Contents

1 Abstract 3
2 Introduction 4
2.1 General 4
2.2 Aims and Objectives 4
3 Literature Research & Review 5
3.1 Construction Processes 5
3.1.1 Stochastic queuing systems 5
3.1.2 The Concrete Pouring Process 5
3.2 Alternative Available Techniques 5
3.2.1 Regression 6
3.2.2 Petri Nets 6
3.2.3 Genetic Algorithm 7
3.3 Neural Networks 7
3.3.1 Introductions 7
3.3.2 Applications 8
3.3.3 Biological Neurons 10
3.3.4 General Principles 11
3.3.5 Alternative Architecture 13
3.3.6 Training 13
3.3.7 Testing & Validating 14
3.3.8 Over-fitting 15
3.3.9 Generalisation 15
3.3.10 Fuzzy Networks 16
3.3.11 Limitations 16
4 Procedure 18
4.1 The Data Set 18
4.1.1 Date Source 18
4.1.2 Format 18
4.1.3 Accuracy & Reliability 18
4.1.4 Limitations 18
4.1.5 Data management 19
4.1.6 Choice of Inputs & Targets 19
4.2 NN design methods 20
4.2.1 Available ‘software’ 20
4.3 Developing NN models 21
4.3.1 Familiarisation 21
4.3.2 Training 21
4.3.3 Testing 21
4.3.4 Feed-Forward Network Investigation 22
4.3.5 Radial Bias Investigation 25
4.3.6 Analysis 25
4.3.7 Sensitivity 25
4.3.8 Fuzzy Networks 26
4.4 Model Variation Methodology 26
4.4.1 Architecture 26
4.4.2 Network Size & Transfer Function 26

Revised 2026: error figures restated as root mean square error.

5 Results 27
5.1 Initial Reconnaissance into Size & Transfer Function 27
5.1.1 Network Size 27
5.1.2 Transfer Functions 30
5.1.3 Input-Output Partnerships 32
5.1.4 Summary 34
5.2 Further Investigation into Successful Networks 34
5.2.1 General Discussion 35
5.3 Predicting a Productivity Probability Histogram 38
5.3.1 Two Layer Networks 38
5.3.2 Three Layer Networks 39
5.4 Radial Basis Network Investigation 39
6 Analysis 42
6.1 Suitability of Neural Networks 42
6.1.1 2 Layer 42
6.1.2 3 Layer 46
6.1.3 Radial Basis Networks 49
6.2 Effect of Network Properties 50
6.2.1 Architecture 50
6.2.2 Size 50
6.2.3 Transfer Function 51
6.3 Probability Distribution Output 51
7 Discussion 54
7.1 Physical meaning 54
7.2 The Accuracy & Reliability of the Final Networks 57
8 Conclusions 62
8.1 Can Neural Networks be used effectively in the Construction Industry? 62
8.2 Specific Findings 62
8.3 Summary of Optimum Results 63
9 Further Work 64
9.1 Increased Data Set 64
9.2 Extensions of work described 64
9.3 Options not investigated 64
10 References 66
11 Graphical Representation of Network Performances 68

1 Abstract

Artificial neural networks are extremely flexible computational devices that are being applied to an ever-diversifying range of applications. The question of whether or not they can be used in the modelling of stochastic construction processes forms the focus of this thesis. Stochastic process and the techniques commonly used in their analyse are briefly discussed before an introduction into neural network fundamentals. The potential of neural networks is illustrated though a comprehensive explanation of network selection, training and validation for a prototype network that predicts the productivity of concrete pours. Variations of feed-forward, radial basis and fuzzy networks have been used for the investigation. Two and three-layer networks are used with varying sizes, transfer functions and input–output pairings. Despite the relatively small data set available a number of very effective networks were developed using three different methods of generalisation, namely; size reduction, regularisation & early stopping.

2 Introduction

2.1 General

Construction processes frequently cannot be modelled by purely deterministic means; they are stochastic systems involving elements of randomness. To estimate the associated productivity and cost effectively, some account of the random distribution must be included. Two such processes are earthmoving and concrete operations; both are cyclic queuing systems that have a substantial effect on most construction projects. Successful modelling of such systems can increase the accuracy of productivity estimates and hence improved cost and task duration assessment. Further more the effects of unpredictable events can be seen and if required taken into account.

Biological neural networks are the mechanisms that enable us to read, move, breath, laugh, in short think. The biological networks present in humans are, not surprisingly, incredibly complicated comprising of some 10^{11} highly interconnected neurons. Each of these is made up of physical tissue and a dynamic chemical state. The exact nature in which the biological neural network functions is still unknown, however it is widely accepted that all biological neural functions (e.g. memory) are stored partly in the neurons themselves and partly by their connections to each other. Learning can be thought of as the altering of connections between neurons as a result of experiences.

The development of the discovery outlined above prompted the question, is it possible to construct artificial neural networks that can be trained to perform specific functions based on the biological model? Towards the end of the 1950's artificial neural networks were produced to solve a limited class of simple pattern recognition problems (Hagan, 1996). The artificial neuron is an incredibly basic simplification of the biological neuron. It consists of a matrix representing its connections to other neurons, an offset and a simple mathematical function. Combining various combinations of these singularly very simple elements allow quite complicated functions to be simulated. One of the advantages of neural networks that hints at its biological inspiration is the fact that a neural network cannot be programmed, it must be taught by example.

2.2 Aims and Objectives

- By means of a full literature survey and review present a critical overview of stochastic construction processes; methods in which these processes can and have been investigated and an in depth review of artificial neural networks and their application to engineering problems.
- Determine the feasibility of using artificial neural networks as a method of analysing non-deterministic construction processes.
- Can artificial neural networks be used to accurately predict the productivity of construction processes? Assuming they can, investigate the alternative methods and their flexibility by developing and testing prototype networks.

3 Literature Research & Review

3.1 Construction Processes

3.1.1 Stochastic queuing systems

Figure 1 below shows the basics of a generic queuing process, the inclusion of the back-cycle allows us to categorise this example as a finite system as the customer returns to the queue having been served after a time related to the systems properties. An infinite system would be one where the customers appeared at the queue on some random basis and leaves the process once served.

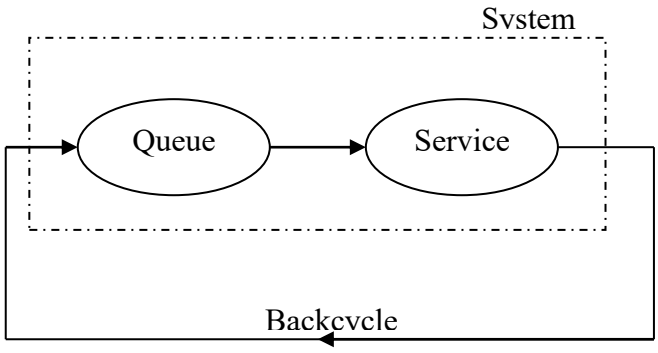


Figure 1: Schematic representation of general queuing process.

If the service time and back-cycle time (inter-arrival time for infinite systems) are constant for each customer then the system is deterministic. For the earthmoving and concrete placing systems considered here clearly both the service and back-cycle time vary considerably. In these cases the system is non-deterministic but has become stochastic. Deterministic approaches are no longer an accurate method for analysis as they fail to take into account the dynamic interaction between customers. It is these interactions that will affect the productivity of the system and therefore a stochastic model is required for reliable productivity estimates.

3.1.2 The Concrete Pouring Process

The concrete process consists of batching, transport and placement of the concrete. The concrete trucks can be assumed to be the customer while the pump may be considered as the server. The placement can be broken down further to the spot time (the time required for the truck to manoeuvre it self into position), the pump time (time required for the pump to empty the trucks contents) and the washout time.

3.2 Alternative Available Techniques

There are a number of techniques that have been used in the past to model or predict the productivity of construction processes, with varying success. This section lists the most significant of these and includes a brief description of the methods, previous applications and highlights the strengths and weaknesses of

each technique. A detailed explanation of each is not intended, however the references included in this section provide further details and specific applications if the reader requires further information.

3.2.1 Regression

3.2.1.1.1 Introduction

Regression analysis attempts to weight explanatory variables and interactions in order to predict the required response. Complications arise in determining which variables and interactions are to be included and then in arriving at suitable coefficients.

A good example of the development of a regression model to predict the effect of earthmoving operations is given by Smith (Smith 1999).

The limitations vary depending upon the situation in which regression is to be used, the major drawback being the linear nature of the technique; any non-linear relationships cannot be modelled, again illustrated by Smith when considering the linear regression used to predict the bunch factor.

3.2.1.1.2 Discussion

A neural network with a linear transfer function effectively acts as a linear regression tool. If comprising of only one neuron no interactions can be modelled, by increasing the number of layers and the number of neurons with in the network a complicated arrangement of interactions are effectively modelled – for this to be consistent with interactions with physical importance requires sufficient amounts of quality data to train the network adequately.

3.2.2 Petri Nets

3.2.2.1.1 Introduction

Petri nets were developed by Carl A. Petri in 1966 and originally consisted of four basic building blocks that can be used to model construction processes graphically and mathematically. The four key elements are places, transitions, arcs and tokens. Places can be thought of as a node that represents a condition such as input data or a resource. Transitions represent a task or activity and can be fed from and/or to a place using arcs as connections. Arcs may have a multiplicity, an integer associated with them determining the amount of tokens required to initiate the transition. The tokens are the dynamic representation in the model; a place may contain zero or a positive integer value of tokens at any one time.

Essentially a transition will move the specified number of tokens from its input place to its output pace when the number of tokens in the input space is equal to the value k, of the relevant arcs multiplicity. Potentially resulting in further triggering of transitions due to the new arrangement of tokens.

A number of additional features have been added to petri nets in order to better model actual systems, for example classical petri nets included no time concept, time delays can now be introduced that are either deterministic or stochastic rather than the instantaneous classical nets. A net including all three time delay types is known as a generalised stochastic petri nets (Marsan 1989). Further advances to the basic blocks described above are also possible to further accurate modelling. For more details on the above or the enhancements consult Sawhney et al 1999.

3.2.3 Genetic Algorithm

3.2.3.1.1 Introduction

Genetic algorithms are a method of arriving at an optimum solution using a trial and error technique driven by random sampling. They are often used in conjunction with another modelling technique to compare various set-ups until the optimum solution is arrived at.

A chromosome, consisting of a number of genes, the first of which is normally reserved for an index, can represent various configurations of the input variables. Every other gene in the chromosomes represents a specific input variable. The range of values the various genes are capable of possessing are defined by the problem. The entire collection of chromosomes is known as the population, however it is possible to include sub-populations of chromosomes to cater for varying conditions between alternative options.

Initial chromosomes can either be generated randomly using the defined gene ranges or specified by the user, perhaps to include likely set-ups or even some extreme possibilities. Once all chromosomes are generated their performance / fitness is estimated using the system or model requiring optimisation. A number of techniques can be used to generate the next generation of chromosomes, the procedure is repeated until the required performance has been reached.

3.3 Neural Networks

3.3.1 Introductions

Artificial neural networks (ANNs) consist of a highly interconnected system of simple processors. The problem definition is supplied to the network as an input vector, the result is the network's output. The mapping of the input vector to the output is achieved through the relative strengths of the connections within the network whose values are arrived at via network training. During training the network is supplied with definition cases (inputs and corresponding targets) that allow the network to iteratively alter the connection strengths while minimising the error between the targets and network outputs. This enables ANNs to approximate functions even when a full understanding of the physical system is unknown, it does however rely on the training data set to define the full range of the input and output spaces

required. The make up of ANNs from individual processors define neural networks as potentially a parallel computational technique, although the use of standard serial computers does not allow the advantages of this to be seen.

Work on ANNS begun in the 1950's all be it for fairly simple problems, during the 1980's interest in ANNs was resurrected due to developments in computers and the discovery of back-propagation. Since this time many thousands of papers have been published with various alterations or additions to neural networks. Applications for the use of ANNs continue to arise and will probably continue to do so as further advances are made. The shear breadth of potential uses of ANNs suggests that this is a technique that will continue to be employed and developed for some time to come. Neural networks are certainly still in their infancy but with an increase in the understanding of their biological inspiration, progress in the associated fields e.g. optimum point searches of performance surfaces, and further computer advancements their success and possible applications should continue to grow.

3.3.2 Applications

3.3.2.1 General

Neural networks range of applications is so broad that a full picture cannot be illustrated here, ranging far beyond the expected fields for a numerical problem solving technique such as engineering, science and maths to medical and business applications to name but a few. A by no means exhaustive list is contained with in Hagan et al, 1996, 1-5. This includes neural networks uses in aircraft autopilots, speech recognition or even text to speech systems, financial market analysis, breast cancer analysis, visual quality inspection systems, defence weapon steering, real-time translation of spoken languages and wind prediction More & Deo. 2003.

The ability of ANNs to learn, classify and approximate functions (known or unknown) explains why they can be used for such a range of problems. They can be adapted to most problems that can be defined as a set of inputs each with corresponding outputs. Perhaps the range of applications would not appear so surprising as the inspiration behind the entire ANN field is that of our own analytical system – biological neural networks.

3.3.2.2 Engineering

In the past ten years numerous engineering problems have used ANNs as a solution technique, civil engineering applications shall form the focus of this section.

There has been previous work to assess the ability of neural networks in modelling construction processes, all of which shall not be discussed here. Flood & Christophilos, 1996 presents a good discussion of how ANNs can be used as an alternative for discrete-event simulation, both static and dynamic networks are assessed in the context of earthmoving systems with reasonable success. Portas & AbouRizk model the productivity for concrete formwork tasks using a static neural network, a number of

different network architectures were considered but the feed-forward, back-propagation network was found to be the most favourable in terms of flexibility and the data availability. Further more the networks were found to fall short of the requirements on two fronts, firstly the network produced reasonable estimates the majority of the time but when the predictions were wrong they were considerably wrong. Secondly, the preference of a fuzzy output to a fuzzy problem was raised and an attempt made to adapt the network accordingly.

Chao & Skibniewski, 1995 uses neural networks as a method of estimating construction technology acceptability based on the new technologies performance attributes. It was found that promising results occurred when the poll size (training data set) was sufficiently large. Liu 1997 uses neural networks in the quality prediction for concrete manufacturing focusing on the ability of networks to learn from examples allowing the various physical processes that had previously been approximated by complex algorithms to be bypassed. Results were found to be better than previous methods of prediction. See also Yeh 1999.

Lu 2002 illustrates hybrid approaches using neural networks to improve the accuracy of Project Evaluation & Review Technique (PERT). The ANN in this case was used to fit beta distribution to the inputs of the model. Another interesting approach using combined techniques is discussed in Yu & Skibniewski, 1999 where genetic algorithms are combined with neural networks, in this case for constructability knowledge acquisition for construction technology evaluation. The author acknowledges the advantages of combining these techniques to produce optimum neural network solutions efficiently for most neural network function approximation techniques.

Other applications include scheduling problems (Leu et al, 1990), construction-site layout optimisation (Yeh 1995) and organisational effectiveness assessment (Sinha & McKim 2000).

Traffic incident detection (Jin 2002) and traffic flow predictions (Faghri & Hua 1995).

In maritime engineering and fluid mechanics neural networks have been used for tide forecasting (Lee & Jeng 2002), and eddy simulations (Sarghini et al 2003).

Further applications in geotechniques, initial design processes, river flow predictions, signal process defection detection, deep excavation forecasting and pipe condition assessment give an idea of the breadth of applications with in civil engineering, for more information on these applications and more see the Appendix.

3.3.3 Biological Neurons

It is useful to start with the biological inspiration behind the development of neural networks as way of an introduction. This brief account has been taken largely from Hagan et al 1996.

The figure below shows a diagrammatic representation of two typical human neurons. Each neuron in itself is very simple, the network however is very much more complicated consisting of some 10^{11} neurons each with 10^4 connections. There are four key terms represented by the figure. Dendrites receive signals from adjacent axons causing electrical signals that are transferred along dendrites into the cell body. If all the signals received by the cell body exceed a threshold value then the neuron is said to be triggered or fired resulting in a signal being released from the cell body along the axon to other adjacent neurons. The transfers of these signals from axons of one neuron to dendrites of another are known as synapses, the strength of which are determined by complex chemical interactions. A combination of the arrangement of neurons and the strength of the connecting synapses determines the ability of the network.

Figure 2: Schematic Drawing of Biological Neurons (Hagan et al, 1996)

The arrangement of the network changes continually through life, generally the network structure is altered more in early life while its connections are created, removed or strengthened in later life. In some senses the potential of the network is it's structure, developed while young but a specific application is controlled by the relative strength of various synapses that can be altered throughout life.

A final point of note raised by Hagan et al, 1996 deals with comparisons between artificial and biological neural networks. Clearly the biological neural

system is an incredibly complicated system, the complete understanding of which would answer not only many physiological and psychological uncertainties but potential also ancient philosophical questions on thought, consciousness and so forth. As described above this is partly due to the immense complexity of the network structure described above coupled with the far greater complexity of a single biological neuron compared with the very simple artificial neuron. In fact the individual biological neurons are much slower than the electronic replicas but due to the parallel nature of biological networks overall 'computing' power is much faster. Artificial neural networks (ANNs) share this property but due to their usual development on conventional computers this potential cannot be exploited.

It should now be clear that present artificial neural networks do not approach any where near the complexity of biological networks however they do borrow two key characteristics. In both cases the networks are made up of many very simple computational elements and the particular function of the network is determined by the connections between these elements.

3.3.4 General Principles

3.3.4.1 The Artificial Neuron

The artificial neuron has been partly introduced above; it is the processor in all ANNs and can be broken down into the key elements below. Figure 3 shows a diagrammatic representation of a typical neuron.

Figure 3: Single-Input Artificial Neuron (Hagan et al, 1996)

3.3.4.1.1 Weights & Bias

The vector inputs to a neuron (p) (either model inputs or the results of a previous neuron layer) are multiplied by the neuron's weight vector (w). These weights represent the strength of the connections. The weight vector must be the same length as the number of inputs. All inputs are multiplied by its corresponding weight vector entry and then summed. The neuron's bias (b) is also added to the summed weighted inputs to obtain the value required by the transfer function (n).

By altering the weights and biases the function approximated by the neuron (or network in multiple neuron cases) are changed. It is arriving at the optimum values of the weights and biases to best fit the required

function that forms much of the discussion on neural networks and is nearly exclusively achieved through some method of training. Training is discussed in section 3.3.6. It is normal to initialise the network with random numbers from between 0-1. The training alters these values until optimum performance is reached.

3.3.4.1.2 Transfer Functions

Each neuron can perform a very simple mathematical function, the nature of which is chosen during network design and determines the nature of the networks potential functions. It can be thought of as equivalent to the firing criteria in biological neurons. The simplest transfer function is a linear function (know as 'purelin' in Matlab) that simply outputs the summed input value (described above). Any network that uses solely linear transfer functions will be limited to linear relationships.

Many transfer functions are available, to model non-linear relationships the inclusion of log-sigmoid or tangential-sigmoid is required. If a network requires training using any variation of back-propagation it is essential that the functions used as transfer functions are differentiable. A table of all the most common transfer functions can be found in Hagan, 1996.

3.3.4.1.3 Inputs

An input vector is required for each run of a neural network, be it in training or during simulation once trained. The order of the input variables within the vector must remain constant. Any number of inputs can be included although the minimum required for accurate predictions should be sought. It is usual for all inputs to be linked to all neurons in the first layer of a network, however alternatives are possible. For example a network may have two input vectors, one attached to the first layer while the second is attached to perhaps the third layer. The inputs provide the network with the problem definition.

3.3.4.1.4 Output

Each neuron produces a single value output (a). The network output is a vector including the output from each neuron in the final layer, often just one. The output is once again chosen while training the network by specifying target values (usually observed cases). The network then uses the specified training technique to alter the weights and biases of the network to reduce the difference between the network prediction and the targets of the training set.

3.3.4.2 A Neuron Layer

Any number of neurons can be grouped together in a layer, each neuron produces a single output that collectively form the output for the layer. All the neurons in a layer receive the same inputs either directly from the input vectors (in the case of the first layer) or from the output vector of the

previous layer. However each neuron has its own weights vector that determine the proportions of the inputs to be summed and added to the bias before being mapped to the neurons output using the transfer function. Successive layers may be built up to create ever more complicated networks.

3.3.5 Alternative Architecture

3.3.5.1 Radial Basis Networks

Radial basis consist of two layers of neurons and function in a different manor to those networks looked at so far. The first layer consists of a number of neurons that are equal to the number of inputs each with a radial basis transfer function. The second layer is simply a linear layer with an equal number of neurons to the number of required outputs.

It is the first layer that differs in its behaviour from previously discussed neural networks. Each network in the first layer produces an output that is an indication of how close the input vector is to each neurons weight vector. If the input vector matches the neurons weight vector exactly the neurons output will be 1, if very far from matching the output will be 0. A value of 1 leads to the neurons second layer weights being passed to the output linear neuron. The input to the radial basis transfer function is the distance between the input vector and the weight vector (0 if equal leading to an out put of 1, large if very different leading to an output of near 0). The sensitivity of the function between these limits is defined by a variable know as ‘spread’. For example if the input vector is a distance of ‘spread’ from the weight vector then the neuron outputs a value of 0.5. It is by varying this value of spread that radial bias networks can be optimised.

3.3.5.2 Recurrent Networks

Recurrent networks are not considered in this study but are similar to feed-forward neural network with layers connected to previous layers, allowing the network to use previous outputs as future inputs. It is anticipated that recurrent networks could be used to model concrete placing and earthmoving systems by simulating the actual truck cycles. Further work is recommended on assessing the suitability of recurrent networks.

3.3.6 Training

3.3.6.1 Supervised Networks

The training function is presented with pairs of input and target vectors that define the required network behaviour. It then alters the weights and biases to produce outputs that are closer to the targets, after several iterations, optimum (or near optimum) values are reached and training terminated. The network is now ready to simulate outputs from new input data, normally a testing set is used first to assess the ‘real’ performance of the network.

Any performance learning consists of two main elements, the performance function chosen (normally mean squared error (MSE) or summed squared

error (SSE)) and the method chosen to search the multi-dimensional parameter space for an optimum solution (Bishop 1997). It is this second element that requires greatest thought. Fortunately many training functions are available within Matlab, and the most suitable for our purposes is discussed below.

The most commonly used method of training is back-propagation that can be used on any multi-layer network, whose transfer functions are differentiable. Generally a gradient descent algorithm is used that moves the independent variables (weights and biases) along the negative gradient of the performance function in an iterative manor – leading to an improved performance. Back-propagation determines the gradient by calculating sensitivities back through the network e.g. from outputs to inputs.

The training of a network is perhaps the most crucial stage of network development and determines the success of the function approximation. The network can only be expected to be able predict future responses from the area of the input space defined by the training set. As a result the training set should include a full a range of possible conditions as is possible, as neural networks are generally poor at extrapolation. With larger networks the training procedure can take some time. Fortunately several faster training algorithms have been developed to minimise the inconvenience caused by lengthy waiting times, the method chosen for use in this study is known as the Levenberg-Marquardt algorithm (or *trainlm* in Matlab) and is one of a number of variations on the Newton method. A detailed account of Levenberg-Marquardt back-propagation is given in Hagan et al, 1996 and shall not be repeated here. Essentially the network is initialised with random values for all weights and biases and the initial error is computed from the training data set. Sensitivities for the various variables are calculated and used to alter their values in an attempt to improve the performance. After a number of iterations a minima is found and the training either achieves its goal or is terminated.

3.3.6.2 Unsupervised Networks

In unsupervised learning target values are not required for training. The algorithms alter the weights and biases based on the training set of only the inputs. Unsupervised learning can be used with good results in pattern recognition and vector quantization as the network learns to group the presented input patterns into discrete categorises. For details on specific unsupervised learning algorithms see (Hagan et al, 1996 – 13.1-16.47). Unsupervised learning will not be considered further.

3.3.7 Testing & Validating

The performance of a network should not be measured on the performance value arrived at during training just as in regression the data used to arrive at the final equation can not be used to assess the success of the model. Unseen data must be presented to the network and its ability to predict these cases accurately will determine its performance.

Ideally three sets of data should be available training, testing and validating. The limitations of data led to the combination of the latter two into a single set.

3.3.8 Over-fitting

An overly complicated network for the required function to be approximated may give very good performance results during training but when a testing set is used to estimate the performance of the network the result may be much poorer than simpler networks. The extra complexity in the network has allowed the network to fit to the training data better than a simpler network may (Bishop 1997). However especially for smaller data sets this may well deviate from the general function that is trying to be derived. When unseen data is presented to such a network major discrepancies may be seen between predicted and target values. This phenomena is known as over-fitting, the function has become two specialised, trying to fit the training data too closely, a method is required that will prevent this and encourage generalisation.

3.3.9 Generalisation

3.3.9.1 Size Reduction

The natural solution to the problem discussed above is the reduction in network size. Since the over-fitting is caused by excessive network complication a reduction in size reduces this and hence the networks ability to over-fit. The question becomes, how big should I make a network for the function approximation required? Clearly there is no absolute answer and various network sizes should be tried with training and testing performances noted.

3.3.9.2 Regularisation

Regularisation actually alters the performance function (the function that drives the training algorithm) in an attempt to reduce the potential over-fitting. The performance function includes the mean square error of the network's outputs (mse), as before, but in addition the mean of the sum of squares of the network biases and weights (msw) are also included. The new performance function is included below:

$$\text{Reg. P.F.} = \gamma mse + (1-\gamma)msw$$

γ is known as the performance ratio and can be automatically optimised if the weights and biases are assumed to be random variables with specified distributions using statistical techniques – again a highly complicated procedure that shall not be explored in detail here.

In effect the inclusion of the mean of the sum of squares of the network's biases and weights encourages the simplest possible solution (by keeping the sum of the independent variables to a minimum) and hence reduces the likely hood of over-fitting. As a result the output from this training function includes an estimate of the number of independent variables required to successfully approximate the function under investigation, this can be particularly useful in the early stages of network design when

choosing the network size which is otherwise difficult. A Bayesian regularisation technique has been used in this study (*trainbr* training function in Matlab). See Ashraf et al (2002) & Vila et al (2000).

3.3.9.3 Early Stopping

Monitoring the performance of the network on a second data set during training can also control over-fitting, the training is prematurely stopped when this performance reaches an optimum value. Early stopping is a good solution but requires a further data set to assess the 'real' performance of the network.

3.3.10 Fuzzy Networks

The networks discussed so far represent standard functions in the sense that for a given input vector there is a specific determinable output, clearly this is not the case for concrete pouring, in as much as the same site conditions could in practice lead to different productivity rates due to the stochastic nature of the process. An exact prediction of productivity is unrealistic and an indication of the possible range of productivity is potentially far more useful. Neural networks can be adapted to produce a histogram of productivity probability. The productivity or output space is divided into a number of zones rather than a single value. The probability of the systems productivity falling within each zone is represented by a number between 0 and 1, the sum of all zone probabilities should not exceed 1. Portas, AbouRizk, 1997 deal with zone creation and representation in some detail and should be referred to for further detail.

This requires only two changes to the feed-forward networks discussed so far. Firstly to produce an output for each zone the network must have an equal number of neurons in its final layer to the number of zones, each neuron producing the productivity probability for a particular zone. The second change concerns the target values that are required for network training. The observed productivity values must be transformed into a probability histogram. In previous cases (Portas, AbouRizk, 1997) fixed values were used to represent the probability for the zone in which the productivity falls and its adjacent zones. It was felt that a method that would better represent the position of the actual productivity in the output space would potentially allow the network to generalise better. A normal distribution centred on the observed productivity with varying standard deviations was considered a better way to produce target probability values for each zone.

3.3.11 Limitations

Neural networks are an extremely powerful tool, still in its infancy when compared to its potential. The use of serial computers to develop neural networks clearly reduces the networks speed. The complexity of determining global minima in large multi-dimensional space still limits the ease of developing complicated networks. Finally, as mentioned throughout this report, the data requirements of neural networks even for simple networks like those included later are substantial and directly effect the accuracy of the final

model. In short the present limitations are slowly being eroded as a better understanding of the field is developed.

4 Procedure

Can artificial neural networks be used to accurately predict the productivity of construction processes? Assuming they can, investigate their potential by developing a prototype network. With this question in mind, the procedure presented here was developed in an attempt to demonstrate the breadth available when using neural networks for this purpose. The procedure also took into account the data set available; while sufficient to demonstrate the abilities of neural networks they would not allow the development of a reliable final model. Therefore wherever possible, general trends and results of interest have been highlighted rather than being solely focused on an optimal solution.

4.1 The Data Set

4.1.1 Date Source

The data set available for this study consisted of historical data collected previously for modelling construction process using techniques other than neural networks. As a result we were limited by the amount of data available and the parameters measured. The data used here falls into two categories;

4.1.1.1 Concrete Pouring Data

All the concrete data was collected by observation from projects near Edinburgh in 2000. The construction of four different wastewater treatment plants was used as sources namely; Fraserburgh, Nigg, Hatton and Peterhead. In total 239 different pour events were available with between 2 and 49 loads recorded in each, the range of total duration of pour events is between 1 and 14 hours.

4.1.2 Format

The data was originally of a form where each pour was contained in a separate excel file. In most cases the required data was available for each pour, any records missing data or of a corrupt nature were removed from the data set prior to the neural network investigation. All pours containing the correct information were collated in a single spreadsheet to ease data handling.

4.1.3 Accuracy & Reliability

As the motivation behind the data collection was initially research based, the accuracy of measurement can be assumed to be acceptable. However care must be taken when using this data as representative of concrete pouring in general, this cannot be assumed. All the site locations are local to Edinburgh limiting areal variation in the data set. Furthermore only data associated with the construction of wastewater treatment plants are considered, again restricting the breadth of relevance of the data set.

4.1.4 Limitations

The limitations are two fold as described above i.e. the size and the specific project type of the data set. In developing a generic concrete pouring working model using neural networks to be used in industry these would prove to be major pitfalls, however for the purpose of determining if neural networks can

play a part in construction process modelling the effects of the limitations can be minimized.

4.1.5 Data management

The large amount of data and the data fluidity required in generating various input, output and target combinations and permutations required in training and testing the networks lead to the creation of a number of macros with in Excel. The summary data for each record was collected in a single spreadsheet. ANNs tend to react far more favourably while handling data in the range of –1 to 1. Clearly negative values have no real meaning when dealing with concrete pours so the input and target data were reduced to values of between 0 – 1. The data was split into training and testing/validation data in the proportions 80:20 in a random fashion. Leaving 191 records to be used to train the network and 40 records making up the testing and validation sets. With a larger total data set the proportions would be split more evenly, ideally 50:50. A program was developed that created text files for use with Matlab - the inputs and outputs were selected from a list on a userform and the destination entered.

4.1.6 Choice of Inputs & Targets

The choice of target was fairly obvious as it was productivity that was of interest to us. Duration was also included as a possible target, as productivity can be determined from this using cumulative volume that is supplied as one of the inputs, in essence saving the network a step. Table 1 below also includes a number of multiple targets; these cater for the fuzzy networks discussed in 3.3.11.

The inputs required more thought; clearly the waiting, positioning, pumping and inter-arrival times would be required. It was decided that average times should be used, as these would be the easiest to estimate for a particular pour in the future. Slump was included in all three input sets as it was the only indication available of concrete variability. The number of trucks was included in input set A along with average truck volume and cumulative volume to provide the network with the information required to determine the total size of pour and the number of loads required. In input sets B & C the number of trucks on the job was omitted and instead the number of loads and average truck volume were included to provide the same information. Input set A & C included month (0.1 (Jan) – 1.0 (Oct)) as a crude indication of season. It was hoped that weather data could be obtained before the investigation began, however it was not included in this study but is recommended that it should be in any further work, perhaps two inputs, one rainfall and the other temperature.

The number of input/output pairs have been kept to a realistic minimum to reduce confusion during the study, however a few differences have been included for specific reasons. Input set B has no measure of season, while sets A & C are provided with a measure of the month, the difference between results obtained from the different sets will help to illustrate the importance of a seasonal dependence. Set A is the only set to include the number of trucks, while sets B & C have only a measure of the number of loads, in other words

sets B & C have no way of distinguishing between 10 trucks each delivering 4 loads or 2 trucks delivering 20 loads each.

				Inputs										Targets				
	Filename	No. of Inputs	No. of Outputs	Month	No. of Trucks on Job / 10	Average Truck volume / 10	Cumulative Volume / 1000	Average Inter-arrival	Average Wait	Average Position	Average Pump	Average Slump	Number of Loads / 100	Overall duration	Productivity	Productivity (0-1)	No. of zones if appropriate	SD if used
Input Set A	AInp9Dur	9	1	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y				
	AInp9Pr	9	1	Y	Y	Y	Y	Y	Y	Y	Y	Y			Y			
	AInp9Pr1	9	1	Y	Y	Y	Y	Y	Y	Y	Y	Y				Y		
	AInp9PrFz	9	7	Y	Y	Y	Y	Y	Y	Y	Y	Y			Y		7	
	AInp9Pr01Fz	9	7	Y	Y	Y	Y	Y	Y	Y	Y	Y				Y	7	
	AInp9PrFz02	9	7	Y	Y	Y	Y	Y	Y	Y	Y	Y			Y		7	0.02
	AInp9Pr1Fz02	9	7	Y	Y	Y	Y	Y	Y	Y	Y	Y				Y	7	0.02
	AInp9PrFz03	9	7	Y	Y	Y	Y	Y	Y	Y	Y	Y			Y		7	0.03
	AInp9Pr1Fz03	9	7	Y	Y	Y	Y	Y	Y	Y	Y	Y				Y	7	0.03
	AInp9PrFz04	9	7	Y	Y	Y	Y	Y	Y	Y	Y	Y			Y		7	0.04
	AInp9Pr1Fz04	9	7	Y	Y	Y	Y	Y	Y	Y	Y	Y				Y	7	0.04
Input Set B	LoadsInp7Dur	7	1			Y		Y	Y	Y	Y	Y	Y	Y				
	LoadsInp7Pr	7	1			Y		Y	Y	Y	Y	Y	Y		Y			
	LoadsInp7Pr1	7	1			Y		Y	Y	Y	Y	Y	Y			Y		
	LoadsInp7PrFz	7	7			Y		Y	Y	Y	Y	Y	Y		Y		7	
	LoadsInp7Pr1Fz	7	7			Y		Y	Y	Y	Y	Y	Y			Y	7	
	LoadsInp7PrFz02	7	7			Y		Y	Y	Y	Y	Y	Y		Y		7	0.02
	LoadsInp7Pr1Fz02	7	7			Y		Y	Y	Y	Y	Y	Y			Y	7	0.02
	LoadsInp7PrFz03	7	7			Y		Y	Y	Y	Y	Y	Y		Y		7	0.03
	LoadsInp7Pr1Fz03	7	7			Y		Y	Y	Y	Y	Y	Y			Y	7	0.03
	LoadsInp7PrFz04	7	7			Y		Y	Y	Y	Y	Y	Y		Y		7	0.04
	LoadsInp7Pr1Fz04	7	7			Y		Y	Y	Y	Y	Y	Y			Y	7	0.04
Input Set C	LoadsInp8Dur	8	1	Y		Y		Y	Y	Y	Y	Y	Y	Y				
	LoadsInp8Pr	8	1	Y		Y		Y	Y	Y	Y	Y	Y		Y			
	LoadsInp8Pr1	8	1	Y		Y		Y	Y	Y	Y	Y	Y			Y		
	LoadsInp8PrFz	8	7	Y		Y		Y	Y	Y	Y	Y	Y		Y		7	
	LoadsInp8Pr1Fz	8	7	Y		Y		Y	Y	Y	Y	Y	Y			Y	7	
	LoadsInp8PrFz02	8	7	Y		Y		Y	Y	Y	Y	Y	Y		Y		7	0.02
	LoadsInp8Pr1Fz02	8	7	Y		Y		Y	Y	Y	Y	Y	Y			Y	7	0.02
	LoadsInp8PrFz03	8	7	Y		Y		Y	Y	Y	Y	Y	Y		Y		7	0.03
	LoadsInp8Pr1Fz03	8	7	Y		Y		Y	Y	Y	Y	Y	Y			Y	7	0.03
	LoadsInp8PrFz04	8	7	Y		Y		Y	Y	Y	Y	Y	Y		Y		7	0.04
	LoadsInp8Pr1Fz04	8	7	Y		Y		Y	Y	Y	Y	Y	Y			Y	7	0.04

Table 1: Combinations of Inputs & Outputs.

4.2 NN design methods

4.2.1 Available 'software'

The software to be used for the investigation was chosen from one of three options; specific neural network packages, development of our own neural network using available languages such as visual basic or the neural network add-on for Matlab.

4.2.1.1 Why Matlab?

It was decided that the third option would be used for a number of reasons. Matlab was available and partly known to the authors and access to the add-on 'Neural Network Toolbox 4' was possible. It was considered to be more flexible than most specific neural network packages while simpler and therefore allowing greater diversity than writing our own networks. Matlab also offered the capability of producing the various graphical outputs required to explain the methodology and present the results as necessary.

4.3 Developing NN models

4.3.1 Familiarisation

An initial familiarisation session allowed the various theoretical neural network concepts to be identified and altered, demonstrating their great versatility. The session also served as an introduction to the various commands required for network manipulation within Matlab. Quite quickly a four layer neural network had been developed with 10, 10, 5 and 1 neurons in layers one to four respectively and transfer functions of logsig, tansig, logsig and linear respectively. Using the training data set a means square error of 0.086 was achieved illustrating the ability of neural networks to at least fit to data successfully very early in the investigation.

4.3.2 Training

The same 80% of the data was used in the training of each network. The data consists of two matrices (a matrix and a vector in cases of only one output), each 181 records long. The input matrix varies in size 9, 7, 8 with input configurations A, B, C respectively. The output matrix is size of the number of outputs, normally one but changes to seven using a fuzzy network to predict a probability histogram as an output.

In general the trainlm (Section 3.3.6) function was used as it provides good and quick results. In the cases of the larger networks the trainbr (3.3.9.2) function is used to reduce any over training that may lead to poorer real performances. Early stopping has also been included but due to the very limited size of the data set proper validation could not be completed, as it would require a further unseen sub data set. The training was generally set for 100 epochs, which in most cases allowed the majority of the improvement in performance to be achieved before this epoch limit was reached. In the fuzzy network investigation a 50 epoch limit was used to limit the time taken to train the networks.

4.3.3 Testing

The testing data set is the remainder 40 records. Again the same data is used in each case. The testing data set was chosen randomly from the full data set available initially. Each trained network tested was simulated using the testing data input matrix as the input while the output was recorded and saved. It is from these results that the mean square error and sum squared, referred to in Section 5, come from as well as the predicted values in Section 6.

4.3.4 Feed-Forward Network Investigation

Initially two, three and four layer networks were investigated. However four layer investigations were suspended partly due to the success of the two and three layer networks and partly due to the difficulties in four-layer analysis with the computers available and their excessive computational needs.

4.3.4.1 Size

SZ	Layer 1	Layer 2
1	20	1
2	19	1
3	18	1
4	17	1
5	16	1
6	15	1
7	14	1
8	13	1
9	12	1
10	11	1
11	10	1
12	9	1
13	8	1
14	7	1
15	6	1
16	5	1
17	4	1
18	3	1
19	2	1
20	1	1

Table 2: Size arrangements for the two-layer networks

SZ	Layer 1	Layer 2	Layer 3
1	20	20	1
2	20	16	1
3	20	12	1
4	20	8	1
5	20	4	1
6	16	20	1
7	16	16	1
8	16	12	1
9	16	8	1
10	16	4	1
11	12	20	1
12	12	16	1
13	12	12	1
14	12	8	1
15	12	4	1
16	8	20	1
17	8	16	1
18	8	12	1
19	8	8	1
20	8	4	1
21	4	20	1
22	4	16	1

23	4	12	1
24	4	8	1
25	4	4	1

Table 3: Size arrangements for the three-layer networks

A number of different size arrangements were decided upon including networks considered too simple and others excessively complicated. For three layer networks some arrangements were included with more neurons in the second hidden layer than the first and vice versa. The sizes used in the initial investigation are included in Tables 2 & 3 above. The column labelled 'SZ' contains the index number that will be used to refer to that size arrangement.

4.3.4.2 Transfer Functions

The transfer functions included in this investigation are linear (L), log-sigmoid (S) and hyperbolic tangent sigmoid (T). The original transfer function configuration for each investigation is a network with all linear transfer functions. The two-layer investigation includes all possible transfer function combinations. The three-layer investigation assumes the third layer of one neuron in most cases to consist of neurons with linear transfer functions. The column labelled 'TF' contains the index number that will be used to refer to that transfer function arrangement.

TF	Layer 1	Layer 2
1	L	L
2	L	S
3	L	T
4	S	L
5	S	S
6	S	T
7	T	L
8	T	S
9	T	T

Table 4: Transfer function arrangement for two-layer networks.

TF	Layer 1	Layer 2	Layer 3
1	L	L	L
2	L	S	L
3	L	T	L
4	S	S	L
5	S	L	L
6	S	T	L
7	T	T	L
8	T	L	L
9	T	S	L

Table 5: Transfer Function for three-layer networks

4.3.4.3 Trials

Each time a specific network configuration is initialised the effect of retraining may be entirely different. Resulting in good reason for the various permutations to be repeated. Generally a 'one off' poor performance from a network, perhaps due to it being 'trained' to a local minimum, can be spotted from adjacent results.

Programs were developed within Matlab that loads the Size and Transfer Function matrices from Tables 2 – 5 above and produces a network for each size – transfer function permutation. The training performance is then noted (using the training method specified in the program) and a predicted testing set simulated using the new network is also recorded.

The two important variables that are created each trial are PerRcd (2D) and Test(4D). PerRcd contains the training performance of each permutation, Test contains the simulated results from each network, referenced as follows Test (Run, 1:n, TF, SZ) where 'Run' distinguishes between different data sets and/or training functions (dependant upon the program) and n is the number of records in the testing set.

The results are to be gathered in four separate investigations, firstly the initial investigation that is designed to get an idea of the sensitivity of the networks to the many variables. The second investigation is aimed to develop the findings of the initial investigation by further developing successful networks. The third investigation involves assessing the suitability and advantages of using a fuzzy output approach and finally the fourth investigation will look at radial bias networks and how they might be used.

4.3.4.4 Approach/Trial Evaluation/Appraisal

The real performance of a network is its ability to estimate unseen data. The testing set was used to create simulated predictions of productivity for each record in the testing set for each trained network. From these the mean square error (MSE) could be calculated and used as a measure of the effectiveness of the network in question. The sum of squared errors is also used where more appropriate. A plot showing observed and predicted results for the testing data set will ultimately be used to assess the effectiveness of a network to estimate productivity from the inputs included.

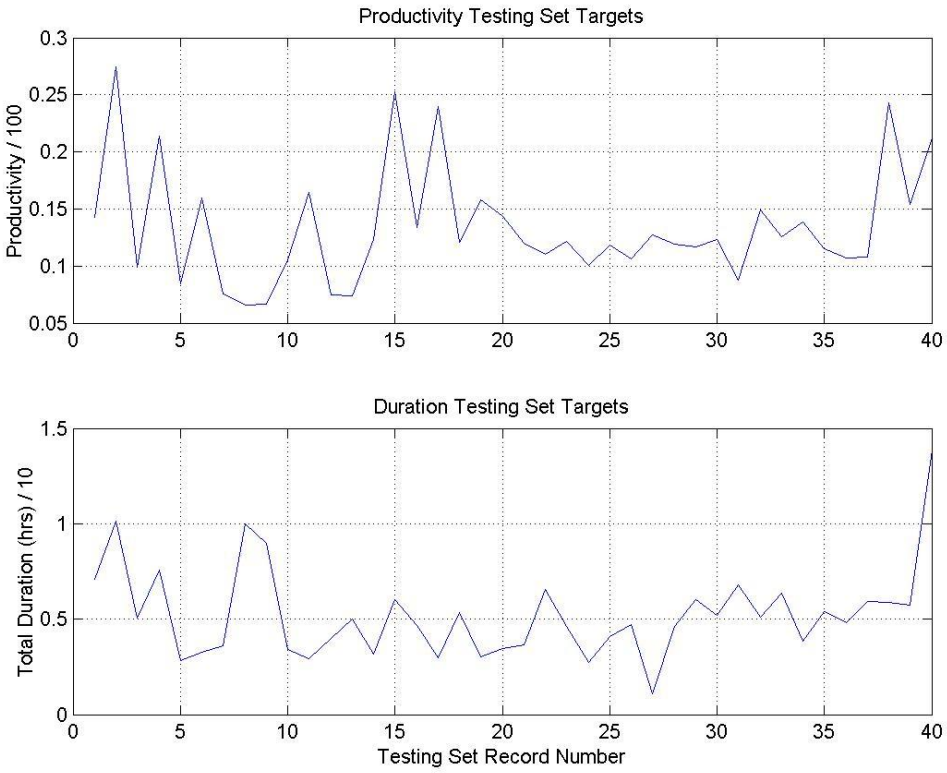


Figure 1: Observed Results (Targets) of the Testing Data Set.

4.3.5 Radial Bias Investigation

The radial bias investigation was relatively straight forward due to the reduced number of variable options. The number of inputs determines the size of the network and the 'spread' is varied to achieve optimum performances. Again all six data pairs were used in the investigation to allow comparisons with the feed-forward techniques.

4.3.6 Analysis

The analysis of an individual network involves assessing the fit between the predicted output from the testing set and the observed actual productivities or durations. The MSE may give false impressions for example if there is one poor result the MSE may be large for a network that fits the rest of the data very well. The best approach to assess the effectiveness of the network is to see a plot of the actual & predicted productivities of the testing set (See Section 6). For the fuzzy networks these had to be represented by colour plots, rather than the more conventional line plots, to represent the extra dimension as clearly as possible.

4.3.7 Sensitivity

The sensitivity of any developed network to changes in one or more of the model inputs can be investigated once it has been trained. A Matlab program was created that generates a series of input vectors with the chosen input variable varied through a specified range while maintaining the other inputs at a constant value, usually equivalent to the mean of the training set. It is possible to create a matrix expressing two dimensional (two varying inputs)

variation, using the chosen predictive model the response variation can be seen over two dimensions. Thus some idea of the parameters that drive any particular model can be sought.

4.3.8 Fuzzy Networks

Rather than try to predict an exact value for the productivity the output range can be broken down into zones and the probability that the productivity will fall in to each zone estimated from the input data. This builds in flexibility to the system as in real situations it is perfectly plausible that exactly the same input conditions could result in two different productivities.

The fuzzy network investigation includes three output sets all created using different productivity standard deviations. As in Portas & AbouRizk, 1997 the zone containing the observed productivity was given the highest probability and the adjacent zones a smaller proportion and so forth. It was decided that rather then use set values the probability in each zone should represent more the actual position of the records productivity not purely which zone it falls within. It was assumed that the probability distribution for each records productivity followed a normal distribution centred on the productivity value with a standard deviation of 0.02, 0.03 and 0.04 $m^3/(hr \text{ X } 10^2)$ respectively. Resulting in higher probabilities in the adjacent zone closest to the observed productivity value.

4.4 Model Variation Methodology

4.4.1 Architecture

Two of the four common neural network architectures have been considered here; the feed-forward and radial basis architectures. Recurrent networks and learning vector quantisation networks are not used in this study but should also be explored as further work in assessing the suitability of neural networks for modelling construction processes. Feed-forward and radial basis networks are two different arrangements of neurons with different transfer function. The design of the feed-forward networks was varied considerably as discussed above, two & three-layer networks were included.

Radial basis networks were considered for all input–output pairs with variations to the spread value made to find the optimum network in each case.

4.4.2 Network Size & Transfer Function

Section 4.3.1 contains the tables listing the various sizes and transfer function arrangements to be looked at. A simple program was used to create networks for every permutation of SZ & TF, each network was altered to except the required number of inputs for each input data set. For each set and every different output the network was trained, the performance recorded, and then used to simulate a set of predictions for the testing set – these were also saved before the network was prepared for the next input-output pair. Clearly such a method generates a large number of results. These can be seen largely in Section 11 and the results section (Section 5) is dedicated to explaining the variations in performance of the array of networks.

5 Results

The results consist of the various network performances from both training and testing data sets. The training performances represent how well the networks have been tailored to fit the training data. The testing performances indicate how the trained networks predict the output of an unseen data set; these values can be used to estimate the relative success of the different networks.

The results are presented in four sections corresponding to the three investigations used in this study, namely; The Reconnaissance, Further, Fuzzy & Radial Basis studies, this is the chronological order and is adopted here as although individual studies they are each developments of their predecessors.

The key figures here are colour plots representing the performance of the different networks calculated from the testing data set. The colour axis scales have been kept constant across comparable figures. They can be thought of as representing relative performance. The dark blue (Min. = 0) represents the best performance while a dark brown (Max. = specified on plot) illustrates an unsuccessful network attempt, for the definition of the full colour transition see Section 11. Section 11 also includes full graphical representation of the results. Where it is appropriate to compare a number of different plots references to Section 11 are made to simplify reading, the pages of Section 11 can be removed individually to allow ease of comparisons.

5.1 Initial Reconnaissance into Size & Transfer Function

As explained above the initial investigation was a reconnaissance mission to gain an appreciation of the sensitivity of network performance in relation to a number of factors, namely; the effect of different inputs and outputs, number of neurons, two or three layers and varying transfer function arrangement.

5.1.1 Network Size

5.1.1.1 Two Layer

Section 11, Figure ‘2 Layer: - 1.1’ and ‘2 Layer: - 1.2’ demonstrate the ability of the various networks to map the training outputs given the training inputs for both productivity and duration respectively. The effect of network size can be seen much clearer in the two layer results, as the increase in size is constant from size number 1 to 20. Looking at the figures referred to above it can be seen that all sizes above number 15 have been able to fit the data as well as each other (excluding the occasional network that fails to fit the data adequately due to poor parameter space optimisation). Sizes 17 & 18 generally seem to give good results for the productivity but appear to be slightly too small for optimum results in predicting duration. The two sizes that consistently give poorer performance are the two smallest, sizes 19 & 20, two neurons and four neurons in the first layer respectively.

It can also be seen that the results from Input Set B corresponds to the worst performance for smallish networks and Set A to the best

performance for small networks. If the size of the network is thought of as the number of independent parameters (biases and weights), the number of inputs play a significant role, especially in a two layer network with so few neurons. In other words Input set B will result in a smaller network than Input set A as the number of inputs are 9 & 7 respectively. This trend is more easily seen in the duration data.

Section 11, Figure '2 Layer: - 1.3' and '2 Layer:- 1.5' illustrate the effect of network size on the performance rather well. Fig. 1.3 Input sets A & C show a very stable constant performance value until size number 15 where the performance value increases almost exponentially. The other input set, B, follows the same general trend but with greater variation in the larger networks. Fig 1.5 seems to follow a slightly different pattern a relatively constant performance value until a network size of 16 or 17 is reached when again a sharp increase in the performance value.

Section 11, Figure '2 Layer: - 1.7' is the most important figure in assessing the success of the various networks. The performance is now based on data that the network has not seen during training and hence gives us a 'real' indication on the ability of the network to predict productivity. The best network from runs one and two have been chosen and their performaces are displayed in these figures.

The opposite trend to the training performance colour plot can be observed with the best results being produced by the smallest networks. The linear networks now produce better results than the majority of the non-linear networks. All the transfer function arrangements give good results for sizes 19 & 20. These results can be explained by the phenomena known as over-fitting see Section 3.3.8. The left centre plot shows a network achieving results as good as any with a SZ of 3 and TF of 9. This could represent a very well trained network as all other networks of this size struggle to predict unseen data with a similar degree of accuracy. The dark blue squares represent a mean square error of approximately 0.0001, which translates to a root mean square error of about 1 m3/hr. This can definitely be thought of as a successful modelling technique if the plots of observed and predicted show a correspondingly accurate correlation.

Section 11, Figure '2 Layer: - 1.8' & '2 Layer: - 1.9' show a break down of the real performances by transfer function. The network size is represented as a series of coloured bars each representing the trend in performance for each TF. The general trend for productivity is high SZ high performance but low SZ very low performance, for duration a similar trend is seen but with indications that the smallest networks are not sufficiently complicated to compute total duration as accurately as a slightly larger network. The similarity that can be seen in the shape of the clusters represents the general size trend seen in Fig. '2 Layer: - 1.7'.

What is actually of concern to us is, of course, the minimum performance values, i.e. the networks that produce the best performance. Section 11, Figures '2 Layer: - 1.10 & 1.12' illustrates the minimum and mean performances of both productivity and duration with respect to size and transfer function arrangement. Fig. 1.10 shows a number of interesting results; for sizes greater than size number 17 the linear transfer networks generally provide the minimum performance values, especially for the input sets A & C. The network with the top performance is of size 20 for each input set again illustrating the problem of over-training but also indicating that accurate predictions of productivity can be produced from a very simple network. The mean results also demonstrate how much more reliable it is to use a network of suitable dimensions rather than apply an overly complicated design to a simple problem, with only sizes 17 – 20 having minimum and mean values of comparable magnitude. Fig. 1.12 illustrates the same information but for the duration networks, here Input sets B & C fair much better than set A. The variation of performance with size is far less for duration than it was with productivity and notably in each case size 20 performs worse than the preceding 5 or so sizes illustrating that a 'P – 2 – 1' network is becoming to small to predict durations accurately. The lack of over-training to the same extent as in the productivity results may be due to a more 'obvious' pattern for determining durations existing, allowing more of the networks to generalise successfully from the training data. Figure '2 Layer: - 1.7' illustrates this especially well for input sets B & C by the more even distribution of blue squares across the non-linear transfer function arrangement space.

5.1.1.2 Three Layer

When considering the trends in network size in the three-layer investigation it must be remembered that the size numbers follow the pattern presented in the table in section 4.3.4.1. i.e. networks do not necessarily increase in size as the size number increase, but rather five different second layer sizes are used for each of the five different first layer sizes. Figure '3 Layer: - 1.1 & 1.2' illustrates the ability of the 3 layer networks under investigation to fit the training data considerably better than the two layer networks were able.

If just the first three TF rows are considered (those with at least one of the first two layers as a linear layer) every 5 size numbers, starting at SZ 5, presents networks of poor performance. These refer to the last size number for each different first layer size configuration, in other words a small second layer of only 4 neurons.

The general trend of the smaller networks performing worst as seen in the two layer plots is repeated with nearly all the networks with only 4 neurons in the first layer performing badly with respect to mapping productivity values, the duration networks perform relatively well with the smaller networks with only SZ 24 & 25 producing the poorer results.

From Fig. '3 Layer: - 1.3 & 1.5' the pattern being described above can be seen clearer. Especially Fig 1.3, Run 1 plots, the plot can be thought of as five sections, each with a similar trend (such as was seen in two layer investigation) with an increase in scale on the preceding set of five. This can be seen to some extent with the duration figure as well, however in this case the performance is maintained at a constant level until the smallest second layer is used, i.e. the fifth, tenth, etc. size number.

Figure '3 Layer: - 1.7' shows us the real picture as far as the three layer networks are concerned for both productivity and duration. The scales in these figures allow a direct comparison with the two-layer investigation (Figure '2 Layer: - 1.7'). Clearly the inclusion of a second hidden layer increases the size of the network (no. of independent variables) to a greater degree than a two layer network with the same number of neurons. Similar results to the two-layer findings can be seen. Dealing first with productivity we see that the all linear network provide excellent results particularly for inputs A & C as can be seen clearly in Figure '3 Layer: - 1.8'. The first-layer linear networks with a small number of neurons in the second layer that produced the poorer training results prove to be some of the most successful networks in predicting the productivity of unseen data. A significant amount of over training can be seen in the larger networks (SZ = 1 – 19) there is however a number of these larger networks that perform well with the testing set, for example Input Set C, TF 6, SZ 6 – one of the largest networks used.

The duration results again show a dramatic improvement for all three data sets especially A & C. The solely linear network is no more accurate than the two-layer linear networks again suggesting that the function to estimate duration cannot be closely approximated to a purely linear function. Nearly all the sizes achieve reasonable results once again indicating that over training occurs less when training the network to predict overall durations, although the better results are still the smaller next work, unlike the two-layer investigation, including the smallest of all SZ 25.

5.1.2 Transfer Functions

5.1.2.1 Two Layer

Section 11, Figure '2 Layer: - 1.1' and '2 Layer: - 1.2' again show us the grid of network performances, perhaps the most obvious observation is the first three brown rows of each plot representing transfer functions 1, 2 & 3. Referring to the table in section 4.3.4.2 it can be seen that these first three transfer function arrangements all use a linear transfer function in the first layer, limiting the flexibility of the network. However it is interesting to note that, again disregarding the odd rogue network, all the linear networks reached a training performance of 0.0006, indicating that there is a single optimum solution to the problem when using linear transfer functions. Clearly the introduction of the more complicated log-sigmoid and tan-sigmoid transfer function allow a nearly infinite number of solutions

including many that may over fit the data considerably. The two layer initial investigation is the only investigation to deal with all transfer functions in the output layer (linear transfer functions were used in the final layer of all three layer networks to reduce the number of permutations to be explored) it is interesting to note that the variation in final layer transfer function has no effect on the performance of the networks when the first layer is made up of linear neurons. The other transfer function arrangements with a linear transfer function in the final layer are numbers 4 & 7, on close inspection of these figures these networks tend to give better results as they are approaching the limiting size, e.g. Input Set A, Run 2 on Fig. 1.1 show transfer function arrangement 7 as the only transfer function arrangement to give good results for networks using size number 20, the same figure with Input Set B, run 2 show similar results with TF 4.

Section 11, Figure '2 Layer: - 1.4' and '2 Layer: - 1.6' show the effect of transfer function arrangement again the results for productivity and duration are very similar, all be it on different scales – productivity will be focused on here. Input sets A & C both show a substantial improvement with a non-linear first layer and large networks. The variation in transfer function arrangement from 4 to 9 is minimal. The middle graphs (input set B) show a far greater variation with size than the other sets often with the performances of the largest networks inferior to that of smaller networks.

Section 11, Figure '2 Layer: - 1.7' shows once again the clear split between those TFs with a linear first layer and those with out. The linear networks have clearly all found the optimum linear relationship between the inputs and productivity prediction and give surprisingly good results only bettered by small networks and the occasional larger one. It is of interest to look at the duration data here as well, again the linear networks have all adopted the same approach but are considerably poorer than the non-linear networks, perhaps indicating a higher dependency on non-linear relationships in the case of duration as opposed to productivity.

Trends within the non-linear networks, with respect to TF are less obvious. Section 11, Figures '2 Layer: - 1.8' & '2 Layer: - 1.9' illustrates this rather better. The first three TFs produce a constant performance for both Productivity and Duration, far closer to the optimum performance for productivity than for duration. Concentrating on productivity, Fig 1.8 a TF of 7 seems to produce the lowest array of values with TFs of 9 and 5 also faring reasonably well.

Section 11, '2 Layer: - 1.11 & 1.13' shows us the minimum and mean network performance distributions by TF. Fig. 1.11 show the linear networks forming the highest values in the minimum plots yet the minimum values in the mean plots illustrating their limitations and yet their reliability. TF 7 produced the lowest results for data sets A & C, while data set B's minimum performance was from TF 9 with TF 7 a close second. TF 5, 6 & 9 scored well looking at the mean plots. Fig. 1.12 again

demonstrates that duration prediction is less suited to a linear approach as many of the other TFs produce an equal or lower mean performance, again very little to choose between the different non-linear TFs.

5.1.2.2 Three Layer

Figures '3 Layer: - 1.1 & 1.2' from Section 11 give away slightly more than their two layer counter parts. The all-linear network is as apparent as ever struggling to fit the data as accurately as the others. Those networks with a linear first layer perform badly with a small second layer as discussed in 5.1.1.2. There is one further observation worthy of note from these figures, namely; the relatively bad performance of TF 2, 3, 5 & 8, all the networks with a first or second linear layer. Interestingly unlike many of the patterns observed in the two-layer investigation all the three-layer trends hold for both the productivity and duration mapping.

Figures '3 Layer: - 1.4 & 1.6' (Section 11) illustrate this point further with TF 4, 6, 7 & 9 all producing good results, especially with large sizes. It also becomes clear that some of the values associated with TF 3, 5 & 8 are very competitive although others are quite large.

In many respects Figure '3 Layer: - 1.7' can be thought of in some respects as the positive, while Figures '3 Layer: - 1.1 – 1.2' as the negatives. The first-layer linear networks with the small second layer agree with this as described in 5.1.1.1 but also the poorer performances seen on TF 2, 3, 5 & 8 can be seen to be reversed in Fig. 1.7 transforming these TFs to the most successful. There are also more good performances scattered in the area normally associated with over fitting often falling in TF 4, 7 & 9.

The three-layer linear results can be seen to be very close to the optimum three-layer solution, and hence the optimum reconnaissance investigation solution, by studying Figures '3 Layer: - 1.7-1.8'. From the latter of these figures and Figure '3 Layer: - 1.11' TF 8 is shown to produce the best performance for input sets A & B, while TF 3 provides the best solution for input set B. The stability and reliability of the linear networks can be seen from the mean plots in Fig. 1.11.

The break down of the TF of the duration performance can be seen from Figures '3 Layer: - 1.9 & 1.13'. Fig. 1.9 shows how the majority of the non-linear TF networks out perform the all-linear networks for nearly each size. TF 3, 5 & 8 (again each with a first or second linear layer) seem to out perform their counterparts, once again input set B produces some of the best performances including the lowest 3-layer duration performance value using TF 2.

5.1.3 Input-Output Partnerships

5.1.3.1 Two Layer

Section 11, Figure '2 Layer: - 1.1' and '2 Layer:- 1.2' illustrate that as a general rule the networks explored perform much better when trying to

predict productivity values rather than duration values. It should be noted that duration was included in these investigation as it was thought of as potentially a way of saving the network unnecessary calculations, however it should also be noted that to convert the duration predictions to productivity values it is required to divide through by cumulative volume (on average 75 m^3) thus reducing the error and increasing the performance. However as the networks achieve such good results for productivity straight off, the focus of the rest of this investigation is on productivity predictions only.

The input sets have been discussed in section 5.1.1.1 with regard to their effect on network size. Other differences are difficult to distinguish, at this stage all three sets will suffice.

Section 11, Figures '2 Layer:- 1.3', '2 Layer:- 1.4', '2 Layer:- 1.5' and '2 Layer:- 1.6' all show a greater variation in the case of input set B. Set B is the only set of inputs that does not include the month or any other measure of season/weather this could explain the less settled response that is observed.

Section 11, Figures '2 Layer: - 1.7','2 Layer: - 1.8' & '2 Layer: - 1.9' tells us much about the relative use of the input sets especially regarding productivity. Looking at Figs 1.7 & 1.8, the linear network performances can be used to give an indication of the input set that best predicts the productivity, giving us a ranking of input sets of A, C, B. A & C produce very similar results suggesting perhaps that it is advantages to include 'month' as an approximate measure of season.

Sections 5.1.1 & 5.1.2 have both considered Section 11, Figures '2 Layer: - 1.10 – 1.13' and commented on the difference between input sets and between productivity and duration estimation.

5.1.3.2 Three Layer

Comparing Figures '3 Layer: - 1.7' and '2 Layer: - 1.7' the most notable advancement is in predicting duration from all three input sets especially A & C. In all three cases a performance of 0.01 or less has been achieved for virtually every network tested, save the all-linear network. It seems as though the effect of over-fitting has been reduced in the more complex three-layer networks contrary to what might have been expected, perhaps a three-layer network models the duration function better than a two-layer network.

The effect of the different input sets seems to have less of an effect in the three layer investigation, as essentially all the same information is being supplied a three layer network can compensate for the particular format of the input data rather better than the two layer networks can. However Figure '3 Layer: - 1.8' shows that the all-linear three-layer network of input sets A & C perform excellently, with the same network but using the

input set B they perform not quite so well. Looking at the same figure for the duration data Fig. 1.9 it can be seen that the all-linear networks performances do not vary between the input sets and are far higher than the performance of most other networks.

Figures '3 Layer: - 1.10 – 1.13' allow us to generalise about the effect of using the different input sets. Productivity results are covered in Fig. 1.10 & 1.11, while the average minimum results are lowest for input sets A & C, input set B produce some of the lowest individual performances. The duration results are somewhat different with input sets B & C clearly giving the better results. From these findings we can draw a hypothesis about the way in which the networks calculate the required output. Productivity is a rate concerned with the ratio of key values; many pours of different lengths could result in the same productivity. Duration however is particular to each pour size and is therefore more suited to the data presented in sets B & C that contain a stronger indication of total pour size.

5.1.4 Summary

The networks that resulted in the lowest performance during training and testing are listed in the tables below.

		Input Set A			Input Set B			Input Set C		
		TF	SZ	MSE	TF	SZ	MSE	TF	SZ	MSE
Productivity										
	Training	7	1	0.000007	5	3	0.000001	7, 9	2, 6	0.000007
	Testing	7	20	0.0001162	9	20	0.0001408	7	20	0.0001395
Duration										
	Training	4-9	1-15	0.0005	4-9	1-15	0.0005	4-9	1-15	0.0005
	Testing	5, 8	18, 19	0.0055	9	18	0.0042	5	17	0.0045

Table 1: 2-Layer optimum training and testing performances.

		Input Set A			Input Set B			Input Set C		
		TF	SZ	MSE	TF	SZ	MSE	TF	SZ	MSE
Productivity										
	Training	4,7	1,2	4.80E-07	7,9	3,11	8.60E-07	7	2	6.00E-07
	Testing	8	21	1.19E-04	3	15	1.80E-04	8	23	1.10E-04
Duration										
	Training	7	1	4.00E-07	6	1	2.50E-06	8,9	2	3.00E-06
	Testing	5,7,8	18,21,25	0.0013	3	10	0.0005	2,4,5,6,8	5,9,20,21,23,25	0.0007

Table 2: 3-Layer optimum training and testing performances.

5.2 Further Investigation into Successful Networks

From the findings of the previous investigation a second investigation was designed. The first problem to be solved was to prevent the over-training seen in section 5.1. As explained in section 3.3.9 there are three main techniques that can be used to improve generalisation and hence reduce over-fitting. Size reduction has already been included in the initial investigation with great

success, however regularisation & early stopping were included in the further investigation to see if further improvements could be made. The TFs and SZs were chosen from those that provided the most promising results in the initial investigation, the complete selection used are included in the tables below.

Tk2 No.	Layer 1	Layer 2	Recon. No
1	40	1	1
2	38	1	2
3	36	1	3
4	30	1	6
5	8	1	17
6	6	1	18
8	4	1	19
9	2	1	20

Table 3: 2-Layer Network Sizes for Second Investigation.

Tk2 No.	Layer 1	Layer 2	Recon. No
1	S	S	5
2	T	L	7
3	T	T	9

Table 4: 2-Layer Transfer Function Arrangements (TF) for Second Investigation.

Tk2 No.	Layer 1	Layer 2	Layer 3	Recon. No
1	20	20	1	1
2	20	16	1	2
3	20	12	1	3
4	12	20	1	11
5	4	20	1	21
6	12	4	1	15
7	4	12	1	23

Table 5: 3-Layer Network Sizes for Second Investigation.

Tk2 No.	Layer 1	Layer 2	Layer 3	Recon. No
1	L	T	L	3
2	S	S	L	4
3	T	T	L	7
4	T	L	L	8
5	L	S	L	9

Table 6: 3-Layer Transfer Function Arrangements (TF) for Second Investigation.

5.2.1 General Discussion

5.2.1.1 Two Layer

Section 11, Figure '2 Layer: - 2.1' shows the results of the training using two different methods to improve generalisation. The similarities between the data sets are improved as the individual networks flexibility have been

limited. From the regularisation plots it can be seen that TF 1 and the mid-sizes produce the best result while SZ 9 performs badly. The early stopping method favours the larger networks particularly of TF 2 & 3. These results can be seen clearer in Fig. 2.2 & 2.3 here the comparative performance of regularisation and early stopping can be seen with the early stopping method producing the lowest performance values.

As in the previous investigation it is how the training performances transfer to testing performances that is of real interest to us, the testing performances are show in the usual forms in Fig. 2.4 – 2.8. It should be noted that the early stopping results used the testing data set to terminate training when the testing set SSE was at a minimum and as a result a direct comparison between the regularisation values and the early stopping values is not entirely fair.

Again Fig. 2.4 can be compared directly to Figure '2 Layer: - 1.7' & '3 Layer: - 1.7' the improvement is clear, although some of the larger early stopping networks still produce some poor performances. As far as the regularisation networks are concerned those of particular interests are those networks that perform well with the training and testing data, for input set A & C TF 1 with SZ 5 & 7. Clearly many other of the networks also give very good results at this stage and more importantly the consistency of finding networks with good performances is greatly improved using both techniques. It is also worth noting that the results obtained from the previously poorer larger networks but using regularisation and early stopping are of the same magnitude as the best performances gained from size reduction as shown in the initial investigation.

Figure '2 Layer: - 2.6 & 2.7' allows us to see the effect of size while over-training is controlled. Fig 2.7 shows that the early stopping method still favours the smaller networks although the larger ones are now far more competitive. Fig 2.6 and the regularisation method have been able to keep the mean performances of the networks down to much closer to the minimum values illustrating high reliability.

Section 11, Figure '2 Layer: - 2.5' demonstrates the distribution of the network performances with transfer function. There is very little to choose between the various TF with all network performances being maintained at a similar level. TF 2 has a linear second layer and achieves the best performance for at least input sets B & C. Fig. 2.8 again illustrates the minimal difference that the TF selection makes in this case. TF 1 achieves the best results for input set C & input set B under early stopping.

	Input Set A			Input Set B			Input Set C		
	TF	SZ	MSE	TF	SZ	MSE	TF	SZ	MSE
Productivity Run 1									
Regularisation	1	4	0.0001299	2	7	0.0001105	2	5	0.000103
Early Stopping	-	-	-	-	-	-	-	-	-
Productivity Run 2									
Regularisation	-	-	-	3	7	0.000108	-	-	-
Early Stopping	1	3	0.0001026	-	-	-	3	7	0.0001309

Table 7: 2-Layer Further Investigation Testing Performances.

5.2.1.2 Three Layer

The training performances of the three layer networks used in the second investigation can be compared with the performances of the two-layer networks from the same investigation (Fig. '2 Layer: - 2.1'). Looking at the results in Fig. '3 Layer: - 2.1' the regularisation can be seen to have performed more poorly than in the two-layer study, while the large early stopping networks, particularly those with TF 2, 3 or 4, seem to perform relatively well.

From Figures '3 Layer: - 2.2 & 2.3' similar trends to those seen in the two-layer results become apparent, the larger early stopping networks performing best, while regularisation produces a more even distribution of successful networks with slightly higher performance values.

Fig. '3 Layer: - 2.4A' compares well to the original 3-layer results. The positive – negative transfer of colours is again appropriate here except for those networks that were unsuccessfully approximated to a suitable function – those marked with a brown square in both Fig 2.4A & 2.1. Over all, the regularisation seems to work best with TF 1, 3 & 5, while the size has little influence. The early stopping results favour the smaller networks with TF 1 & 5 producing most of the best performances.

Figure '3 Layer: - 2.5' illustrates well how both methods lead to similar consistent performances across most of the networks. The minimum values can be seen to come from input sets B & C with SZ 7 using regularisation. Fig. 2.8 shows the limited effect of varying the TF, however it can be seen that in most cases TF 3 & 5 produce either the most consistent or best results.

	Input Set A			Input Set B			Input Set C		
	TF	SZ	MSE	TF	SZ	MSE	TF	SZ	MSE
Productivity									
Initial Investigation	4	5	0.000119	1	6	0.0001802	4	7	0.0001096
Productivity Run 2									
Regularisation	-	-	-	2	7	0.0001025	3	7	0.0001063
Early Stopping	5	7	0.0001266	-	-	-	-	-	-

Table 8: 3-Layer Further Investigation Testing Performances.

The introduction of regularisation and early stopping improve the results slightly but more importantly increase the chance of converting a chosen network to approximate the function required.

5.3 Predicting a Productivity Probability Histogram

It was assumed that those networks best suited to predicting a single productivity value could adapt to predict the probability of the productivity arising in each of the seven zones, as a result no initial fuzzy investigation was carried out, rather those networks used in the further investigation were also used in the fuzzy investigation, of course altered to include seven neurons in the final layer. The size matrices are not included again as they are identical to Tables 4 & 6 in section 4.3.4 with the 1 in the final layer in each case replaced with a 7. Again Section 11 contains the plots referenced below for ease of plot access. However no direct comparisons can be made using MSE or SSE to non-fuzzy networks as the fuzzy networks output vectors rather than an individual number. The plots vary from those discussed above as they include results for the three values of standard deviation (SD) used to distribute the probability. A SD of 0.2 (refers to an actual SD of 0.02) relates to the majority of the probability being distributed to a single zone while an SD of 0.4 (refers to an actual SD = 0.04) leads to a distribution of probability across many zones, as a result the networks tend to predict higher SD distributions better as essentially it is asking them to be less exact.

5.3.1 Two Layer Networks

Figure '2 layer: - 3.1' illustrates a few key points quite effectively. The higher value standard deviation runs are trained with more success than the lower values. TF 2 seems to be the most apt, followed by TF 3. SZ 8 & 9 seem to be too small for accurate results suggesting that the function required for a fuzzy network is more complicated than that used to produce a single output result. The 'jump' in sizes between SZ 5 & 6 (from 30 to 8 neurons in the first layer) can be seen clearly for a SD of 0.2 and to a lesser extent in the other plots.

Figure '2 Layer: - 3.4' shows the results of the trained networks in predicting the testing set distributions. The higher the standard deviation of the probability distribution the better the results are (but the less useful). The vast majority of accurate networks use TF 1, despite its apparent shortcomings in the training. SZ 8 appears to be too small for most cases although coupled with TF 1 can predict the SD of 0.2 results remarkably well. The results discussed above, from Fig. 3.1, can be used with those from Fig. 3.4 to distinguish those networks that produces good results for both data sets, as these networks may generalise better. For example; Input Set C, SD 0.2 networks TF 1 SZ 1 may not generalise as well as TF 3 SZ3 because the latter performs better with the training data set (Note: the training data set is over four times the size of the testing set.)

	SD = 0.2			SD = 0.3			SD = 0.4		
	TF	SZ	MSE	TF	SZ	MSE	TF	SZ	MSE
Input Set A	1	1, 2, 3, 5, 7, 8	0.0003	1	2, 7	0.0001	1	5	0.0001
Input Set B	1	1	0.0000	1	2	0.0000	1	1, 4, 5, 6	0.0002
							2	7	
Input Set C	1	3, 5	0.0000	3	2	0.0001	1	3	0.0001
	2	6							

Table 9: Fuzzy Investigation, Best 2-Layer Network Testing Performances.

5.3.2 Three Layer Networks

Figure '3 Layer: - 3.1' generally shows a good ability for three layer networks to fit to the fuzzy data. Again the 'jump' in size from SZ 4 to SZ 5 can be seen to have a large effect. TF 2 & 3 seem to produce the best training performances due largely to the lack of a linear layer as one of the first two layers.

Figure '3 Layer: - 3.4' shows similarities to Fig. '2 Layer: - 3.4', not lease in the poor results in predicting the SD 0.2 probability distributions from input set A. However a number of the larger networks using input sets B & C produce very good results even for a SD = 0.2, summarised in the table below. The other network of note is TF 5 SZ 7 that seems to predict distributions with a SD 0.2 using input set B surprisingly well.

	SD = 0.2			SD = 0.3			SD = 0.4		
	TF	SZ	MSE	TF	SZ	MSE	TF	SZ	MSE
Input Set A	2	4	0.0000	1	7	0.0001	1	2, 3	0.0002
	3	6		2	2		3	5	
	5	6		5	3		4	1	
Input Set B	1	2	0.0000	2	1, 3, 4	0.0001	3	4	0.0001
	2	2, 3							
	3	4		3	2, 7		5	5	
	5	8							
Input Set C	2	2	0.0000	2	2	0.0000	5	7	0.0001

Table 10: Fuzzy Investigation, Best 3-Layer Network Testing Performances.

5.4 Radial Basis Network Investigation

As discussed above the design of radial bias networks is very simple - the major advantage of radial bias networks. The only decisions required during their design are input and output choices and the spread value. Two investigations were carried out one attempting to predict productivity and the other duration. Figure 1 below illustrates the effect of input sets A, B & C in trying to predict productivity using radial basis networks.

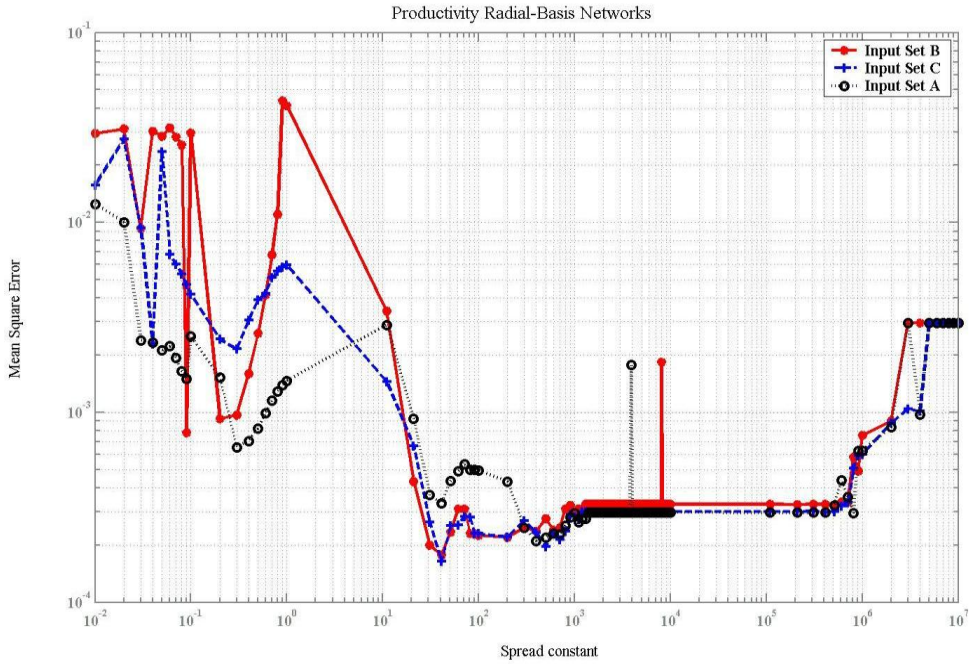


Figure 1: The effect of spread on Productivity Radial Basis Networks

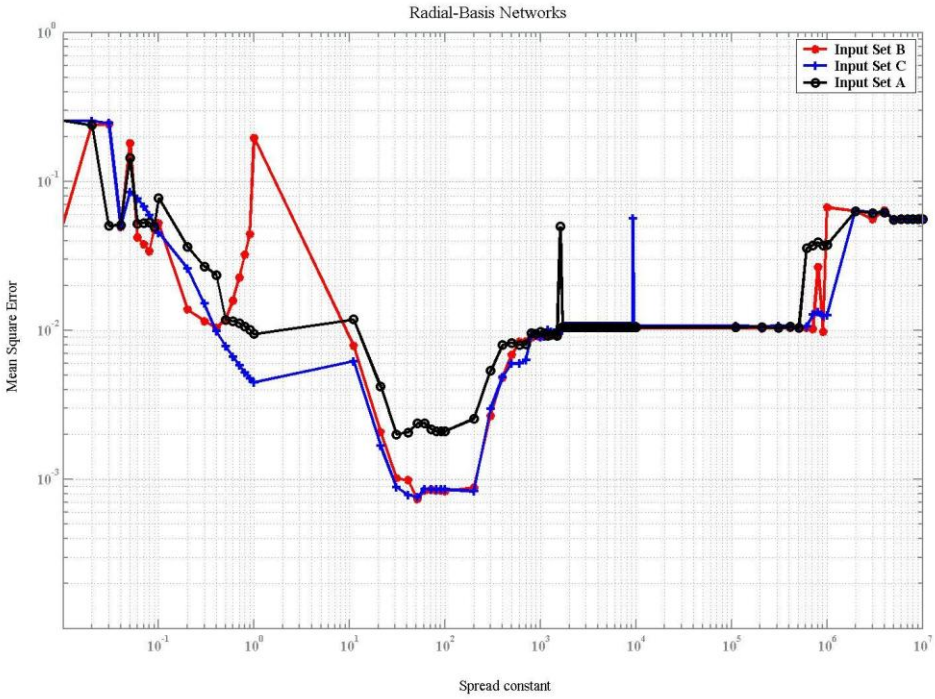


Figure 2: The effect of spread on Duration Radial Basis Networks

To demonstrate the effect of spread requires a log-log plot of MSE against the spread constant. Input set B & C reach a very similar MSE minimum at a spread of approximately 40, while input set A performs less well with a minimum MSE at a spread value of about 400. Radial basis networks are well known for performing better with fewer inputs and perhaps the inclusion of the month in input set C just over compensates for the extra input over set B.

Once again input sets B & C produce very similar results with minimum MSE values achieved with a spread of about 50. Input set A again performs considerably poorly in comparison. As was found with the feed-forward networks the productivity networks produced the lowest MSE values overall. The radial basis chosen, as the most successful was therefore the Productivity, Input Set B, spread 100. Although Input Set C produced fractionally better results it was decided that the difference did not justify the need for a larger network.

6 Analysis

In the last section the results of the three investigations were presented and discussed with key findings illustrated. The focus was on finding the most suitable network type. A large number of different networks have been tried and tested, in this section the specific findings will be presented in more detail. The results of individual networks shall be looked at.

6.1 Suitability of Neural Networks

The primary aim of this study is to determine the suitability of neural networks to stochastic construction processes. Section 5 – 'Results' has compared the relative performance of the different networks in some depth, however to deal with this crucially important objective we need to assess the ability of the neural networks to calculate outputs to largely coincide with the observed findings. Using the networks discussed in Section 5 a number of alternative predictive models have been used to generate estimates of either productivity or total duration using only the inputs of the testing set.

6.1.1 2 Layer

The best results from the initial investigation were very promising indeed when considering the smaller networks, i.e. improved generalisation due to size reduction. The best way to compare individual network performances is by examining a plot of observed and predicted output values. The MSE & SSE values are a good guide to indicate which networks are likely to produce the best results, however to distinguish between a number of networks with similar performances these more detailed plots are particularly useful. The first plot demonstrates the two-layer linear network attempts from the initial investigation (see Section 11, Fig. '2 Layer: - 1.7').

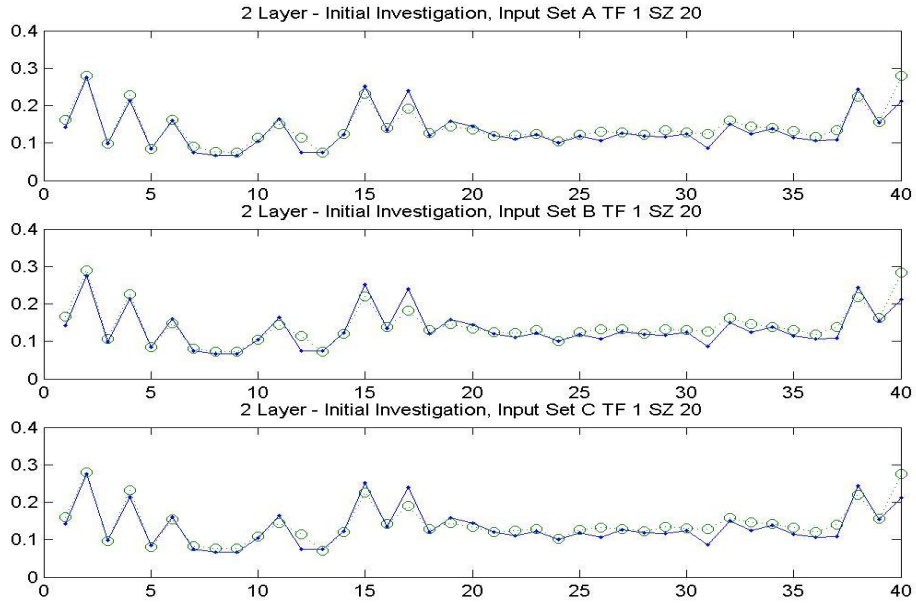


Figure 1: Initial Investigation - Productivity, 2-Layer linear networks. Observed (Dots) and Predicted (Circles).

The results as illustrated in Figure '3 Layer: - 1.7' of Section 11 are encouraging. The networks predict the productivity with very good accuracy for the majority of the records in the testing set, there are however a number of exceptions most notably record numbers 12, 17 & 31. The inaccuracy in these cases could be as a result of non-linear interactions in these cases, poor data or due to the omission of key variables (in this case for example, weather). The following graphs address the first of these possibilities by using non-linear networks.

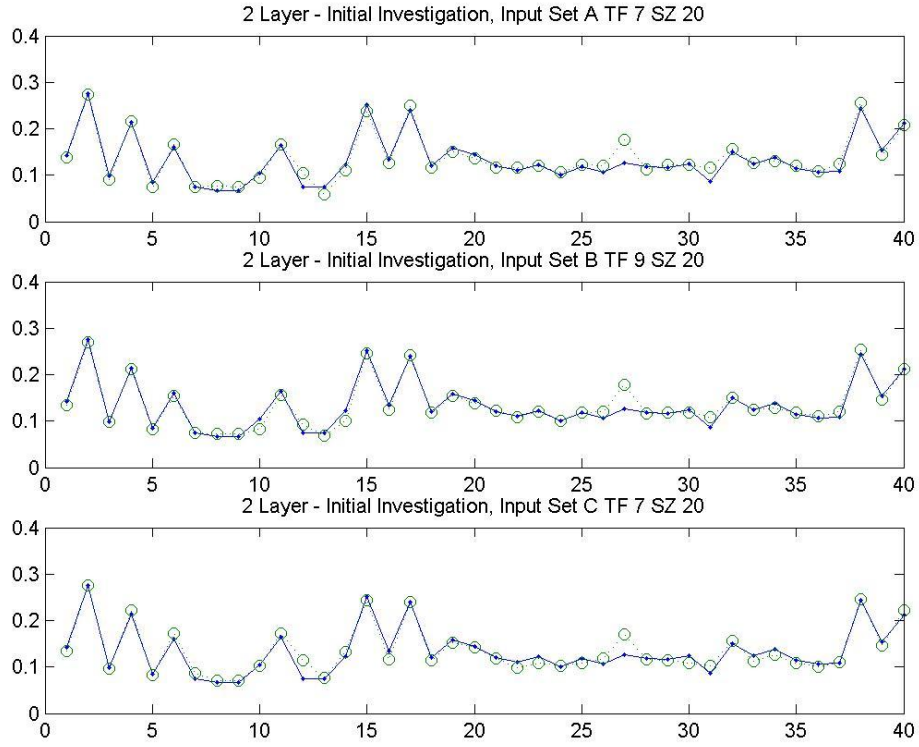


Figure 2: Initial Investigation - Productivity, 2-Layer best performance networks. Observed (Dots) and Predicted (Circles).

Using non-linear networks has helped the network predict the productivity even more accurately, Figure 2 above shows how record 17 is now predicted accurately for each input set, with records 12 & 31 also being greatly improved, however in each case the prediction of record 27 has decreased significantly reducing the overall improvement of the networks (Record 27 represents a very small pour of just two 7 m^3 trucks, it would be imprudent to become overly concerned with a record of such minor importance).

From figure '2/3 Layer: - 1.7' it is already clear that a linear approximation is not particularly accurate when trying to predict total durations. Figure 3 below shows the results of the linear 2-layer networks and indicate how close a linear approximation can fit the data.

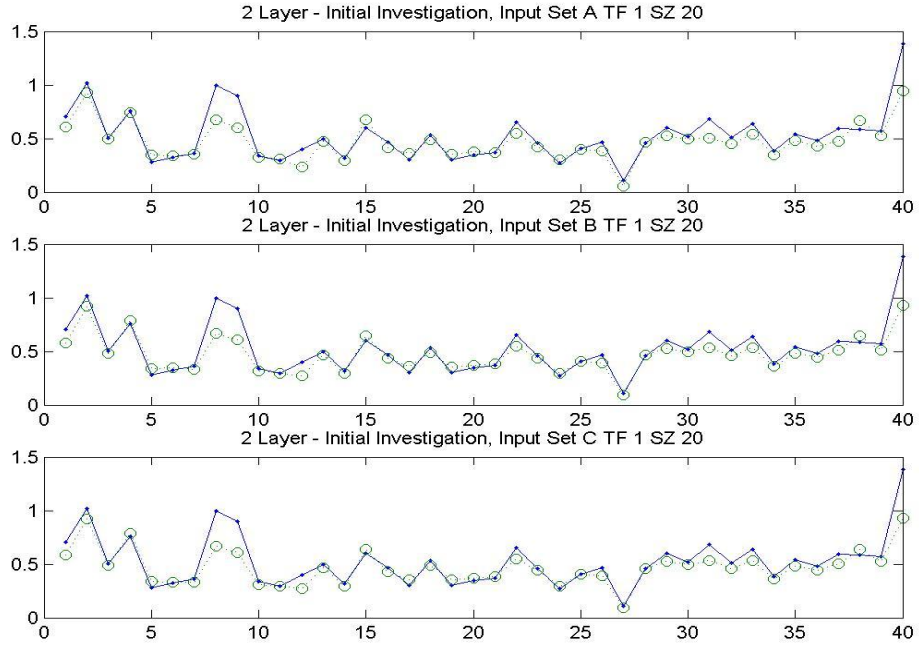


Figure 3: Initial Investigation, 2-Layer linear networks. Observed (Dots) and Predicted (Circles).

The results are better than might be expected having studied the results presented in Section 11, it is clear however that although the majority of the variation is picked up much of the subtlety is lost.

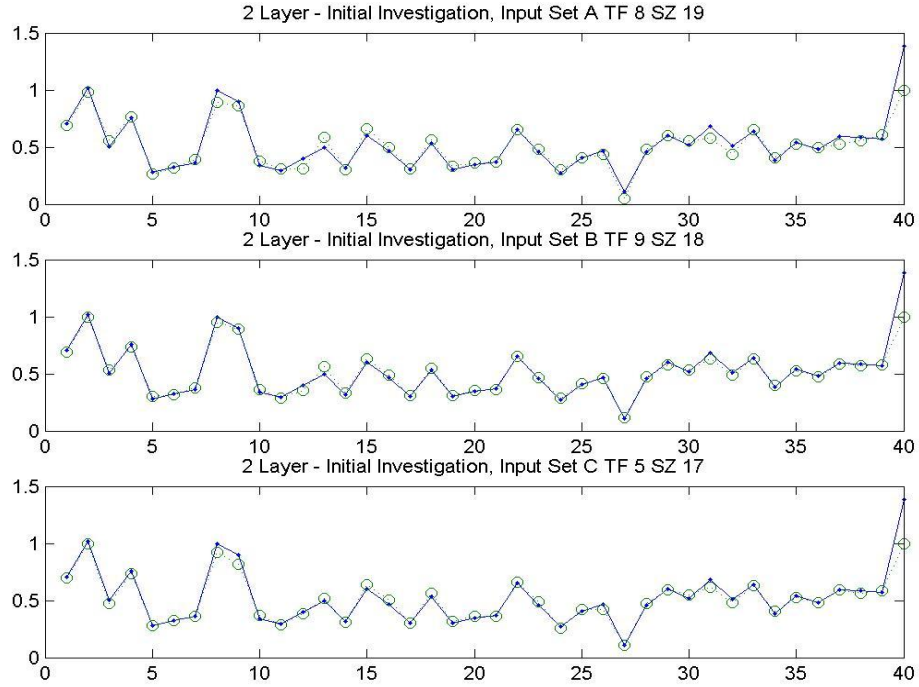


Figure 4: Initial Investigation - Duration, 2-Layer best performance networks. Observed (Dots) and Predicted (Circles).

Figure 4 shows the substantial improvement gained by using the best non-linear networks. Input set B seems to produce the best results although all three input sets predict the target durations with excellent accuracy until the last record that is slightly underestimated.

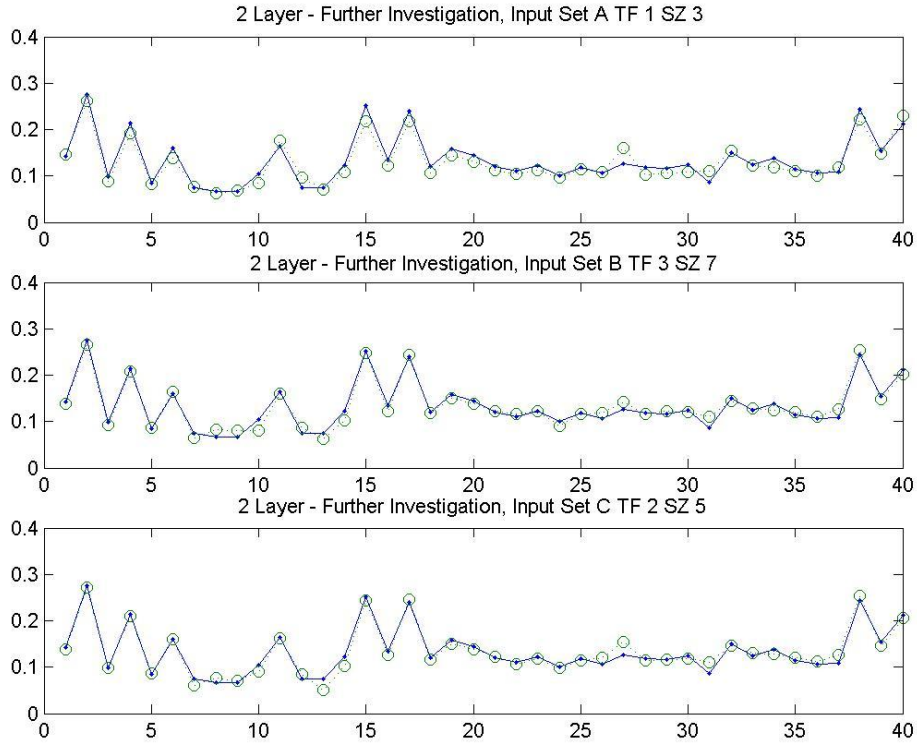


Figure 5: Further Investigation, 2-Layer best performance networks. Observed (Dots) and Predicted (Circles).

Using regularisation and early stopping the performance of the networks can be slightly improved as shown in Figure 5. The problem shown in Figure 2 can be seen to be partly corrected above, most notably in the case of Input Set B where record 27 is fairly well predicted, however it does appear to be at the expense of slightly poorer predictions between records 5 and 15. It is of note at this point to highlight the usefulness of the three generalisation techniques. All three techniques lead to networks with very similar performance to each other as a result no huge improvement was found over the initial investigation's networks, as size reduction had already been explored fairly comprehensively. However both regularisation and early stopping were found to greatly increase the chance of a network (even if over-sized) to competently predict values close to the target values. Enabling good results to be achieved with out having to experiment with large numbers of different networks as done in this study.

6.1.2 3 Layer

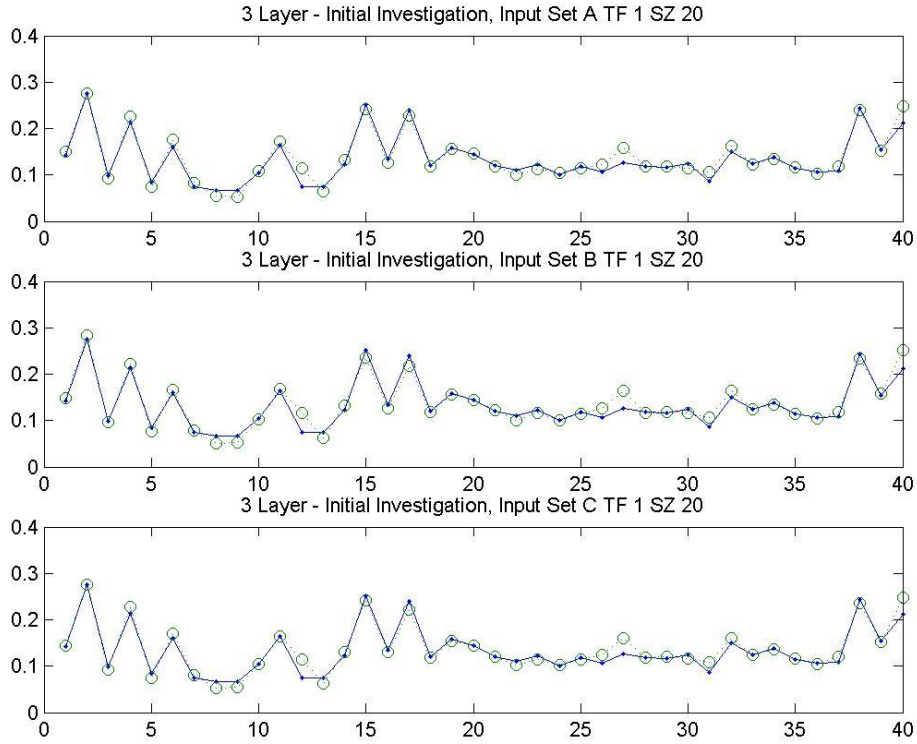


Figure 6: Initial Investigation, 3-Layer linear networks. Observed (Dots) and Predicted (Circles).

Figure 6 above shows how the three layer linear network produce results rather similar to the non-linear 2-layer networks, again over-estimating record 27. However record 12 is still over estimated as it was by the linear 2-layer networks. The addition of a second hidden layer allows the network to better estimate non-linear functions although both layers still contain linear transfer functions. Figure 7 shows what further improvements are achieved if both multi-layers and non-linear transfer functions are combined.

The results are very similar to those shown in Fig. 6, however the estimates for records 8, 9 & 12 are much closer to their observed values. Input set C also minimises the error in the estimate of record 27, all be it slightly at the expense of the estimate of record 26. However the predictions being generated from these networks are exceptionally accurate and despite slight discrepancies between some network predictions and the observed value a better fit could not be expected, a larger testing set however would provide a more rigorous examination.

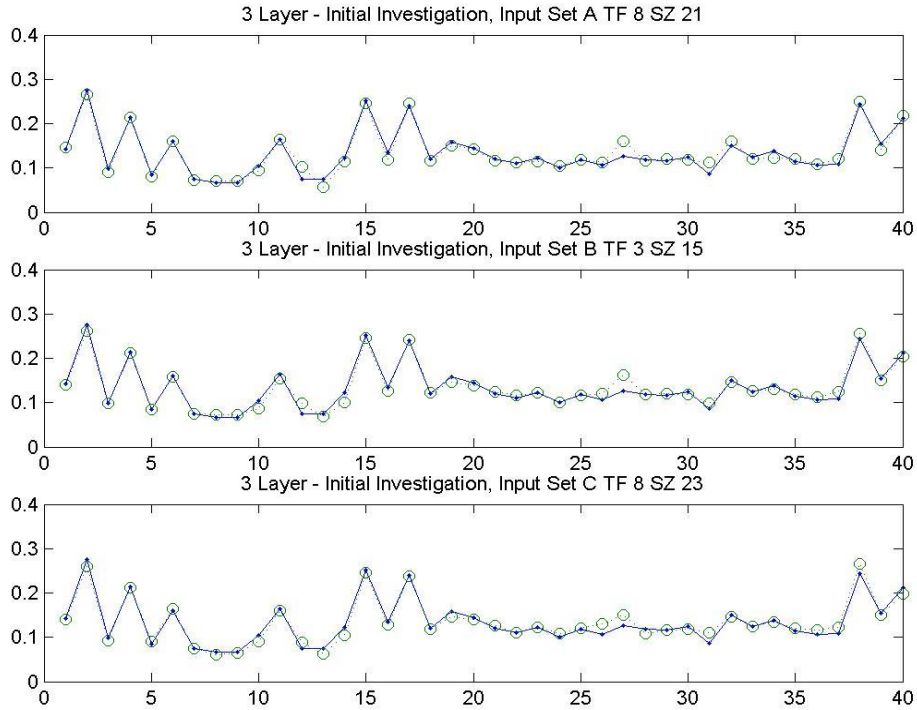


Figure 7: Initial Investigation, 3-Layer best performance networks. Observed (Dots) and Predicted (Circles).

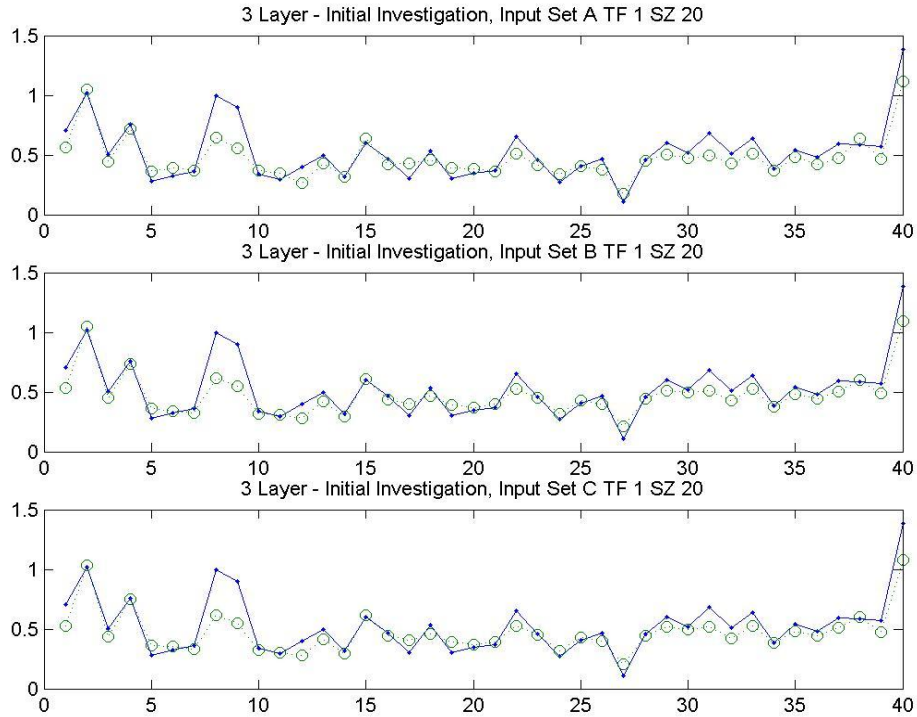


Figure 8: Initial Investigation - Duration, 3-Layer linear networks. Observed (Dots) and Predicted (Circles).

Again the linear approximation to the duration function can be seen to be lacking considerably, the results in Figure 8 above are very similar to those in Section 3.1.1 Figure 3 although some improvements are noticeable, record 40 is much closer to the observed duration, for example.

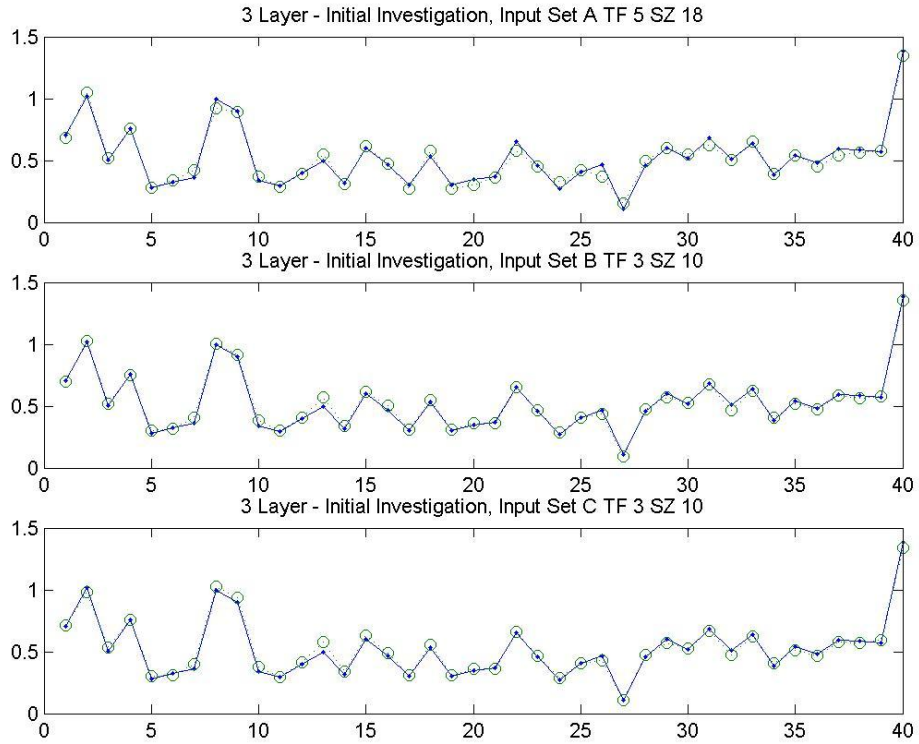


Figure 9: Initial Investigation - Duration, 3-Layer best performance networks. Observed (Dots) and Predicted (Circles).

Arguably the best results presented so far are those contained in Figure 9, all three input sets display very good results indeed. The middle plot in Fig. 9 shows an exceptional fit, with record 13 producing the poorest estimate – just fractionally too high. The biggest difference between the two-layer and three-layer networks is shown in records 40.

Figure 10 below shows the best results of the three-layer further investigation. The B set of inputs shown here actually represents the most successful of all the single output networks tried, although only by a fraction (a MSE margin of 1.0E-7). If record 27 was removed from the testing set many other networks would surpass this performance by a relatively large margin once again underlining the importance of high quality data.

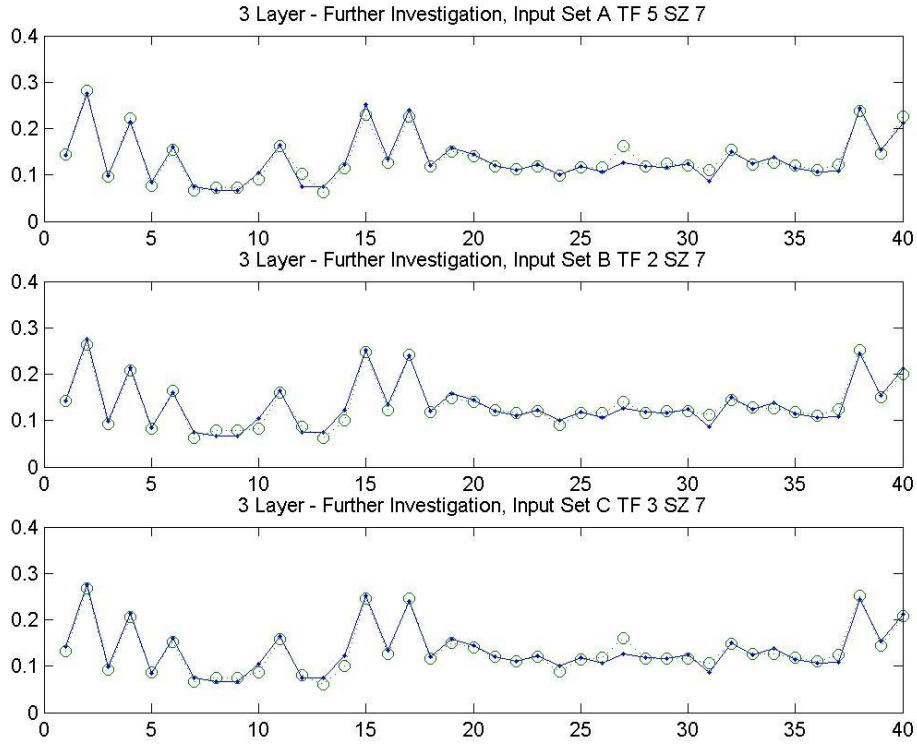


Figure 10: Further Investigation - Productivity, 3-Layer best performance networks. Observed (Dots) and Predicted (Circles).

6.1.3 Radial Basis Networks

Arriving at an optimum solution using a radial basis network is far simpler than with feed-forward networks, the complexity of the network may be greater – as each radbas neuron is only responsive to much smaller areas of the input space compared to transfer functions logsig, tansig etc.

The graph illustrating the optimum spread value is contained in the previous section, Fig. 11 below shows the plots representing the best results obtained from radial basis networks.

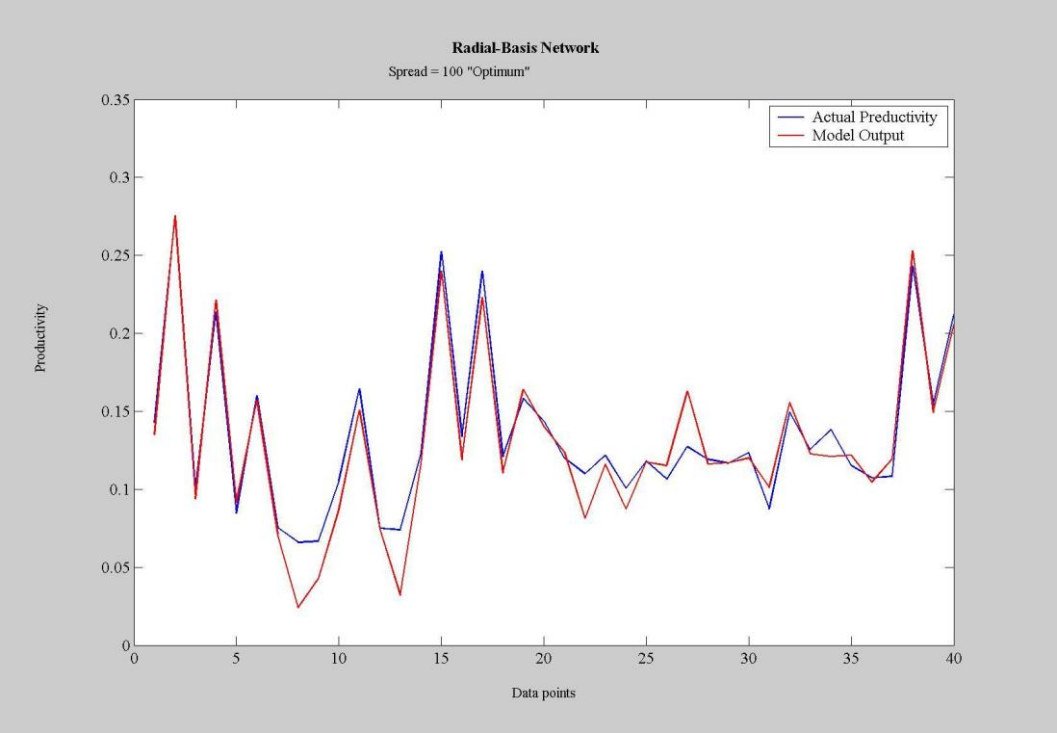


Figure 11: The optimum Productivity Radial Basis Network predictions based on the testing data set.

6.2 Effect of Network Properties

6.2.1 Architecture

The number of layers increases the complexity of a network but in a different sense to that caused by increasing the neuron numbers. The addition of a second hidden layer enables interactions between input-interactions to be included. It should be remembered that although this study investigated the basic variations possible in feed-forward networks a nearly infinite number of variations are possible, for example: some inputs can be supplied to the first layer while others can be supplied only and directly to the second layer. In theory a large enough network with all inputs being supplied to the first layer could be trained to effectively model such system, all that would be required is a first layer neuron with weights of zero on all but the input that was required in the second layer.

6.2.2 Size

The effect of network size has been well discussed in the various sections previous to the present one. The more neurons included in a network design the higher complexity of function that can be modelled, however using an excessively large network (especially with a data set of a limited size) can lead to over-training or over-fitting. In other words the network has the ability to not only fit to the general function but also to the specifics of the individual data resulting in poorer performances when used to estimate outputs from unseen data.

6.2.3 Transfer Function

The clear distinction here is between linear and non-linear functions as described in detail in Section 5. Linear networks generally find the same function to approximate while, it has be seen that, when using non-linear transfer functions the alternatives are too great to ensure different networks (or different attempts to train the same network) will approximate the same function as each other as shown by the variability of the performance colour plots in the non-linear network space.

6.3 Probability Distribution Output

The task asked of the network during the fuzzy investigation was considerably more complex than those previous, stretching the data set further. The number of independent variables increases significantly in most cases. It should be noted that a MSE value referring to a fuzzy network does not necessarily indicate the same goodness of fit as a similar MSE value referring to previous investigations, due mainly to the larger output space. For example the blue squares in the top left hand plot of Section 11, Fig. '2 Layer: - 3.4' represents networks whose output neurons return a constant value[*], however certain others produced more promising results. Only the networks using the data set constructed using a standard deviation of 0.2 shall be considered here as it provides the most rigorous test and is of the most value. Colour plots shall again be used to illustrate the results as three dimensions are required, the colour transition continues as before with scales of 0 – 1.

Figure 12 shows one of the most accurate (by inspection) two-layer networks, despite Section 11, Fig. '2 Layer: - 3.4' marking the relevant square brown i.e. a MSE greater than 0.001. The probability distribution is known to add up to one, therefore zone predictions can justly be scaled up so that their sum equals one. In such a case it is the pattern returned by the network that is more important than the MSE. Thus the assessment of performance relies on visual inspection (or more neural networks!). The fit shown in Fig. 12 is very good with the main zone correct in most and the two most likely zones correct in nearly all cases, excluding the usual 27 spike. Figure 13 shows the other alternative two-layer network that nearly manages to model pour 27 correctly. The output neurons can be thought of as independent, each requiring training, thus the peripheral zones are relying on three or four non-zero training pours.

[*] See Appendix

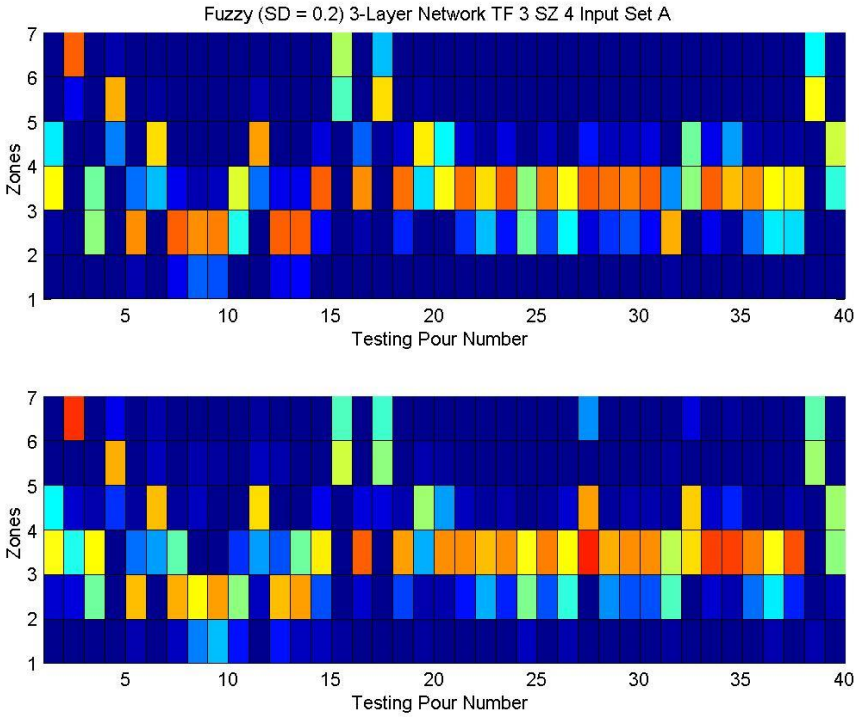


Figure 12: Successful 2-Layer Fuzzy Network, Outputs (Bottom), Targets (top).

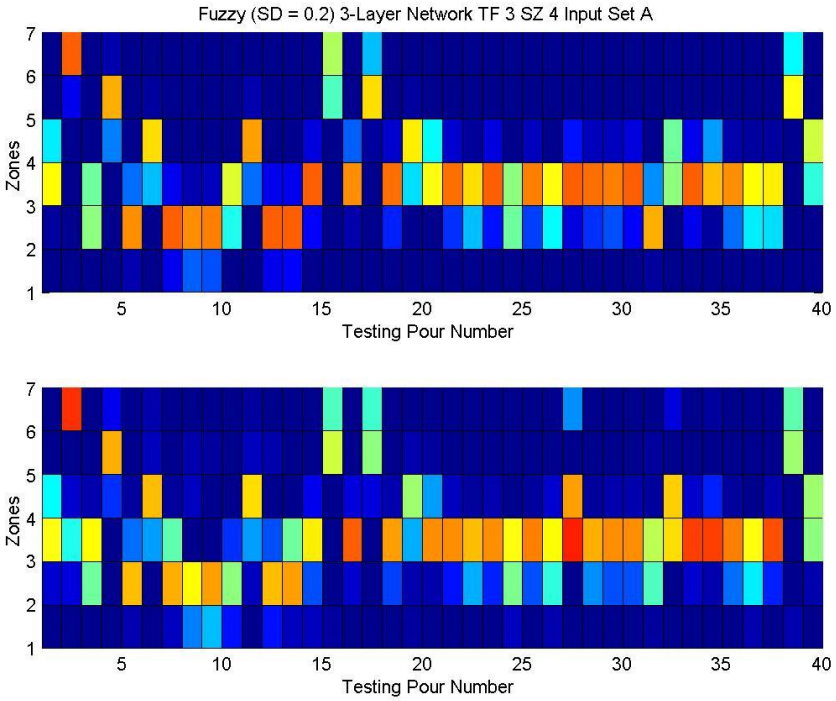


Figure 13: Successful 2-Layer Fuzzy Network, Outputs (Bottom), Targets (Top).

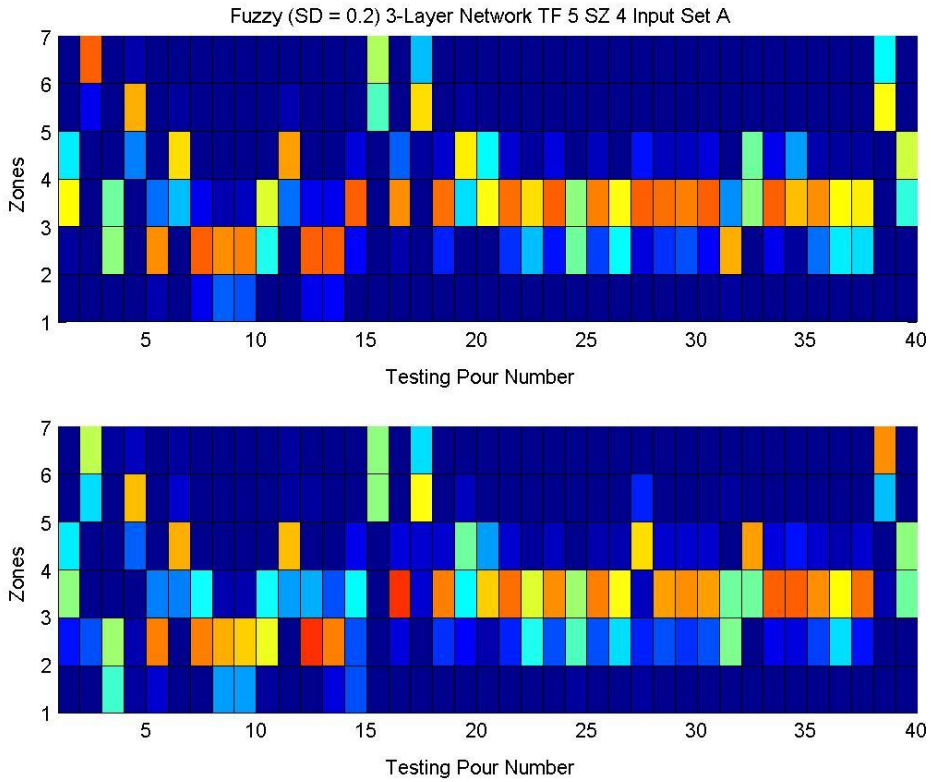


Figure 14: Successful 3-Layer Fuzzy Network, Outputs (Bottom), Targets (top).

Figure 14 shows the results of the best performing three-layer network, probably the best fuzzy network results. The MSE does register in the performance charts at a value of around 0.00065. Ignoring the results of pour 27 the network shows an excellent fit, including most of the low value zones. Pours 3 & 14 are the only slight inaccuracies, but the prediction for pour 14 would be nearly correct once scaled up.

7 Discussion

7.1 Physical meaning

In many ways neural networks can be thought of as a two-dimensional series of connected mini-neural networks and essentially they are. Any smaller processes operating within the concrete pouring and affecting its productivity could be being modelled within the network. Wherever possible a single cycle should be modelled and if necessary combine it with a second network to produce the desired effect. For example in earthmoving, the hauler and loader cycles could be modelled separately and then combined either within a network or as multiple networks. By the same argument you could claim that once neural network models are developed for the majority of key construction processes a construction site could be dynamically modelled by their sum, with appropriate interdependencies.

At the moment the results from the simplest network that satisfactory predict the productivity will suffice for demonstration purposes. With a two-layer linear network quite simply we have n interactions between the inputs and then a final interaction of all n results. Where n is the number of neurons in the first layer, in this case two. The weights and bias of a two layer linear network with 9 inputs, two neurons in the first layer and one in the second are shown in Fig. 1 below.

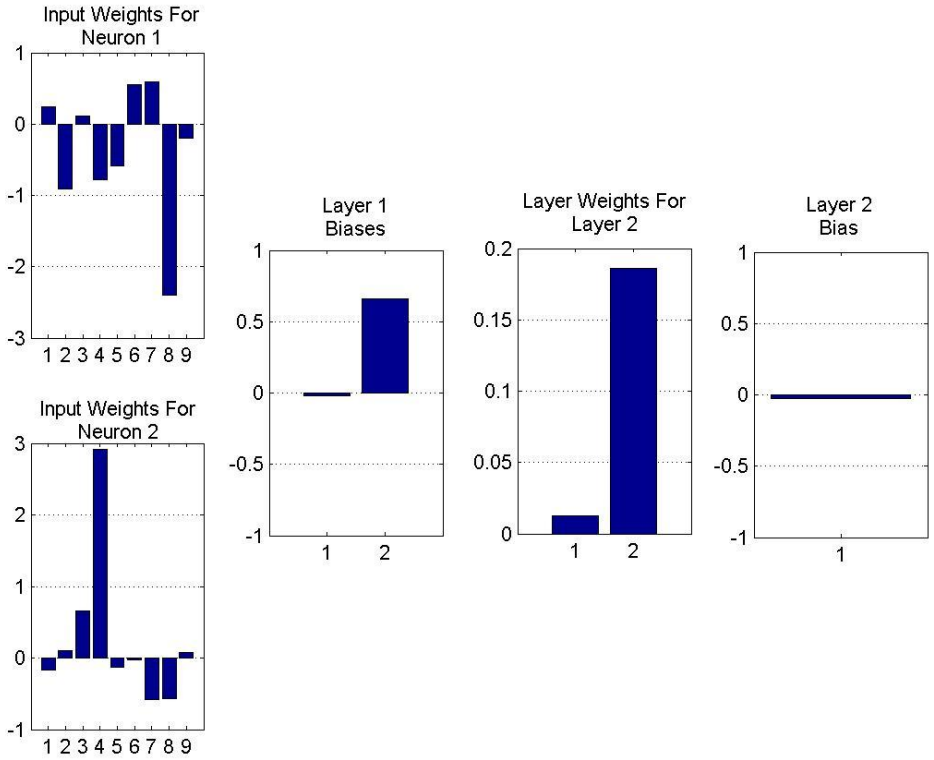


Figure 1: Weights and biases for a linear 2-Layer network (TF 1 SZ 20).

It is easier to understand how the network calculates the output by working backwards through the network. The bias in the second layer can virtually be ignored as clearly the network estimates the productivity largely from the weighted sum of the outputs from the neurons in the first layer. The layer two weights indicate that the second neuron in the first layer is responsible for the majority of the estimate with a weighting nearly 18 times greater than that of the first neuron in the first layer. The output of the second neuron in the first layer appears to consist of four major input influences, however the range of each of the inputs should be considered. Using the weighting values displayed above and the mean values of each of the inputs for the entire data set Fig. 2 illustrates the effective influence of the nine inputs on the variable part of the output in neuron 2, layer 1.

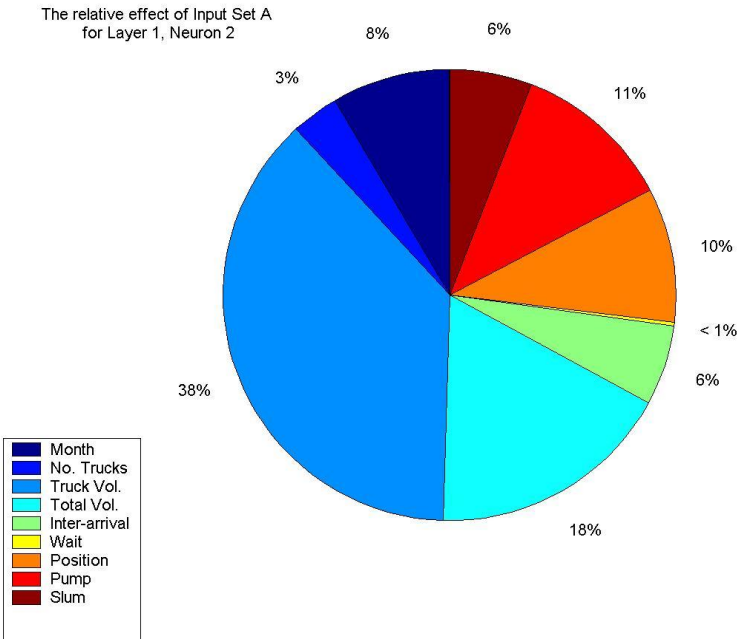


Figure 2: The influence of the different inputs for the second neuron of the first layer in the simple 2-layer network.

Before the effects of the different inputs are discussed the entire network should be examined. Figure 3 below shows the relative effects of the input on the variable proportion of the first layer, first neurons output. Figure 4 shows the effect each input has on the variable part of entire networks output. (Note: Figures 2-4 can only give an indication of the effects of the different inputs, as average input values were assumed and the modulus taken of negative figures to create an estimate of relative input effect.)

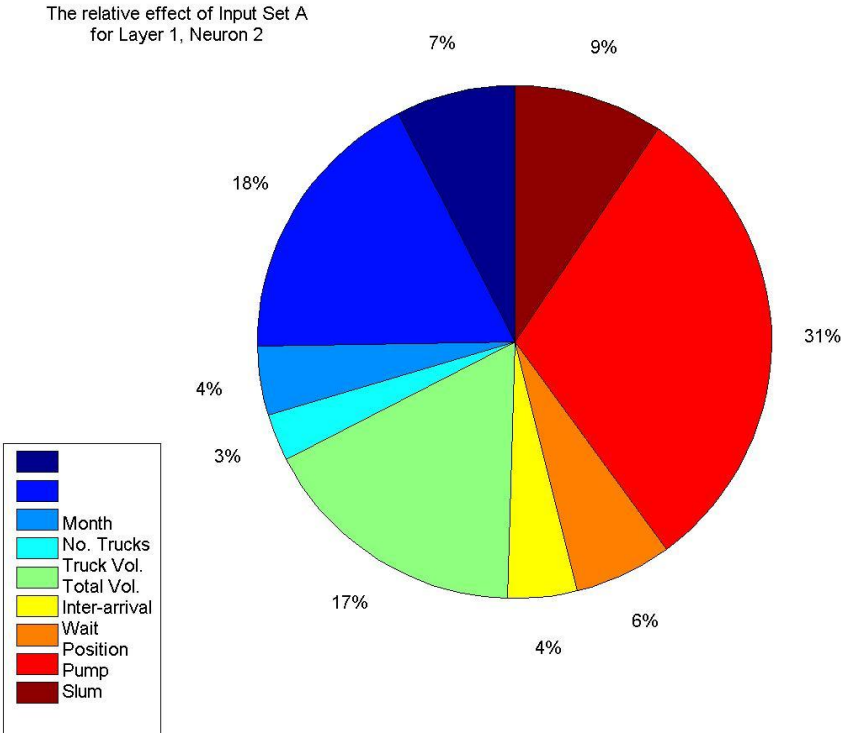


Figure 3: The influence of the different inputs for the first neuron of the first layer in the simple 2-layer network.

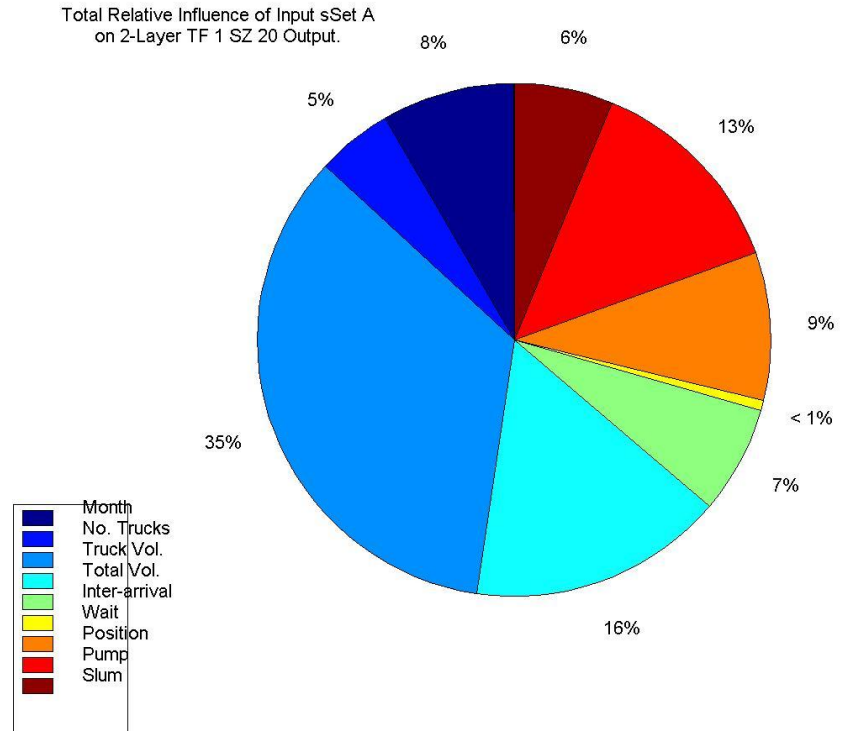


Figure 4: The influence of the different inputs on the linear 2-layer network.

Comparing Fig. 4 with Fig. 2 it is confirmed that as displayed in Fig. 1 the first layer, second neuron plays the most crucial role in the network. However those

inputs heavily influencing the first layer, first neuron (such as 'Pump') do increase their overall effect slightly.

The key inputs for this network are truck volume, total volume and pump time, respectively. As the majority of the time leading to inefficiency remains constant per truck, it makes sense that the variation in truck volume would have a significant effect. The highest productivities were usually also the longest pours justifying the second most influential input, total volume. The pumping time is clearly a key input, as the pump is the bottleneck of the system. All the other inputs had similar influences at just under 10% apart from the number of trucks and average waiting time. The waiting time was very small in almost all cases preventing the network from being able to estimate the effect of its variation. The number of trucks has only a 5% effect, perhaps again due to the range of values the network was trained with. Clearly as all the data was recorded from real pours the number of trucks will always be within a certain range to maximise productivity - this will prevent the network from being presented with the more extreme examples illustrating the effect of either very few or far too many trucks on a particular pour.

7.2 The Accuracy & Reliability of the Final Networks

It has been shown that a number of alternative neural network designs can be used to predict productivity to a similar degree of accuracy. The MSE figures quoted throughout this commentary refer to productivity divided by 100 or total duration divided by 10. This section will assess the most promising results in a manor allowing direct comparisons between alternatives, rating their relative success by converting the MSE values to average errors with a physical meaning e.g. average error in m^3 / hr. This will give an indication on the accuracy of the various networks although the reliability of the indication will only hold for pours whose inputs and output fall in the input and output space covered by the training set.

Figure 5, below, shows the two networks resulting in the best performance as measure using the testing data set. Both networks managed to reduce the MSE for the testing set predictions to a fraction over $1E^{-4}$, this can be translated into a root mean square error of 1.01 m3 / hr. The average productivity within the testing set was calculated to be 13 m^3 / hr giving an error of just under 8 % - a good level of accuracy for such a variable process.

Figure 6 represents the three-layer network that just out performed the two two-layer networks shown in Fig. 5. Once again resulting in a root mean square error of 1.01 m3 / hr. In comparing the plots contained in Figs. 5 & 6 it can be seen that the source of error occurs in different places for each network, with some similarities such as testing pour number 27. This raised the question, could predictions be improved by combining different networks? Figure 7 illustrates the effect of combining the three networks in Figs. 5 & 6.

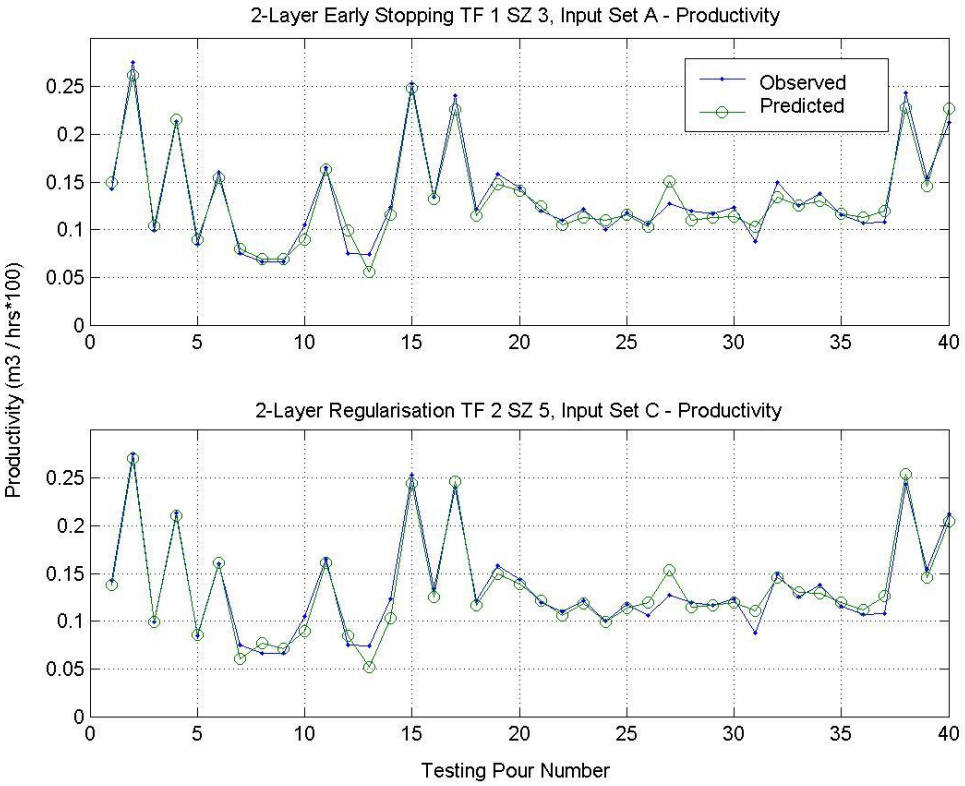


Figure 5: 2-Layer Networks Producing The Lowest Performance.

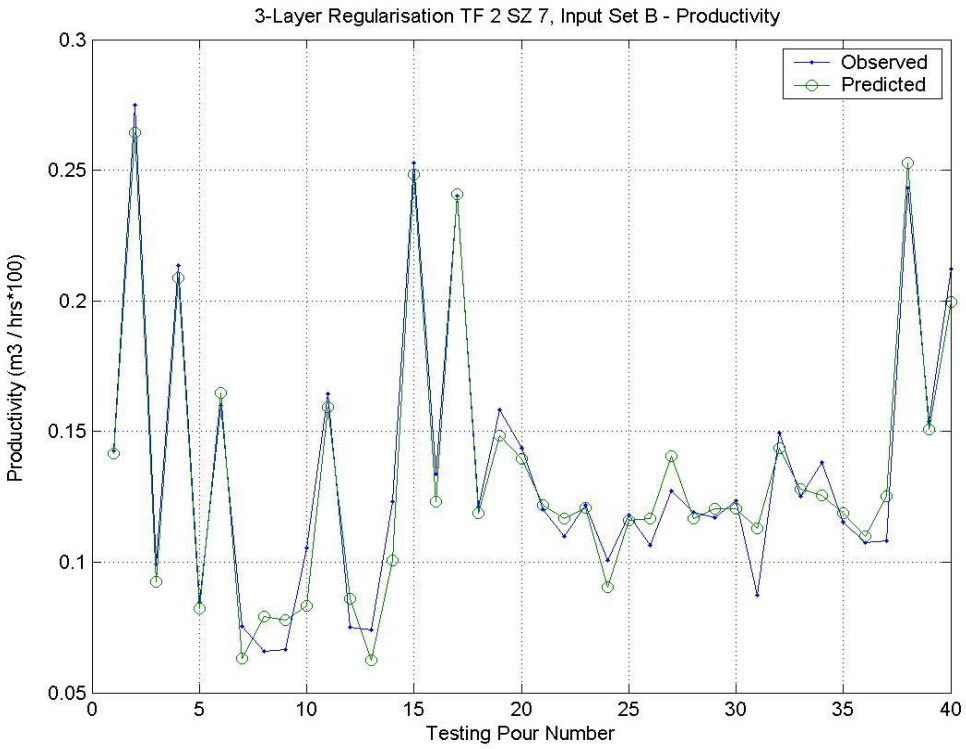


Figure 6: 3-Layer Network Producing the Lowest Performance.

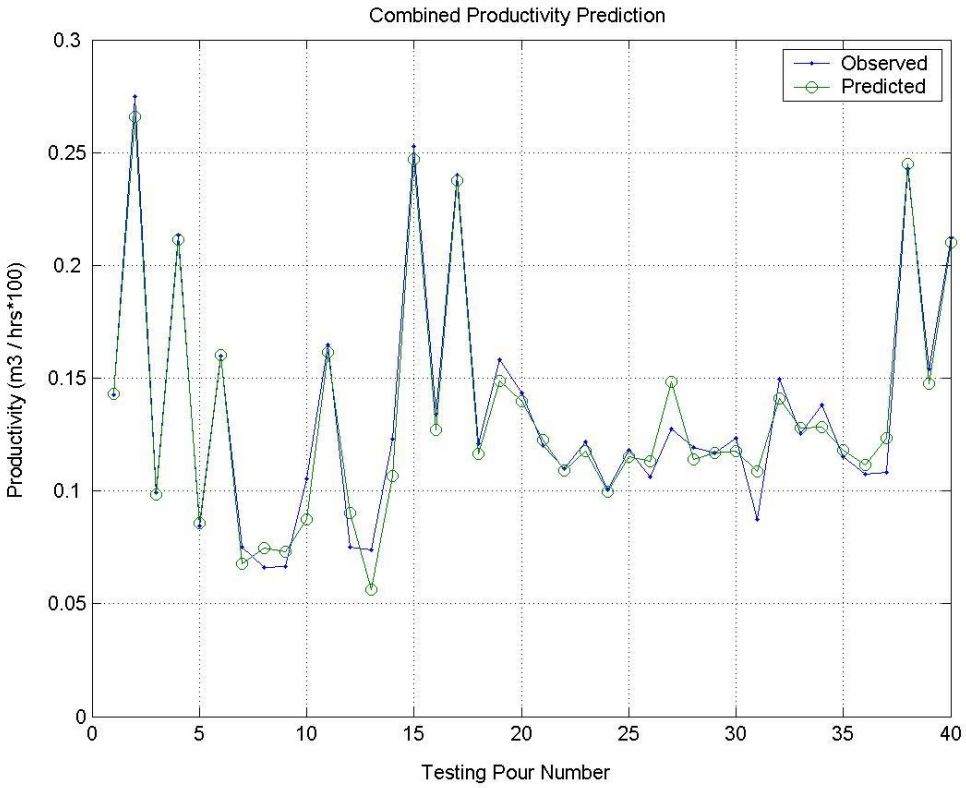


Figure 7: Results of Combining Three Different Networks.

The fit has certainly improved a little, with a MSE of now $7.8381E^{-5}$. Reducing the root mean square error over the testing set to 0.89 m3 / hr, an error of 6.8 %. Many more networks could be arranged together to further reduce the error, the advantage of such an approach is that a single network is not required to generalise perfectly. Returning to the biological inspiration, the shear complexity of the biological neural network suggests that real neural networks could well use a similar technique.

The final comparison concerning single valued productivity output networks (fuzzy networks prove difficult to compare) involves the duration prediction networks. As discussed at the beginning of the report the idea behind including total duration as a potential output was to save the network unnecessary computation, as productivity is simply total duration divided by cumulative volume. Using the three-layer network that produced the best duration predictions (Fig. 8) productivity values were calculated by dividing through by cumulative volume and are presented in Fig. 9 showing an even better fit than the combined approach. The MSE value is calculated as $4.086E^{-5}$, which translates to a root mean square error of 0.64 m3 / hr, an error of only 4.9 %.

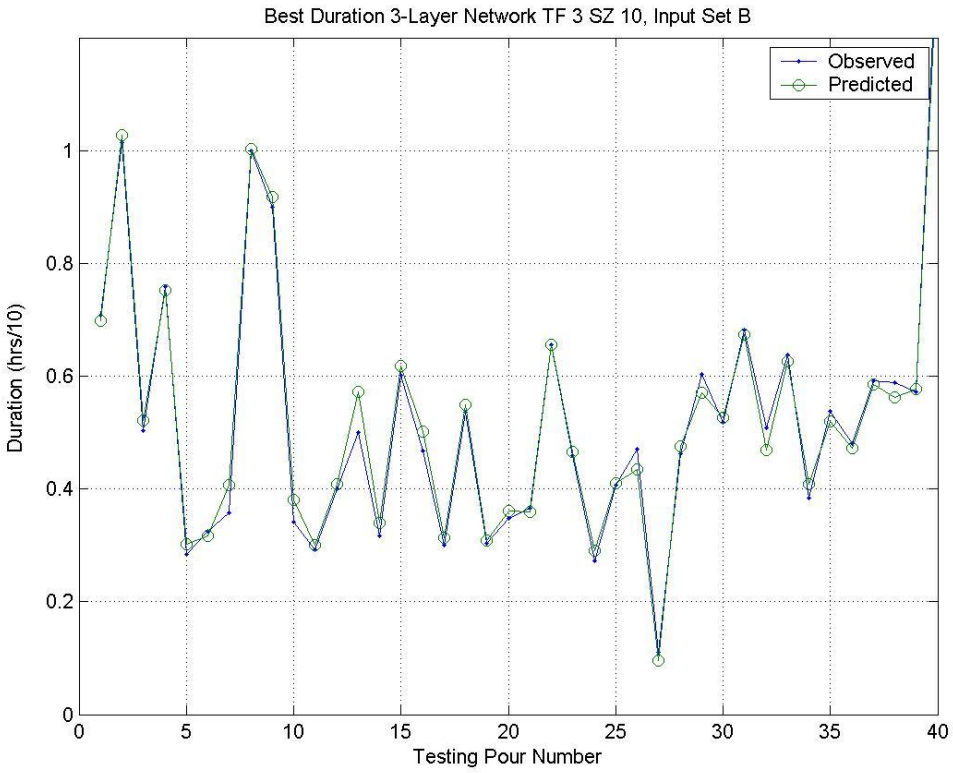


Figure 8: Best Network for Duration Prediction.

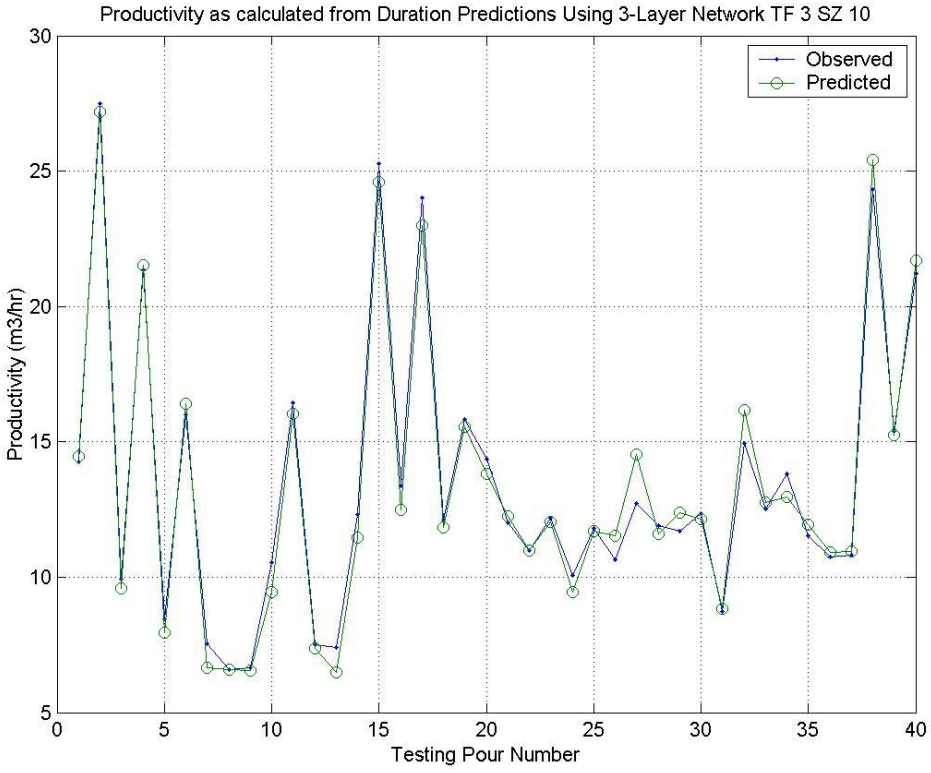


Figure 9: Productivity Values Calculated from Duration Predictions Shown in Fig. 8

Testing pour 27 once again causes the majority of what error is left, although the duration appears to be estimated very accurately. On closer inspection the duration error is under eight and a third minutes, however as the entire duration is only 1 hour and 6 minutes the percentage error is over 12 %, causing a more noticeable error when converted to productivity values. However the need to predict the productivity of a concrete pour that consists of only two loads and only lasts a fraction over an hour is questionable and errors resulting from testing record 27 should not be cause for any concern. The removal of record 27 would further increase the accuracy of the networks. It should also be noted that the further investigation did not include duration, however the initial two-layer training results suggest a near optimum relationship was found by the majority of the networks tested.

8 Conclusions

8.1 Can Neural Networks be used effectively in the Construction Industry?

Neural networks are a fast immerging computational technique that in many ways can be considered to still be in its infancy. The applications are essentially limitless and potential applications will become evident as and when new developments within neural networks and those fields surrounding them become available.

The ability to train a number of different networks to successfully predict the productivity of concrete pouring activities across a number of different projects managed by different contractors has been illustrated throughout this report, particularly Sections 5, 6 & 7 with excellent results. The error over the forty testing pours was reduced to under 8 % using a number of different networks, and to under 5 % by predicting duration. A root mean square error of about 1 m3 / hr was consistently achieved. The accuracy of the results and the numerous networks that could achieve them lead to a categorical conclusion that neural networks most certainly can be used with great success in predicting the efficiency of construction processes.

8.2 Specific Findings

- Feed-forward networks out performed radial basis networks, however with a larger data set radial basis could produce very accurate results, as they are capable of reducing the error within the training set to zero.

- Duration predictions resulted in the best results; during the investigations duration produced more consistent results across the various networks suggesting that the function for duration is more easily approximated than the productivity function. Input sets B & C were found to produce the best results. It was also seen that the productivity function could be better approximated to a linear function than the duration function.

- A three-layer network with 16, 4 & 1 neurons in the first, second & third layer respectively, using a linear first and last layer and a second layer hyperbolic tangent sigmoid transfer function was found to achieve the lowest performance value for productivity estimation by first predicting the total duration of each pour. (For results see Section 7, Fig. 9.)

- A general understanding of how neural networks can be adapted to model construction processes has been presented, illustrating the effect of; the number of layers, network size, transfer function, feed-forward versus radial bias networks, over-fitting, three generalisation techniques and various input/output pairs.

- The ability of neural networks to produce a productivity probability distribution has also been illustrated with good results.

- Section 7.1 analyses a simple two-layer linear network and breaks down the output into input influences, demonstrating what inputs drive the network and to what extent.

- To develop a model that could be used reliably within the industry a greater data set is required. The network's accuracy and reliability depends largely on the size of the data set. The range over which the network can be used confidently is also determined by the range covered in the training data set.

8.3 Summary of Optimum Results

Network	Input Set	Generalisation Method	TF (Initial)	SZ (Initial)	Duration	Productivity		
					MSE	MSE	RMS Error (m3 / hr)	% Error
2-Layer	C	Regularisation	2 (7)	5 (17)	-	1.030E-04	1.015	7.81
2-Layer	A	Early Stopping	1 (5)	3 (3)	-	1.026E-04	1.013	7.79
3-Layer	B	Regularisation	2 (4)	7 (23)	-	1.025E-04	1.012	7.79
3-Layer	B	None	(3)	(10)	0.0005	4.086E-05	0.639	4.92

Table 1: Optimum Network Performances

9 Further Work

9.1 Increased Data Set

As has been stressed though out the above report one of the fundamentals limitations of neural networks is the size of the data set it is presented with. It is this data set that largely determines the ability of the network to generalise successfully enabling it to predict previously unseen cases. To maximise the use of neural networks in predicting the efficiency of construction processes a far larger data set than used in this study is required. If a serious model was to be constructed an automated data recording system should enable very good results indeed. Telemetry fitted to the pump and trucks, although clearly carrying an initial cost of installation, would provide accurate and continuous data. Unfortunately the advantages of such an approach would only be seen once the expense had been incurred reducing the likely hood of the installation.

9.2 Extensions of work described

With an increased data set all the options explored could be improved and more definite conclusions reached. With a sufficiently large data set pattern recognition neural networks could be used rather than function approximation networks, returning the productivity learnt for the relevant input pattern.

A further investigation into three-layer duration networks may result in a further improvement to the model. However any error reduction is likely to be small.

Fuzzy networks could be designed with more zones producing a more accurate probability histogram.

9.3 Options not investigated

Clearly there are many alternative ways in which neural networks could be used in predicting the efficiency of construction process that were not explored in the above report.

- Recurrent Networks: Recurrent networks should be able to model construction processes in such a way that closely reflects the physical nature of the systems. See Flood & Christophilos (1996) & Nerrand (1994).

- Additional features could be added to the networks explored within this study, for example competitive neurons, delays and alternative transfer functions.

- The network could be defined more closely to represent the physical processes. For example, targets can be entered in any layer to any neuron, as can inputs. In other words the network can be designed so different sections produce sub-outputs that can then be combined in such a manor

that the final output is reached. In this study all inputs were supplied to all first layer neurons and a single target was used for the entire network.

- Use genetic algorithms as a means of variable selection in future neural network investigations, including network properties, inputs/outputs selection.

10 References

Ashraf M. Abdelbar, Sahar Attia and Gene A. Tagliarini (2002) *A hybridization of Bayesian and neural learning.* Neurocomputing, Volume 48, Issues 1-4, October 2002, Pages 443-453

Bishop, C.M (1998) *Neural Networks and Machine Learning*. NATO ASI Series, Series F: Computer and System Sciences, Vol. 168.

Chao L., Skibniewski M. (1996) *Neural Network Method of Estimating Construction Technology Acceptability.* Journal of Construction Engineering & Management, ASCE.

Faghri A, Hua J (1995) Ro*adway seasonal classification using neural networks.* Journal of Computing in Civil Engineering 9 (1995) 209-215

Flood I., Christophilos P., (1996) *Modelling construction processes using artificial neural networks.* Automation in Construction 4 (1996) 307-320, ELSEVIER

Hagan, M.T., Demuth, H.B., Beale, M. (1996). *Neural Network Design*. PWS Publishing Company.

Jin X. Cheu R.L. Srinivasan D. (2002) *Development and adaptation of constructive probabilistic neural network in freeway incident detection.* Transportation Research Part C: Emerging Technologies, Volume 10, Issue 2, April 2002, Pages 121-147

Kartman, N., Flood, I., Garrett, J.H. (1997). *Artificial Neural Networks for Civil Engineers: Fundamentals and Applications.* ASCE, New York.

Lee T.L. Jeng D.S. (2002) *Application of artificial neural networks in tide-forecasting,* Ocean Engineering, Volume 29, Issue 9, August 2002, Pages 1003-1022

Leu SS, Chen AT, Yang CH, *Fuzzy optimal model for resource-constrained construction scheduling,* Journal of Computing in Civil Engineering 13 (1999) 207-216

Marsan, M.A. (1989) *Stochastic petri nets: An elementary introduction: Advances in petri nets.* Lecture Notes in Computer Science 424. Springer-Verlag, New York.

More A, Deo MC *Forecasting wind with neural networks.* Marine Structures (2003) Vol. 16, 35-49

Nerrand O, Poussel-Ragot P, Urbani D, Personnaz L, Dreyfus G (1994) *Training Recurrent Neural Networks: Why and How? An illustration in dynamical process modelling* IEEE Transactions in Neural Networks, Vol. 5, No. 2, (1994) March

Portas J., AbouRizk S. (1997) *Neural Network Model for Estimating Construction Productivity.* Journal of Construction Engineering & Management, December 1997, 399-410.

Sarghini, F. de Felice, D. Santini, S (2003) *Neural networks based subgrid scale modeling in large eddy simulations.* Computers & Fluids, Volume 32, Issue 1, January 2003, Pages 97-108

Sawhney A., Abudayyeh O., Chaitavatputtiporn T. (1999) *Modelling and Analysis of Concrete Production Plant Using Petri Nets.* Journal of Computing in Civil Engineering, ASCE, July, 178-186.

Sinha S.K., McKim R.A. (2000) *Artificial neural network for measuring organizational effectiveness.* Journal of Computing in Civil Engineering 14 (2000) 9-14

Smith S.D. (1999) *Earthmoving Productivity Estimation Using Linear Regression Techniques.* Journal of Construction Engineering & Management, May/June, 133-141.

Yeh IC (1995) *Construction-site layout using annealed neural-network.* Journal of Computing in Civil Engineering 9 (1995) 201-208

Vila J.P, Wagner V., Neveu P. (2000) *Bayesian Nonlinear Model Selection & Neural Networks: A Conjugate Prior Approach.* IEEE Transactions on Neural Networks, Vol. 11, No. 2, March (2000) 265-278

Yeh IC (1999) *Design of high-performance concrete mixture using neural networks and nonlinear programming.* Journal of Computing in Civil Engineering 13 (1999) 36-42

11 Graphical Representation of Network Performances

