based sorting, each individual member $\mathbf{x}$ in the decision space $\mathbf{X}$ is assigned a strength $S(\mathbf{x})$ which is simply a measure of the number of solutions $\mathbf{x}$ dominates. Following strength assignment, in domination-count based soaring; each individual $\mathbf{x}$ is assigned a raw fitness as a measure of the number of solutions $\mathbf{x}$ is dominated by. Thus, if a solution is not dominated by any other solutions, its raw fitness is 0 and if it is dominated by many individuals, its raw fitness is high. Therefore, in Fig. 7.9 the raw fitness of the black dots would be 0, the raw fitness of the red dots would be the number of solutions represented by the black dots, and the raw fitness of the blue dots would be the total number of the black and red dots.

Although sorting the population in terms of ranks or raw fitness provides the niching mechanism based on the Pareto dominance, it may still fail when most members do not dominate each other. In evolutionary algorithms, such a situation is called "genetic drift" (Goldberg 1989). Developments of all stochastic search methods are based on the assumption that the population size is infinite. However, in practice, the population size must be finite, and any stochastic search process using a finite population size has a tendency to cluster near a single solution over time. Notice that such a clustering is desirable when the solutions are near the global optimum, but it must be avoided at all other times to avoid premature convergence to a local optimum. Therefore, the next step in multi-objective optimizations is to maintain diversity in the population so that the clustering can be avoided. Thus, each member is assigned a measure of its diversity. This is usually done in two ways, (1) computing the crowding distance which is simply the normalized distance of a member from its nearest neighbors along different objective axes, or (2) computing the distance of a member from its k-th nearest neighbor, for a suitably chosen value of k (Silverman 1986), also measured along different objective axes. The inverse of the distance of a member form its k-th nearest neighbor is then treated as its measure of population density and added to the raw fitness value as a measure of its fitness. Thus, out of any two individuals with the same rank or raw fitness, the one that is less crowded or less densely populated is preferred over the other.

After non-dominated sorting and diversity computations, a population of new members or solutions is created. In MOEA, creating the new population of members from the old one is called advancing from one generation into the next, and is performed using the GA operations of tournament selection, crossover, mutation, and elitism as described earlier. There are two popular approaches to create such a new set: (1) without using any external archive and (2) using an external archive. In the first approach, the tournament selection reproduces a set of members from the old set in proportion to their levels of non-dominance and diversity. In crossover, the members after the tournament selection are treated as parents, and an entirely a new set of members, called the "children" or "off-springs" are produced. In mutation, parameters of the children are partially changed using a given probability

of mutation. After computing the objectives of the mutated children, in elitism the parents and their children are combined into a single set, sorted according to their non-dominance levels and diversity, and are compared to select the best set of members to advance to the next generation, and this process is repeated until a predefined stopping criterion is reached. In the second approach, an external set of archive members are maintained along with the current population. The current population and the archive population are sorted according to their non-dominance and diversity, and the best members are copied into the archive. This is followed by the tournament selection from the archive, and crossover and mutation to create the children. Once the children are created, the entire population is replaced by the children, which is again combined with the archive to repeat the same process until some predefined stopping criterion is reached.

Even though all multi-objective optimizations are based on the general ideas described above, their specific implementations are algorithm-dependent, and out of many multi-objective methods proposed to date including the hybrid implementations, the two most promising ones are the fast non-dominated sorting genetic algorithm or NSGA II (Deb et al. 2002) and the improved version of the strength Pareto evolutionary algorithm or SPEA2 (Zitzler et al. 2001). NSGA II, shown in the flow diagram of Fig. 7.10 uses ranks for non-dominated sorting, crowding distance for population diversity, and does not use any external archive in generating the new members. SPEA2, shown in Fig. 7.11, uses domination-count for non-dominated sorting, *k*-th nearest neighbor for population diversity, and uses an external archive.

7.4.2.3 Geophysical Examples of Multi-objective Optimization

Application of multi-objective optimization to solve geophysical inverse problems is relatively new. Estimation of an anisotropic model of mantle lithosphere using the splitting parameters of teleseismic shear waves and P wave residual spheres using a multi-objective optimization method was reported by Kozlovskaya et al. (2007). For wave-equation migration velocity inversion, Singh et al. (2008) jointly minimized the semblance and differential semblance using MOEA. To estimate earthquake focal mechanisms, Heyburn and Fox (2010) jointly inverted the tele-seismic body wave data and regional surface wave amplitude spectra using a multi-objective optimization method.

The first application of multi-objective optimization to invert the multicomponent seismic waveform data was carried out by Padhi and Mallick (2013, 2014). They implemented the original version of NSGA II and demonstrated that the method is capable of extracting the elastic properties and density from multicomponent seismic data, under both isotropic and anisotropic subsurface conditions. The anisotropy was however a special type called vertical transverse isotropy or VTI, where the medium is horizontally isotropic and vertically anisotropic (Thomsen 1986). In addition, Padhi and Mallick (2014) also developed a practical way to approximately calculate the uncertainties associated with estimating each

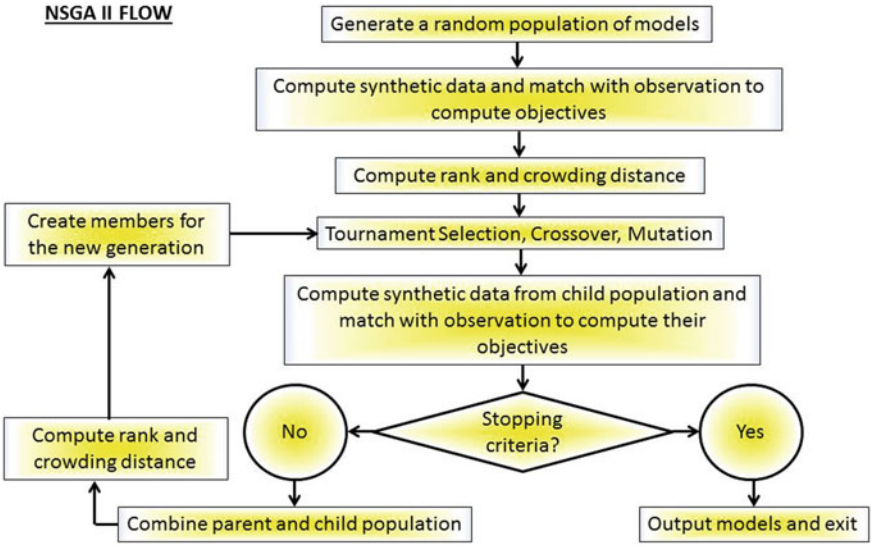


Fig. 7.10 NSGA II workflow based on non-dominated sorting in ranks, crowding distance for population diversity, and does not use an external archive

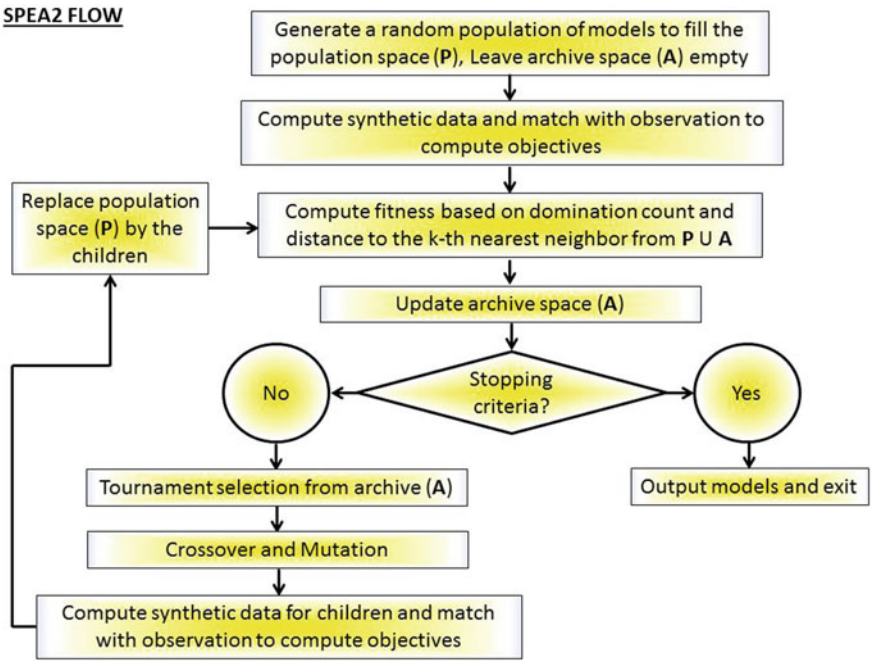


Fig. 7.11 SPEA2 workflow, based on domination count, distances to the k-th nearest neighbor for population diversity, and uses an external archive

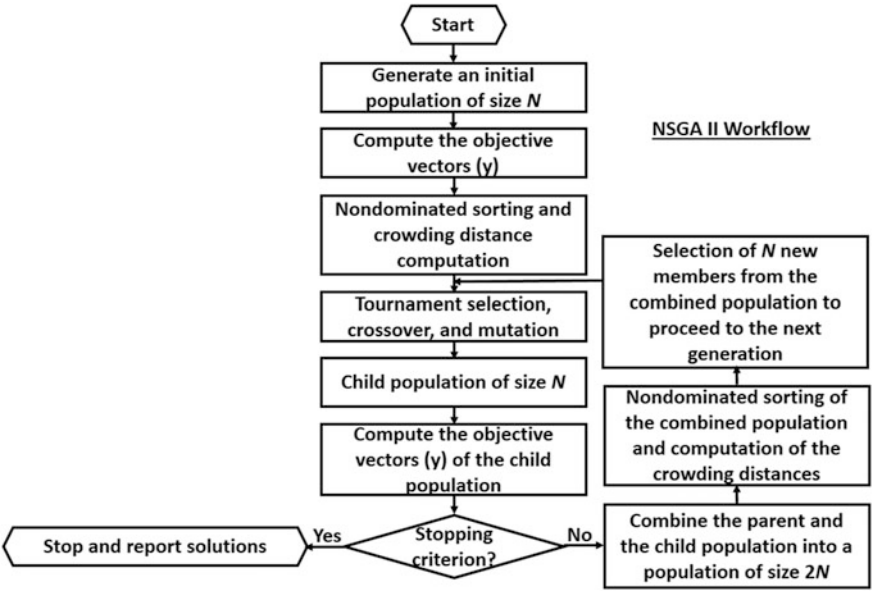


Fig. 7.12 NSGA II implementation of Padhi and Mallick (2013, 2014). Reproduced from Padhi and Mallick (2014)

model parameter. Figure 7.12 is the implementation of Padhi and Mallick (2013, 2014). The inversion was implemented in theτ-p domain, and another important concept introduced by Padhi and Mallick (2013, 2014) is the implementation of a real-coded GA, which was briefly discussed earlier, but needs further clarifications.

Coding of the models in the decision space is traditionally done using a binary coding (Goldberg 1989; Stoffa and Sen 1991; Mallick 1995). We assume that the model vectors $\mathbf{x}^{1,t} = \left[x_1^{1,t}, x_2^{1,t}, \ldots x_n^{1,t}\right]$ and $\mathbf{x}^{2,t} = \left[x_1^{2,t}, x_2^{2,t}, \ldots x_n^{2,t}\right]$, in which $x_i^{1,t}$ and $x_i^{2,t}$ are the model parameters for the i-th component of the respective model vectors represent two parents in any given generation, say t, randomly chosen out of the population pool. As previously noted, each component of the model vector is represented as a string of binary digits (bits) in binary coded GA. Then, by choosing a random crossover point within each model parameter, the bits on the right-hand side of these crossover points are swapped between the parents with a given probability of crossover P_c to produce two children. Crossover is followed by mutation in which each bit in both children are sequentially visited and changed with a predefined probability of mutation P_m. As previously noted, binary coded GA can only discretely sample the decision space, which, in turn, is controlled by the number of bits that are used to define each model parameter and their minimum and maximum search bounds chosen. Considering the fact that the decision space is continuous, it is desirable to develop a parameter coding capable of continuous sampling of this space. Although there are different ways to implement such continuous sampling, the

real coded GA with simulated binary crossover (SBX) and real parameter mutation (RPM) operators developed by Deb and Agrawal (1995, 1999) is straightforward and were used by Padhi and Mallick (2013, 2014) for seismic waveform inversion. The exact procedures for SBX and RPM are as follows.

Here we describe how does the SBX and RPM work with each model parameter $x_i^{1,t}$ and $x_i^{2,t}$ of the parent population and produce the corresponding model parameters $x_i^{1,t+1}$ and $x_i^{2,t+1}$ for their children. By repeating exactly the same procedure for all model parameters will thus produce two children from their parents. Before proceeding further, it is worth mentioning that strictly, we should not use the superscript $t + 1$ to represent the children. Once a set of N children are created in any generation t, they are combined with the parents to choose the best set of N members for the next generation $t + 1$ from the combined population via ranking and diversity (see Fig. 7.12). But, here we use $x_i^{1,t+1}$ and $x_i^{2,t+1}$ to represent the i-th model parameter for the two children, corresponding to the i-th model parameters $x_i^{1,t}$ and $x_i^{2,t}$ of their parents, and the superscripts t and $t + 1$ do not represent two different generations.

SBX, implemented by Padhi and Mallick (2014), first calculates a parameter β_{qi}, given as

$$\beta_{qi} = \begin{cases} (u_i\alpha)^{\frac{1}{\eta+1}}; & u_i \le \frac{1}{\alpha} \\ \left[\frac{1}{2-u_i\alpha}\right]^{\frac{1}{\eta+1}}, & \text{otherwise.} \end{cases} \tag{7.11}$$

In Eq. 7.11, u_i is given as a random number between 0 and 1, and η is called the crossover distribution index, which is a non-negative real number. Note that choosing a large value of η gives a higher probability of selecting the child solutions near the parent solutions. And, if a small value of η is chosen, child solutions tend to lie far away from their parents. In an ideal scenario, one must therefore use a small value of η in the early generations and then gradually increase it later when the solutions merge to the global optimum. In their waveform inversion applications however, Padhi and Mallick (2013, 2014) used a constant η for all generations. Finally, α in Eq. (7.11) is given as

$$\alpha = 2 - \frac{1}{\beta^{\eta+1}}, \tag{7.12}$$

with

$$\beta = 1 + \frac{2}{x_i^{2,t} - x_i^{1,t}} \min\left[\left(x_i^{1,t} - x_i^{L,t}\right), \left(x_i^{U,t} - x_i^{2,t}\right)\right]. \tag{7.13}$$

In Eq. (7.13), $x_i^{L,t}$ and $x_i^{U,t}$ are respectively the lower and upper search limits for the model parameter i in the parent population. Once β_{qi} is defined via Eqs. 7.11 through 7.13, the child solutions are

$$x_i^{1,t+1} = 0.5\left[\left(x_i^{1,t} + x_i^{2,t}\right) - \beta_{qi}\left(x_i^{2,t} - x_i^{1,t}\right)\right], \tag{7.14}$$

and

$$x_i^{2,t+1} = 0.5\left[\left(x_i^{1,t} + x_i^{2,t}\right) + \beta_{qi}\left(x_i^{2,t} - x_i^{1,t}\right)\right]. \tag{7.15}$$

It is explicitly assumed that in Eqs. 7.14 and 7.15 $x_i^{2,t} > x_i^{1,t}$. It is however straightforward to write a similar set of equations for $x_i^{1,t} > x_i^{2,t}$, but in practice it is not necessary. In any practical application, we can always compare the model parameters for both parents before crossover and denote the one with higher value as parent 1 and the other as parent 2 and directly use Eqs. 7.14 and 7.15. From the Eqs. 7.13 and 7.17, it must also be noted that the SBX creates the children that are uniformly distributed between their parents. This is deliberate so that there is no bias in producing the children towards either of the parents (Deb and Agrawal 1995). Also, like the binary coded GA, the real coded GA also performs the crossover with a given probability P_c.

In RPM, implemented by Padhi and Mallick (2014), the model parameter i for a mutated child $x_i^{mutated}$ is produced from the original one x_i^{child} as follows, and by repeating the same for all model parameters and for all children generates the mutated child population:

First, we define $\bar{\delta}_i$ as

$$\bar{\delta}_i = \begin{cases} \left[2r_i + (1-2r_i)(1-\delta)^{\kappa+1}\right]^{\frac{1}{\kappa+1}} - 1; & r_i \le 0.5 \\ 1 - \left[2(1-r_i) + 2\left(r_i - \frac{1}{2}\right)(1-\delta)^{\kappa+1}\right]^{\frac{1}{\kappa+1}}, & \text{otherwise.} \end{cases} \tag{7.16}$$

In Eq. 7.16, r_i defines a random number between 0 and 1, and κ is called the mutation distribution index. Similar to η in Eq. 7.11, it is a non-negative number; a small value of which produces the parameter of the mutated solution far away from the original and a large value produces solution close to it. Thus, ideally κ should be varied over the generations, but Padhi and Mallick (2013, 2014) used a constant κ for all generations. The parameter δ in Eq. 7.16 is defined as

$$\delta = \frac{\min\left[\left(x_i^{child} - x_i^L\right), \left(x_i^U - x_i^{child}\right)\right]}{x_i^U - x_i^L}, \tag{7.17}$$

where x_i^U and x_i^L are the upper and lower search limits for the model parameter x_i. By defining $\bar{\delta}_i$ as above in Eq. 7.17, the mutated model parameter $x_i^{mutated}$ is computed from x_i^{child} as

$$x_i^{mutated} = x_i^{child} + \left(x_i^U - x_i^L\right)\vec{\delta}_i \tag{7.18}$$

Again, similar to the binary coded GA, the real coded GA also performs the mutation using a probability of mutation P_m.

Figures 7.13 and 7.14 show the inversion results from Padhi and Mallick (2014) on noisy synthetic seismic data, generated from a real well-log. Two-component (vertical and radial) synthetic data were simultaneously inverted using the multi-objective scheme to obtain these results.

Figure 7.13 shows the inversion result for two-component seismic data under isotropic assumptions. Note that an isotropic elastic earth model is described by three model parameters-the P-wave velocity (V_P), the S-wave velocity (V_S), and density (ρ). Padhi and Mallick (2014) however parameterized their model in

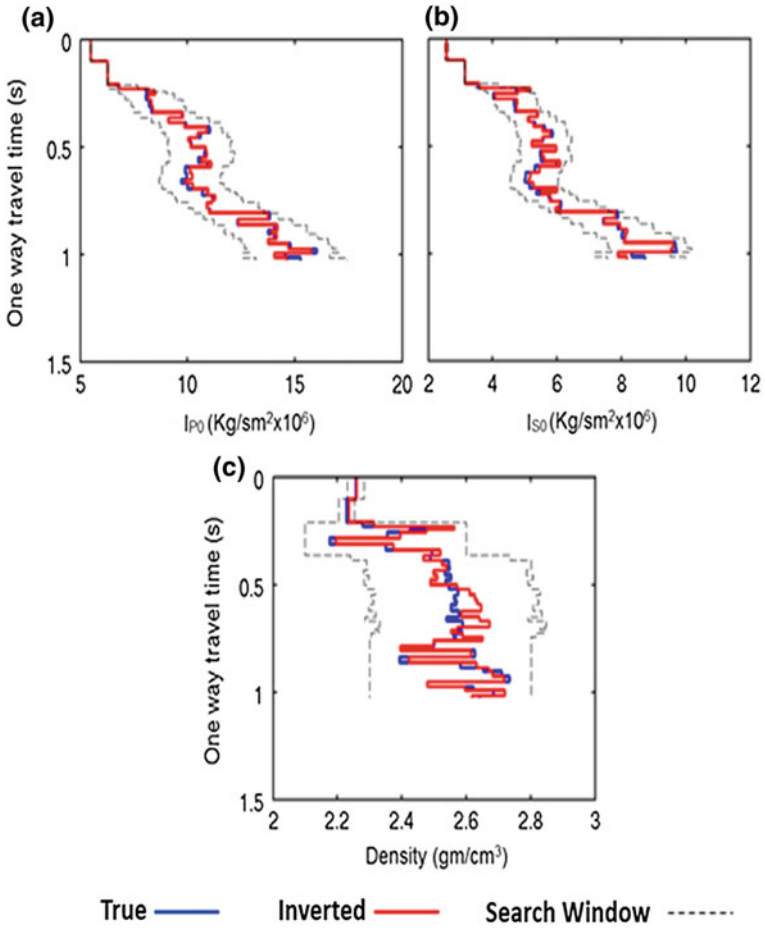


Fig. 7.13 Isotropic inversion of the two-component (vertical and radial) noisy synthetic data. The true model is shown in blue, the inverted model is shown in red, and the dashed lines represent the upper and lower search limits for each of the model parameters. This figure is reproduced from Padhi and Mallick (2014)

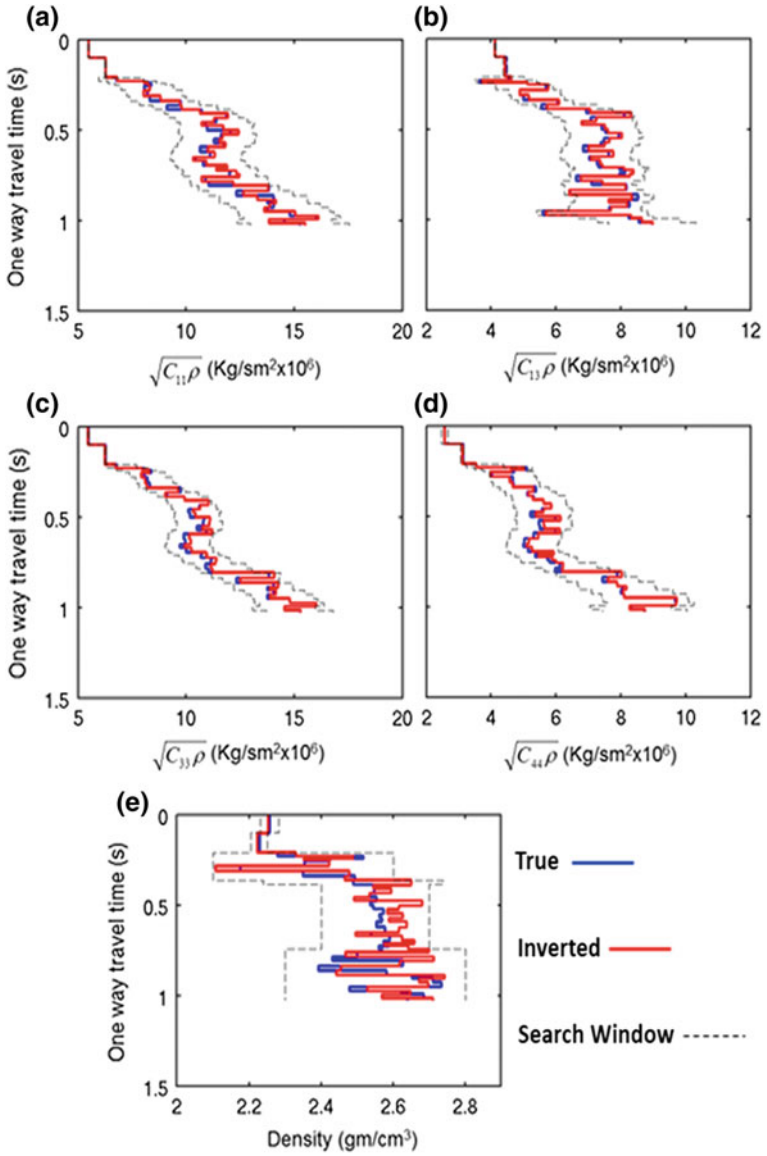


Fig. 7.14 Same as Fig. 7.13, but for the VTI model parameters. The figure is reproduced from Padhi and Mallick (2014)

impedance units (P and S-wave impedance, shown in Fig. 7.13 as I_{P0} and I_{S0}) and density. The inversion results shown in Fig. 7.13 used a model population size of 400, and the maximum number of generations of 800. Additionally, it used the crossover probability (P_c) = 0.9, the mutation probability (P_m) = 0.08, crossover distribution index (η) = 20, and the mutation distribution index (κ) = 10.

The inversion result for two-component noisy synthetic data with a VTI assumption is shown in Fig. 7.14. A VTI (Vertical Transverse Isotropy) model is described by five elastic constants (C_{11}, C_{13}, C_{33}, C_{44}, C_{66}) and density. Alternatively, in geophysical literature, is also common to describe a VTI model with Thomsen parameters- vertical P- and S-wave velocities (V_{P0}, V_{S0}), three dimensionless parameters (ε, δ, and γ) and density (Thomsen 1986). Again, as shown in Fig. 7.14, Padhi and Mallick (2014) parameterized their VTI inversion in impedance units and density. Additionally, because C_{66} (or γ) does not influence the vertical and horizontal (radial) data components, in Fig. 7.14, inversion results for only four out of five elastic coefficients and density are shown. The VTI inversion result of Fig. 7.14 used a population size of 800, maximum number of generations of 1400, $P_c = 0.9$, $P_m = 0.05$, $\eta = 20$, and $\kappa = 10$.

Padhi and Mallick (2014) took their multi-objective inversion further to compute an approximate a posteriori probability density (PPD) in the model (decision) space for each model parameter. The procedure to approximately compute the PPD for single objective problems was outlined by Sen and Stoffa (1991) and Mallick (1995), and it is possible to extend the same concept to find the PPD for each model parameter associated with each objective for multi-objective problems, which in fact, could be called a Pareto-optimal PPD. But, as the number of objective increase, interpreting such a Pareto-optimal PPD becomes difficult. In dealing with multi-objective problems, Mosegaard and Tarantola (1995) points out that plotting all model solutions in the model space in a normalized histogram display is a reasonable approximation to the true PPD. Following this idea, Fig. 7.15 is a subset of the solutions in the model (decision) space, shown in a normalized histogram display as the approximate representation of the PPD for the VTI inversion problem of the noisy synthetic data. Thus, in each plot shown in Fig. 7.15, the vertical axis represents value of the specific model parameter and the horizontal axis represents the layer number, and each curve represents a normalized histogram display of the chosen subset of solutions. In addition, Fig. 7.16 is an estimate of the Pareto-optimal front of rank 1 at the end of the predefined number of generations for the two-component VTI inversion problem. Padhi and Mallick (2014) claim that combining the histogram display (Fig. 7.15) with the Pareto optimal front (Fig. 7.16) is an effective way to assess the quality of the inverted results and quantify the uncertainty in estimating each model parameter.

For most geologic areas of exploration interest, a VTI-type anisotropy is typically caused by thin layers of sediments whose thickness are much smaller than the wavelength of the seismic waves propagating through them. These propagating waves do not see these individual layers, but their overall effect is manifested by a VTI type of behavior on the recorded seismic data (Schoenberg 1983). In most areas, these thin layers are also associated with fractures and/or maximum and minimum in situ horizontal in situ stress fields S_{Hmax} and S_{hmin} (Zoback 2010). These fractures/stress-fields on top of layering make the subsurface azimuthally anisotropic with orthorhombic (ORT) properties where the medium is isotropic only

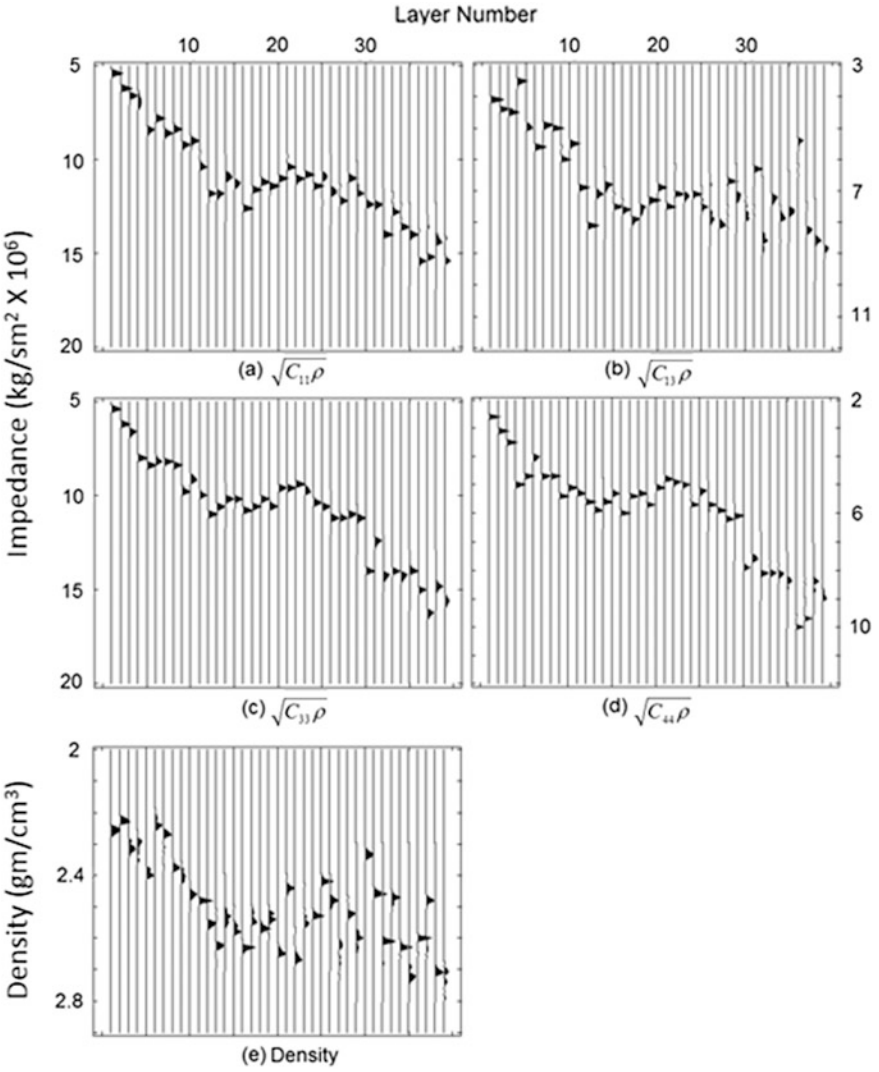


Fig. 7.15 Estimated PPD (posteriori probability density) for all layers and for **a** $\sqrt{(C_{11}\rho)}$, **b** $\sqrt{(C_{13}\rho)}$, **c** $\sqrt{(C_{33}\rho)}$, **d** $\sqrt{(C_{44}\rho)}$ and **e** ρ for the VTI inversion. This figure is reproduced from Padhi and Mallick (2014)

along three mutually orthogonal symmetry axes and anisotropic elsewhere (Schoenberg and Douma 1988; Thomsen 1988). Fractures and in situ stress fields are closely related to one another and estimating them from seismic data is of practical importance in the exploration and exploitation of the unconventional, naturally fractured, and CO_2 sequestered reservoirs. For unconventional reservoirs, knowledge of S_{Hmax} and S_{hmin} allow proper placement of hydraulic fractures for an

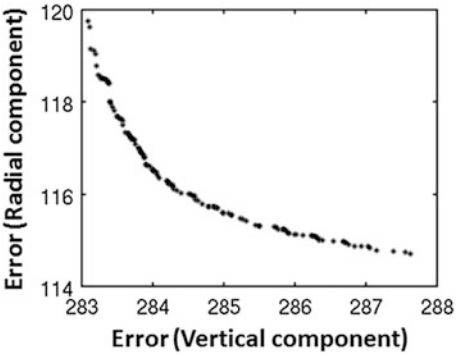


Fig. 7.16 Pareto-optimal front of rank 1, estimated at the end of the VTI inversion. The figure is reproduced from Padhi and Mallick (2014)

optimum recovery of the hydrocarbon resources with the least damage to the environment (Zoback 2010). For naturally fractured reservoirs, knowledge of the fracture parameters such as their orientation, density, etc., allows an optimum placement of horizontal wells for maximum resource recovery. As CO_2 injection can potentially alter the in situ subsurface stress fields, their knowledge is important in designing an injection scheme such that the fractures can be avoided in the overlying sealing formations and at the same time be initiated within the injecting reservoirs to enhance additional storage (Campbell-Stone et al. 2012). Because the fracture parameters, S_{Hmax} and S_{hmin} can be calculated directly from the estimated ORT properties (Gray et al. 2012), developing an inversion method for estimating the subsurface properties to a complexity of an ORT symmetry is necessary.

In view of the above, Li and Mallick (2015) extended the work of Padhi and Mallick (2013, 2014) to develop a multi-objective inversion method for ORT media properties. To handle ORT properties, the original NSGA II was completely modified and rewritten by Li and Mallick (2015) as follows:

- Each aspect of NSGA II was carefully analyzed and the entire method was parallelized. As shown in Fig. 7.17, this parallel NSGA II distributes the parent population into different nodes, evaluates the objectives and performs the non-dominated sorting, and sends them back to the master node for subsequent processing.
- The raw objectives were first linearly scaled (Goldberg 1989; Mallick 1995) and these scaled objectives were used for the non-dominated sorting of the population into their respective ranks.
- The crowding distance to preserve population diversity was defined not only as the function of the scaled objectives but also as a function of how much a given model is diverse from the others in the model space. Note that in this respect, this modified NSGA II utilizes the SPA2 concept of the k-th nearest neighbor as a measure of population diversity where k is provided from the model space. For the multi-objective optimization problems where the total number of parameters to be estimated is much higher than the number of objectives as is the case for

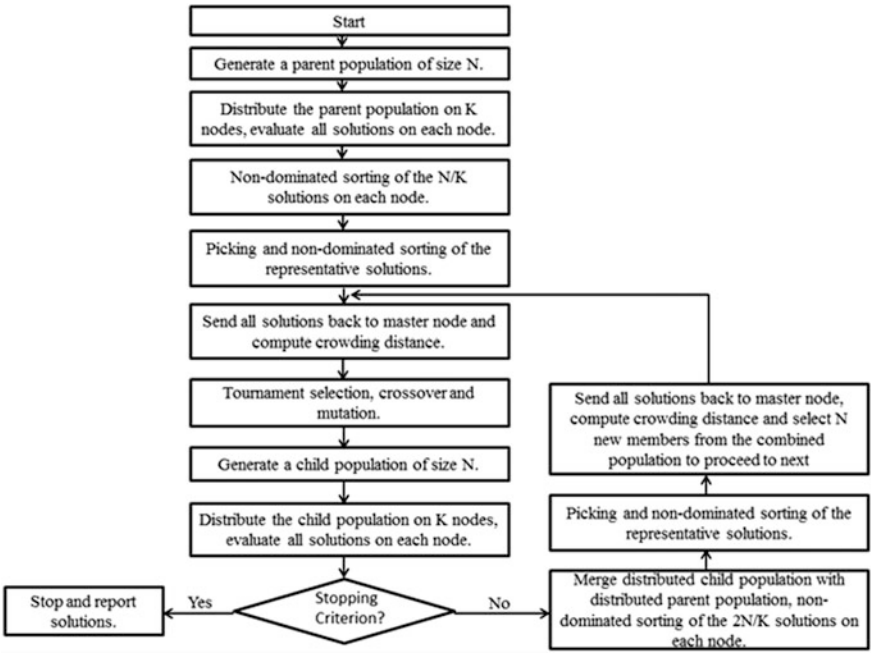


Fig. 7.17 Workflow of an improved and parallelized version of multi-objective optimization. The figure is reproduced from Li and Mallick (2015)

the ORT inverse problem, a modification of the NSGA II to adapt to this concept of SPEA2 was necessary.

- To achieve continuous sampling of the model space, a real coded GA was used following the procedure outlined above. In this real-coded GA, the crossover and the mutation probabilities (P_c and P_m) and their respective indices (η and κ) were varied over the generations for a fast convergence to the true solutions (see below for details).
- Computing the approximate PPD curves as histogram displays as implemented by Padhi and Mallick (2014) was also implemented to approximately quantify the uncertainty associated for each model parameter estimate.

Although this modified parallel implementation of Li and Mallick (2015) utilizes a few concepts from SPEA2, in essence it still is an NSGA II methodology because this method does not maintain an external archive like SPEA2. Therefore, we prefer to call this multi-objective optimization method to be an improved and parallel version of NSGA II.

An ORT (Orthrhombic) medium is described by eleven model parameters- nine elastic constants, density, and the direction of the symmetry axis. The nine elastic constants can be described either as the stiffness constants, C_{11}, C_{12}, C_{13}, C_{22}, C_{23}, C_{33}, C_{44}, C_{55}, and C_{66} or as Thomsen-Tsvankin parameters (Thomsen 1986; Tsvankin 1997) given as the vertical P- and S-wave velocities (V_{P0}, V_{S0}), and seven

dimensionless parameters ε_1, ε_2, δ_1, δ_2, δ_3, γ_1, and γ_2. In contrast with the impedance parameterization of Padhi and Mallick (2013, 2014), Li and Mallick (2015) parameterized their inversion using the Thomsen-Tsvankin parameterization. Also, to efficiently implement inversion for ORT subsurface properties, Li and Mallick (2015) used a two-step approach for inversion. First, they used the near-offset vertical and radial component along a single azimuth and inverted the data under an isotropic assumption to obtain an estimate of V_{P0}, V_{S0}, and density. This isotropic inversion used a population size of 800 and a maximum number of generations of 930. Additionally, the crossover and mutation probabilities (P_m and P_c) and their indices (η and κ) were kept exactly the same as those used by Padhi and Mallick (2014). Figures 7.18, 7.19, and 7.20 are the results of this step 1 inversion. Note from these Figs that although the isotropic near-offset inversion managed to get a reasonable estimate of V_{P0} and V_{S0}, it failed to obtain a reasonable estimate of the density.

Figure 7.21 shows the evolution of solutions over generations for the two component inversion results shown in Figs 7.18, 7.19, and 7.20 to demonstrate how the random models initially generated and shown as black dots that are widely distributed in the objective space slowly converge to the most optimal set of solutions over the generations.

Following isotropic inversion of the near offsets, Li and Mallick (2015) used the three-component (vertical, horizontal inline, and horizontal crossline) full offset data along two azimuths to invert for the density and other anisotropic parameters. In this inversion, the V_{P0} and V_{S0}, estimated from the near offset inversion (Figs. 7.18, 7.19, and 7.20) were used as constraints. Note that this inversion simultaneously optimized three-component data along two azimuths, i.e., for a total of six objectives. Figures 7.22, 7.23, 7.24, 7.25, 7.26, 7.27, 7.28, 7.29, and 7.30 show these inversion results. Note that using full offset data allows estimating density and other anisotropic parameters to a reasonable accuracy.

Both isotropic near offset and anisotropic full offset inversions were carried out in the τ-p domain. The anisotropic inversion however needed a population size of 4000 and a maximum number of generations of 5000. For the anisotropic inversion, Li and Mallick (2015) also found that varying P_c, P_m, η, and κ over generations provide better convergence instead of keeping them constant. The probability of crossover P_c was varied linearly with generation as $P_c = 0.7 - 0.1 \times \frac{t}{t_{max}}$, and the crossover distribution index η, was varied as $\eta = 1.0 + 19.0 \times \frac{t}{t_{max}}$. In these linear functions, t and t_{max} respectively denote the current generation number and the maximum number of generations. Note that based on the previous arguments, this choice of generation-dependent P_c and η, were deliberately chosen such that P_c decreases and η increases with generation such that the model space is widely sampled at the beginning and is slowly reduced over generations as the algorithm approaches the true solution. Following Deb and Agrawal (1999), to get a mutation effect of 1% perturbation in the solutions out of the entire population, the mutation index was varied as $\kappa = 100 + t$ and the probability of mutation was varied as $P_m = \frac{1}{n} + \frac{t}{t_{max}} \times \left(1 - \frac{1}{n}\right)$, where n is the total number of variables (model parameters). Note here that like the choice of the generation-dependent P_c and η, the choice

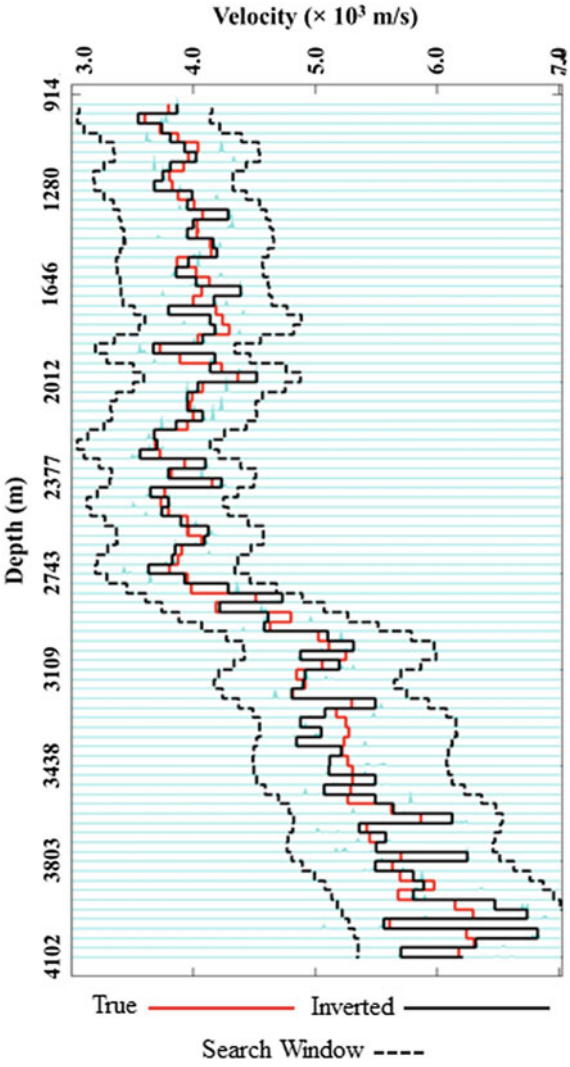


Fig. 7.18 Comparison between the true P-wave velocities with the inversion results and the search window used. Here, the near-offset prestack data were inverted using an isotropic assumption. Normalized PPD of the inverted results is also plotted in light blue (cyan). The width of each PPD curve helps to quantify the uncertainties associated with the estimates of each model parameter for each layer. Solutions with the highest likelihood were picked as best inversion results from these approximate PPD plots. The figure is reproduced from Li and Mallick (2015)

of the generation-dependent P_m and κ as above was also motivated by maintaining diversity in the population at early stage and slowly reducing it over time as the solutions near the global minimum. Another important modification made by Li and Mallick (2015) in the algorithm is the way of computing the crowding distance. In the standard implementation of NSGA II the crowding distance is computed as the normalized Euclidean distance in the objective space. Thus, for a given model in the objective space, its distance from all its neighbors is computed purely in the objective space and normalized to assign the model a value of its crowding distance. For problems where the number of model parameters to be solved for is

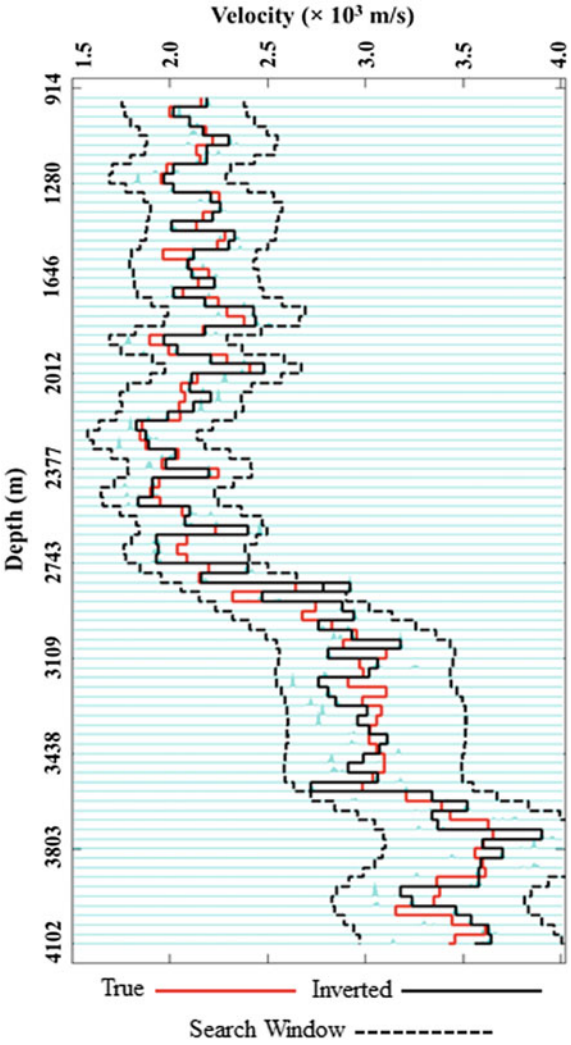


Fig. 7.19 Same as Fig. 7.18, but for the S-wave velocity. The figure is reproduced from Li and Mallick (2015)

much higher than the number of objectives, which is typically the case for most geophysical problems, Li and Mallick (2015) found that basing crowding distance only in the objective space does not necessarily guarantee that the actual models in the model space are diverse. Combining the normalized distance measured in the scaled objective space with that measured in the model space provided a better way to maintain diversity within the generations than using the objective space alone.

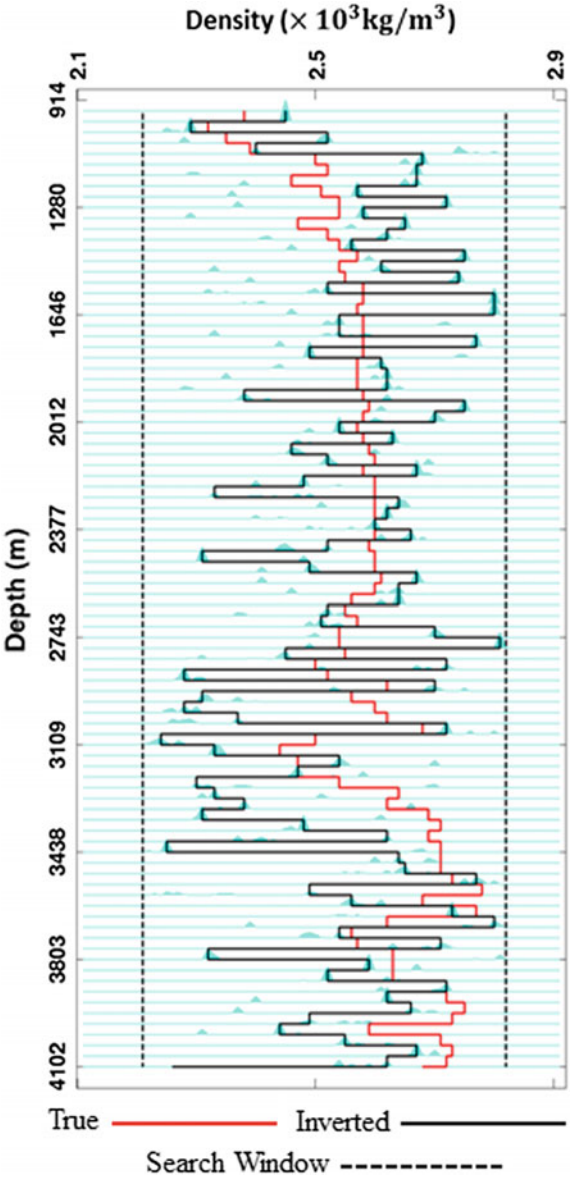


Fig. 7.20 Same as Fig. 7.18, but for the density. Note that applying an isotropic inversion failed to capture the density trend. The figure is reproduced from Li and Mallick (2015)

Recently, Li et al. (2016) used the methodology developed by Li and Mallick (2015) to invert wide-azimuth real vertical seismic profile (VSP) data. In this application, they used well-data to constrain V_{P0}, V_{S0}, and density and inverted for anisotropic subsurface properties. Figure 7.31 shows this inversion result. These results were verified to be correct by comparing them with the results obtained from the slowness polarization analysis developed by Grechka and Mateeva (2007).

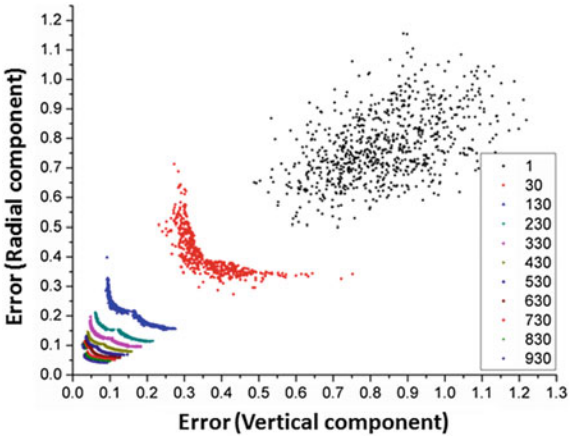


Fig. 7.21 Evolution of all solutions through generations under the defined error functions or objectives given as the misfit between synthetic data and real data measured as a cross-correlation. As shown in the legend, the dots in different colors represent different generations. Note how the solutions, widely distributed in the objective space, slowly converge to the most optimal set of solutions over generations. The figure is reproduced from Li and Mallick (2015)

7.4.3 The Future of Multi-objective Optimization in Geophysics

Multi-objective methods provide an elegant way of combining different data types. In above, we have restricted our examples to seismic inversions only. It is however possible to combine seismic with other geophysical data (gravity, magnetic, electromagnetic etc.), in joint inversion schemes using the multi-objective methods described here. We have noted above that there have been some recent interests in such joint inversions for reservoir description. Although there are a few attempts at such joint inversions, they tend to cast these inversions into a single-objective optimization (for example, Du and MacGregor 2010; Karaoulis et al. 2012). In the near future, we would expect to see these joint inversions cast as multi-objective optimizations.

Another potential application of multi-objective optimization could be in combining geophysical data with dynamic reservoir models in production history matching. Dynamic reservoir models are built via reservoir flow simulation and geomechanical modeling. Starting from the baseline reservoir model, flow simulation and geomechanical modeling are iteratively run over time to match the production data. Traditionally, the time-lapse geophysical (typically seismic) data are only used as additional constraints to these simulations (Huang et al. 1998 for example). These iterative methods use the single-objective, gradient-based approach as the optimization tool and suffer from many shortcomings, specifically for the dependence of the final model with the initial (baseline) model. Being

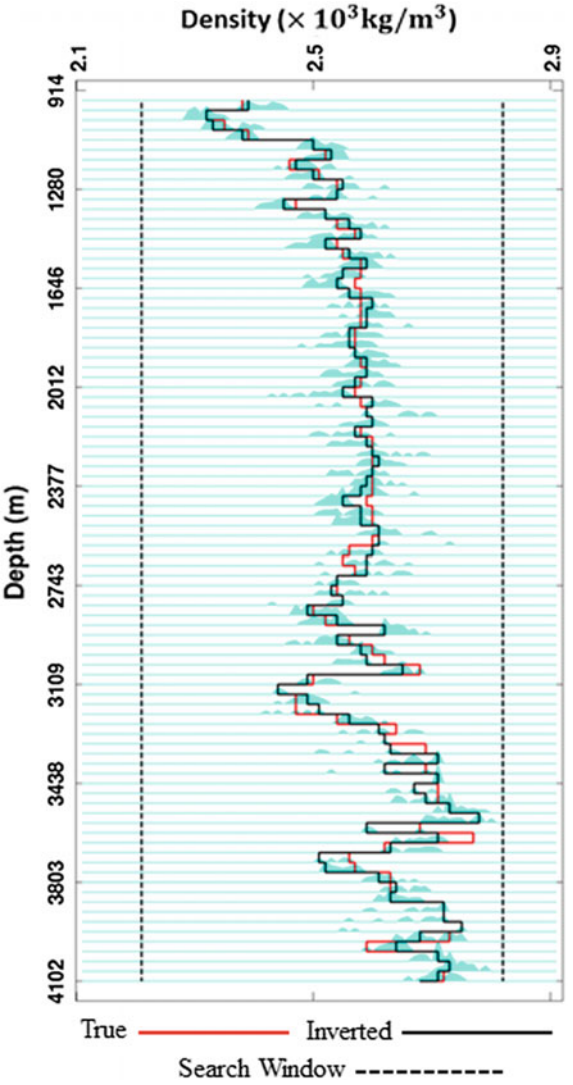


Fig. 7.22 Comparison between the true densities with the inversion results after 5000 generations. Here, the entire prestack data were inverted under an azimuthally anisotropic assumption. Normalized PPD of the inverted results is also plotted in light blue (cyan). Figure reproduced from Li and Mallick (2015)

linearized and gradient-based, they cannot correctly predict if the dynamic model is far from the baseline model. To circumvent this limitation, dynamic models are built under the assumption that only the reservoir is changed over the two repeated time-lapse surveys. The time-lapse seismic data used as constraint, are also processed such that they are identical to the baseline seismic data everywhere except at the reservoir zone so that they can satisfy the conditions explicitly imposed in the iterative flow simulation and geomechanical modeling runs. This condition, termed as "repeatability" in time-lapse seismology is seldom true. Production results in the in situ stress fields to change not only within the reservoir volumes but also in the

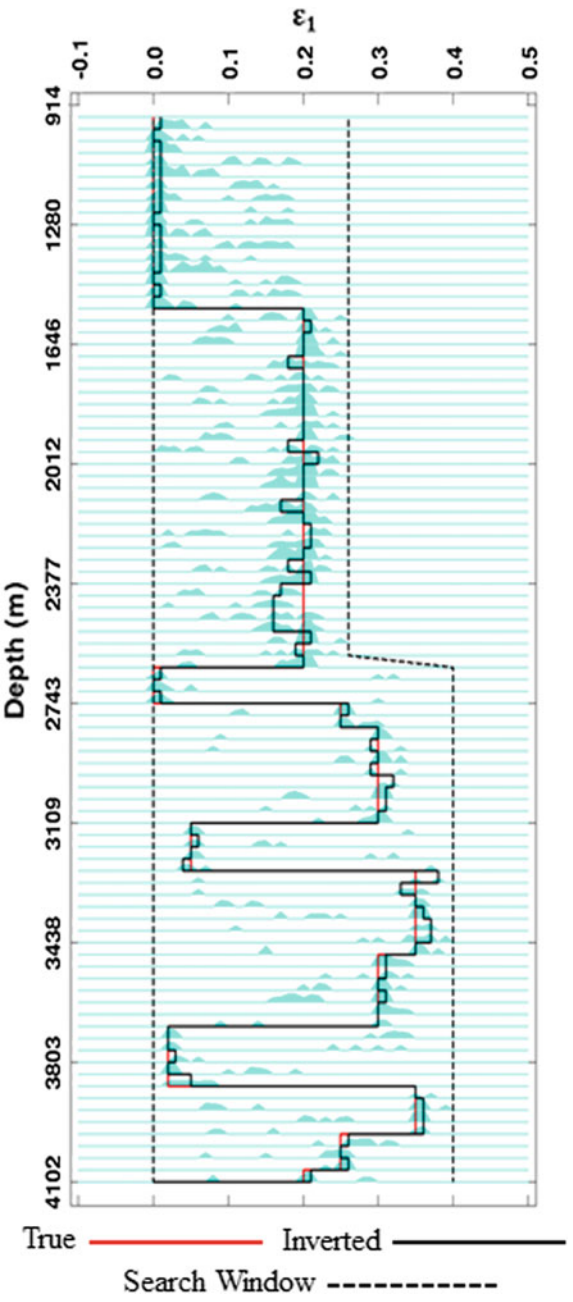


Fig. 7.23 Same as Fig. 7.22, but for ε_1. Figure reproduced from Li and Mallick (2015)

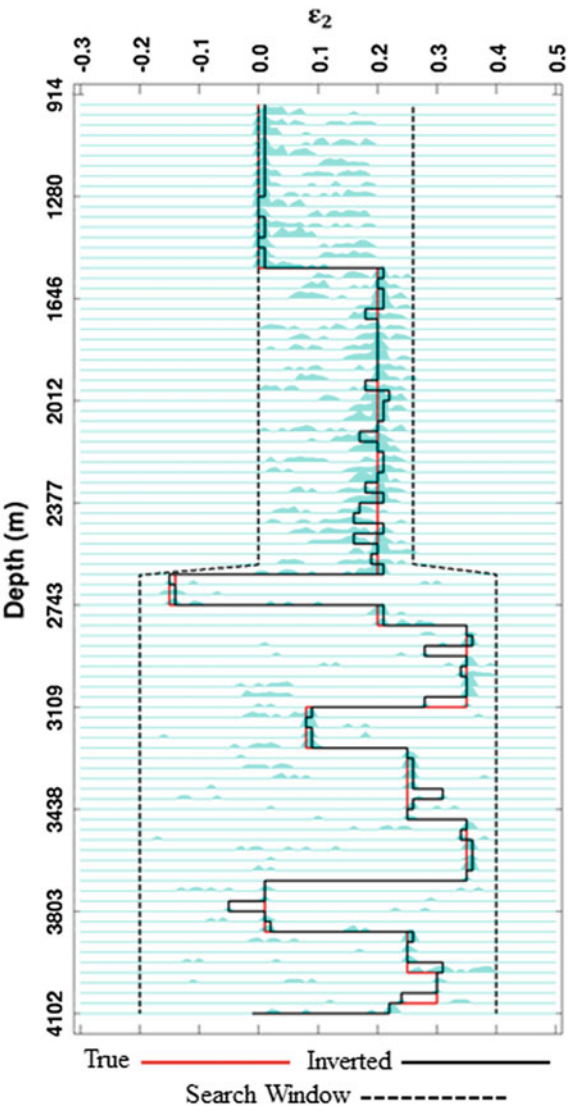


Fig. 7.24 Same as Fig. 7.22, but for ε_2. Figure reproduced from Li and Mallick (2015)

surrounding formations. Because, the stress field is directly related to anisotropy, the geophysical properties must also be different, and therefore the repeatability condition must be relaxed, which could be achieved if geophysical and engineering data are combined in a joint multi-scale/multi-physics, multi-objective optimization.

Figure 7.32 is one possible approach for integrating the geophysical component with the engineering component including the production data between two time-lapse geophysical surveys, and extending this to more time-lapse surveys is identical. As shown in Fig. 7.32, the baseline geophysical data, in conjunction with the geological and petrophysical information can be first inverted using a

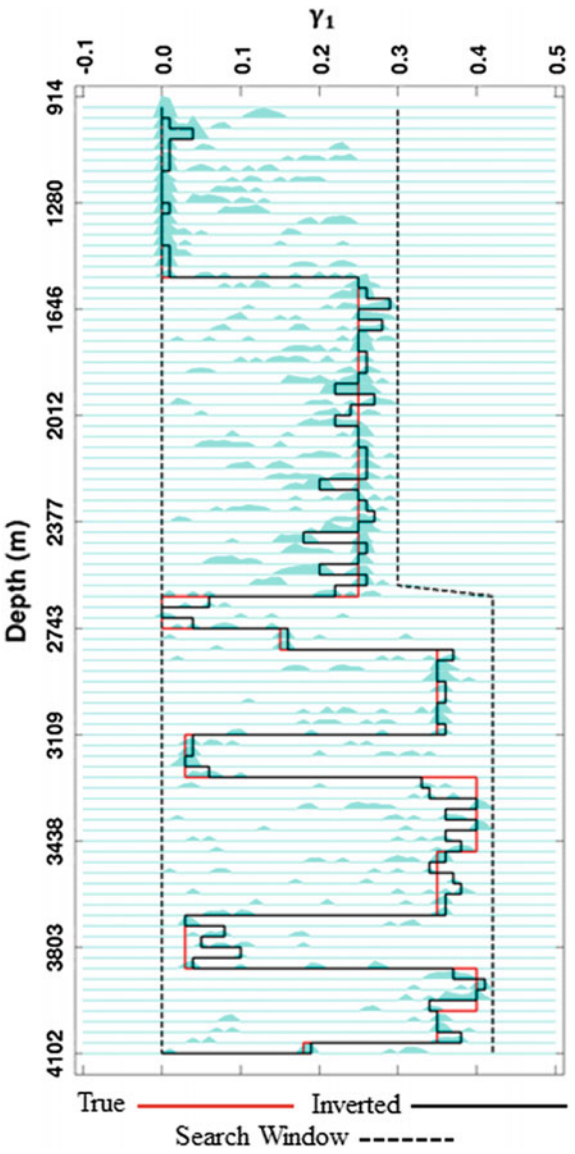


Fig. 7.25 Same as Fig. 7.22, but for γ_1. Figure reproduced from Li and Mallick (2015)

multi-objective approach to provide the baseline geophysical model. This geophysical inversion can then be followed by an anisotropic rock-physics inversion to map the inverted geophysical model into the reservoir model of porosity and fluid saturation. In fact, if these inversions are cast in a Bayesian framework (Lang and Grana 2015), they would also provide the uncertainties associated with each parameter estimate and therefore the reservoir model at this step will be the baseline reservoir model with uncertainty. This baseline reservoir model and the uncertainty

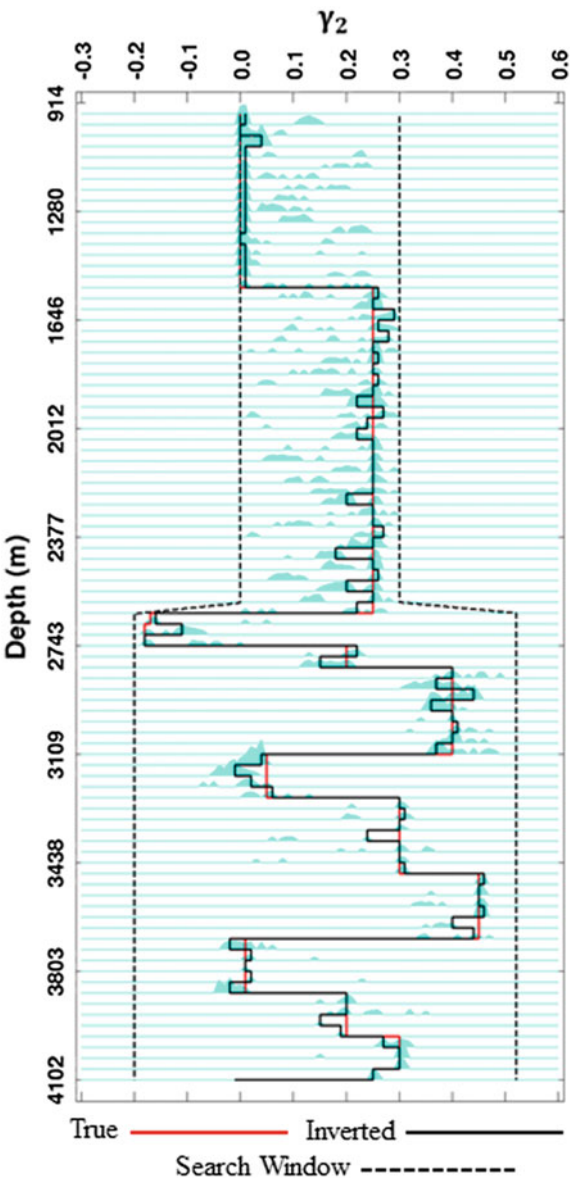


Fig. 7.26 Same as Fig. 7.22, but for γ_2. Figure reproduced from Li and Mallick (2015)

estimates can then be fed into the joint reservoir simulation/geomechanical/geophysical inversion methodology which will simultaneously match two sets of observations (1) production data at the monitoring wells and (2) the time-lapse geophysical data. Because flow-simulation and geomechanical modeling require reservoir properties while geophysical inversion requires geophysical properties, rock-physics must play a vital role in this joint inversion. Such a multi-objective

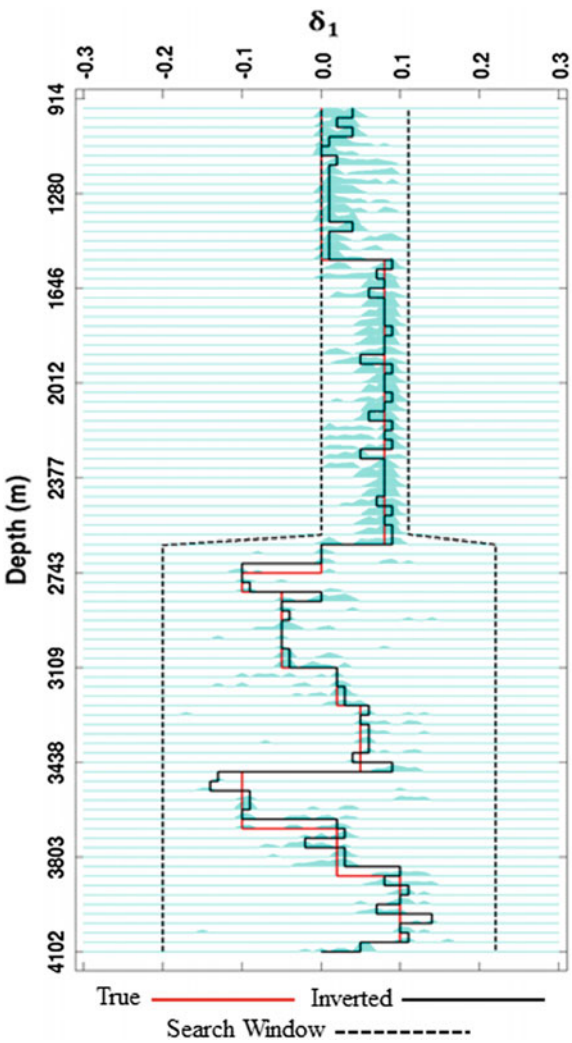


Fig. 7.27 Same as Fig. 7.22, but for δ_1. Figure reproduced from Li and Mallick (2015)

approach can lead to a new generation technology for reservoir characterization, monitoring, and history matching, which, in turn, would be a true success of multi-objective optimization.

Appendix of Chapter Seven: Pseudo Codes for GA

Basic steps of a simple genetic algorithm are as follows:

1. Define the search space: $m_i^{\min} \le m_i \le m_i^{\max}$ with resolution Δm_i. Note that a model vector is represented as $\mathbf{m} = [m_1 m_2 m_3 \ldots m_M]^T$.

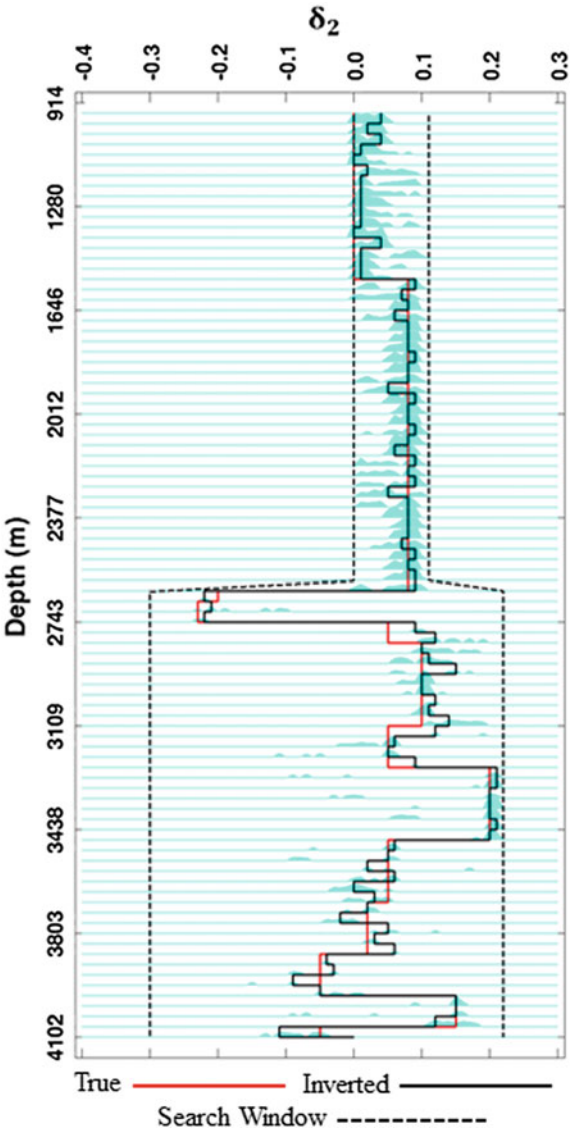


Fig. 7.28 Same as Fig. 7.22, but for δ_2. Figure reproduced from Li and Mallick (2015)

2. Given the search space, draw N number of model vectors $\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3, \ldots, \mathbf{m}_N$.
3. Represent each model in coded forms.
4. Evaluate the fitness function values for each of the model.
5. *Selection*: Draw N models with a probability that is proportional to fitness.
6. *Crossover*: For each pair of model from 5, select a crossover site and apply crossover with probability P_{cross}.

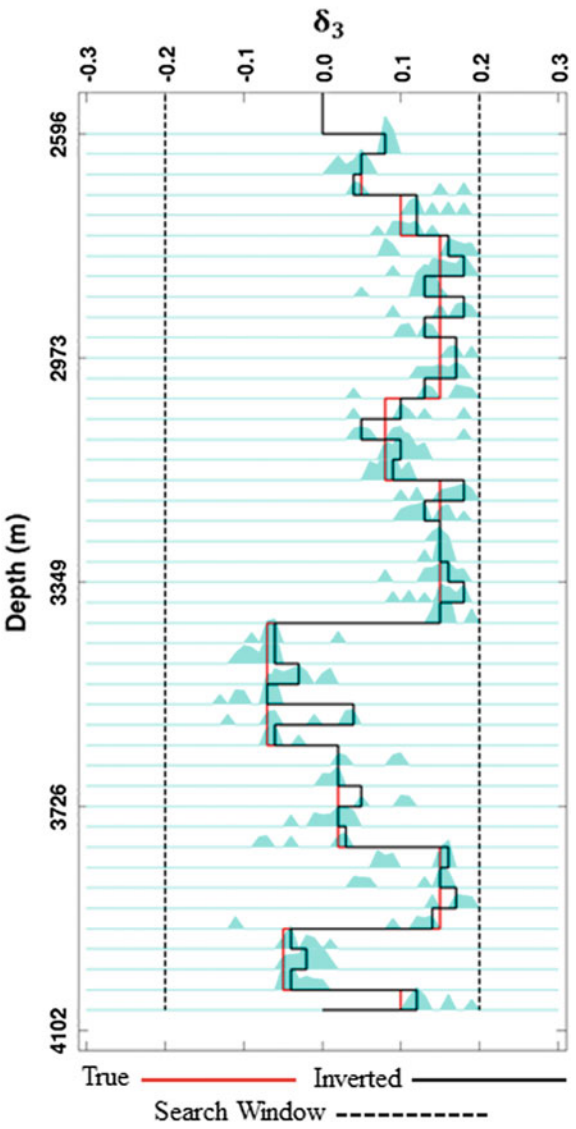


Fig. 7.29 Same as Fig. 7.22, but for δ_3. Figure reproduced from Li and Mallick (2015)

7. *Mutation*: Apply mutation on each of the models with P_{mu}.
8. Now we have N new models. Go to step 4 and repeat this process until the population becomes nearly homogeneous.

Pseudo Fortran codes for binary coding, cross over and mutation are given below.

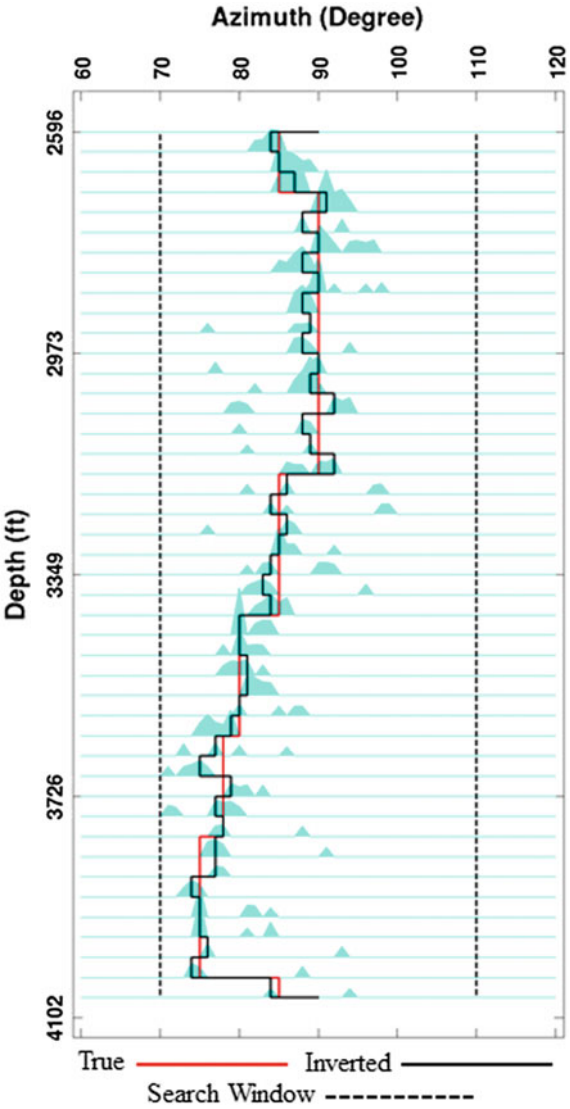


Fig. 7.30 Same as Fig. 7.22, but for the azimuthal angle of vertical symmetry planes. Figure reproduced from Li and Mallick (2015)

****

A pseudo-code for determination of the number of bits required to represent a model parameters.

Input: m_min, m_max and dm: minimum and maximum values of the model parameter, and resolution of the model parameter.

****

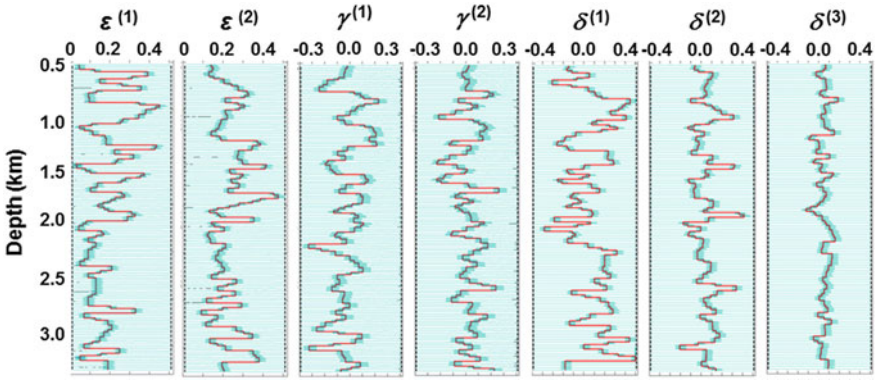


Fig. 7.31 Anisotropic inversion of wide-azimuth VSP data. The figure reproduced from Li et al. (2016)

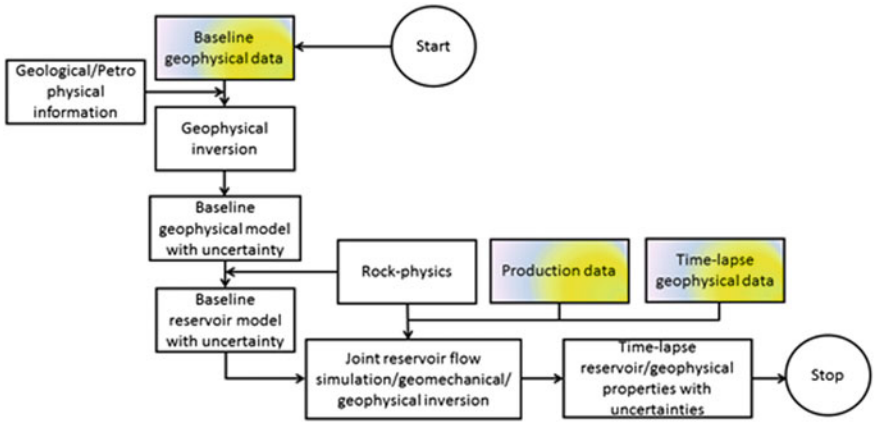


Fig. 7.32 A possible approach for combining time-lapse geophysical data with the engineering data in a multi-objective optimization scheme

$$\text{N\_bits} = (\text{m\_max} - \text{m\_min})/\text{dm}$$

Binary coding: a model parameter to be represented as a string of bits. Each bit can be coded as a logical variable (true or false)

Input: M = model parameter value

```
Temp = M - m_min
for j = 1, N_bits
db = dm*2**(j-1)
```

```
  test = int(temp/dm)
  if (test >=1) then
    bits(j) = true.
    temp = temp – db*dm
  end if
end
```

Crossover: Note that the crossover is done with the probability p_cross. First a crossover site is selected at random and then the bits are swapped between the pair of models to the right of the crossover site.

Bits1: string for model 1
Bits2: string for model 2
Cbits1: string for model 1 after crossover
Cbits2: string for model 2 after crossover

```
icross = nbits*ran2(irand)    ! Selecting a crossover site

for j = 1,icross
       Cbits1(j) = Bits1(j)
       Cbits2(j) = Bits2(j)
End for
for j = icross + 1,nbits
       Cbits1(j) = Bits2(j)
       Cbits2(j) = Bits1(j)
end for
```

Mutation: Note that mutation is done with probability p_mu

Bit: the bit that needs to be changed

Mutate: bit after mutation

```
Mutate = .not.Bit ! simply flip the bit
```

References

Backus, G. E., and Gilbert, F., 1967, Numerical applications of a formalism for geophysical inverse problems, Geophys. J. R. Astr. Soc., 13, 247–276.

Campbell-Stone, E., L. Shafer, S. Mallick, D. Mukherjee, S. Adhikari, and R. Lynds, 2012, Subsurface geomechanical analysis, comparison with prestack azimuthal AVO analysis, and implication for predicting subsurface geomechanical properties from 3-D seismic data: Soc. Petrol. Geophys., Expanded Abstracts, February 16–18, 2012, Hyderabad, India.

Deb, K. and R.B. Agrawal, 1995, Simulated binary crossover for continuous search space: Complex Syst., **9,** 115–148.

Deb, K., and S. Agrawal, 1999, A niched-penalty approach for constraint handling in genetic algorithms: Proceedings of the ICANNGA-99, Portoroz, Slovenia.

Deb, K., A. Pratap, S. Agarwal, and T. Meyarivan, 2002, A fast and elitist multi-objective genetic algorithm: NSGA-II: IEEE Transaction on Evolutionary Computation, **6(2)**, 181–197.

Dosso, S. E., J. Dettmer, G. Steininger, and C. W. Holland, 2014, Efficient trans-dimensional Bayesian inversion for geoacoustic profile estimation: Inverse Problems, 30, 114018, https://doi.org/10.1088/0266-5611/30/11/114018.

Du, Z. and L.M. MacGregor, 2010, Reservoir characterization from joint inversion of marine CSEM and seismic AVA data using Genetic Algorithms: a case study based on the Luva gas field: SEG Technical Program Expanded Abstracts, **29,** 737–741.

Giagkiozis, I., R.C. Purshouse, and P.J. Fleming, 2015, An overview of population-based algorithms for multi-objective optimisation: International Journal of Systems Science, **46(9)**, 1572–1599.

Gray, D., P. Anderson, J. Logel, F. Delbecq, D. Schmidt, and R. Schmid, 2012, Estimation of stress and geomechanical properties from 3D seismic data: First Break, 30, 59–68.

Grechka V. and A. Mateeva, 2007, Inversion of P-wave VSP data for local anisotropy: Theory and case study: Geophysics, **72**, 69–79.

Goldberg, D.E., 1989, Genetic algorithms in search, optimization and machine learning: Addison-Wesley Pub. Co., Inc.

Hajela, P., and C.-Y. Lin, 1992, Genetic search strategies in multicriterion optimal design: Structural Optimisation, **4**, 99–107.

Heyburn, R. and B. Fox, 2010, Multi-objective analysis of body and surface waves from the Market Rasen (UK) earthquake: Geophysical Journal International, **181**, 532–544.

Hong, T., and M. K. Sen, 2009, A new MCMC algorithm for seismic waveform inversion and corresponding uncertainty analysis, Geophysical Journal International, 177(1), 14–32.

Holland, J. H., 1975, Adaptation in Natural and Artificial Systems, Univ. of Michigan Press, Ann Arbor, Michigan.

Karaoulis, M., A. Revil, J. Zhang, and D.D. Werkema, 2012, Time-lapse joint inversion of crosswell DC resistivity and seismic data: a numerical investigation:Geophysics, **77**, D141–D157.

Kennett, B.L.N., 1983, Seismic wave propagation in stratified media, Cambridge University Press.

Kirkpatrick, S., Gelatt, C. D., Jr., and Vecchi, M. P., 1983, Optimization by simulated annealing, Science, 220, 671–680.

Konak, W., C. David, and E.S. Alice, 2006, Multi-objective optimization using genetic algorithms: A tutorial: Reliability Engineering and System Safety, **91**, 992–1007.

Kozlovskaya, E., L. Vecsey, J. Plomerova, and T. Raita, 2007, Joint inversion of multiple data types with the use of multiobjective optimization: problem formulation and application to the seismic anisotropy investigations: Geophysical Journal International, **171(2)**, 761–779.

Lang, X, and Grana, D., 2015, Bayesian rock physics inversion of acoustic and electrical properties for rock-fluid property estimation: SEG Technical Program Expanded Abstracts, 2931–2935.

Li, T., and S. Mallick, 2015, Multicomponent, multi-azimuth pre-stack seismic waveform inversion for azimuthally anisotropic media using a parallel and computationally efficient non-dominated sorting genetic algorithm: Geophysical Journal International, **200**, 1136–1154.

Li, T., S. Mallick, N. Tamimi, and T. Davis, 2016, Inversion of wide-azimuth multicomponent vertical seismic profile data for anisotropic subsurface properties: SEG Technical Program Expanded Abstracts, 1252–1257.

Mallick, S., 1995, Model-based inversion of amplitude-variations-with-offset data using a genetic algorithm: Geophysics, **60**, 939–954.

Mallick, S., 1999, Some practical aspects of prestack waveform inversion using a genetic algorithm: an example from the east Texas Woodbine gas sand: Geophysics, **64**, 326–336.

Malinverno, A., 2002, Parsimonious Bayesian Markov chain Monte Carlo inversion in a nonlinear geophysical problem: Geophysical Journal International, 151, 675–688, https://doi.org/10.1046/j.1365-246x.2002.01847.x.

Menke, W., 1984, Geophysical Data Analysis: Discrete Inverse Theory, Academic Press.

Miettinen, K., 1999, Nonlinear Multiobjective Optimization: Kluwer Academic Publishers.

Mosegaard, K. and Tarantola, A., 1995, Monte Carlo sampling of solutions to inverse problems: Journal of Geophysical Research, **100(B7)**, 12, 431–12, 447.

Padhi A., and S. Mallick, 2013, Accurate estimation of density from the inversion of multicomponent prestack seismic waveform data using a nondominated sorting genetic algorithm: The Leading Edge, **32(1)**, 94–98.

Padhi, A., and S. Mallick, 2014, Multicomponent pre-stack seismic waveform inversion in transversely isotropic media using a non-dominated sorting genetic algorithm: Geophysical Journal International, **196**, 1600–1618.

Padhi, A., S. Mallick, W. Fortin, W.S. Holbrook, and T.M. Blacic, 2015, 2-D ocean temperature and salinity images from pre-stack seismic waveform inversion methods: an example from the South China Sea: Geophysical Journal International, **202**, 800–810.

Press, W. H., Flannerry, B. P., Teukolsky, S.A., and Vetterling, W. T., 1989, Numerical Recipes: The Art of Scientific Computing, Cambridge University Press.

Ritzel, B.J, J.W. Eheart, and S. Ranjithan, 1994, Using genetic algorithms to solve a multiple objective groundwater pollution containment problem: Water Resources Research, **30(5)**, 1589–1603.

Sambridge, M.S., and Drijkoningen, G.G., 1992, Genetic algorithms in seismic waveform inversion, Geophys. J. Int., 109, 323–342.

Scales, J.A., Smith, M.L., and Fisher, T.L., 1992, Global optimization methods for highly nonlinear inverse problems, *J. Comp. Phys.*, *103*, 258–268.

Schoenberg, M., 1983, Reflection of elastic waves from periodically stratified media with interfacial slip: Geophysical Prospecting, **31**, 265–292.

Schoenberg, M., and J. Douma, 1988, Elastic wave propagation in media with parallel fractures and aligned cracks: Geophysical Prospecting, **36**, 571–590.

Smith, M., Scales, J., and Fisher, T., 1992, Global search and genetic algorithms, The Leading Edge of Exploration, 11(1), 22–26.

Sen, M.K., and P.L. Stoffa, 2013. Global optimization methods in geophysical inversion, Cambridge University Press.

Sen, M. K., and Stoffa, P. L., 1991, Nonlinear one-dimensional seismic waveform inversion using simulated annealing, *Geophysics*, ***56***, 1624–1638.

Sen, M. K., and Stoffa, P. L., 1992, Rapid sampling of model space using genetic algorithms: Examples from seismic waveform inversion, *Geophys. J. Int.*, ***108***, 281–292.

Sen, M.K., and P.L. Stoffa, P.L., 1996, Bayesian inference, Gibbs' sampler and uncertainty estimation in geophysical inversion: Geophysical Prospecting, **44,** 313–350.

Sen, M. K., and Biswas, R., 2017, Trans-dimensional seismic inversion using the reversible jump Hamiltonian Monte Carlo Algorithm, Geophysics, 82(3), R119–R134.

Silverman, B.W., 1986, Density estimation for statistics and data analysis: Chapman and Hall.

Singh, V.P., B. Duquet, M. Léger, and M. Schoenauer, 2008, Automatic wave-equation migration velocity inversion using multiobjective evolutionary algorithm: Geophysics, **73**, VE61–VE73.

Spall, J.E., 2000, Adaptive stochastic approximation by the simultaneous perturbation method, IEEE Transactions On Automatic Control, Vol. 45, No. 10, 1839-1853.

Stoffa, P.L, and M.K. Sen, 1991, Nonlinear multiparameter optimizations using genetic algorithms: Inversion of plane-wave seismograms: Geophysics, **56**, 1794–1810.

Tang, L., and X. Wang, 2013, A hybrid multiobjective evolutionary algorithm for multiobjective optimization problems: IEEE Transactions on Evolutionary Computation, **17(1)**, 20–46.

Tarantola, A., 2000, Inverse Problem Theory, Methods of Data Fitting and Model Parameter Estimation, SIAM publications.

Thomsen, L., 1986. Weak elastic anisotropy: Geophysics, **51**, 1954–1966.

Thomsen, L., 1988, Reflection seismology in azimuthally anisotropic media: Geophysics, **53**, 304–313.

Tsvankin, I., 1997, Anisotropic parameters and P-wave velocity for orthorhombic media: Geophysics, **62**, 1292–1309.

Whitley, D., 1991, The genetic algorithm and selection pressure: Why rank-based allocation of reproductive trials is best, In J. D. Schaffer (Ed.), Proceedings of the Third International Conference on Genetic Algorithms, 116–123.

Zitzler, E., M. Laumanns, and L. Thiele, 2001, SPEA2: Improving the strength Pareto evolutionary algorithm: Comput. Eng. Networks Lab., Swiss Fed. Inst. Technol., Zurich, Switzerland, Tech. Rep. 103.

Zoback, M., 2010, Reservoir Geomechanics: Cambridge University Press.