UNIVERSITY OF STRATHCLYDE

DEPARTMENT OF BIOENGINEERING

MATHEMATICAL MODELLING OF BLOOD GLUCOSE FOR SHORT-TERM DIABETES THERAPY (USING ARTIFICIAL NEURAL NETWORKS)

BY:

SMRITI RAMNARAYAN VISWANATH

Email id: smriti.viswanath@strath.ac.uk

smritirv@gmail.com

This thesis is submitted in fulfilment of the requirements for the degree of Masters in Bioengineering.

August 2012.

DECLARATION OF AUTHORS RIGHTS

Signed:

Date:

ACKNOWLEDGEMENTS

I would like to thank my supervisor Prof. William Sandham for his continuous support and guidance throughout this project. I would like to thank Gavin Robertson for his help in rectifying my code. I would like to thank the entire staff of the Bioengineering Unit and the University of Strathclyde for the infrastructure and making this project possible.

I would like to thank Shruti Vishwanath for her valuable inputs in proof reading the thesis. Also I thank my family and friends for their blessings and support.

ABSTRACT

Diabetes is a major global health issue. Proper control of the blood glucose level (BGL) is very important. Insulin type, dose, stress levels, diet, exercise, height to weight ratio, metabolism rate, etc. all affect the BGL. Sometimes, inconsistency in measurement of the BGL values by the patient or the inability to interpret the data obtained from the system can lead to problems in the management of diabetes. It is therefore necessary to design a system which can assist patients in managing their diabetes.

The aim of this project was to design an improved artificial neural network (ANN) which will predict the short-term BGL and improve diet, exercise and insulin regimens. It should also enable training to be continuously updated with each new dataset. An important advantage of using ANNs is that they do not require a comprehensive overview of the problem, but are trained to recognise the patterns in the input dataset which are stored effectively.

In this project, an ANN was modelled using the Neural Network Toolbox^{TM} in the MATLAB^{TM} software. The automated insulin dosage advisor (AIDA) is a simple model of the interaction of glucose and insulin in the body and is mainly intended for Type-1 diabetes. The data obtained from the AIDA software was used to train the ANN in order to predict the BGL. The R2011b version of MATLAB^{TM} was used in this project. Functions such as 'adaptwb' and 'trainlm' were used to update the weights while training.

The architecture used was a slightly modified version of the Elman architecture and it was found that the best results obtained were at 82 epochs. However, the effects of the individual inputs given to the network were still unknown.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ANN- Artificial Neural Network

AIDA- Automated Insulin Dosage Advisor

BGL- Blood Glucose Level

DI- Diabetes Insipidus

GDM- Gestational Diabetes Mellitus

IDDM - Type-1 (insulin dependent) diabetes mellitus

LM - Levenberg-Marquardt

NIDDM- Type-2 (non-insulin dependent) diabetes mellitus

SOM – Self-Organizing Map

CHAPTER 1: INTRODUCTION

1.1 INTRODUCTION

Diabetes is a chronic disease and a major global health issue in today’s times. Researchers have found that in the United Kingdom alone, around 2.9 million people have been diagnosed with diabetes and around 8.5 million people may have the condition but are unaware of it [5]. It is also projected that in 2030, around 366 million people worldwide will have the disease, which is an estimated 4.4% of the population [14]. There is an increasing trend in the development of this condition and hence, significant research is being carried out for improving management of diabetes.

The complications associated with the condition may be short- or long-term. The short-term effects are hyper- or hypo-glycaemia, whereas the more serious damaging long-term effects are cardiovascular diseases, retinopathy, neuropathy, nephropathy and peripheral tissue damage [6].

Insulin type, dose, stress levels, diet, exercise, height-to-weight ratio, metabolism rate, etc. all affect the BGL. Sometimes, inconsistency in the measurement of BGL values by the patient or inability to interpret the data obtained from the system can lead to problems in diabetes management.

Continuous blood glucose monitors (CGMs) and infusion pumps (IPs) reduce the effort needed by the patient for blood glucose management. However, these are open loop systems and it is ideal for a system to be closed loop as it is preferred to have minimum intervention from the patient for accurate management of diabetes. Thus, there is a need to design an efficient closed loop system which will assist patients in effective diabetes management.

The main advantages to diabetic patients for such a system are that it would be patient-centric and they could have an improvised therapy. Precise BGL predictions would not only enhance the patient’s understanding of their condition, but also prove to be useful for learning purposes at various medical centres [14].

1.2 AIM OF THE PROJECT

The aim of this project is mainly two-fold:

- To design an improved artificial neural network (ANN) paradigm for BGL prediction and improve diet, exercise and insulin regimens. Also to enable training to be updated with each new dataset.
- ANN training and evaluation with the use of the Automated Insulin Dosage Advisor (AIDA) BGL simulator.

The ANN is modelled using the Neural Network Toolbox™ present in the MATLAB™ software. AIDA is a simple model of the interaction of glucose and insulin in the body and is mainly intended for Type-1 diabetes. BGL training and prediction is carried out using the data obtained from the AIDA software. However, the neural network may also accommodate Type-2 diabetes.

1.3 OUTLINE OF THE THESIS

The various topics included in this report are as follows:

Chapter 1 outlines the project with its aims and objectives.

Chapter 2 introduces diabetes, its types and associated complications caused along with few treatment measures.

Chapter 3 describes ANNs including types of architectures, learning processes, general applications and its use in BGL prediction, advantages and disadvantages.

Chapter 4 gives a short summary of AIDA, its online/downloadable versions, the benefits and limitations.

Chapter 5 explains the experimental methodology; mainly collecting the database and training the network.

Chapter 6 includes the results obtained from the experiment and the inference obtained.

Chapter 7 concludes the project and projects a probable way forward for the scope of this project.

The thesis also comprises the references and an appendix which mainly includes a sample of the MATLAB code and the importance of the functions that were used for writing the code of this project.

CHAPTER 2: DIABETES

2.1 DIABETES – AN INTRODUCTION

Diabetes mellitus is a condition in the body in which the sugars accumulate in the blood and urine as a result of faulty glucose metabolism [6]. It affects the protein, carbohydrate and fat metabolism in the body, and mainly occurs if (a) the pancreas is not producing any or sufficient insulin for the blood glucose to enter the body cells, or (b) the uptake of insulin is not carried out efficiently by the cells [5].

Insulin is a hormone which is produced by the beta cells in the pancreas. It enables the target cells to obtain the required amount of glucose which is used as a fuel for energy to carry out various activities. Insulin also stimulates the formation and storage of lipids and glycogen. Glucagon is also another important hormone, produced by the alpha cells in the pancreas and mainly promotes the synthesis of glucose and the breakdown of glycogen in the liver [6].

The normal range of BGL is 4.0–8.0 mmol/l. For a non-diabetic person, the range of BGL is between 4.0–5.9 mmol/l before meals and is under 7.8 mmol/l after meals. For normal children the range is between 4.0–8.0 mmol/l before meals and is under 10.0 mmol/l after meals [16].

2.2 GLUCOSE-INSULIN MECHANISM

Every cell in the body requires a constant supply of glucose for energy. Since glucose is transported around the body to the cells, it is essential to regulate the BGL in the body. The organs that are primarily involved in this regulation are the liver, pancreas and organs of the digestive system.

If the blood sugar level rises (usually occurring after meals), the homeostasis is disturbed and the beta cells secrete insulin. The insulin has an effect on various organs of the body, increasing cell permeability to glucose and increasing enzyme activity in the cells, allowing the glucose to be absorbed and stored. The liver and muscle tissues convert this glucose to glycogen and then store it, which reduces the BGL and restores homeostasis. In addition, the breakdown of fats in fat cells is inhibited, so that the glucose will be used preferentially for energy.

If the blood sugar level falls, again the homeostasis is disturbed and the alpha cells secrete glucagon. Glucagon effects are antagonistic to insulin. Glucagon increases the conversion of stored glycogen into glucose in the liver and muscles thereby increasing the blood sugar levels. Glucagon also increases the uptake of amino acids and glycerol into the liver so that more glucose can be synthesized; thus maintaining the homeostasis [6].

The problems related with diabetes occur mainly when this homeostasis of glucose and insulin becomes disrupted.

2.3 TYPES OF DIABETES

There are various different categories of diabetes. The most well-known and common types are Type-1 and Type-2 diabetes mellitus.

2.3.1 Type-1 Diabetes Mellitus (IDDM)

Type-1 (insulin dependent) diabetes mellitus (IDDM) is the condition where there is inadequate insulin production due to destruction of the beta cells of the pancreas. Since insulin is a vital hormone for the use of glucose by the body cells, lack of the hormone causes an increase in the build-up of glucose in the body. The real cause as to how the insulin-producing cells get destroyed is still under research. However, the most likely cause is the abnormal reaction to the cells which may be due to a virus or infection. People having Type-1 diabetes generally require multiple injections of insulin every day or continuous infusion through an insulin pump or other such devices [6]. The target range of BGL values for a Type-1 diabetic patient is between 4.0–7.0 mmol/l before meals and under 8.5 mmol/l after meals [16]. Type-1 diabetes generally accounts for 5-10 % of the total number of people suffering from this condition but usually occurs before the age of 40 or occurs in childhood [6].

The most common symptoms of IDDM includes excess thirst, frequent urination, unusual weight reduction, fatigue, nausea, poor healing of cuts or sores, dry mouth, blurred vision, etc. which may develop very quickly sometimes over a few weeks or in certain cases, a few days as well [17].

2.3.2 Type -2 Diabetes Mellitus (NIDDM)

Type-2 (non- insulin dependent) diabetes mellitus (NIDDM) is a type of diabetes this is prevalent among the majority of population having this condition (85–95%). In this type of diabetes, the body can still make sufficient insulin; however, the insulin is insufficient or the produced insulin is inefficiently used in the body. Generally, this type of diabetes is treated with a healthy diet, increased exercise, and medication and in a few cases, insulin injections [6]. The target range of BGL values for a Type-2 diabetes patient is between 4.0–7.0 mmol/l before meals and under 8.5 mmol/l after meals [16]. Usually this condition occurs in people over the age of 40, but it often may appear from around 25 years of age. An increasing trend has been observed in it being more common among children and young people of all ethnicities.

NIIDM often does not cause any symptoms but it may be identified at the time of a routine screening. Other symptoms are similar to those that are prevalent in Type-1 diabetes mellitus [14].

2.3.3 Gestational Diabetes Mellitus (GDM)

Gestational diabetes mellitus (GDM) is a condition that usually arises at the time of the second or third trimester during pregnancy. It mainly occurs because during pregnancy, the body is unable to produce enough insulin to meet the requirements. It may also be prevalent among people during the first trimester; however, in these women, the condition may have existed before pregnancy. Similar to Type-2 diabetes, gestational diabetes may only require making changes in the diet, where in few cases, medications and injections may also be needed. Malformations to the baby may occur in this type of diabetes, but the later the diabetes sets in during pregnancy; the lesser are the chances of malformations in the baby. The Oral Glucose Tolerance Test (OGTT) is used to detect GDM which involves a blood test before and after breakfast [16].

2.3.4 Diabetes Insipidus (DI)

Diabetes Insipidus (DI) is a form of diabetes mainly resulting from a temporary disorder in the endocrine system - mainly the pituitary gland. It is not very common and occurs chiefly due to the lack of production of the antidiuretic hormone (ADH). Due to less ADH, the kidneys are unable to conserve water, which causes several complications. DI caused due to lack of ADH is called central diabetes insipidus and

DI caused if the kidneys fail to respond to the ADH is called nephrogenic diabetes. The main cause of central DI is due to the damage to the hypothalamus which may be due to head injury, infection, surgery, tumour, or loss of blood supply to the gland. It may also be genetic in certain cases. Nephrogenic DI could be due to certain specific drugs, kidney diseases or abnormally high levels of calcium in the body. DI can be treated with various medications which help monitor and keep the diabetes in check [1].

2.3.5 Type-3 diabetes

Type-3 diabetes is still very new and under research but has been discovered by scientists in the United States. Other names for this are latent autoimmune diabetes in adults (LADA) or 1.5 diabetes. This new type has been discovered following the latest findings of insulin being produced in the brain as well. It is believed to not affect the blood sugar and is only related to the levels of the brain insulin and this condition is found in adults. However, this form of diabetes was only discovered very recently and the complete symptoms and forms of treatment are still under research; but oral medications might be the effective treatment in this case [16].

2.4 TREATMENT METHODS FOR DIABETES MELLITUS

People with diabetes are highly vulnerable to several complications. Due to the varied nature of this condition (diabetes), it is difficult to give acute treatment and long-term management requires a lot of effort.

One of the treatment methods is prevention with the probable use of immunosuppressant; but it is found to have several side effects in the long-term, increasing the complications. Some of the immunosuppressive drugs used were azathioprine and cyclosporine. The common side effect of immunosuppressive drugs is post-transplant diabetes mellitus (PTDM) [12].

Pancreas transplant is a highly effective procedure, however, lack of donors and the cost of the operation and treatment incurred has increased the demand for stem cells in this field as well. Stem cell research in the field of diabetes has been considered to be the chief potential to provide the ultimate unlimited source of cells for research, to replace the missing and damaged insulin-producing cells or the other cells damaged by diabetes. Stem cell replacement in the case of type-1 diabetes is not very simple. Various scientists and doctors are attempting to cure the disease through the injection of the pancreatic islet cells. There has been a very small amount which is effective primarily as there is a need to supress the immune system [16].

Edmonton protocol was initially used for the transplantation of the islet cells for type-1 diabetes patients. It was found that among the people who received the transplant, 85% were insulin independent for many years. However, the main disadvantages were that immunosuppressant had to be taken for life, enough cadaver pancreas were not available to meet the demand and an alternative source for insulin producing cells was required [4].

Most of the types of diabetes (mainly Type-2 diabetes mellitus) are kept within control by oral medications and insulin regimes. Also the BGLs can be monitored by usage of various glucose meters which provide accuracy in the measurement such as AccuChek®.

Some of the ways to help the management of Type-1 diabetes mellitus apart from the ones mentioned earlier are:

- Insulin injections or insulin pump
- Balanced diet
- Exercise
- Regular monitoring of BGL’s.

CHAPTER 3: ARTIFICIAL NEURAL NETWORKS

3.1 INTRODUCTION

An artificial neural network (ANN) is a modelling technique of the human brain to perform a certain task or function of interest. Over the past few decades, their usage has been increasing owing to their varied range of system applications due to their strong pattern recognition capabilities. The principal advantage of ANNs is that the system is non-linear and does not require a comprehensive idea of the problem. Some of its other benefits include its compatibility with Very Large Scale Integration (VLSI) and Field Programmable Gate Array (FPGA) technologies, its adaptivity, fault tolerance, uniformity in analysis and design, evidential response, and proper input-output mapping [7]. A neural network can be defined as follows [7]:

"A neural network is a massively parallel distributed processor made up of simple processing units that has a natural propensity for storing experiential knowledge and making it available for use. It resembles the brain in two respects:

- *Knowledge is acquired by the network from its environment through a learning process*
- *Interneuron connection strengths, known as synaptic weights, are used to store the acquired knowledge"*

An ANN develops its computational capability from its parallel structure and its unique ability to generalise. Generalization of the network provides a realistic output for unknown inputs. They have to be trained with many sets of patterns displaying the distinctive characteristics of the system, after which the ANN generalises. The input is then characterised by a weighting element and then given to the layers. A weight matrix is where the knowledge of the training is stored in the ANN. They can also adapt to new data through dynamic learning, on a continuous basis [14].

3.2 THE NEURON

Neurons are the basis for designing the family of neural networks and are the information-processing unit in neural networks [7]. Figure 3.1 shows a standard model of the neuron that consists of inputs with its own weighting element, a summing junction, an activation function and the output of the network. They are modelled compared to the biological neuron where the dendrites act as inputs, the synapses as the weights, and the axon as the output transmission path. It is very important to write the weights in the appropriate order since they have explicit significance. The summing junction is used to add the weighted inputs. The importance of the bias is to control the response of the neuron to an input. If the bias is less than the summation value, there will be no output from the neuron, but if the value of the bias exceeds the summation value, there will be an output which is provided to the activation function.

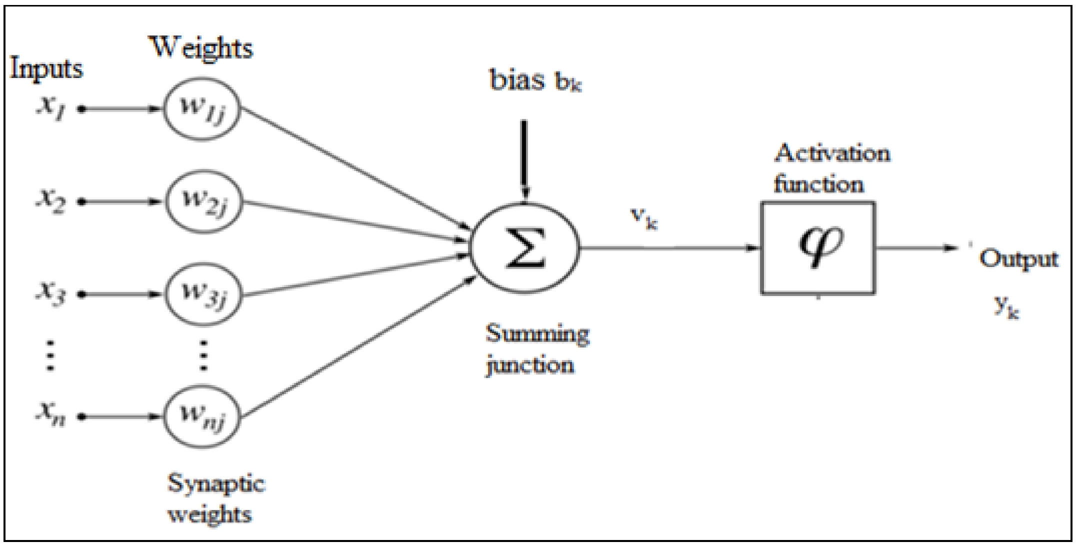


Figure 3.1: Model of a neuron.

The activation function limits the value of the output of the neuron and confines it to a particular range. There are mainly two types which are used: threshold (may be known as Heaviside function) and sigmoid functions. The sigmoid function does not have a sharp edge and it is strictly an increasing function whereas the threshold function is defined by the saturation of maximum or minimum values. Also, the sigmoid functions are differentiable everywhere i.e. they do not have any discontinuities, which is prevalent in the threshold function. Purelin, hardlim, tansig and logsig are some of the sigmoid functions which are used in several applications. Tansig is most extensively used in the hidden layer and the output is generally a linear function [7].

The threshold function is as shown in the Figure 3.2. If the value of *n* is greater than a certain threshold (in this case it is 0), then the function takes the value of 1 else it is 0.

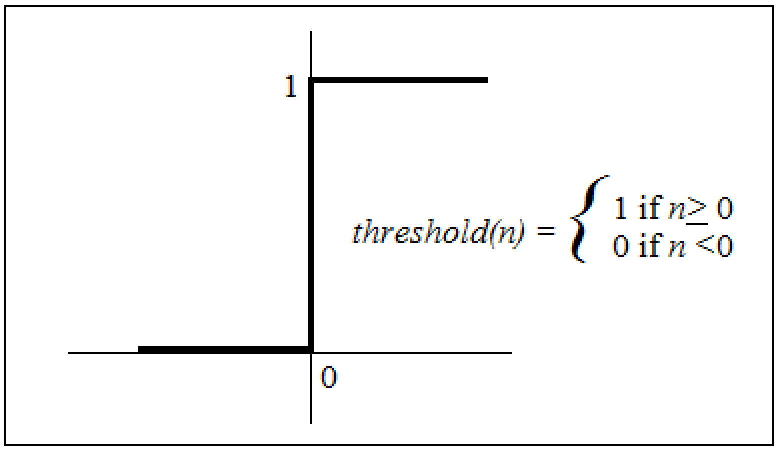


Figure 3.2: Threshold function.

Logsig is a sigmoid transfer function which limits the output range between 0 and 1 for the inputs which range from negative to positive infinity [9].

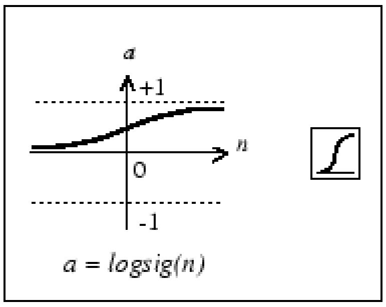


Figure 3.3: Logsig function.

Tansig as mentioned earlier, is widely used in neural networks mainly because of the efficiency in its use in pattern recognition glitches [9].

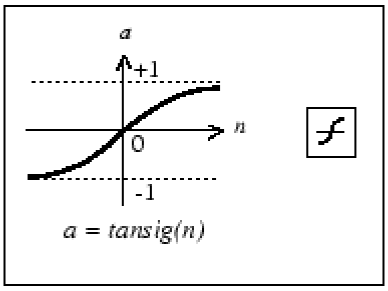


Figure 3.4: Tansig function.

Linear activation functions are generally used for output functions and generally to solve fitting problems. Purelin is a type of linear activation function which can be observed in the Figure 3.5.

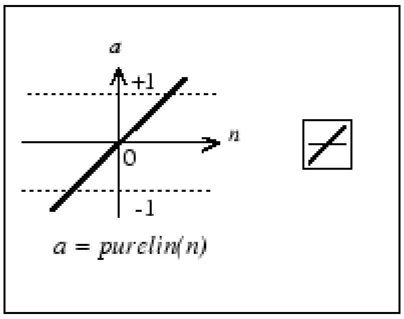


Figure 3.5: Purelin function.

3.3 ANN ARCHITECTURES

There may be a number of ANN structures that are in use today or have been used earlier; however, there are generally classified as follows:

- Single layer feed forward networks
- Multilayer feed forward networks
- Recurrent networks
- Self-Organizing map

3.3.1 Single Layer Feed Forward Networks

The single layer feed forward network is one of the simplest and most basic networks. The most primitive configuration of the single layer network is a data input which is connected to the output neuron [7]. The arrow in Figure 3.6 indicates the direction of the data flow, showing that the data can flow only in the forward direction. At the time of counting the number of layers, the input layer is not considered since it acts only as a distribution layer. The network may have one or more input nodes but it will have only one layer; generally in this case an output layer [13].

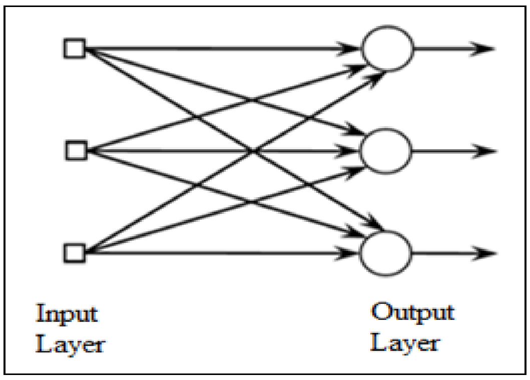


Figure 3.6: Single layer feed forward network.

3.3.2 Multilayer Feed Forward Networks

Figure 3.7 shows a multilayer feed forward network. The difference between single and multilayer feed forward networks is the addition of one or more layers which are termed hidden layers. It is mainly referred to as hidden because these cannot be seen directly either from the input or the output of the network. The main advantage of the hidden layer is that it interfaces the input and output layers in a beneficial way. Addition of the hidden layers allows the network to extract higher-order statistics [7].

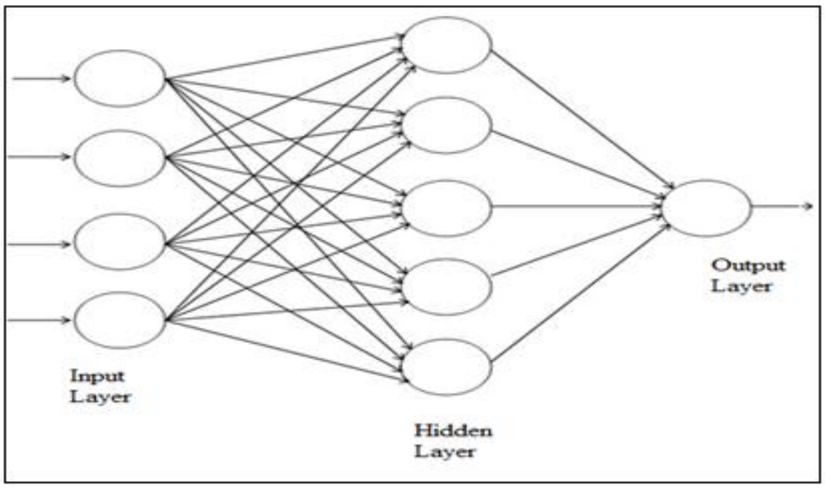


Figure 3.7: Multilayer feed forward network.

3.3.3 Recurrent Networks

In recurrent networks, there is a feedback path from one layer to another. Feedback is very constructive in most cases as it affects the learning and performance of a network. In the network shown in Figure 3.8, the feedback which connects the output to the input of the neurons in the hidden layer is represented by dashed lines. This feedback also incorporates a time delay, thereby enabling the network to work more efficiently [7].

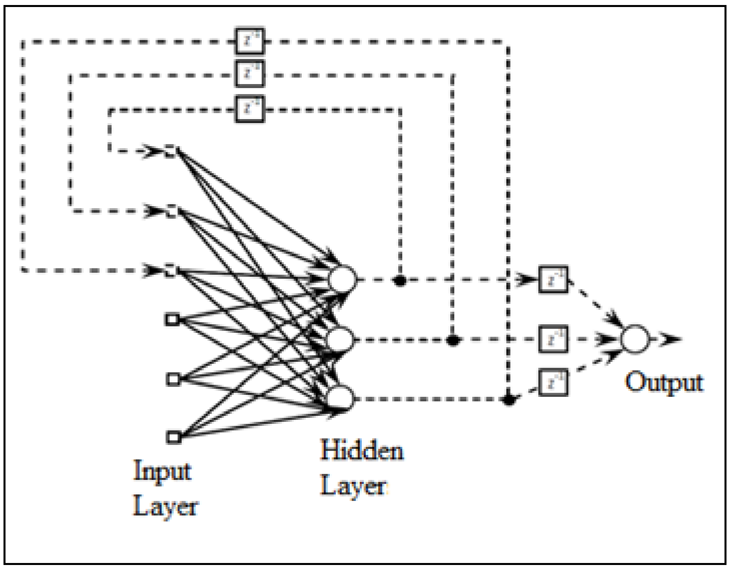


Figure 3.8: Recurrent network.

3.3.4 Self-Organizing Map (SOM)

Another important architecture in neural networks is the self-organizing map (SOM) which is trained using unsupervised learning. It produces generally an output which is 2-D and is representation of the input training samples called a map. They mainly use neighbourhood functions. Since the model was first described by professor Kohonen, it is also known as the Kohonen map.

The Kohonen neural network is very different in comparison to the feedforward back propagation networks chiefly in the manner in which recalling and training occurs. Kohonen network does not use an activation function nor does it sort weight bias. On presenting a pattern to the network, one of the output neurons is selected and is given as the output. The main advantage of these networks are that they are simple to construct and can be trained quickly and easily [10].

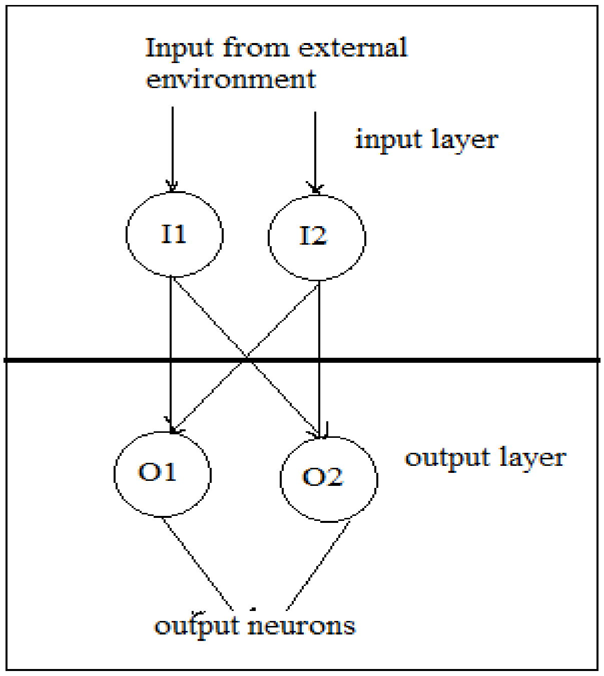


Figure 3.9: Kohonen network [10].

3.3.5 Other Networks

Some of the networks that are widely used in several applications are as follows:

3.3.5.1 Hopfield network

The Hopfield network is one of the oldest artificial neural networks and is a type of recurrent neural network. These networks consist of a model which is for the human memory. They use binary threshold units. It uses threshold activation function and the output value is determined whether the units input exceed a particular threshold value. Therefore, the network units can take a value of either 0 or 1 or values of -1 to 1. For training the network, since the network consists of a content addressable memory system, the energy states should be lowered so that the network would remember. In this network, all the nodes were interconnected to each other. It is not very often used today due to more advanced architectures available.

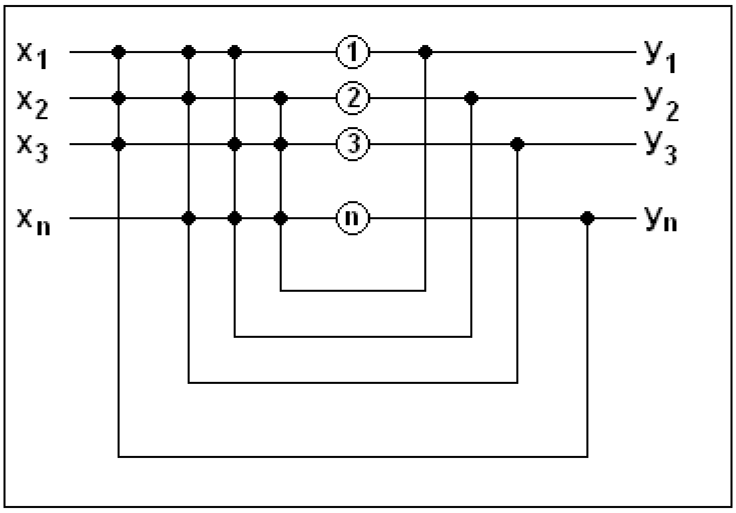


Figure 3.10: Single layer n-neuron Hopfield network [18].

3.3.5.2 Elman recurrent network

The Elman recurrent is a network which has been used to mainly solve the prediction glitches. It consists of a feedback from the outputs of some neurons in the hidden layer or the output layer to a context layer. This context layer is in addition to the input layer. Context nodes are the feedback from the output of the neurons in the hidden layer to the context layer [7]. Generally tansig activation function is used for the hidden layers and purelin is used for the output layer [3].

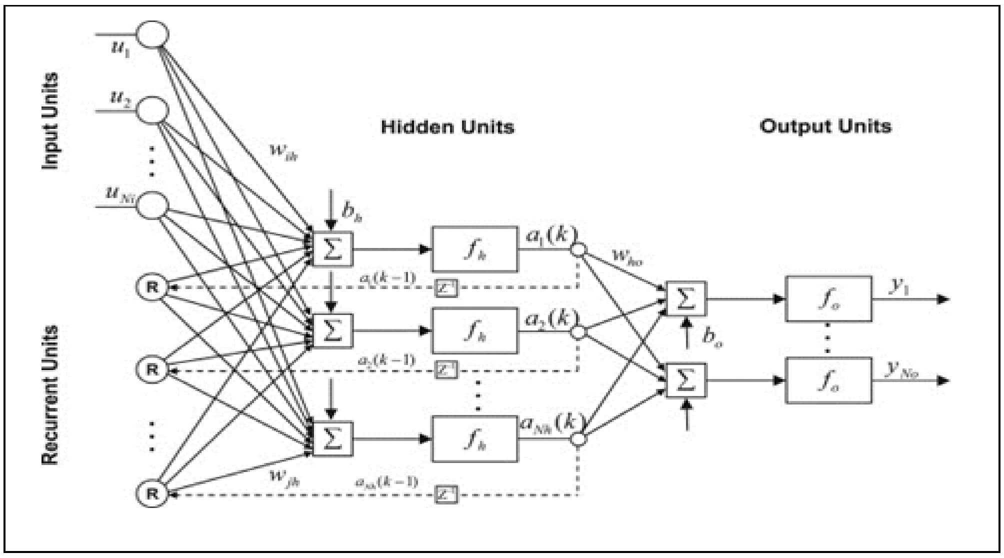


Figure 3.11: Elman recurrent network.

3.4 LEARNING PROCESSES

The major advantage of using neural networks is that they can be trained or programmed for a specific application according to the type of network created and weights used. The neural networks are trained using the dataset called training data and are tested using the test dataset. The learning algorithm in an ANN is a process which is used to train the network. It is possible to train the network through two main learning processes:

- Learning with a teacher (Supervised training)
- Learning without a teacher (Unsupervised training)

3.4.1 Learning with a Teacher

Learning with a teacher, also called supervised learning is a learning technique where both the inputs and the outputs are given to the network. Once the inputs are given to the network, it compares the desired and calculated outputs. The network training algorithm consists of a feedback process such that the weights and the biases are adjusted so that the error obtained between the calculated and desired output is minimal [3]. Learning with a teacher is one of the most widely used techniques in the training of ANN's and is used in commercial applications as it is more accurate in predicting the output.

The back propagation algorithm is primarily learning or training algorithm in neural networks. It is considered as one of the classic algorithms as many new networks are based on it. It is a type of the supervised learning process. In order to train any network, it is important for the weights to be adjusted in such a manner that the error is the least. The network thus has to calculate the error derivative of the weights. The back propagation algorithm mainly learns by examples. The examples are given to the network to do the changes as required and it changes the weights such that the desired output is obtained at the output of the training. Back Propagation is essentially useful in pattern recognition techniques and Mapping tasks [7].

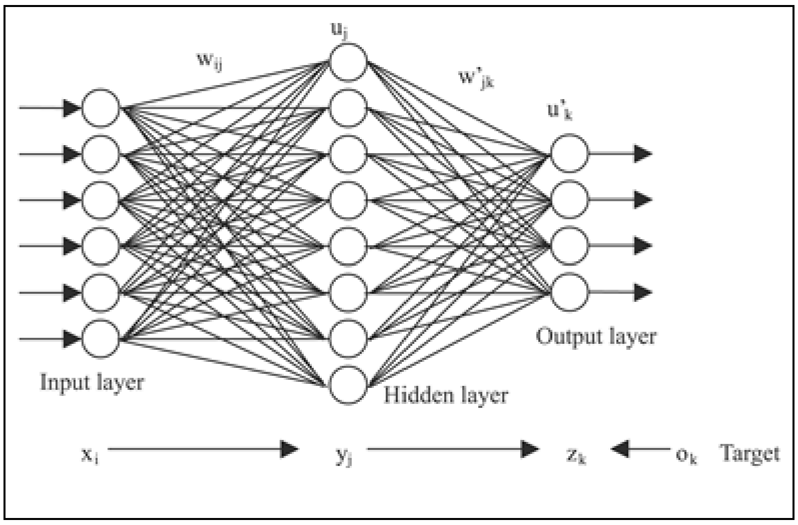


Figure 3.12: Back Propagation neural network [11].

3.4.2 Learning without a Teacher

In this type of learning process, there is no example of the input-output which is given to the network and is independent of target data for learning purposes. The input data is fed into the network which in turn builds a map which is stored in the neurons. Since in this case the output is very independent, there may be times when the output deviates immensely compared to the required output. Learning without a teacher is not used for time-series forecasting analysis since it is a random process [7]. In this project, it will not be used for training the network.

3.5 ANN USE IN BGL PREDICTIONS

There have been a number of attempts to use Information Technology (IT) in the treatments of various medical conditions. Several advancements in the attempt to help in management of diabetes have been made to prevent both short and long-term complications. Significant development is observed primarily due to the implementation of electronic instruments such as blood glucose meters, insulin pumps, and use of software databases or various compartmental models, algorithms, for the understanding, predicting and controlling the glucoregulatory process [3], [13],[14].

ANNs can be very useful as it is possible to train the network to perform complex functions and model them in a non-linear glucoregulatory system. It therefore generally makes accurate predictions for an individual patient’s BGL profile. Also, with proper training, ANNs can be used for patients having Type-2 diabetes mellitus, but this has not been carried out in this project.

CHAPTER 4: AUTOMATED INSULIN DOSAGE ADVISOR (AIDA)

4.1 INTRODUCTION

The Automated insulin dosage advisor (AIDA) is freely available software which is chiefly a simulator program of the interaction of glucose-insulin, insulin dose and the modifications of the dietary requirements for patients suffering from diabetes mellitus (mainly Type-1). It virtually gives the blood glucose profile with variations in the level of insulin and diet [2].

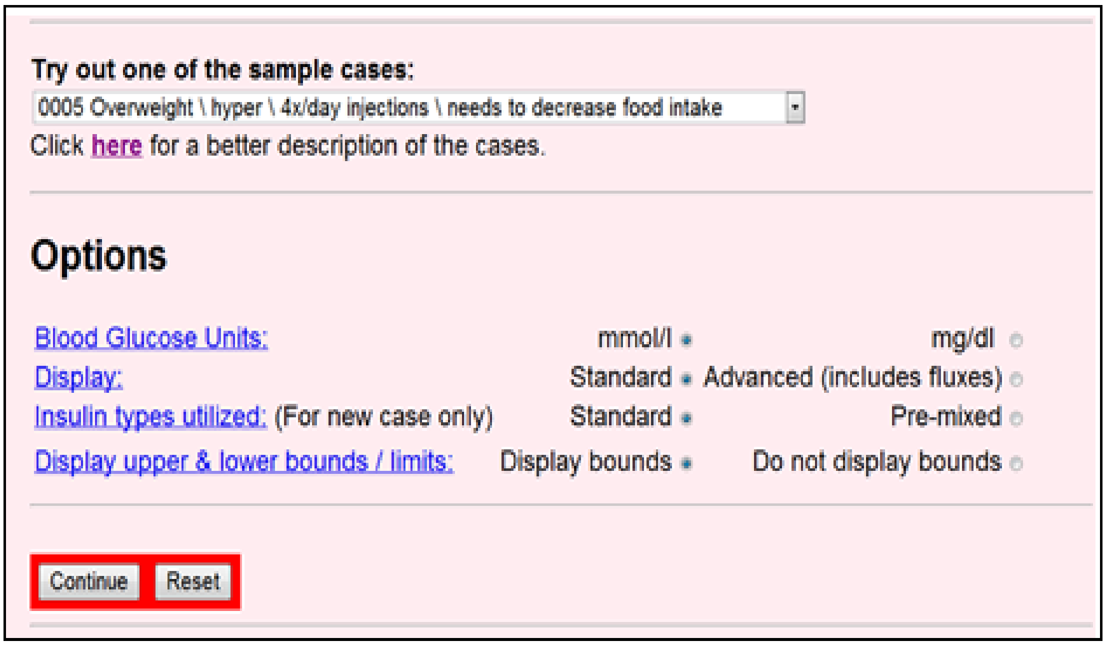
AIDA can be accessed by either downloading the software as a stand-alone package which is generally compatible to the latest Windows PC's or Apple Macintosh computers or it can also be used online from the website www.2aida.net. The main intent of the software is for educational purposes only and cannot be used for therapeutic purposes.

The AIDA software consists of 40 different cases although further additions can be entered by users. AIDA has a simple knowledge-based system which identifies problems in different cases and presents a theoretical solution. AIDA allows 24 hour simulations of the glucose profile for the patients. It helps IDDM patients to have an electronic log which may help them in giving assistance about the insulin dosage or the carbohydrate content in the diet [2].

The model is dependent on various inputs and considering a number of factors namely the amount and duration of intake of carbohydrates, type, amount and timings of the insulin dosages, the function of the kidney and the insulin sensitivity. The time duration is generally (by default) in hours, the carbohydrate intake in grams and the insulin amount in units. The AIDA model uses a compartmental architecture and the blood glucose is displayed over a time period of 24 hours in steps of 15 minutes.

4.2 ONLINE AIDA VERSION

The online AIDA version is a more updated version than the one available for downloading and hence has been used in the project for obtaining the initial input data for the neural network. AIDA is quite user-friendly. As soon as the ‘AIDA on-line’ option is chosen, the screen enables the user to choose the type of case and other specific options for the type of units for glucose and insulin as shown in Figure 4.1. Figure 4.2 shows that variations in the values of the input parameters can be achieved and the values of the meal size, insulin type and value can be changed as per an individual’s requirement [2].

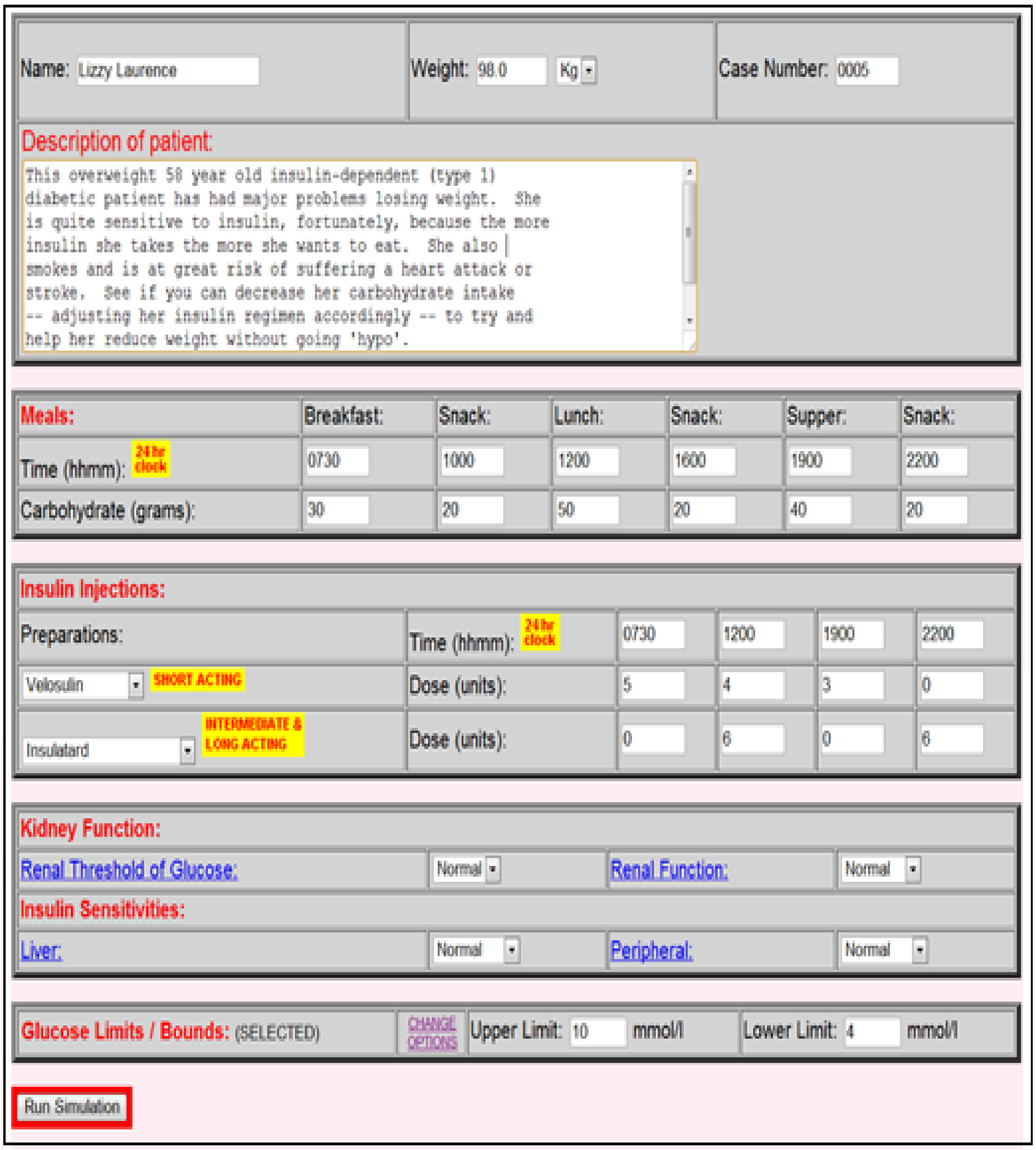

Try out one of the sample cases:
0005 Overweight \ hyper \ 4x/day injections \ needs to decrease food intake

Click here for a better description of the cases.

Options

Blood Glucose Units:	mmol/l ●	mg/dl ○
Display:	Standard ●	Advanced (includes fluxes) ○
Insulin types utilized: (For new case only)	Standard ●	Pre-mixed ○
Display upper & lower bounds / limits:	Display bounds ●	Do not display bounds ○

Figure 4.1: AIDA case options.



Name: Lizzy Laurence	Weight: 98.0 Kg	Case Number: 0005

Description of patient:

This overweight 58 year old insulin-dependent (type 1) diabetic patient has had major problems losing weight. She is quite sensitive to insulin, fortunately, because the more insulin she takes the more she wants to eat. She also smokes and is at great risk of suffering a heart attack or stroke. See if you can decrease her carbohydrate intake -- adjusting her insulin regimen accordingly -- to try and help her reduce weight without going 'hypo'.

Meals:	Breakfast:	Snack:	Lunch:	Snack:	Supper:	Snack:
Time (hhmm): 24 hr clock	0730	1000	1200	1600	1900	2200
Carbohydrate (grams):	30	20	50	20	40	20

Insulin Injections:

Preparations:	Time (hhmm): 24 hr clock	0730	1200	1900	2200
Velosulin SHORT ACTING	Dose (units):	5	4	3	0
Insulatard INTERMEDIATE & LONG ACTING	Dose (units):	0	6	0	6

Kidney Function:

Renal Threshold of Glucose:	Normal	Renal Function:	Normal

Insulin Sensitivities:

Liver:	Normal	Peripheral:	Normal

Glucose Limits / Bounds: (SELECTED)	CHANGE OPTIONS	Upper Limit: 10 mmol/l	Lower Limit: 4 mmol/l

Figure 4.2: Case 5 screen.

Figure 4.3 is the screen that appears when 'Run simulation' is selected. It gives the blood glucose profile with respect to the carbohydrate intake and the insulin uptake for a period of 24 hours. The exact values of blood glucose are observed when 'Data' is selected on the screen and the resultant values are as shown in Figure 4.4.

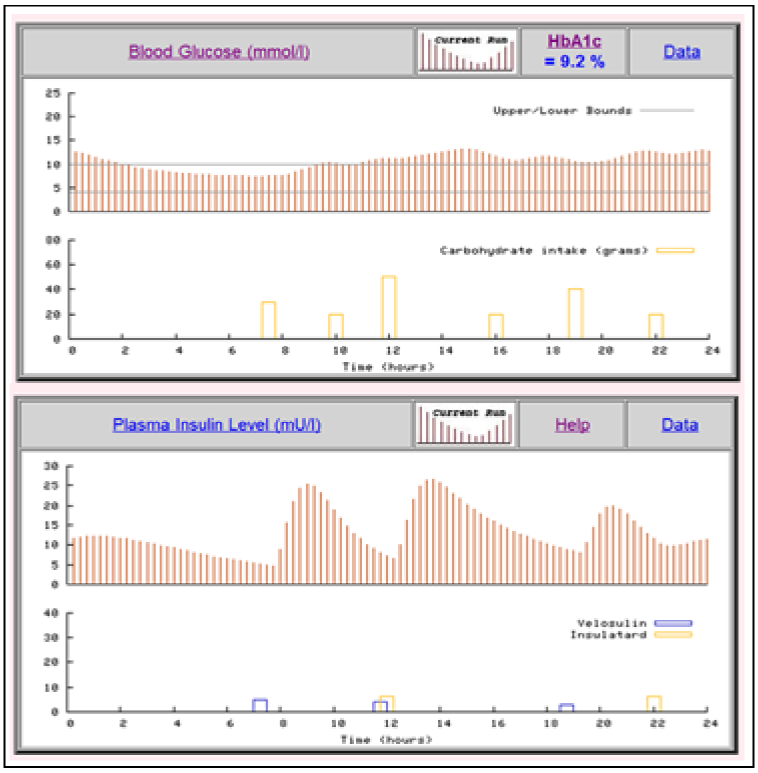
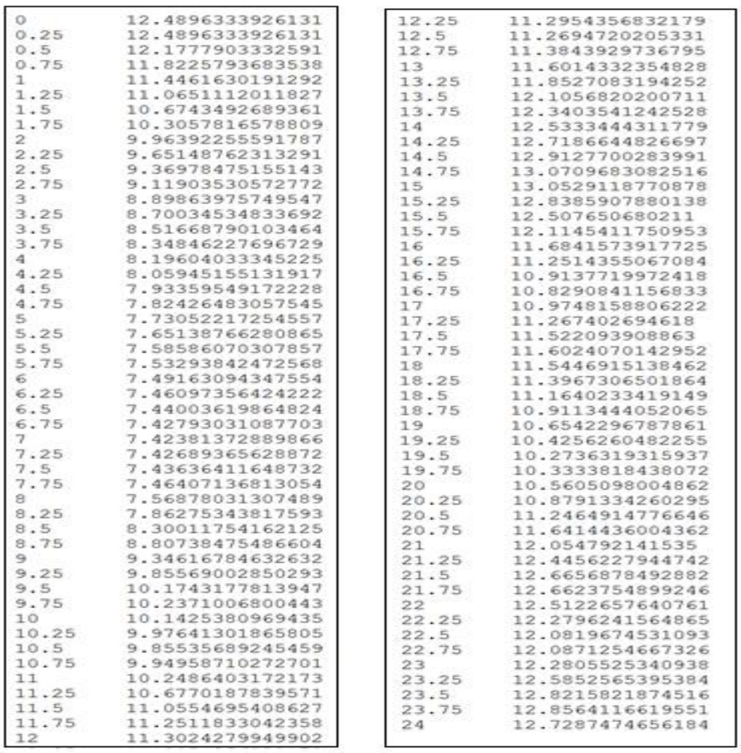


Figure 4.3: Output graphs obtained.



```
0          12.4896333926131
0.25       12.4896333926131
0.5        12.1777903332591
0.75       11.8225793683538
1          11.4461630191292
1.25       11.0651112011827
1.5        10.6743492689361
1.75       10.3057816578809
2          9.96392255591787
2.25       9.65148762313291
2.5        9.36978475155143
2.75       9.11903530572772
3          8.89863975749547
3.25       8.70034534833692
3.5        8.51668790103464
3.75       8.34846227696729
4          8.19604033345225
4.25       8.05945155131917
4.5        7.93359549172228
4.75       7.82426483057545
5          7.73052217254557
5.25       7.65138766280865
5.5        7.58586070307857
5.75       7.53293842472568
6          7.49163094347554
6.25       7.46097356424222
6.5        7.44003619864824
6.75       7.42793031087703
7          7.42381372889866
7.25       7.42689365628872
7.5        7.43636411648732
7.75       7.46407136813054
8          7.56878031307489
8.25       7.86275343817593
8.5        8.30011754162125
8.75       8.80738475486604
9          9.34616784632632
9.25       9.85569002850293
9.5        10.1743177813947
9.75       10.2371006800443
10         10.1425380969435
10.25      9.97641301865805
10.5       9.85535689245459
10.75      9.94958710272701
11         10.2486403172173
11.25      10.6770187839571
11.5       11.0554695408627
11.75      11.2511833042358
12         11.3024279949902
```

```
12.25      11.2954356832179
12.5       11.2694720205331
12.75      11.3843929736795
13         11.6014332354828
13.25      11.8527083194252
13.5       12.1056820200711
13.75      12.3403541242528
14         12.5333444311779
14.25      12.7186644826697
14.5       12.9127700283991
14.75      13.0709683082516
15         13.0529118770878
15.25      12.8385907880138
15.5       12.507650680211
15.75      12.1145411750953
16         11.6841573917725
16.25      11.2514355067084
16.5       10.9137719972418
16.75      10.8290841156833
17         10.9748158806222
17.25      11.267402694618
17.5       11.522093908863
17.75      11.6024070142952
18         11.5446915138462
18.25      11.3967306501864
18.5       11.1640233419149
18.75      10.9113444052065
19         10.6542296787861
19.25      10.4256260482255
19.5       10.2736319315937
19.75      10.3333818438072
20         10.5605098004862
20.25      10.8791334260295
20.5       11.2464914776646
20.75      11.6414436004362
21         12.054792141535
21.25      12.4456227944742
21.5       12.6656878492882
21.75      12.6623754899246
22         12.5122657640761
22.25      12.2796241564865
22.5       12.0819674531093
22.75      12.0871254667326
23         12.2805525340938
23.25      12.5852565395384
23.5       12.8215821874516
23.75      12.8564116619551
24         12.7287474656184
```

Figure 4.4: Output data screen.

4.3 LIMITATIONS

The AIDA model has limitations which are generally associated with a few conceptual issues and its existing implementation. It has been concluded that although AIDA consists of a model which can calculate the amount of insulin required, it is a very simple model and is not refined enough to help coping with other important processes like stress, and exercise, which significantly affect the BGL in the body. It also does not allow the simulation of the transient conditions which could well alter the graphs. Also, the counter regulatory mechanisms of hormones such as glucagon at times of low BGL have not been integrated in this model. It has been noted that the parameters which were valid for a patient cannot be used to model patients in all conditions. The authors of AIDA also accept that the current knowledge of the processes involved in the absorption of food from the gut is limited. Also, since AIDA is used mainly for educational purposes, this limits its implementation [2].

CHAPTER 5: PROJECT METHODOLOGY

5.1 OVERVIEW

The method of implementation of this project was carried out, keeping in mind the seven primary steps in designing the neural networks as provided by the User's Guide for the Neural Networks Toolbox Design Book issued by MATLAB [9]. These steps are collecting the data, creating, initializing and configuring the network, initializing weights and biases, training, validating and implementing the network.

5.2 COLLECTING THE DATA AND INITIALIZING WEIGHTS

Data collection is the first step and it is done using AIDA, which has been described in Chapter 4. The 40 case studies that are provided by AIDA vary on the basis of sex, weight, the renal function and threshold, and the insulin sensitivity. It can incorporate patient specific information and also can integrate 6 meals for a period of 24 hours. The carbohydrates are entered in grams and the maximum value that can be entered is 80 grams. The maximum possible variations in the glucose levels are up to 40 mmol/l. There is an option to include 8 different values of the insulin units, 4 for each short and long acting dosage a variety of types of insulin are also provided to suit the different cases, which add more accuracy to the solution.

In this project, case 5 was chosen to make the datasets. The description of the case was as follows: "This overweight 58 year old insulin-dependent (type-1) diabetic patient has had major problems losing weight. She is quite sensitive to insulin. Unfortunately, the more insulin she takes the more she wants to eat. She also smokes and is at great risk of suffering a heart attack or stroke. See if you can decrease her carbohydrate intake - adjusting her insulin regimen accordingly - to try and help her reduce weight without going 'hypo'" [2]. The daily food had carbohydrate consumption 6 times a day and the insulin dosages (including both short and long-term) were given 5 times a day. The type of short acting insulin was velosulin and the long acting insulin was insulatard.

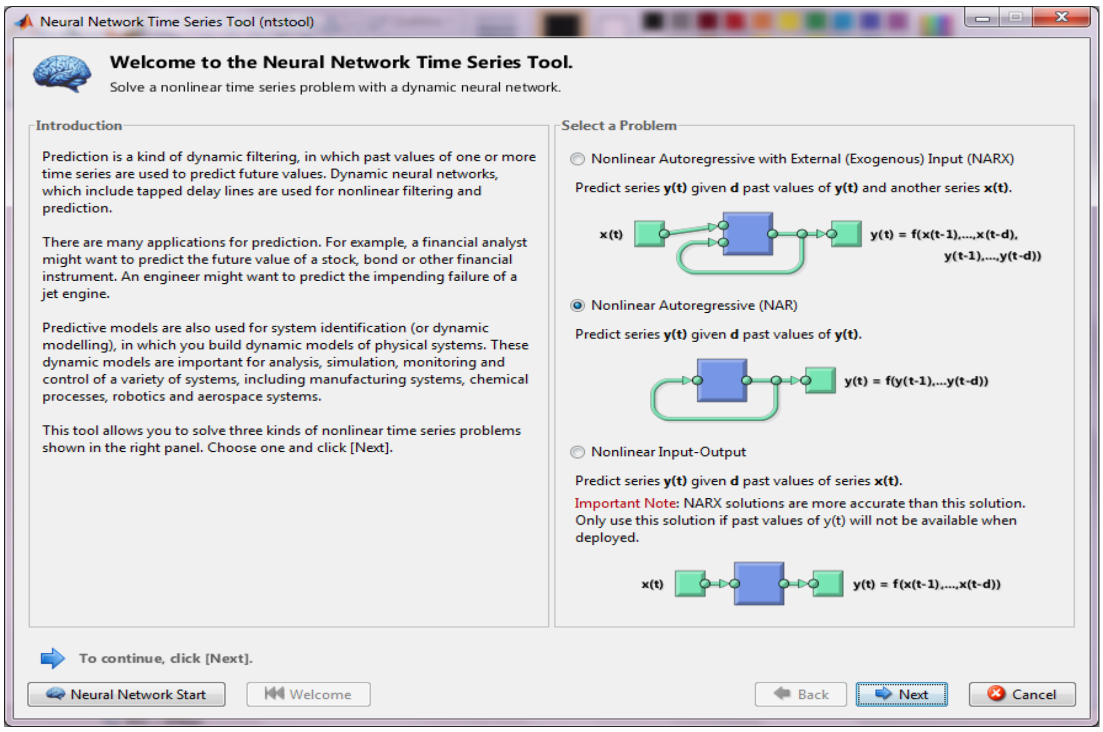
At the time of collection of the data, certain parameters like sex, weight, renal threshold, insulin type, renal function and insulin sensitivity were kept constant. The meal timings, amount of carbohydrate intake and dosage of insulin were varied to a small extent. The values obtained were merged into a dataset, which provided information about the insulin dosages, carbohydrate values, and blood glucose levels for duration of 24 hours. Any one of the 40 cases could be chosen as they all represent the data of people having the condition of Type-1 diabetes mellitus. To vary the day-to-day values of the carbohydrate values, meal times, etc. random data was created which acted as an input to the AIDA simulator and the different values of BGL were noted. Graphs such as Figure 4.3 were obtained which showed the BGL profiles for the different input values that were generated by the AIDA simulator. The obtained values of the input parameters such as insulin dosage, meal timings, etc. and the output BGL and plasma level values were stored in an MS EXCEL document.

The main importance of collecting the dataset through AIDA was to test the new network and its application in helping to solve the problem of management of diabetes. Since AIDA was validated with patients having real data, it is possible to use the obtained dataset to test the new network.

5.3 CREATING THE NETWORK

MATLAB, in comparison to other programming languages such as C and C++, is superior in problem solving. It may be used for a wide range of applications such as signal and image processing, communications, neural networks, financial analysis, etc. [9]. The neural network was created in this project using MATLAB^{TM} version 7.13.0.564 (R2011b) and the Neural Networks Toolbox version 7.0.2 (R2011b).

The network called narnet was used initially which is one of the default networks for the time series prediction in the neural network toolbox in MATLAB^{TM}. Nonlinear autoregressive (NAR) neural networks can be trained to predict the time series from past values of that series [9]. When the 'ntstool' command was given to the Command Window in MATLAB, the neural network time series window is displayed on the screen.


Neural Network Time Series Tool (ntstool)

Welcome to the Neural Network Time Series Tool.

Solve a nonlinear time series problem with a dynamic neural network.

Introduction

Prediction is a kind of dynamic filtering, in which past values of one or more time series are used to predict future values. Dynamic neural networks, which include tapped delay lines are used for nonlinear filtering and prediction.

There are many applications for prediction. For example, a financial analyst might want to predict the future value of a stock, bond or other financial instrument. An engineer might want to predict the impending failure of a jet engine.

Predictive models are also used for system identification (or dynamic modelling), in which you build dynamic models of physical systems. These dynamic models are important for analysis, simulation, monitoring and control of a variety of systems, including manufacturing systems, chemical processes, robotics and aerospace systems.

This tool allows you to solve three kinds of nonlinear time series problems shown in the right panel. Choose one and click [Next].

Select a Problem

○ Nonlinear Autoregressive with External (Exogenous) Input (NARX)

Predict series **y(t)** given **d** past values of **y(t)** and another series **x(t)**.

x(t) y(t) = f(x(t-1),...,x(t-d), y(t-1),...,y(t-d))

● Nonlinear Autoregressive (NAR)

Predict series **y(t)** given **d** past values of **y(t)**.

y(t) = f(y(t-1),...y(t-d))

○ Nonlinear Input-Output

Predict series **y(t)** given **d** past values of series **x(t)**.

Important Note: NARX solutions are more accurate than this solution. Only use this solution if past values of y(t) will not be available when deployed.

x(t) y(t) = f(x(t-1),...,x(t-d))

To continue, click [Next].

Figure 5.1: Neural Network Time Series Tool (ntstool).

After selecting the second option (narnet as shown in Figure 5.1), the inputs and the targets are given to the network. However, with various attempts in providing the inputs and the targets, it was found that since the network has a continuous feedback to the input, the output consists of 5 outputs and not just the one as desired. Also, the individual effects of the inputs were not known. Therefore, eventually in the project, a different architecture was implemented.

The new developed architecture is loosely based on the Elman recurrent network, because of its ability to predict in time series analysis. In this network, five inputs are given and it consists of two layers and an output. One layer acts as a hidden layer whereas the other layer is the output layer. There is a feedback given by the output layer to the input so that the network has the memory or the knowledge of the previous events and also may enable to know the effects of the inputs. The neurons of the hidden layer have 'tansig' activation while the output has a linear activation function.

The network is created using the function 'network' where the number of inputs, and layers are entered using the net.numInputs and net.numLayers. The manner in which the inputs and the layers are connected with each other is given by inputConnect and layerConnect. The weights and biases are initialized by default but they can also have a specific value using the functions 'initnw' and 'initb'. The complete layout of the network can be viewed using the function 'view'. A sample code for the initialization of the network is given in the Appendix. The network was as shown in Figure 5.2.

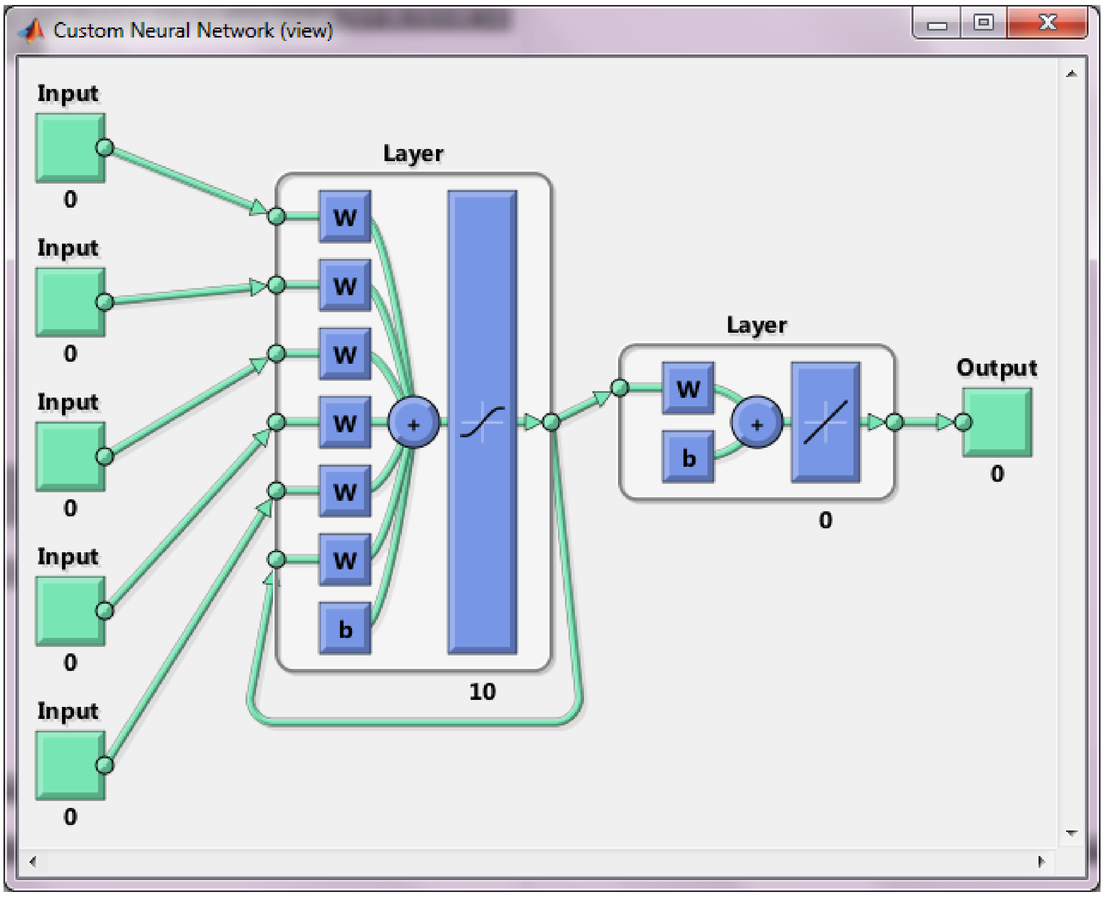


Figure 5.2: The neural network.

The input vector consists of past events. An event is a time step, which provides information of the blood glucose level, the amount of carbohydrates, the short and long acting insulin. The target is the blood glucose level, which is given at a specific prediction length along with the input vector at the time of training.

5.4 TRAINING THE NETWORK

There are mainly two types of training procedures viz. batch training and incremental training. Incremental training and batch training are applied to both static and dynamic networks; however, in several MATLAB environments, batch training is more often used. In incremental training, whenever an input is applied to the network, the weights and biases are updated. The function ‘adapt’ is used for the purpose of incremental training. In batch training, all the inputs and targets are presented and only then the weights and biases are updated. The function ‘trains’ is generally associated with batch training. In this project, the function used is ‘adaptwb’ which updates according to the weights and bias rules [9].

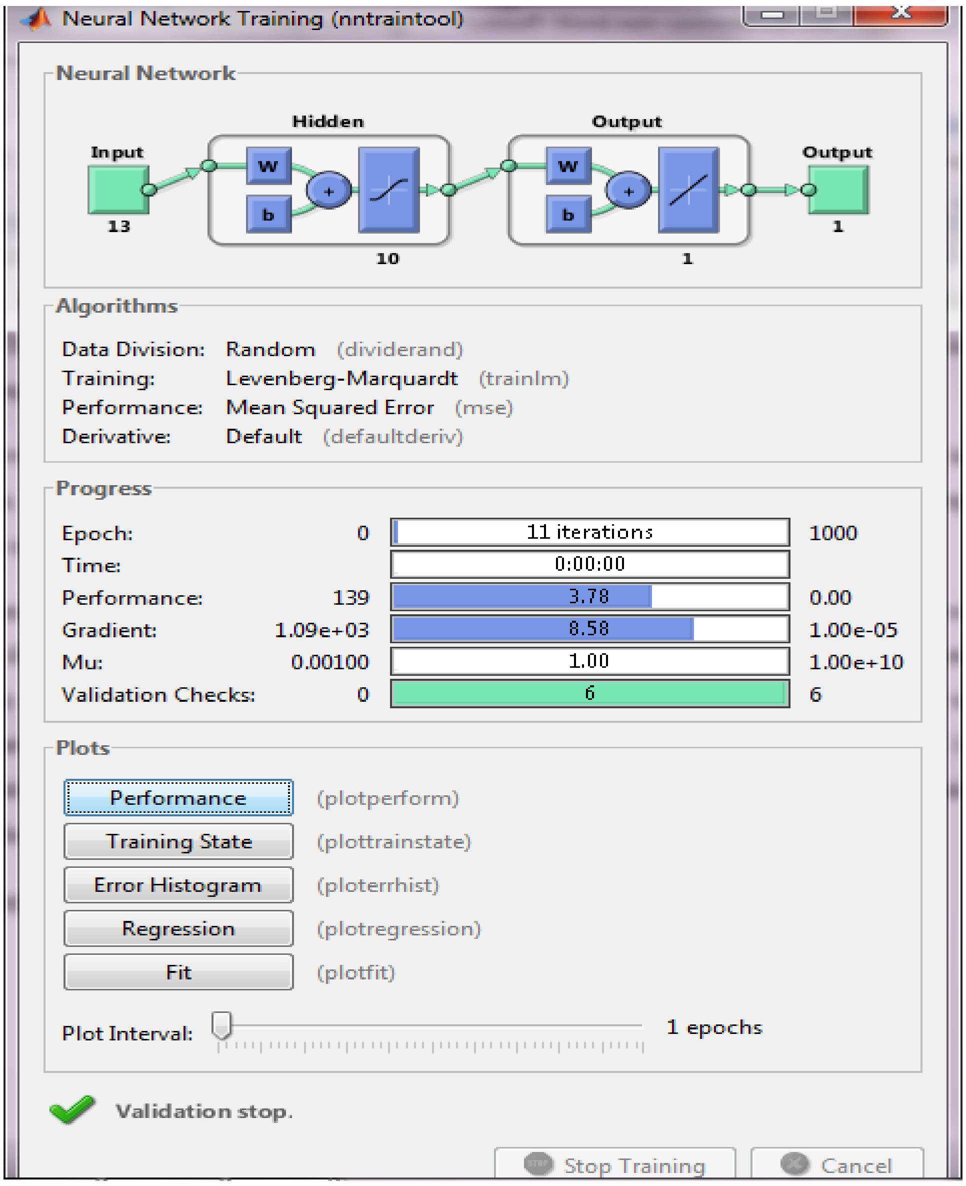
The Levenberg-Marquardt (LM) algorithm is the most commonly used training algorithm in neural networks and is the standard for several networks in MATLAB^{TM}. It is an iterative method and traces the minimum value obtained by the function which is expressed as the sum of squares of the non-linear functions. It is very widely used to solve the problems of non-linear least squares. In the case of this project, the training was undertaken using the LM algorithm. Although it consumes more memory, it helps in faster training [8].

The data allocation was achieved using the function ‘dividerand’ which, as the name suggests, divides the input and target vectors into three sets namely: training, validation and testing. The ratio is randomly split up as 70%, 15% and 15% for training, validation and testing respectively. There is also a provision to change these default settings [9].

The weights are updated using the performance function ‘trains’ or in this case using ‘adaptwb’, which is an incremental algorithm. It gives the value of network inputs and weights with each time step which is called an epoch [9].

The performance is measured using mean square error (MSE), which is the average error squared difference between the targets and outputs. The lower the value the better and zero means no error. It could also be measured using regression values (R) where the targets and outputs are correlated. A value of 0 means there is no fixed relation whereas 1 indicates a close relationship. The training was stopped when the maximum number of training epochs was reached [9].

A sample of the training window can be observed in the Figure 5.3 which is for a default feed forward architecture which has one input layer, one hidden layer and output.


Neural Network Training (nntraintool)
Neural Network
Input
13
Hidden
W
b
10
Output
W
b
1
Output
1

Algorithms

Data Division: Random (dividerand)
Training: Levenberg-Marquardt (trainlm)
Performance: Mean Squared Error (mse)
Derivative: Default (defaultderiv)

Progress

Epoch:	0	11 iterations	1000
Time:		0:00:00	
Performance:	139	3.78	0.00
Gradient:	1.09e+03	8.58	1.00e-05
Mu:	0.00100	1.00	1.00e+10
Validation Checks:	0	6	6

Plots
Performance (plotperform)
Training State (plottrainstate)
Error Histogram (ploterrhist)
Regression (plotregression)
Fit (plotfit)
Plot Interval: 1 epochs

Validation stop.

Figure 5.3: Training Window in MATLAB (nntraintool).

CHAPTER 6: RESULTS AND DISCUSSION

The data which was obtained using AIDA was used for testing. To test the architecture, a range of neurons in the hidden layer were narrowed down. From the previous literature written by Robertson [13], it was observed that although the number of neurons was limited to a number as small as just 3 or 7, the network still worked very well. Therefore, in this project, it was decided to select the default value for the number of neurons which is 10.

In total, the data obtained from AIDA consisted of 30 days of training data and 5 days of testing data. The input vectors that were given to the network consisted of the blood glucose value, carbohydrate intake and insulin values with the timings at which these readings were observed. It was noted that for the case considered in this project, short acting insulin was given three times a day to the patient (breakfast, lunch, supper) and long acting insulin was given twice a day (lunch, snacks). The number of events can be varied such as 2, 3, 4 and 6 events for prediction times of 1 hour, 2 hours, 4 hours and 8 hours. However, in this project, for all practical purposes, 3 events would be generally considered for a prediction time of 8 hours.

To test the working of the network, the number of events was considered as 3 events with the prediction period of 8 hours. The target vectors were also included so that the network trains properly and an average mean square error was programmed to be the parameter of extent of proper training. The 5 days test data was used to evaluate the performance of the network. The lesser the error value, superior was the functioning of the network.

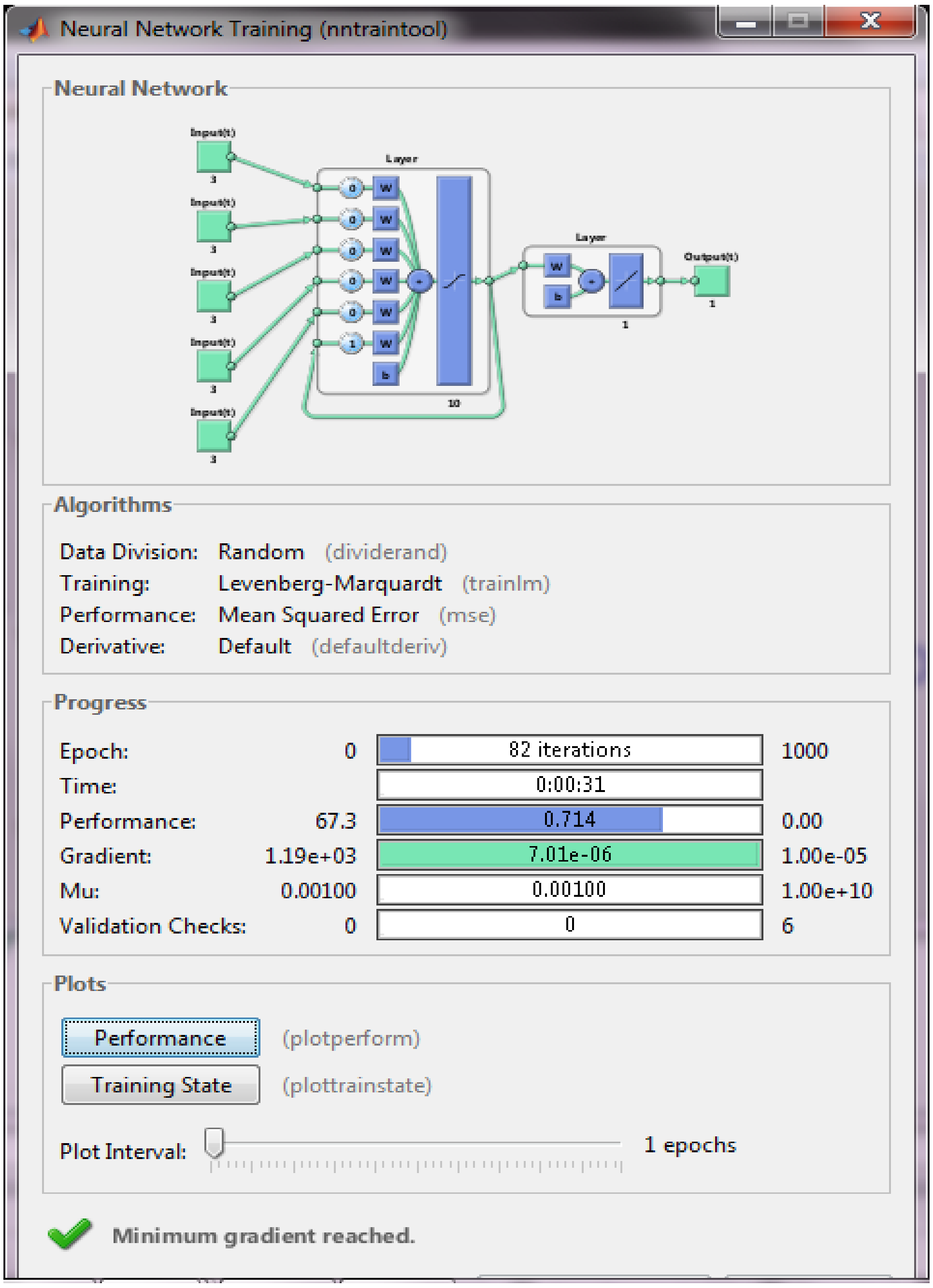
For the purpose of implementing the obtained database and to choose the correct architecture to check the effects of inputs on the outputs, several architectures were considered. Some of them were multilayer feedforward, cascade, Elman, Hopfield and the Kohonen model. The Elman is one of the most advanced types of the recurrent feedforward type and it works very efficiently in the time series analysis. The Hopfield network is an older type of architecture and with the advances in technology today, it would be better to use a relatively new and current method. The cascade model is made up of several interconnections between the inputs, layers and outputs and hence the effect of each input would not be apparent. Other new types of dynamic architectures which are offered in the newer versions of MATLAB such as timedelaynet, narxnet and narnet all work well for specific functions such as time delay and in certain cases which have just one input to the network, the output values are fed back to the input through a reasonable delay. However, it has been found that the multilayer feedforward enabled the proper mapping of the input-output data flow and also consisted of a feedback path. A model which is loosely based on the Elman recurrent network was eventually implemented.

To check if the training could be updated with a new dataset, a relatively small sample of data (only day1 data) were considered as input to the network which was trained for a small period where the number of epochs was 25. It could be inferred that the weights and biases can be updated with the datasets in an efficient manner. However, it increases the utilization of the memory space. This problem can be resolved by using the newer versions of MATLAB and specific commands are stated in the user guide for the R2012a version [9].

All the input vectors were given to the network which consisted of 10 neurons and the training of the network was done. The input vectors were imported from the file on to the m-file from which there is a ‘Generate Script’ where the values of the database were stored and given as the input. The sample to get a database from another file type onto MATLAB has been attached in the Appendix. The network was created and the training was performed. However, there was an error and the output did not yield the desired results. By default, the training stops by default when the minimum epochs was reached and in this case, was programmed to stop when maximum epochs was reached. There was an error given out by MATLAB which did not give the result.

The main error was found to be in a particular train command which was unable to get out of the default delay which was zero. The error was found to be in the network which had been unable to get out of the default delay and was not accepting the training function. Several attempts to the debugging of the network were attempted. The network worked very well and the code was able to generate the neural network and the inputs and with the help of disp( ) command in MATLAB, every line in the code was tested for prevention of the error. With the help of disp( ), the line could be tested and the output of the line could be observed in the command window. Certain changes such as increasing the delay between the input and the hidden layer, using a variety of training functions which are given in the user guide of the software, different methods to give an input and targets were also made for debugging the code, but it did not yield the desired result.

The code was then given to Gavin Robertson who rectified the code. The results were obtained and it was found that with the input data (in this case 15 days of training was given with the target), it was found that the best training obtained was 0.71443 at the 82th epoch. The Figure 6.1, Figure 6.2, Figure 6.3 and Figure 6.4 were the results obtained for the training with the improvised code which is attached in the appendix.



Algorithms

```
Data Division:  Random  (dividerand)
Training:       Levenberg-Marquardt  (trainlm)
Performance:    Mean Squared Error  (mse)
Derivative:     Default  (defaultderiv)
```

Progress

```
Epoch:                    0   82 iterations   1000
Time:                         0:00:31
Performance:           67.3   0.714           0.00
Gradient:          1.19e+03   7.01e-06        1.00e-05
Mu:                 0.00100   0.00100         1.00e+10
Validation Checks:        0   0               6
```

Plots

Performance (plotperform)
Training State (plottrainstate)
Plot Interval: 1 epochs

Minimum gradient reached.

Figure 6.1 : Output of the nntraintool.

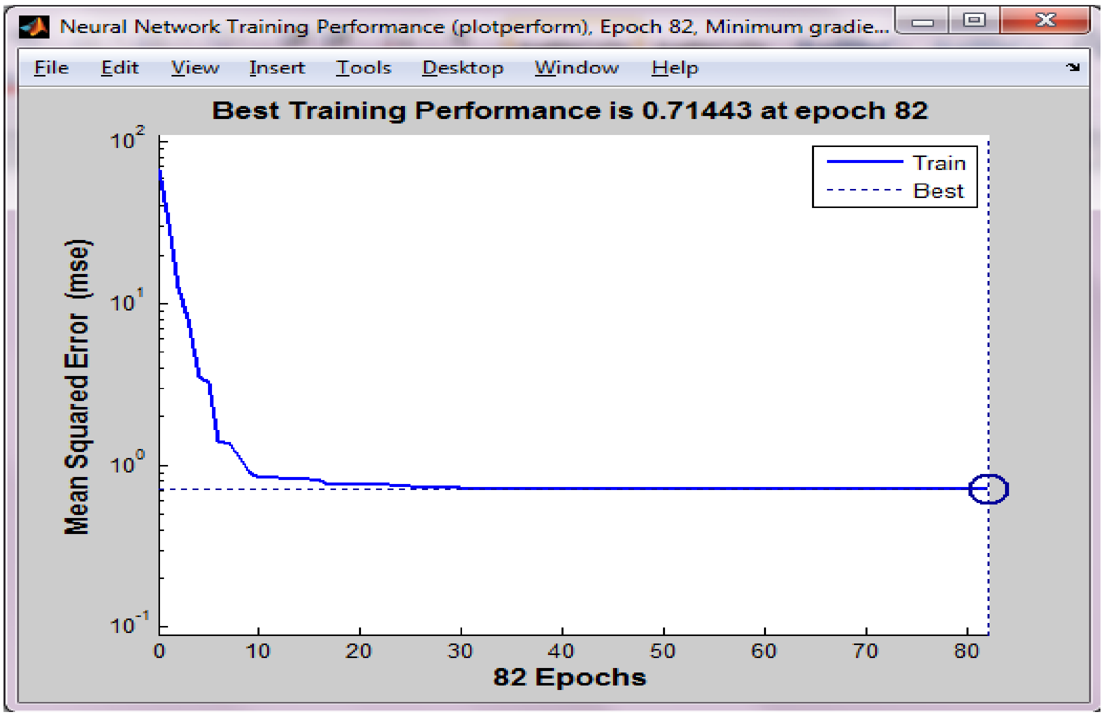


Figure 6.2: Performance plot of the training.

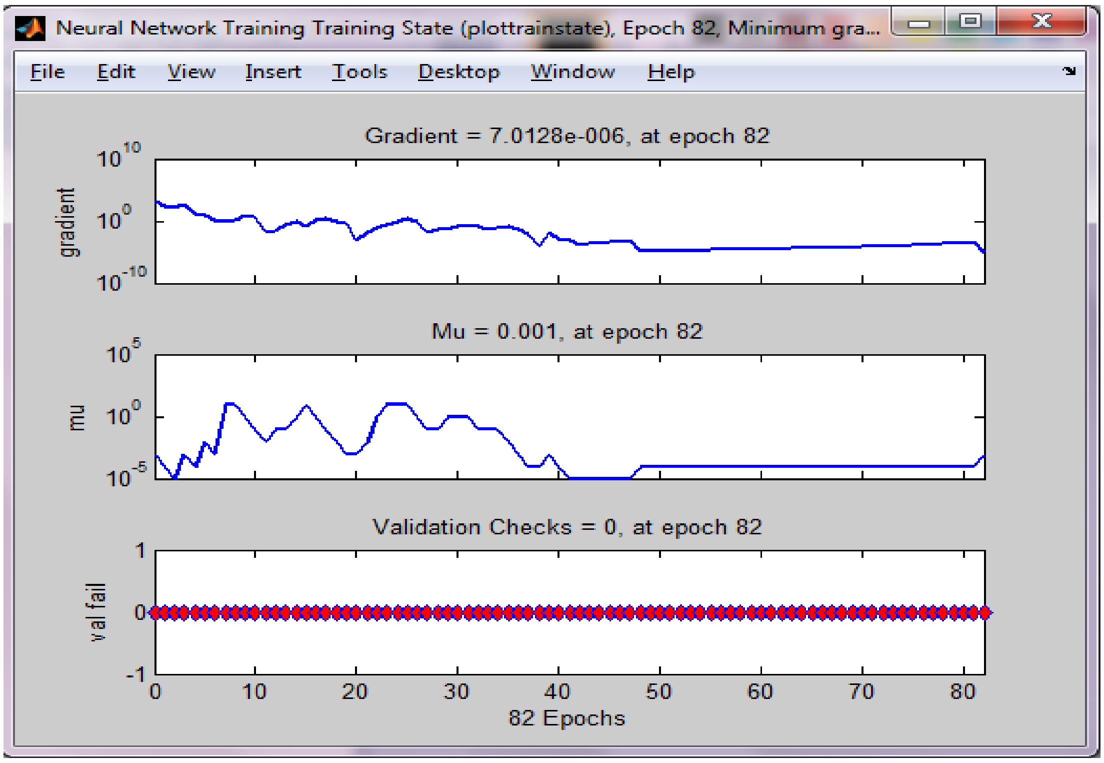


Figure 6.3: Graphs obtained from the trainstate.

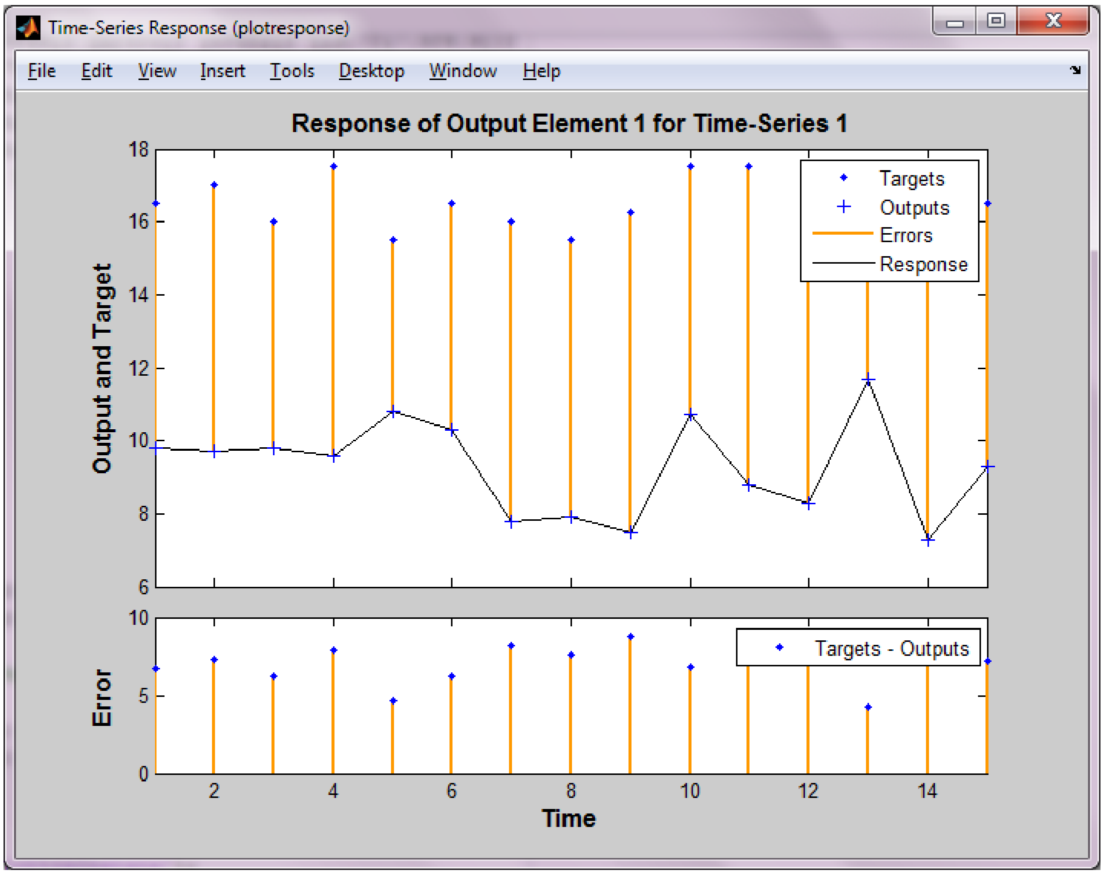


Figure 6.4: Response of Output and Target and Error.

CHAPTER 7: CONCLUSIONS AND FUTURE WORK

7.1 CONCLUSIONS

One of the aims of this project was to increment the dataset without retraining the entire network. This has been found possible with the new upcoming versions of MATLAB by writing a code using the function ‘adaptwb’ or ‘trainlm’ as the main training function. It was observed that it was possible to update the dataset with the proper declaration of this function and the associated parameter function in the training procedure. Another aim of the project was to develop a new neural network to differentiate the inputs. For this, a new architecture loosely based on the Elman network was used. Due to an error in the code and since several attempts to debug the program were unsuccessful earlier at the stage of the project, it did not yield the desired results. The obtained results were not very satisfactory as the network would be able to predict the effect of the individual inputs was unknown.

There were also some other error possibilities in obtaining the data. Since the data obtained for several days as input was not real data, there is a possibility for the effects not being present in the outputs that were obtained in the project. Also, if the duration was slightly extended, it might have been possible to check the code for the errors and get a result which could have been compared for the desired outputs.

Since the amount of extensive research being carried out in this field and new discoveries every passing year, it is possible to help management of diabetes using neural networks where the predictions are rather accurate. However, the individual effects of the inputs still need to be investigated and more time should be allotted for the same.

7.2 FUTURE SCOPE

There is a lot of scope for the development of better neural networks and also there is a lot of scope for improving the validation in the existing ones. One of the chief problems associated in this project was that the data was taken from AIDA which is a compartmental model and has its set of limitations with respect to various parameters such as insulin sensitivities, etc. Also, factors such as variations in meal timings, noise obtained from the sensors, duration of working hours were not taken into consideration in the compartmental model. Therefore, before implementing this in real patients, the testing must be done on real patients. One of the ways to reduce these problems is by taking directly real patient data.

More research could be conducted in the future to find the effects of the individual inputs on the overall performance of the network. The knowledge of the effects of the individual inputs would help greatly in the treatment of diabetes.

With every new updated version, the Neural Network toolbox in MATLAB is upgrading the functions with the passing time. For example, fitnet and feedforwardnet are used in the current versions instead of newff, newfit and newpr [9]. The updates often are very beneficial and could help in future research into ANN. Therefore, it might be possible to enable many more complex inputs (such as stress) individually and see the effects on the overall blood glucose level, which would help greatly to solve the problem of management of diabetes.

REFERENCES

[1] A.D.A.M Medical encyclopedia, 2012. Diabetes Insipidus. [Online] , Available at: http://www.ncbi.nlm.nih.gov/pubmedhealth/PMH0001415/, Last Accessed on 15 July 2012.

[2] AIDA. [Online]. Available at: http://www.2aida.net, Last accessed on 10 July, 2012.

[3] Bhansali, J., (2005). Mathematical modelling of blood glucose for short-term diabetes therapy, Bioengineering Unit, University of Strathclyde.

[4] Brown University, (2004). Edmonton Protocol- The Procedure. [Online] Available at: http://biomed.brown.edu/Courses/BI108/BI108_2004_Groups/Group09/procedure.htm, Last accessed on 30 August 2012.

[5] Diabetes UK. [Online]. Available at: http://www.diabetes.org.uk, Last accessed on 15 July, 2012.

[6] Fredric H. Martini, J. L. N. E. F. B., 2012. Fundamentals of Anatomy and Physiology. 9th ed. Boston: Benjamin Cummings.

[7] Haykin, S. (1999). Neural Networks - A Comprehensive Foundation, Prentice-Hall Inc., New Jersey.

[8] Lourakis, M. I. A., (2005). A Brief Description of the Levenberg-Marquardt Algorithm Implemented by levmar. [Online]. Available at: http://www.ics.forth.gr/~lourakis/levmar/levmar.pdf, Last accessed on 15 June 2012.

[9] MathWorks, 2012. MATLAB *overview*. [Online] Available at: http://www.mathworks.co.uk/products/matlab/ Last Accessed on 27 July 2012.

[10]Muhammad Ardalani-Farsa, S. Z., (August 2010). Chaotic time series prediction with residual analysis method using hybrid Elman–NARX neural networks. Neurocomputing, 73(13-15), pp. 2540-2553.

[11] Paolo Priore, D. d. l. F. ,. P. J. P., (2003). Dynamic scheduling of flexible manufacturing systems using neural networks and inductive learning. Integrated Manufacturing Systems, 14(2), pp. 160-168.

[12] Penfornis A, K. -P. S.,(2006). Immunosuppressive drug-induced diabetes. Diabetes and metabolism, 5(Part 2), pp. 539-546.

[13] Robertson, G.,(2010). Blood glucose prediction using artificial neural networks, Bioengineering Unit, University of Strathclyde.

[14] Robertson, G., Lehmann, E.D., Sandham, W., Hamilton, D. (2011). Blood glucose prediction using artificial neural networks trained with the AIDA diabetes simulator: a proof-of-concept pilot study, Journal of Electrical and Computer Engineering, Volume 2011, Article ID 681786.

[15] Scott, K.D., (August 1999). Blood glucose prediction for diabetes therapy using artificial neural networks. MSc Thesis, Bioengineering Unit, University of Strathclyde.

[16] The global diabetes community. [Online]. Available at: http://www.diabetes.co.uk, Last accessed 26 July 2012.

[17] WebMsD. [Online]. Available at: http://www.webmd.boots.com, Last accessed 28 July 2012.

[18] Young,J.H., (2007) . Artificial Intelligence .[Online]. Available at: http://comsci.us/ai/notes/chap06.html,[Accessed 25 July 2012].

APPENDIX

MATLAB CODE

IMPORTING DATA

The Microsoft Excel document which had the database was imported in MATLAB:

File →Import Data → Specify the file name → Import Wizard dialogue box appears.

The sample code generated when the file is imported:

```
function [newData1] = importfile(fileToRead1)
%IMPORTFILE(FILETOREAD1)
%  Imports data from the specified file
%  FILETOREAD1:  file to read
%  Auto-generated by MATLAB on 21-Jul-2012 15:16:39
% Import the file
sheetName='Sheet1';
[numbers, strings] = xlsread(fileToRead1, sheetName);
if ~isempty(numbers)
    newData1.data =  numbers;
end
if ~isempty(strings)
    newData1.textdata =  strings;
end
```

Once the file has been imported, the generated sample script has to be saved. If the script is saved in the same file as that of the program, then just the name of the M-file is sufficient to load the data on to the current M-file and the data or the desired results could be observed.

CREATING A NETWORK

```
net = network;% making a custom network
net.numInputs = 5;%defining the number of inputs
net.numLayers = 2;%defininf the number of layers
net.inputConnect = [1 1 1 1 1;0 0 0 0 0];
net.layerConnect = [1 1; 1 0];
net.outputConnect = [0 1];
net.biasConnect = [1; 1];
net.layerWeights{1}.delays = 1;
net.layers{1}.transferFcn = 'tansig';
view(net);% viewing the network
```

TRAINING THE NETWORK

```
net.adaptFcn = 'adaptwb';% used instead of trains; it’s an updated
command
net.divideFcn = 'dividerand';% randomly allots the data for
training, validation and testing; generally 70% training,
15%validation and 15% testing.
net.initFcn = 'initlay';
net.performFcn = 'mse';% measuring the performance of the network
using mean square error
```

```
[net,tr] = train(net,inputs,outputs);%training the network
```

NETWORK DIAGRAM

The neural network can be displayed with the help of the view (name of the network) command. For example:

```
View (net);
```

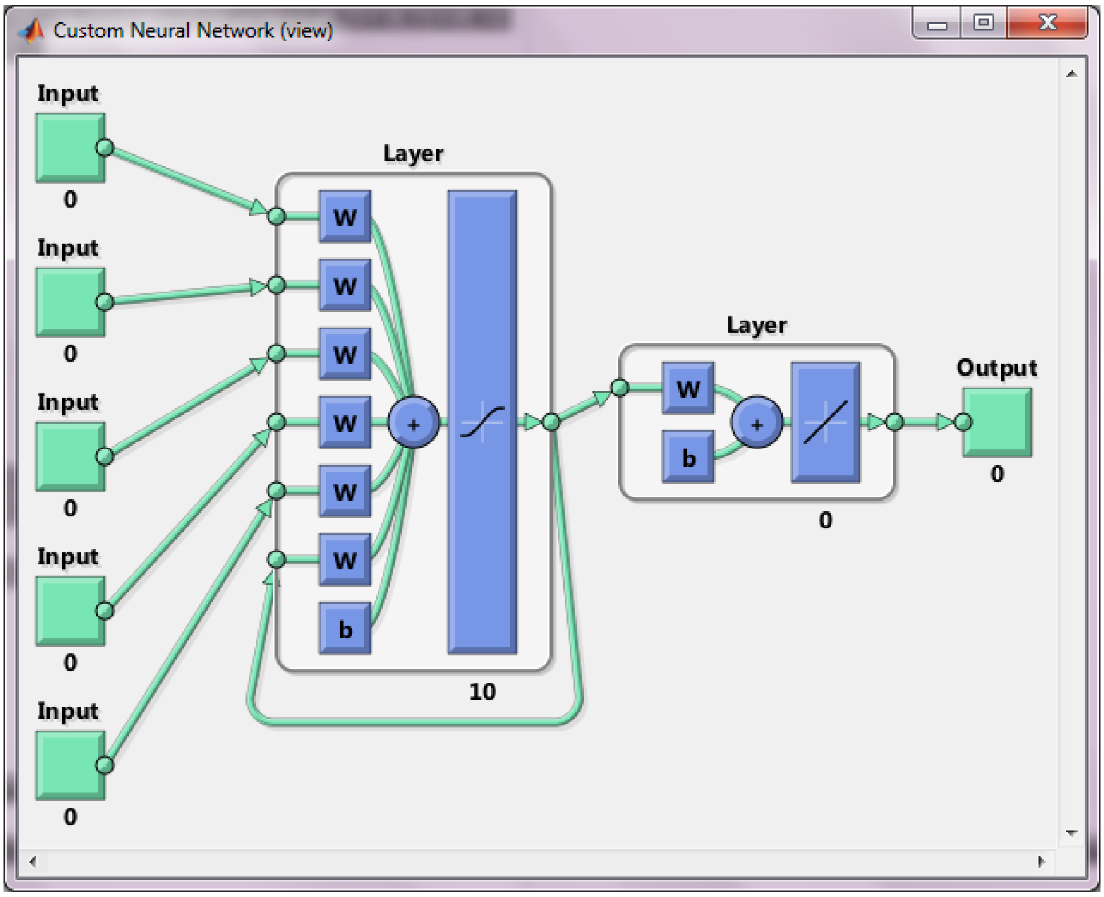


Figure A: Sample network.

THE COMPLETE CODE:

```
net = network;
net.numInputs = 5;
net.numLayers = 2;
net.inputConnect = [1 1 1 1 1;0 0 0 0 0];
net.layerConnect = [1 0; 1 0];
net.outputConnect = [0 1];
net.biasConnect = [1;1];
net.layers{1}.transferFcn = 'tansig';
net.layers{1}.size = 10;
view(net);
```

```
P1= {[16.50;19.50;22.00]; [20;30;40]; [0;3;0];[0;0;6];
[12.8;14.7;13.6]}  %needs to a cell input to network training try:
P2= {[17.00;19.50;23.00]; [15;45;30]; [0;4;0];
[0;0;5];[12.5;14.6;13.6]}
P3= {[16.00;19.50;22.50]; [20;55;40]; [0;5;0]; [0;0;3];
[11.7;14.2;13.6]}
P4= {[17.50;19.50;22.50]; [15;30;40]; [0;3;0]; [0;0;4];
[14.2;12.7;13.3]}
P5= {[15.50;20.50;22.00]; [30;50;40]; [0;5;0]; [0;0;6];
[13.6;15;8.5]}
P6= {[16.50;20.50;23.00]; [5;40;25]; [0;4;0]; [0;0;2]; [11.6;15;12]}
P7= {[16.00;20.00;22.50]; [30;40;10]; [0;4;0]; [0;0;5];
[11.3;14.9;12]}
P8= {[15.50;19.50;22.00]; [30;40;15]; [0;3;0]; [0;0;4];
[10.5;14.5;11.9]}
P9= {[16.25;19.00;22.00]; [10;50;05]; [0;5;0]; [0;0;4];
[13.7;12.5;11.6]}
P10= {[17.50;20.00;22.50]; [40;60;15]; [0;6;0]; [0;0;3];
[13.7;17.5;9.2]}
P11= {[17.50;21.00;23.00]; [40;50;10]; [0;5;0]; [0;0;2];
[13.7;17.5;8.9]}
P12= {[16.50; 20.00; 22.50]; [40; 75; 10]; [0;6;0]; [0;0;4];
[13.4;17.4;8.9]}
P13= {[16.00;21.50;23.00]; [40;25;10]; [0;3;0];
[0;0;4];[12.4;16.7;8.8]}
P14= {[16.00;20.50;23.00]; [15;55;10]; [0;5;0]; [0;0;6];
[16.1;14.5;8.2]}
P15= {[16.50;19.00;22.00]; [20;40;20]; [0;3;0]; [0;0;6];
[10.6;12.5;11.6]}
T1={9.8}
T2={9.7}
T3={9.8}
T4={9.6}
T5={10.8}
T6={10.3}
T7={7.8}
T8={7.9}
T9={7.5}
T10={10.7}
T11={8.8}
T12={8.3}
T13={11.7}
T14={7.3}
T15={9.3}
P=[P1 P2 P3 P4 P5 P6 P7 P8 P9 P10 P11 P12 P13 P14 P15];
T=[T1 T2 T3 T4 T5 T6 T7 T8 T9 T10 T11 T12 T13 T14 T15];

net.layerWeights{1}.delays = 1; %need this to define delay!

net.layers{1}.initFcn = 'initnw';
net.layers{2}.initFcn = 'initnw';
net.divideFcn = 'dividerand';
net.initFcn = 'initlay';
net.performFcn = 'mse';
net.plotFcns = {'plotperform','plottrainstate'};
net.trainFcn = 'trainlm';
net = train(net,P,T); %training the network
plotresponse (P,T);
```