Prediction of Friction Stir Weld Quality using Self Organising Maps.

Abhishek Bhowmick

A Thesis submitted to
Auckland University of Technology
in fulfilment of the requirements for the degree of
Master of Philosophy (MPhil)

2010
School of Engineering
Primary Supervisor: Zhan Chen

Acknowledgments

The author would like to express sincere gratitude to supervisor Dr. Guy Littlefair for his excellent guidance and patience both of which contributed significantly throughout the duration of this study.

A notable acknowledgment goes to Associate Prof Zhan Chen for his valuable inputs, discussion and assistance with FSW trials.

The author is grateful to the technicians at AUT’s workshop for their time and guidance during the FSW trials.

Finally, none of it would have been possible without my loving partner, Jesal’s unconditional support and encouragement through trying times.

Dedicated to my parents in love and gratitude

Swati & Animesh Bhowmick

Abstract

Interest in using artificial neural networks for predicting & forecasting has led to a tremendous surge in research activities in the past decade. Self organising maps, also commonly known as unsupervised neural networks, are known to generate their topology during learning. This leads to a network structure which converts complex, nonlinear relationships between multi-dimensional data into simple geometric relationships on a low-dimensional display.

In this study, a Self Organising Map (SOM) is employed to predict the quality of welds using the Friction Stir Welding (FSW) process. FSW is a relatively novel welding technology, which has caught the interest of many industrial sectors due to its many advantages and clear industrial potential. Despite the successful deployment of FSW in industry, research relating to friction stir weld quality is developing rather gradually. This is mainly due to the non-deterministic nature of the environment in which the system must function.

The study is aimed to demonstrate and apply the most important property of the SOMs to FSW - orderliness of input-output mappings. Experimental data was collected by performing a series of FSW trials on Aluminium alloy AA2024 and A253 against a selected range of parameters. The SOM system was trained using the prepared training set. The generated model was tested against an unseen set of data captured during the FSW trials. The network results were in good agreement with the previously unseen data. It has been demonstrated that the SOM algorithm can be used as reliable tool for predicting weld quality.

Table of Contents

List of Figures

List of Tables

List of Acronyms

1. TWI – The Welding Institute

2. FSW – Friction Stir Welding

3. ANN – Artificial Neural Network

4. CNC – Computer Numerical Control

5. SOM – Self Organising Map

6. MLP – Multi Layered Perceptron

7. TAR – Threshold Autoregressive Cointegration

8. FSP – Friction Stir Processing

9. TRS – Tool Rotation Speed

10. HAZ – Heat Affected Zone

11. RSM – Response Surface Methodology

12. BP – Back Propogation

13. HRC – Hardness Rockwell C

14. PSD – Power Spectral Density

Statement of Originality

“I hereby declare that this submission is my own work and that, to the best of my knowledge and belief it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the qualification of any other degree or diploma of a university or other institution of higher learning, except where due acknowledgement is made in the acknowledgements.”

………………………………. (Signed)

..………………………………. (Date)

1 Introduction

1.1 Welding: A Brief Overview

Welding is the process of joining materials, usually metals or thermoplastics, by softening with heat and applying pressure. ‘The majority of welding processes rely on heat more than on pressure to accomplish joining by creating atomic bonding across the joint interface’ [31]. There are two types of welding: fusion and non-fusion welding. During fusion welding, sufficient heat causes melting, and significant melting is necessary for welding to take place. In non-fusion welding, heat is only enough to soften the material in the solid state to facilitate plastic deformation or speed up the material solid phase diffusion [31].

Welding technology has advanced quickly during from the early 20th century - several modern welding techniques were developed, including friction stir welding, gas metal arc welding, submerged arc welding and electroslag welding [1]

1.1.1 Background of Friction Stir Welding

Friction stir welding (FSW) is a novel welding technique invented by The Welding Institute (TWI) in 1991. It was invented and experimentally proven by Wayne Thomas and a team of his colleagues at The Welding Institute UK in December 1991 [5]. TWI holds a number of patents on the process, the first being the most descriptive. FSW is in fact a solid-state joining process that is a combination of extruding and forging and is not a true welding process [5].

1.1.2 Key benefits of FSW

Since the process occurs at a temperature below the melting point of the work piece material, FSW has several advantages over fusion welding. Some of the process advantages are given in the following list [7]:

Metallurgical benefits	Environmental benefits	Energy benefits
▪ Solid phase process	▪ No shielding gas required	▪ Improved materials use (e.g., joining different
▪ Low distortion of work piece.	▪ No surface cleaning required	▪ thickness) allows reduction in weight
▪ No loss of alloying elements	▪ Eliminate grinding wastes	▪ Only 2.5% of the energy needed for a laser weld
▪ Excellent metallurgical properties in the joint area	▪ Eliminate solvents required for degreasing	▪ Decreased fuel consumption in light weight aircraft, automotive and ship applications
▪ Absence of cracking.	▪ Consumable materials saving, such as rugs, wire or any other gases	
▪ Fine Microstructure		
▪ Good dimensional stability and repeatability		

Table 1 Benefits of FSW [3]

These advantages have generated interest from many key industries, particularly in the aerospace industry (such as Boeing, NASA – Marshall Space Flight Centre), the marine industry (e.g. fast ferries, fishing boats and yachts etc), and also transportation like the A-train from Hitachi [3-4]

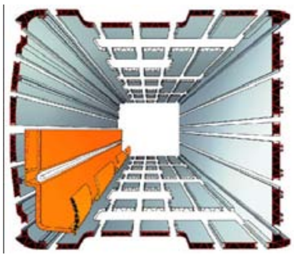


Figure 1-1 Aluminium double skin of the A-Train [4]

1.1.3 Principle of Operation

In friction stir welding, a cylindrical-shouldered tool, with a profiled threaded/unthreaded probe (nib or pin) is rotated at a constant speed and fed at a constant traverse rate into the joint line between two pieces of sheet or plate material, which are butted together. The parts have to be clamped rigidly onto a backing plate in a manner that prevents the two joint faces from being forced apart. The length of the pin is slightly less than the depth of weld and the tool shoulder is kept in close contact with the work surface [6].

The pin is moved against the work, or vice versa. Frictional heat is generated between the wear-resistant welding tool shoulder and pin, and the material of the work pieces. This heat, along with the heat generated by the mechanical mixing process and the adiabatic heat within the material, cause the stirred materials to soften without reaching the melting point (hence cited as a solid-state process), allowing the traversing of the tool along the weld line [6,11]

As the pin is moved in the direction of welding, the leading face of the pin, assisted by a special pin profile, forces plasticised material to the back of the pin while applying a substantial forging force to consolidate the weld metal. The welding of the material is facilitated by severe plastic deformation in the solid state, involving dynamic re crystallization of the base material [6, 11]

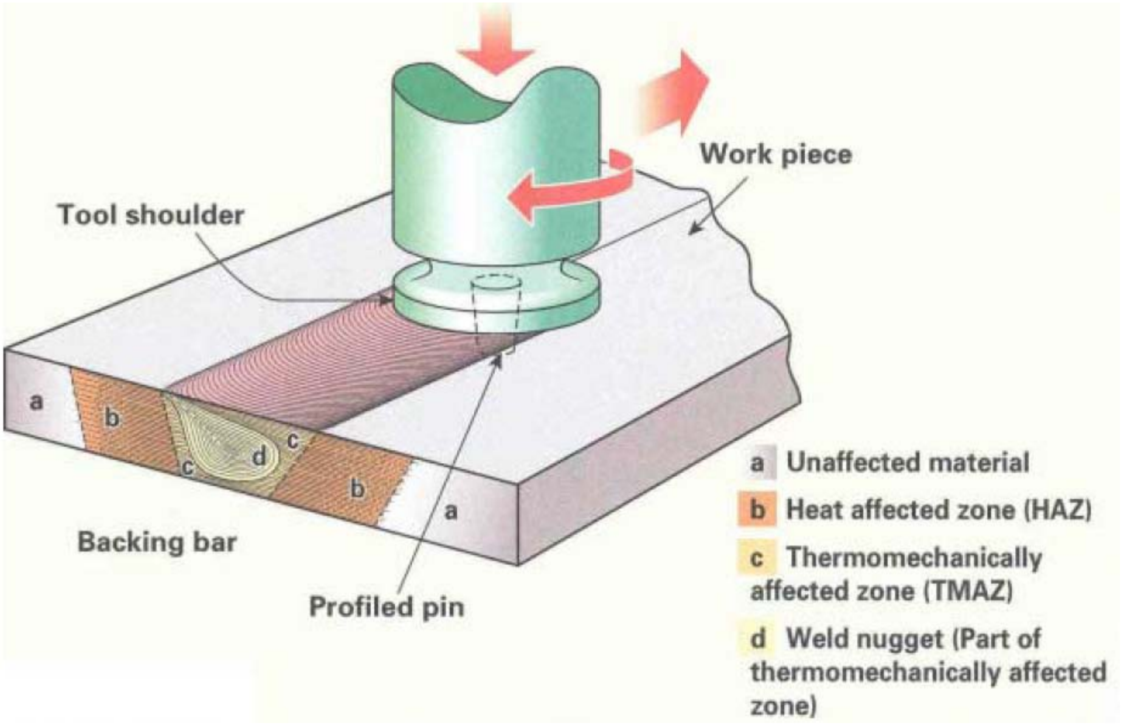

Work piece
Tool shoulder
a
b
c
d
c
b
a
Backing bar
Profiled pin
a Unaffected material
b Heat affected zone (HAZ)
c Thermomechanically affected zone (TMAZ)
d Weld nugget (Part of thermomechanically affected zone)

Figure 1-2 Schematic illustration of the FSW process [5]

1.1.4 Shortcomings in Friction Stir Welds

One of the major drivers for the use of FSW in aluminium welding is the low incidence of weld flaws as compared to that produced by conventional arc welding [17]. However, the process does have its own characteristic flaws. A number of different process variables affect the quality of the joint produced, for example - tool design, travel speeds, welding gap, thickness mismatch & plate thickness variation, tool plunge depth & tilt angle.

Successful, reproducible welds may be produced by operating within the process “windows” However, defects in weld quality spring up when the welding conditions deviate from the standard operating window. Currently, there are no set standards for evaluating the quality of friction stir welds [29].

1.2 Introduction to Artificial Neural Networks

Artificial neural networks emerged after the introduction of simplified neurons by McCulloch and Pitts in 1943. These neurons were presented as models of biological neurons and as conceptual components for circuits that could perform complex computational tasks. Inspired by the biological nervous system they evolved in to an information processing paradigm.

The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems [10].

Artificial neural networks (ANN) are amongst the newest signal-processing technologies in the engineer's toolbox. The field is highly interdisciplinary, but this study will restrict the view to the engineering perspective. In engineering, neural networks serve two important functions: as pattern classifiers and as nonlinear adaptive filters [7, 8]

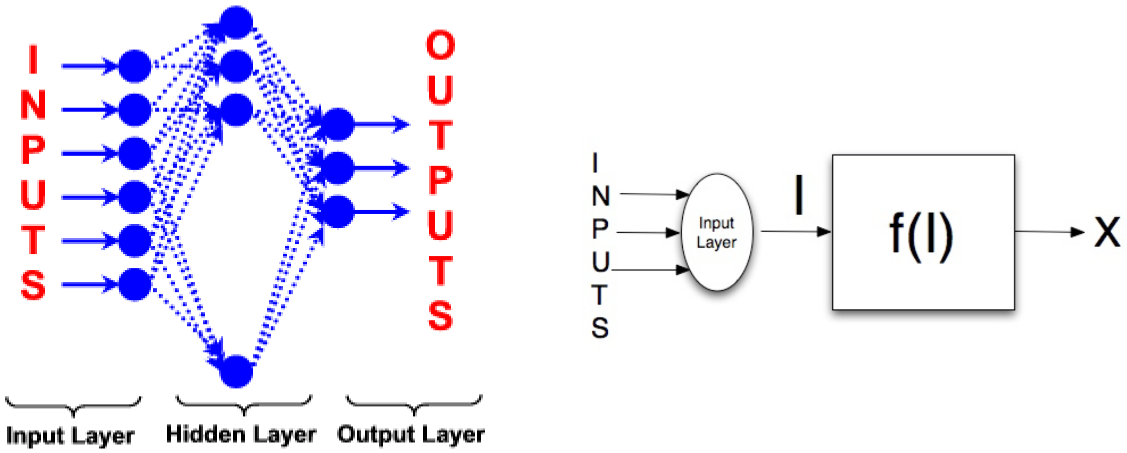


Figure 1-3 General architecture of Artificial Neural Networks [12]

1.3 Artificial Neural Networks versus Conventional Computing

Artificial Neural networks take a different approach to problem solving than conventional computers. Conventional computers use an algorithmic approach i.e. the computer follows a set of instructions in order to solve a problem. Unless the specific steps that the computer needs to follow are known or given, the computer cannot solve the problem [10]. This restricts the problem solving capability of conventional computers to problems that we already understand and know how to solve. However, standard computer algorithms would be so much more useful if they could do things that we don't exactly know how to do.

Neural networks process data and information in a way similar to the human brain [8]. The network is composed of a large number of highly interconnected processing elements (neurones) working in parallel to solve a specific problem [7, 8]. Neural networks learn by example. They cannot be programmed to perform a specific task. The examples must be selected carefully otherwise useful time is wasted or even worse the network might be functioning incorrectly.

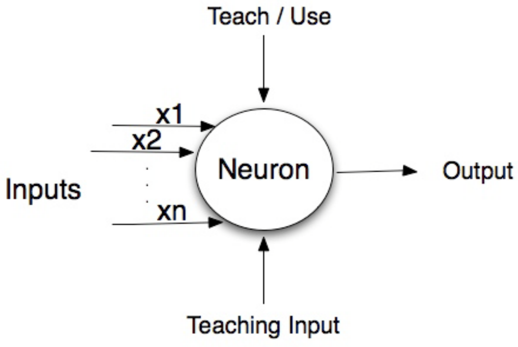


Figure 1-4 Basic neuron model

On the other hand, conventional computers use a cognitive approach to problem solving; the way the problem is to be solved must be known and stated in small unambiguous instructions. These instructions are then converted from a high level language program into machine code that the computer understands. These machines are totally predictable; if anything goes wrong it's due to a software or hardware fault.

Neural networks and conventional algorithmic computers are not in competition but complement each other. There are certain tasks are more suited to an algorithmic approach like arithmetic operations and tasks that are more suited to neural networks. Even more, a large number of tasks, require systems that use a combination of the two approaches (normally a conventional computer is used to supervise the neural network) in order to perform at maximum efficiency [27].

1.3.1 Why use Self Organising Neural Networks

Self organising neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. A trained self organising neural network can be thought of as an "expert" in the category of information it has been given to analyse. This expert can then be used to provide projections given new situations of interest and answer "what if" questions [30, 38].

Other advantages include [42]:

- *Adaptive learning*: An ability to learn and mimic tasks based on the data given for training or initial experience.
- *Self-Organisation*: An ANN can create its own organisation or representation of the information it receives during learning time.
- *Real Time Operation*: ANN computations may be carried out in parallel, and special hardware devices are being designed and manufactured which take advantage of this capability.
- *Fault Tolerance via Redundant Information Coding*: Partial destruction of a network leads to the corresponding degradation of performance. However, some network capabilities may be retained even with major network damage.

1.3.2 Real World Applications of Self Organising Artificial Neural Networks

- A great deal of research is being undertaken on neural networks worldwide. There are many different types of neural networks, each with it's own strengths particular to it's field of applications. The abilities of different networks can be related to their structure, dynamics and learning methods. Since neural networks are best at identifying patterns or trends in data, they are well suited for prediction or forecasting needs including [20]:
- Industrial process control
- Data validation
- Risk management
- Sales forecasting

- Self organising ANNs are also used in the following specific paradigms: recognition of speakers in communications; diagnosis of hepatitis; undersea mine detection; texture analysis; three-dimensional object recognition; hand-written word recognition; and facial recognition [27, 20]

1.4 Research Objectives

The ability to determine the quality of a weld is important, and in some applications critical.

Traditionally, it has been necessary to determine the weld input parameters for every new welded product to obtain a welded joint with the required specifications. This requires a time-consuming trial and error development effort, with weld input parameters chosen by the skill of the engineer or machine operator. Then welds are examined to determine whether they meet the specification or not. However, the likely applications for this technology make destructive testing economically infeasible. Also, what is not achieved or often considered is an optimised welding parameters combination, since welds can often be produced with very different parameters [21]. In other words, there is often a more ideal welding parameter combination, which can be used if it can be determined.

A number of studies have focussed on the quantitative relationship between the primary FSW inputs and measured outputs (responses) [13, 14, 15, 16]. The measured responses are the most descriptive in understanding FSW process fundamentals and may help in understanding possible monitoring and control schemes.

The goal of this research is to determine if the weld quality can be ascertained from feedback generated by the welding equipment. The friction stir welding process provides a wealth of feedback information which may be used to evaluate weld quality in a non-destructive manner. The rationale behind this study is to develop a self organising artificial neural network approach to establish predictive equations & relationships that may be used in a similar setup to pre determine weld quality. The breadth of the neural network approach is sufficient to allow the results to be truly representative of FSW.

1.4.1 Collecting Experimental Data

In this work, two types of aluminium alloys were considered for collecting data from the FSW process: AA2024 and A356 (AA2024 and A356 which have different strength characteristics near peak FSW temperatures) [19].

FSW experiments were carried out under a range of rotation and welding speeds. For each material, 22 sets of parameters were used. The resulting parameters were recorded for all welds using an external weld monitoring system plus software to display real time numerical values of forces, torque, the temperature adjacent to the system electronics and (if desired) the tool temperature. This enabled the measure of key weld parameters.

Tools were made by CNC machining, and heat treated. A normal milling machine with stepped up rotation/linear speed settings was used. “Bead on plate” FSP was conducted on 290mm long, 85 mm wide, 6.35 mm thick A356 (Al-7Si-0.3Mg) cast aluminium plates.

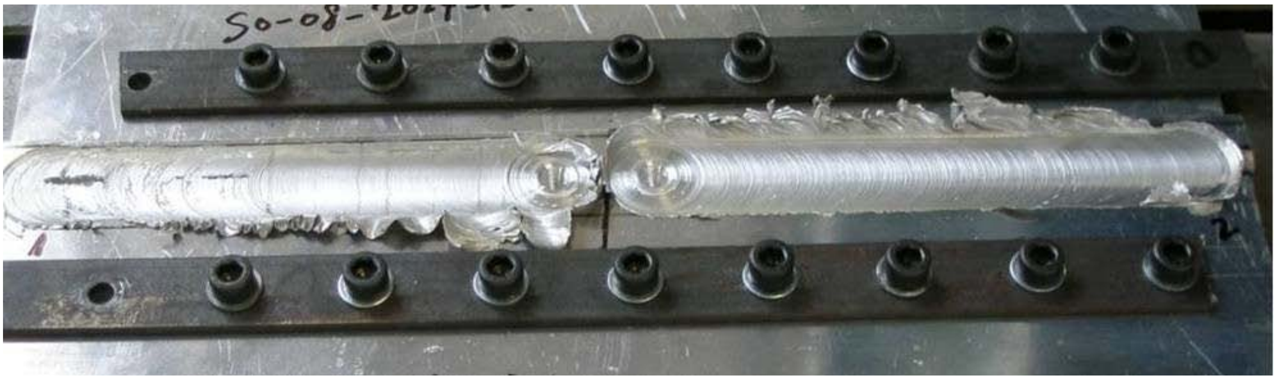


Figure 1-5 Bead on plate setup

1.4.2 ANN application and modeling data with network

The data obtained experimentally was used to explore the utilisation of Kohonen self organizing maps or SOM, which is a type of self organising neural network, for mapping and forecasting the relationships between widely different FSW parameters to assess weld quality. The value of the SOM analysis is to observe interrelationships that exist between the various FSW variables that were experimentally obtained and thereby provide a basis for generating a trained model on real data that can be experimentally examined.

The SOM does not replace existing statistical tools, but complements our ability to examine relationships between disparate types of variables in a visual presentation of the data. Visual inspection of the component planes in SOM show that there are common patterns between many sets of variables. In this model, the primary concern is weld quality. The model presented here has demonstrated the type of system which can successfully be employed to predict FSW weld quality — an artificially intelligent program, providing information on a variety of input & generated parameters, which can be successfully applied to a personal computer.

1.4.3 Evaluation of results

To develop a neural network with good performance, an adequate quantity of experimental data needs to be collected [22, 23]. The training process involves minimising the sum of square error between actual and predicted outputs, using the available training data, by continuously adjusting and finally determining the weights connecting neurons in adjacent layers.

During the training and testing process, the structure, learning algorithm and other parameters of the neural network should also be optimised to the specific problem under investigation. When the neural network is sufficiently optimally trained based on the available data, it then becomes possible to generate satisfactory results when presented with any new input data it has never experienced before.

Summing up, the objectives of this research were to:

Carry out a series of FSW trials on Aluminium alloy A253 and AA2024 against a select range of parameters.

Obtain raw feedback data from the sensory machine during a weld.

Develop an self organising neural network for simulating real world experiments

Train and test the self organising map with real world data

Evaluation of experimental results

2 Literature Review

Chapter 1 gave a brief introduction and background to FSW, self organising neural network and an outline of the research objectives. This chapter reviews the state of the art on Artificial Neural Networks and the reasons to use growing self organising networks are investigated with in the context of prediction & optimisation of FSW weld quality. Furthermore, it aims to identify the knowledge gaps that have been incorporated into the research objectives.

2.1 Artificial Neural Network – review of state-of-the-art

An artificial neural network (ANN) is a mathematical model or computational model based on biological neural networks [9]. An ANN model consists of a number of highly interconnected processing elements organized into layers, the geometry and functionality of which have been likened to that of the human brain.

ANNs learn by experience, generalize from previous experiences to new ones, and can make decisions [8, 9].

In general terms, neural networks are non-linear statistical data modelling tools. They can be used to model complex relationships between inputs and outputs or to find patterns in data. In most cases an ANN is an adaptive system that changes its structure based on external or internal information that flows through the network during the learning/training phase.

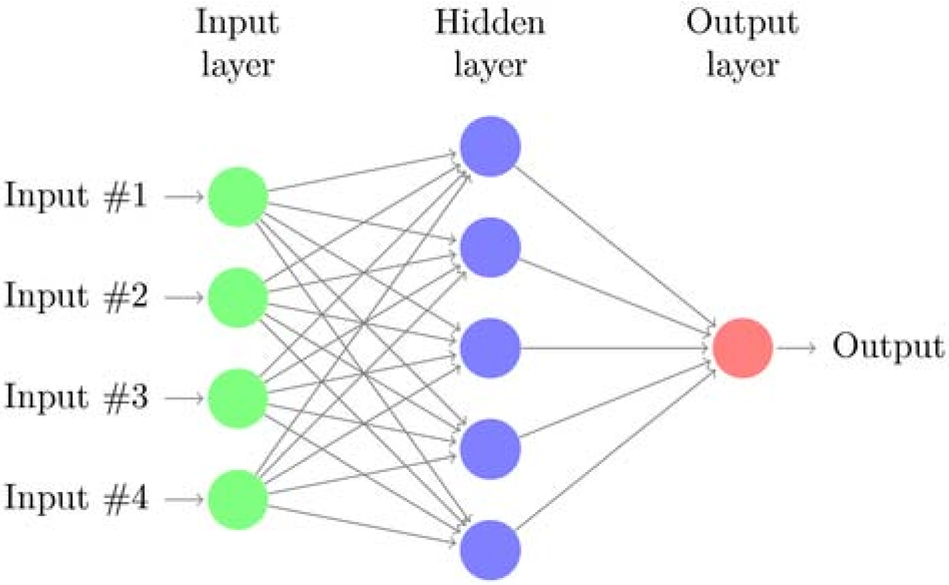


Figure 2-1 Basic neural network architecture [10]

2.1.1 Learning Paradigms

There are three major learning paradigms, each corresponding to a particular abstract learning task. These are supervised learning, unsupervised learning and reinforcement learning. Usually any given type of network architecture can be employed in any of these categories.

2.1.1.1 Supervised learning

In supervised learning, the classifier for a given input is able to classify it with respect to some kind of classification. For a system to be using supervised learning, a teacher must help the system in its model construction by defining classes and providing positive and negative examples of objects belonging to these classes. The system is then to find out common properties of the different classes, and what separates them, in order to make correct classification for other objects. Supervised document classifiers are commonly referred to as statistical document classifiers because they make use of statistical properties of category features during classification [13].

For example, say, we are given a set of example pairs and the aim is to find a function *f* in the allowed class of functions that matches the examples. In other words, we wish to infer the mapping implied by the data; the cost function is related to the mismatch between our mapping and the data and it implicitly contains prior knowledge about the problem domain.

Tasks that fall within the paradigm of supervised learning are pattern recognition (also known as classification) and regression (also known as function approximation). The supervised learning paradigm is also applicable to sequential data (e.g., for speech and gesture recognition). This can be thought of as learning with a "teacher," in the form of a function that provides continuous feedback on the quality of solutions obtained thus far [13, 27]

2.1.1.2 Unsupervised learning

This learning technique identifies groups or clusters, of related documents as well as the relationships among them. This approach is commonly referred to as clustering; because this approach eliminates the need for tagged training documents and also does not require a preexisting taxonomy or category structure. However, clustering algorithms are not always good at selecting categories that are intuitive to human users. For this reason, clustering generally works hand-in-hand with the previously described supervised learning. Kohonen’s Self Organizing Map is an unsupervised learning technique. By using Kohonen’s SOM, the data dimensionality can be reduced from a very high dimension data into 2 or 3 dimensional space. This reduction is dimensionality enables us to interpret the results easily and instinctively.

In unsupervised learning we are given some data *x*, and the cost function to be minimized can be any function of the data *x* and the network's output, *f*. The cost function is dependent on the task (what we are trying to model) and our assumptions (the implicit properties of our model, its parameters and variables) [13].

Tasks that fall within the paradigm of unsupervised learning are in general estimation problems; the applications include clustering, the estimation of statistical distributions, compression and filtering [13, 27]

2.1.1.3 Reinforcement learning

In reinforcement learning, data *x* is usually not given, but generated by an agent's interactions with the environment. At each point in time *t*, the agent performs an action *yt* and the environment generates an observation *xt* and an instantaneous cost *ct*, according to some (usually unknown) dynamics. The aim is to discover a policy for selecting actions that minimises some measure of a long-term cost, i.e. the expected cumulative cost. The environment's dynamics and the long-term cost for each policy are usually unknown, but can be estimated. ANNs are frequently used in reinforcement learning as part of the overall algorithm [7, 13].

Tasks that fall within the paradigm of reinforcement learning are control problems, games and other sequential decision making tasks.

2.1.2 Useful properties of Artificial Neural networks

2.1.2.1 Computational Power

The multi-layer perceptron (MLP) is a universal function approximator, as proven by the Cybenko theorem. However, the proof is not constructive regarding the number of neurons required or the settings of the weights [32].

Work by Hava T. Siegelman [25] has provided proof that a specific recurrent architecture with rational valued weights (as opposed to the commonly used floating point approximations) has the full power of a Universal Turing Machine. They have further shown that the use of irrational values for weights results in a machine with trans-turing power.

2.1.2.2 Capacity

Artificial neural network models have a property called 'capacity', which roughly corresponds to their ability to model any given function. It is related to the amount of information that can be stored in the network and to the notion of complexity [32].

2.1.2.3 Convergence

Nothing specific can be said in general about convergence since it depends on a number of factors. Firstly, multiple local minima may exist. This depends on the cost function and the model. Secondly, the optimization method used might not be guaranteed to converge when far away from a local minimum. Thirdly, for a very large amount of data or parameters, some methods become impractical. In general, it has been found that theoretical guarantees regarding convergence are not always a very reliable guide to practical application [32].

2.1.2.4 Generalisation and statistics

In applications where the goal is to create a system that generalises well in unseen examples, the problem of overtraining has emerged. This arises in over-complex or over-specified systems when the capacity of the network significantly exceeds the needed free parameters. There are two schools of thought for avoiding this problem: The first is to use cross-validation and similar techniques to check for the presence of overtraining and optimally select hyper parameters to minimise the generalisation error. The second is to use some form of regularisation. This is a concept that emerges naturally in a probabilistic (Bayesian) framework, where the regularisation can be performed by putting a larger prior probability over simpler models; but also in statistical learning theory, where the goal is to minimise over two quantities: the 'empirical risk' and the 'structural risk', which roughly correspond to the error over the training set and the predicted error in unseen data due to over fitting [25, 32].

By assigning a softmax activation function on the output layer of the neural network (or a softmax component in a component-based neural network) for categorical target variables, the outputs can be interpreted as posterior probabilities. This is very useful in classification as it gives a certainty measure on classifications.

2.1.3 Network Architecture

An ANN is typically composed of layers of nodes. The popular Multi Layer Perceptron network, all the input nodes are in one input layer, all the output nodes are in one output layer and the hidden nodes are distributed into one or more hidden layers in between. In designing an MLP, one must determine the following variables:

2.1.3.1 The number of input nodes

The number of input nodes corresponds to the number of variables in the input vector used to predict future values. For casual prediction, the number of inputs is usually transparent and relatively easy to choose. In a time series prediction problem, the number of input nodes corresponds to the number of lagged observations to make forecasts for future values. However, currently there is no suggested systematic way to determine this number [32]. The selection of this parameter should be included in the model construction process. Ideally, a small number of essential nodes are desired which can unveil the unique features embedded in the data. Too few or too many input nodes can affect either the learning or the prediction capability of the network [36].

2.1.3.2 The number of hidden layers and hidden nodes

The hidden layer and nodes play very important roles for successful applications of neural networks. It is the hidden nodes in the hidden layer that allow neural networks to detect the feature, to capture the pattern in the data, and to perform complicated nonlinear mapping between input and output variables. It is clear that without hidden nodes, simple perceptrons are equivalent to linear statistical forecasting models. The most common way in determining the number of hidden nodes is via experiments or by trial-and error. Several rules of thumb have also been proposed, such as, the number of hidden nodes depends on the number of input patterns and each weight should have at least ten input patterns (sample size) [36].

The issue of determining the optimal number of hidden nodes is a crucial yet complicated one. In general, networks with fewer hidden nodes are preferable as they usually have better generalization ability. But networks with too few hidden nodes may not have enough power to model and learn the data [32].

2.1.3.3 The number of output nodes.

The number of output nodes is relatively easy to specify as it is directly related to the problem under study. For a time series forecasting problem, the number of output nodes often corresponds to the forecasting window. There are two types of forecasting: one-step-ahead (which uses one output node) and multi-step-ahead forecasting.

The first is called the iterative forecasting as used in the Box-Jenkins model in which the forecast values are iteratively used as inputs for the next forecasts. In this case, only one output node is necessary [32].

The second called the direct method is to let the neural network have several output nodes to directly forecast each step into the future.

Results from Zhang et al. [34] show that the direct prediction is much better than the iterated method. However, Weigend et al. [43] report that the direct multi-step prediction performs significantly worse than the iterated single-step prediction for the sunspot data. Hill et al. [42] conclude similar findings for 111 M-competition time series.

In any case, the selection of these parameters is basically problem-dependent. Furthermore none of these methods can guarantee the optimal solution for all real forecasting problems. To date, there is no simple clear-cut method for determination of these parameters. Guidelines are either heuristic or based on simulations derived from limited experiments [36].

2.1.4 Training and Data Normalisation

Training and test samples are typically required for building an ANN predicter model. The training sample is used for ANN model development and the test sample is adopted for evaluating the forecasting ability of the model. Sometimes a third one called the validation sample is also utilised to avoid overfitting problem or to determine the stopping point of the training process [32].

2.1.4.1 Data Normalisation

Data normalisation is often performed before the training process begins when nonlinear transfer functions are used at the output nodes, the desired output values must be transformed to the range of the actual outputs of the network. Even if a linear output transfer function is used, it may still be advantageous to standardize the outputs as well as the inputs to avoid computational problems to meet algorithm requirement, and to facilitate network learning

Four methods for input normalization were summarised by Weigend et.al's research [43]:

1. Along channel normalization: A channel is defined as a set of elements in the same position over all input vectors in the training or test set. That is, each channel can be thought of as an ‘‘independent’’ input variable.

The along channel normalization is performed column by column if the input vectors are put into a matrix.

2. Across channel normalization: This type of normalization is performed for each input vector independently, that is, normalization is across all the elements in a data pattern.
3. Mixed channel normalization: As the name suggests, this method uses some kind of combinations of along and across normalization.
4. External normalization: All the training data are normalized into a specific range.

The choice of the above methods usually depends on the composition of the input vector. It should be noted that, as a result of normalizing the target values, the observed output of the network will correspond to the normalized range. Thus, to interpret the results obtained from the network, the outputs must be rescaled to the original range. From the user’s point of view, the accuracy obtained by the ANNs should be based on the rescaled data set. Performance measures should also be calculated based on the rescaled outputs. However only a few authors clearly state whether the performance measures are calculated on the original or transformed scale [31, 43, 44].

2.1.4.2 Training Algorithm

The neural network training is an unconstrained nonlinear minimization problem in which arc weights of a network are iteratively modified to minimize the overall mean or total squared error between the desired and actual output values for all output nodes over all input patterns. The existence of many different optimization methods [10] provides various choices for neural network training.

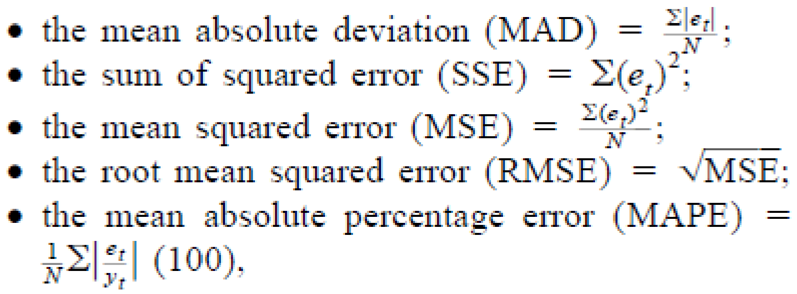
There is no algorithm currently available to guarantee the global optimal solution for a general nonlinear optimization problem in a reasonable amount of time. As such, all optimization algorithms in practice inevitably suffer from the local optima problems, and the most we can do is to use the available optimization method which can give the ‘‘best’’ local optima if the true global solution is not available.

Aleksander.I and Morton.H [9] conclude that the training parameters play a critical role in the performance of ANNs. Using different learning parameters, they re-tested the performance of ANNs for several time series which have been previously reported to have worse results with ANNs. They find that for each of these time series there is an ANN with appropriate learning parameters, which performs significantly better. Zhang et al. [31] also study the effect of training parameters on the ANN learning. They report that high learning rate is good for less complex data and low learning rate with high momentum should be used for more complex data series. However, there are inconsistent conclusions with regard to the best learning parameters. In light of the weakness of the conventional back propagation algorithm, a number of variations or modifications of back-propagation, such as the adaptive method [42], and Kohonen’s feature maps have been proposed [39].

Among them, the kohonen’s feature maps are more efficient for nonlinear optimisation methods [39]. Their faster convergence, robustness, and the ability to find good local minima make them attractive in ANN training. Kohonen’s feature maps are motivated by the self-organising behavior of the human brain

2.1.5 Performance Measures

Although there can be many performance measures for an ANN forecaster like the modelling time and training time, the ultimate and the most important measure of performance is the prediction accuracy it can achieve beyond the training data. However, a suitable measure of accuracy for a given problem is not universally accepted by the forecasting academicians and practitioners. An accuracy measure is often defined in terms of the forecasting error which is the difference between the actual and the predicted value. There are a number of measures of accuracy in the forecasting literature and each has advantages and limitations [31, 44]. The most frequently used are;



- the mean absolute deviation (MAD) $= \frac{\Sigma|e_t|}{N}$;
- the sum of squared error (SSE) $= \Sigma(e_t)^2$;
- the mean squared error (MSE) $= \frac{\Sigma(e_t)^2}{N}$;
- the root mean squared error (RMSE) $= \sqrt{\text{MSE}}$;
- the mean absolute percentage error (MAPE) $= \frac{1}{N}\Sigma\left|\frac{e_t}{y_t}\right|(100)$,

where *et* is the individual forecast error; *yt* is the actual value; and *N* is the number of error terms. Because of the limitations associated with each individual measure, one may use multiple performance measures in a particular problem. However, one method judged to be the best along one dimension is not necessarily the best in terms of other dimensions [31].

There are many inconsistent reports in the literature on the performance of ANNs for forecasting tasks. The main reason is, that a large number of factors including network structure, training method, and sample data may affect the forecasting ability of the networks. For some cases where ANNs perform worse than linear statistical models, the reason may simply be that the data is linear without much disturbance. We cannot expect ANNs to do better than linear models for linear relationships. In other cases, it may simply be that the ideal network structure is not used for the data set. Table 2-1 summarizes the literature on the relative performance of ANNs.

Study	Data	Conclusions
Brace et al. (1991)	8 electric load series (daily)	ANNs are not as good as traditional methods
De Groot and Wurtz (1991)	Sunspots activity time series (yearly)	ANNs are not the best but comparable to the best linear or nonlinear statistical model.
Hann and Steurer (1996)	Weekly and monthly exchange rate	ANNs outperform the linear models for weekly & monthly data.
Kohzadi et al. (1996)	Monthly live cattle and wheat prices	ANNs are considerably and consistently better and can find more turning points
Srinivasan et al. (1994)	One set of load data	ANNs are better than regression and liner models
Marquez et al. (1992)	Simulated data for 3 regression models	ANNs perform comparatively as well as regression models
Nam and Schaefer (1995)	One airline passenger data (monthly)	ANNs are better than time series regression and exponential smoothing
Gorr et al. (1994)	Student grade point averages	Slight improvement with ANNs in predicting students' GPAs over linear models
Weigend et al. (1992)	Exchange rate (daily)	ANNs perform better than TAR and bilinear models. ANNs are significantly better than random walk model.
Lachtermacher and Fuller (1995)	4 stationary river flow and 4 non-stationary electricity load time series (yearly)	For stationary time series, ANNs have a better overall performance than traditional methods; for non-stationary series ANNs are almost much better than other linear models.

Table 2 Neural network applications to non-linear data [42]

2.1.6 Application Areas of ANN

Generally, ANNs are more robust and outperform other computational tools in solving a variety of problems from seven categories.

2.1.6.1 Pattern Classification

Pattern classification deals with assigning an unknown input pattern, using unsupervised or supervised learning, to one of several pre-specified classes based on one or more properties that characterize a given class, as shown in Fig 2-2. Classification applications from the area of microbiology include classification of commodities based on their microbiological characteristics [32]. Unlike discriminant analysis in statistics, ANNs do not require the linearity assumption and can be applied to nonlinearly separable classes [44].

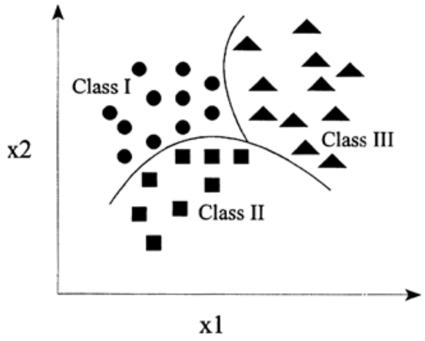


Figure 2-2 Pattern classification

2.1.6.2 Clustering

Clustering is performed via unsupervised learning in which clusters (classes) are formed by exploring the similarities or dissimilarities between the input patterns based on their inter-correlations (Fig. 2-3). The network assigns ‘similar’ patterns to the same cluster. Example applications from microbiology include sub-species discrimination using pyrolysis mass spectrometry and Kohonen network [38].

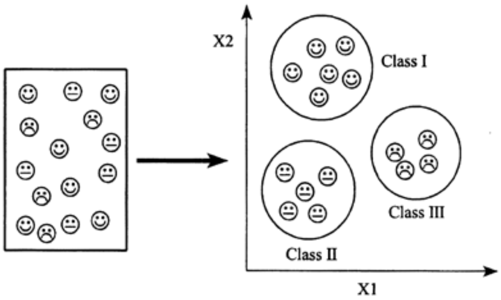


Figure 2-3 Clustering

2.1.6.3 Function Approximation

Function approximation (modeling) involves training ANN on input–output data to approximate the underlying rules relating the inputs to the outputs. Multilayer ANNs are considered universal approximators that can approximate any arbitrary function to any degree of accuracy [41], and thus are normally used in this application. Function approximation is applied to problems (i) where no theoretical model is available, i.e., data obtained from experiments or observations are utilised, or (ii) to substitute theoretical models that are hard to compute analytically by utilizing data obtained from such models

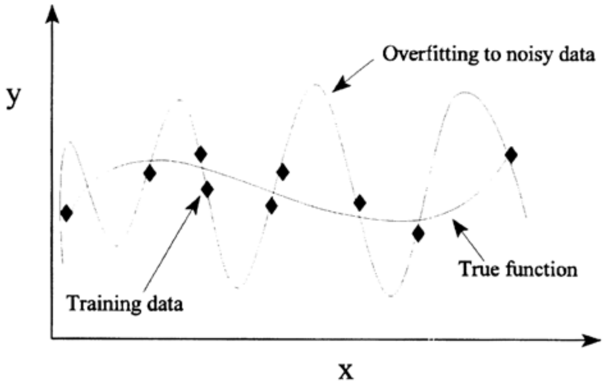


Figure 2-4 Function Approximation

2.1.6.4 Forecasting

Successful applications of ANNs in forecasting were reported by G Simona et.al and Hill et.al [60, 44]. Using two deterministic chaotic time series generated by the logistic map and the Glass-Mackey equation, they designed the feed-forward neural networks that can accurately mimic and predict such dynamic nonlinear systems. Their results show that ANNs can be used for modeling and forecasting nonlinear time series with very high accuracy.

The sunspot series has long served as a benchmark and has been well studied in statistical literature. ANNs have been used for forecasting business failure and traffic [60, 62].

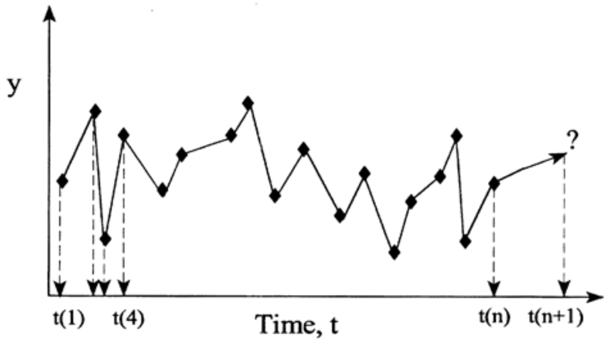


Figure 2-5 Forecasting

2.1.6.5 Optimisation

Optimization is concerned with finding a solution that maximizes or minimizes an objective function subject to a set of constraints. Optimization is a well-established field in mathematics. However ANNs, such as the Hopfield network, were found to be more efficient in solving complex and nonlinear optimization problems [60]

2.1.6.6 Association

Association involves developing a pattern association ANN by training on ideal (noise-free) data and subsequently using this ANN to classify (noisy) corrupted data. The associative network may also be used to correct (reconstruct) the corrupted data or complete the missing data (or image), as shown in Fig. 2-6. Hopfield and Hamming networks are especially used for this application [63], and to a lesser degree multilayer back-propagation ANNs trained on patterns with identical input and output [41].

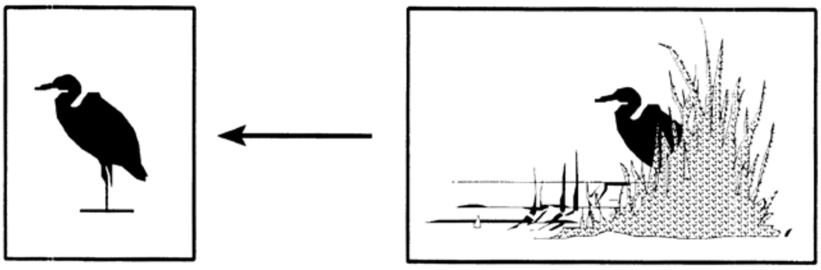


Figure 2-6 Image Completion

2.1.6.7 Control

Control is concerned with designing a network normally recurrent, which will aid an adaptive control system to generate the required control inputs such that the system will follow a certain trajectory based on system feedback [58].

2.2 State of the Art in Friction Stir Welding

Welding input parameters play a very significant role in determining the quality of a weld joint. This section reviews the literature on FSW within the context of this study. In this section, the present understanding of mechanical processes during FSW is reviewed.

2.2.1 Process Parameters

FSW/FSP involves complex material movement and plastic deformation. Welding parameters, tool geometry, and joint design exert significant effect on the material flow pattern and temperature distribution, thereby influencing the microstructural evolution of material [14]. In this section, a few major factors affecting FSW/FSP process, such as tool geometry, welding parameters, joint design are addressed.

2.2.1.1 Welding Parameters

For FSW, two parameters are very important [6]:

- *Tool rotation rate (v, rpm)* in clockwise or counter clockwise direction and
- *Tool traverse speed (n, mm/min)* along the line of joint.

The rotation of tool results in stirring and mixing of material around the rotating pin and the translation of tool moves the stirred material from the front to the back of the pin and finishes welding process. Higher tool rotation rates generate higher temperature because of higher friction heating and result in more intense stirring and mixing of material as will be discussed later. However, it should be noted that frictional coupling of tool surface with work piece is going to govern the heating.

In addition to the tool rotation rate and traverse speed, another important process parameter is the angle of spindle or tool tilt with respect to the work piece surface [18]. A suitable tilt of the spindle towards trailing direction ensures that the shoulder of the tool holds the stirred material by threaded pin and move material efficiently from the front to the back of the pin. Further, the insertion depth of pin into the workpieces (also called target depth) is important for producing sound welds with smooth tool shoulders. The insertion depth of pin is associated with the pin height. When the insertion depth is too shallow, the shoulder of tool does not contact the original work piece surface. Thus, rotating shoulder cannot move the stirred material efficiently from the front to the back of the pin, resulting in generation of welds with inner channel or surface groove [33]. When the insertion depth is too deep, the shoulder of tool plunges into the work piece creating excessive flash. In this case, a significantly concave weld is produced, leading to local thinning of the welded plates [35]. Such tools are particularly preferred for curved joints.

Preheating or cooling can also be important for some specific FSW processes. For materials with high melting point such as steel and titanium or high conductivity such as copper, the heat produced by friction and stirring may be not sufficient to soften and plasticise the material around the rotating tool [33]. Thus, it is difficult to produce continuous defect-free weld.

As for tilt angle (θ), Fujii et al [36] observed that increase θ from 1.5 to 3 degrees increased the welding resistant force by 20 percent in a conventional shoulder tool FSW.

2.2.1.2 Tool Geometry

Tool geometry is the most influential aspect of process development. The tool geometry plays a critical role in material flow and in turn governs the traverse rate at which FSW can be conducted. An FSW tool consists of a shoulder and a pin as shown schematically in Fig. 2-6. As mentioned earlier, the tool has two primary functions: (a) localized heating, and (b) material flow.

In the initial stage of tool plunge, the heating results primarily from the friction between pin and work piece. The tool is plunged till the shoulder touches the work piece. The friction between the shoulder and work piece results in the biggest component of heating [28]. From the heating aspect, the relative size of pin and shoulder is important, and the other design features are not critical. The shoulder also provides confinement for the heated volume of material.

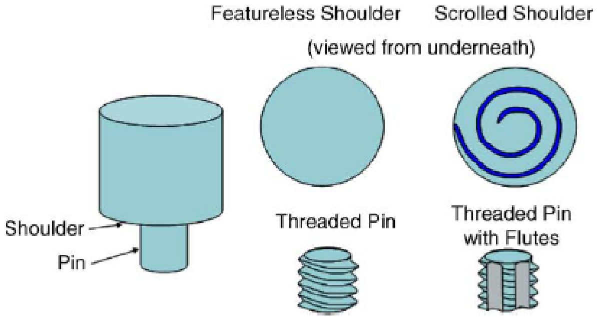


Figure 2-7 Schematic drawing of the FSW tool [11]

The second function of the tool is to ‘stir’ and ‘move’ the material. The uniformity of microstructure and properties as well as the process loads are governed by the tool design [11].

2.2.2 Welding Forces

During welding a number of forces will act on the tool [14]:

- A downwards force is necessary to maintain the position of the tool at or below the material surface. Some friction-stir welding machines operate under load control but in many cases the vertical position of the tool are preset and so the load will vary during welding.
- The traverse force acts parallel to the tool motion and is positive in the traverse direction. Since this force arises as a result of the resistance of the material to the motion of the tool it might be expected that this force will decrease as the temperature of the material around the tool is increased.
- The lateral force may act perpendicular to the tool traverse direction and is defined here as positive towards the advancing side of the weld.
- A torque is required to rotate the tool, the value of which will depend on the down force and friction coefficient (sliding friction) and/or the flow strength of the material in the surrounding region (sticking friction).

In order to prevent tool fracture or simply to minimise excessive wear and tear on the tool and associated machinery it is desirable to modify the welding cycle such that the forces acting on the tool are as low as possible and that sudden changes are avoided [14]. In order to find the best combination of welding parameters it is likely that a compromise must be reached since the conditions that favour low forces (e.g. high heat input, low travel speeds) may be undesirable from the point of view of productivity and weld properties [14, 33].

2.2.3 Process Modelling

Computational and non linear models are essential to the future of FSW. These will help elucidate many of the fundamental principles presently unknown [28]. They can be summarised as:

- Relationships between process inputs and measured outputs leading to good weld quality [17]
- Characterisation of the nature of heat generation and dissipation [21]
- Assessing the physics of the tool/weld metal interface [19]
- Marker studies [18]

An understanding of mechanical and thermal processes during FSW/FSP is needed for optimising weld quality. In this section, the present understanding of mechanical and thermal processes during FSW is reviewed.

2.2.3.1 Metal flow

The material flow during friction stir welding is quite complex depending on the process parameters, tool geometry and material to be welded. It is of practical importance to understand the material flow characteristics for optimal tool design and obtain high structural efficiency welds. This has led to numerous investigations on material flow behaviour during FSW [15, 19]. A number of approaches, such as tracer technique by marker, welding of dissimilar alloys/metals, have been used to visualize material flow pattern in FSW [18].

2.2.3.2 Temperature distribution

The effect of FSW parameters on temperature was further examined by Robert and Messler

[28]. They reported that for a given tool geometry and depth of penetration, the maximum temperature was observed to be a strong function of the rotation rate (v, rpm) while the rate of heating was a strong function of the traverse speed (n, rpm). It was also noted that there was a slightly higher temperature on the advancing side of the joint where the tangential velocity vector direction was same as the forward velocity vector. They measured the average maximum temperature on 6.35 mm aluminium plates as a function of the pseudo ‘‘heat index”. It was demonstrated that for several aluminium alloys a general relationship between maximum welding temperature and FSW parameters can be explained by $\frac{T}{T_{\mathrm{m}}} = K\left(\frac{\omega^2}{\nu \times 10^4}\right)^{\alpha}$, where the exponent α was reported to range from 0.04 to 0.06, the constant K is between 0.65 and 0.75, and Tm (degree celsius) is the melting point of the alloy.

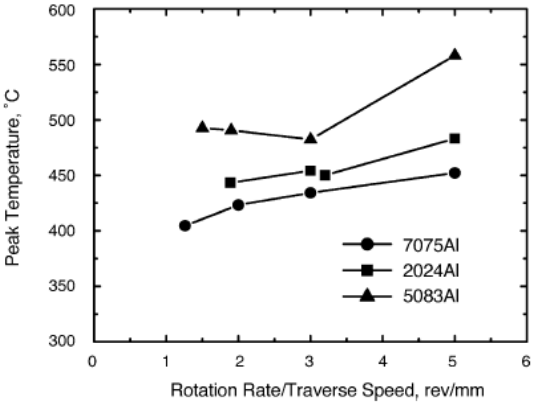


Figure 2-8 Effect of tool rotation rate/traverse speed (v/n) ratio on peak temperature of FSW 2024Al-T6, 5083Al-O, and 7075Al-T6 [28]

2.2.4 Factors affecting the Weld Quality

Weld Quality is strongly dependent on welding parameters, base material & tool geometry [29]. Factors affecting weld zone formation and weld imperfections will be reviewed in this section

2.2.4.1 Welding Parameters & Tool Geometry

Fujii et al [33] studied the effect of FSW tool shapes and welding parameters on the weld quality. The experiments were conducted using three different tool pin profiles as shown in figure 2-9.

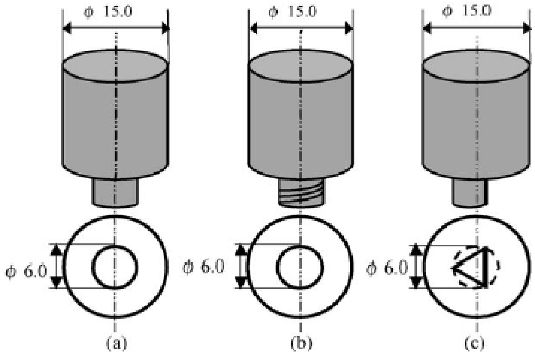


Figure 2-9 FSW tool shape (a) column without threads, (b) column with threads, (c) triangular prism [33]

Figure 2-10 shows the macrostructure of welds produced at various travelling speeds and tool shapes, as obtained by Fuji et al. It can be observed from the figures, there is a defect appearing in the weld obtained by a triangular prism tool FSW 38 at 1500rpm (J) and 1000mm/min (V). However, there are no defects in the welds produced by the same tool rotating at the same speed (J) but a lower travelling speed of 100 and 400mm/min. This indicates that a faster tool travelling speed (V) results in an increase in the likelihood of welding defects which correlates with another study conducted by Kim et al [34].

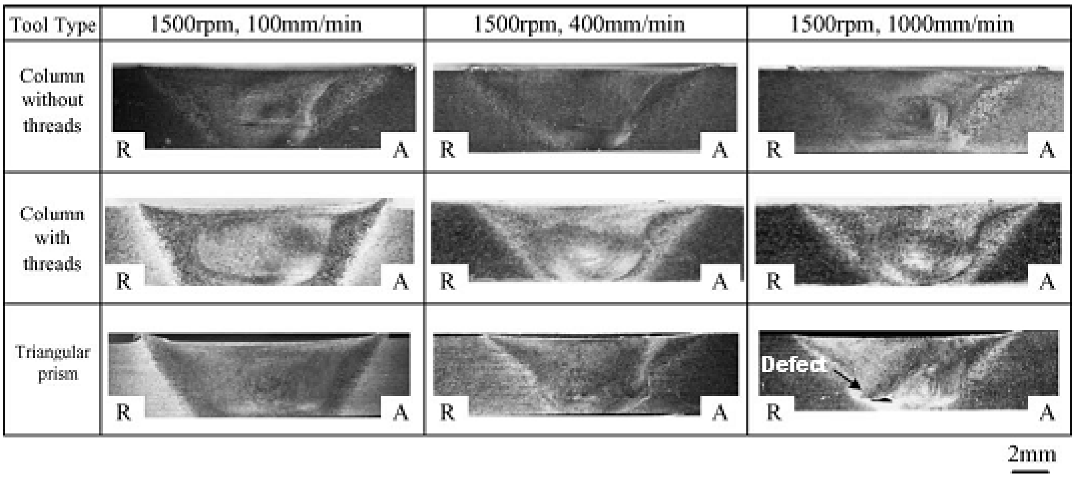


Figure 2-10 Macrostructure of cross sections of 6061 Al FSW joints, showing the weld quality at various welding parameters and tool types [33]

Shoulder flow zone forming mechanisms were studied by using a special experimental setting [35]. Figure 2-11 shows the experimental setup, where the interaction between the conventional shoulder tool with the base material is increased progressively, leading to a greater depth of shoulder penetration.

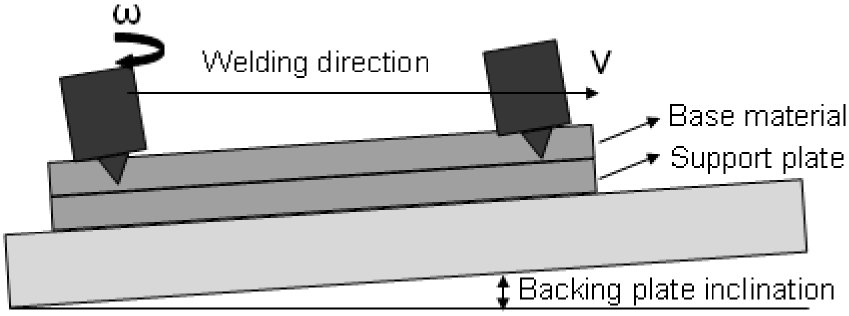


Figure 2-11 Schematic illustration of the experimental set up; the backing plate is kept at an angle such that the axial load can be linearly increased (from 4 to 10.9 kN) [35]

The primary reason for the defect in the welds, at the initial stages, where the axial load was less than 7.4 kN, was shallow shoulder penetration (lack of shoulder contact with the base material). When the depth of shoulder penetration was increased the axial load increased and when the axial load was above 7.4 kN, the shoulder flow zone material from the leading edge was confined in the weld cavity, and then a sufficient amount of frictional heat and hydrostatic pressure was generated to produce a defect-free weld.

Accordingly, the research concluded that deeper shoulder penetration lead to a significant decrease in the likelihood of welding defects.

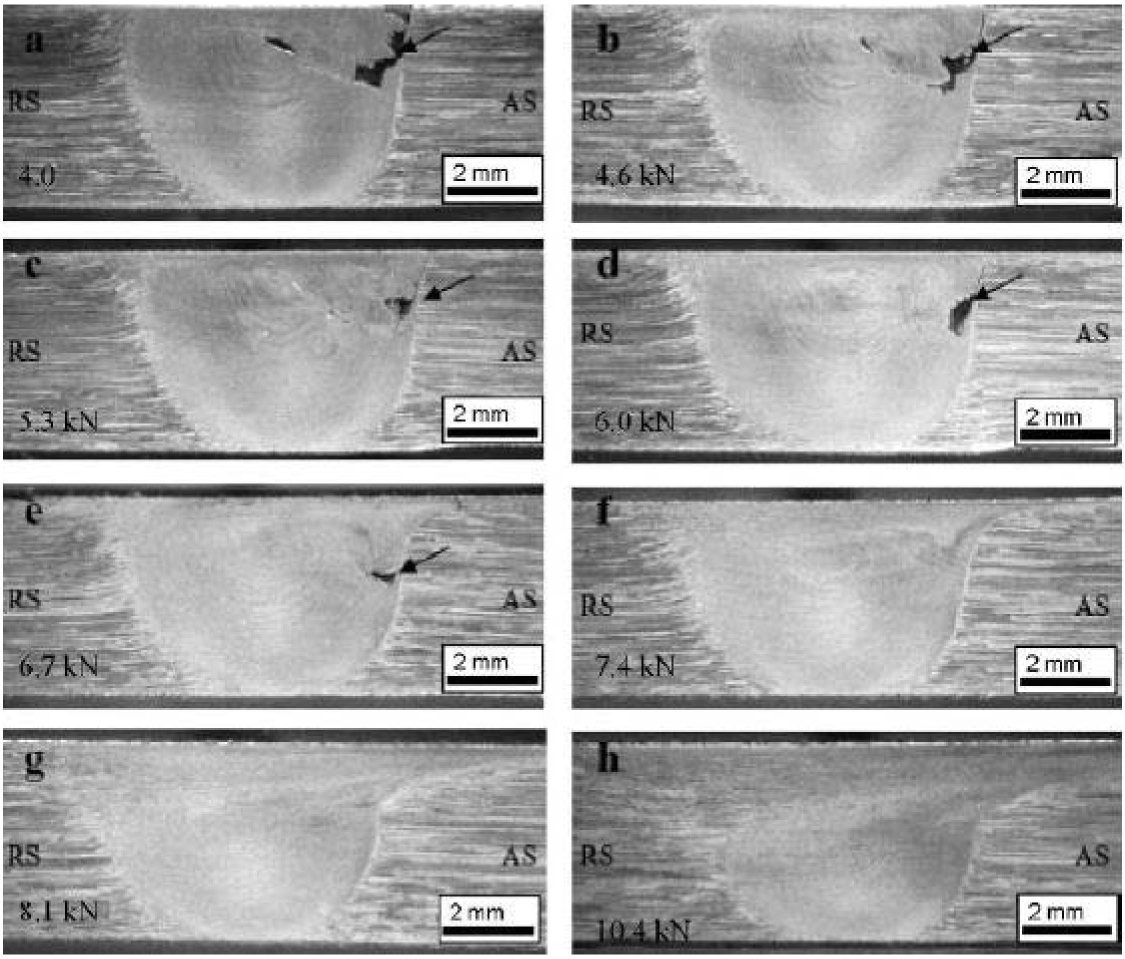


Figure 2-12 Evolution of a shoulder flow zone formation as a function of the downward force, arrow marks indicate the presence of voids in the weld [35]

2.2.5 FSW Defects & Controls Mechanisms

Leonard et al [29] investigated the imperfections of FSW welds, and they reported that there are three common defects encountered during FSW. Figure 2-13 shows these three typical defects, which are internal voids, joint line remnants, and root flaws. They observed that internal voids normally occurred on the advancing side of the weld, and internal void formation was due to insufficient forging pressure and excessive welding speed.

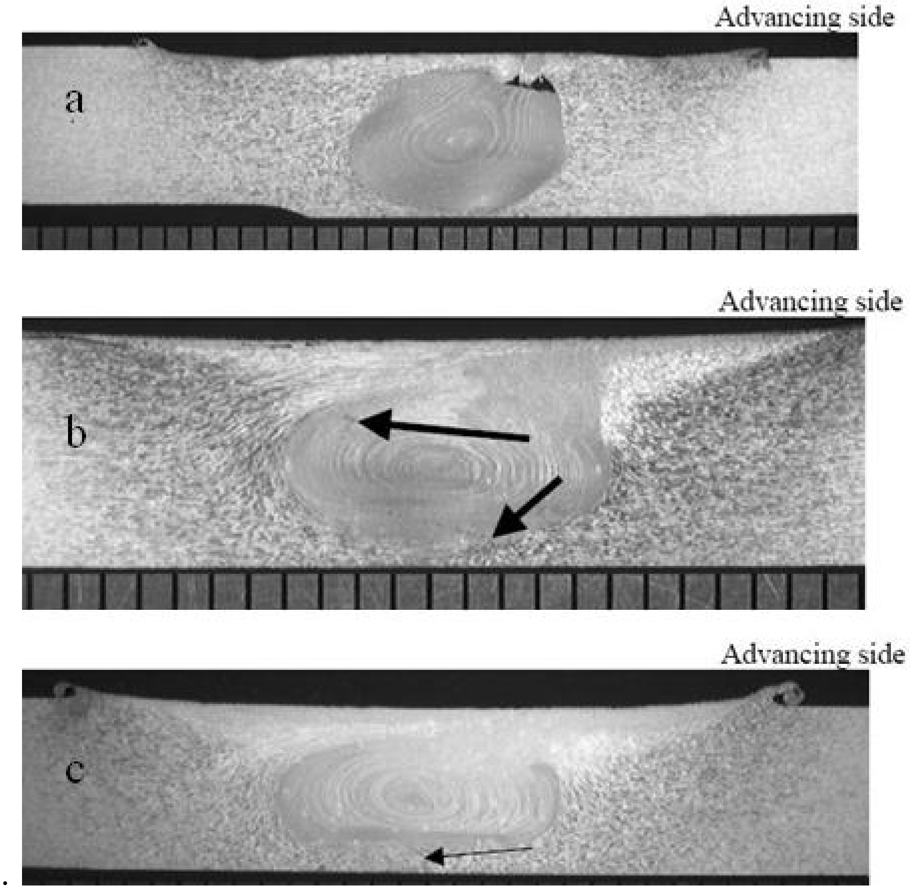


Figure 2-13 Cross sections of defect weld (a) internal voids, (b) joint line remnants, and (c) root

Leonard et al also recommended several strategies to reduce this type of weld imperfection, such as sufficient cleaning of the workpiece, correct tool location versus weld joint-line, suitable shoulder diameters and tool welding speeds etc.

Further, they also concluded that there were quite a few causes of root flaws, including variation of plate thickness, incorrect tool position, and improper tool design. They advised that a sufficient depth of shoulder penetration could eliminate root defects. Leonard et al's findings and suggestions are well supported by Kim et al [34] in investigation of FSW weld defect mechanism.

2.3 History of Employing Artificial Neural Network in related research

Since neural networks are best at identifying patterns or trends in data, they are well suited for predicting and optimising. Their ability to learn by example makes them very flexible and powerful. Furthermore there is no need to devise an algorithm in order to perform a specific task; i.e. there is no need to understand the internal mechanisms of that task.

They are also very well suited for real time systems because of their fast response and computational times which are due to their parallel architecture [10]. A recent survey conducted on optimisation of different welding parameters using various types of modelling approaches has revealed a high level of interest in adapting Artificial Neural Networks to predict response(s) and optimise friction stir welding [20,27].

Anderson et al. [22] have explained some concepts related to neural networks and how they can be used to model weld-bead geometry, in terms of equipment parameters, in order to evaluate the accuracy of neural networks for weld modelling. They carried out a number of simulations and they used actual data for this purpose.

The data consisted of values for voltage, current, electrode travel speed and wire feed speed and the corresponding bead width, penetration, reinforcement height and bead cross-sectional area. The performance of neural networks for weld modelling was presented and evaluated using actual welding data. It was concluded that the accuracy of neural networks modelling is fully comparable with the accuracy achieved by more traditional modelling schemes.

Evaluation of ANN for monitoring and control of the plasma arc welding process was carried out by Cook et al. [24]. Three areas of welding application were investigated in this work: weld process modelling, weld process control and weld-bead profile analysis for quality control.

A network was constructed to determine the torch standoff, forward current, reverse current and travel speed for desired crown width and root width. The base material was 2219 aluminium alloy in the form of plates 6.35 mm thick; the joint type was bead-on-plate. It was confirmed that ANNs are powerful tools for analysis, modelling and control of such applications.

Furthermore, the results obtained when analysing weld profile data suggested that ANNs can yield real-time results of equal or better accuracy and reliability than previously used data analysis algorithms.

Y.K. Yousif et al. [36] developed for the analysis and simulation of the correlation between the friction stir welding (FSW) parameters of aluminum (Al) plates and mechanical properties.

The input parameters of the model consist of weld speed (Ws) and tool rotation speed (Rs). The outputs of the ANN model include property parameters namely: tensile strength, yield strength and elongation. The aim of this paper was to show the possibility of the use of neural networks for the calculation of the mechanical properties of welded Al plates using FSW method. Results showed that, the networks can be used as an alternative way in these systems.

The model can be used to calculate mechanical properties of welded Al plates as functions of weld & rotation speeds. The combined influence of weld & rotation speeds on the mechanical properties of welded Al plates was simulated.

A comparison was made between measured and calculated data. It is found that the correlations between the measured and predicted values of tensile strength, bending stress were in good agreement with measured data.

H. Okuyucu et al. [24] developed an artificial neural network (ANN) model for the analysis and simulation of the correlation between the friction stir welding (FSW) parameters of aluminium (Al) plates and mechanical properties.

The input parameters of the model consist of weld speed and tool rotation speed (TRS). The outputs of the ANN model include property parameters namely: tensile strength, yield strength, elongation, hardness of weld metal and hardness of heat effected zone (HAZ). Good performance of the ANN model was achieved. The model can be used to calculate mechanical properties of welded Al plates as functions of weld & tool rotation speeds. The combined influence of weld speed and TRS on the mechanical properties of welded Al plates was simulated. A comparison was made between measured and calculated data. The calculated results were in good agreement with measured data. The aim of the paper was to show the possibility of the use of neural networks for the calculation of the mechanical properties of welded Al plates using FSW method. Results showed that, the networks can be used as an alternative in these systems.

Vitek et al. [49] have developed a model to predict the weld pool shape parameters (penetration, width, width at half-penetration and cross-section area) in pulsed Nd- YAG laser welds of Al-alloy 5754 using neural network.

They considered the following process parameters; travel speed, average power, pulse energy and pulse duration. The accuracy of the model was excellent. They concluded that this approach allows for instantaneous results and therefore, offers advantages in applications where real-time predictions are needed and computationally intensive predictions are too slow.

L Fratini and G Buffa [50] studied the continuous dynamic re-crystallisation phenomena occurring in the FSW of Al alloys. A good agreement with the experimental results was obtained using the ANN model. In regard to ANNs, it noted that ANNs perform better than the other techniques, especially RSM when highly non-linear behaviour is the case. Also, this technique can build an efficient model using a small number of experiments; however the technique accuracy would be better when a larger number of experiments are used to develop a model.

2.4 Self Organising Map

The name Self-Organizing Map (SOM) signifies a class of neural-network algorithms in the unsupervised-learning category. In its original form the SOM was invented by the founder of the Neural Networks Research Centre, Professor Teuvo Kohonen in 1981-82, and numerous versions, generalizations, accelerated learning schemes, and applications of the SOM have been developed since then [54].

With a conventional ANN approach an input vector is presented to the network (typically a multilayer feed-forward network) and the output is compared with the target vector. If they differ, the weights of the network are altered slightly to reduce the error in the output. This is repeated many times and with many sets of vector pairs until the network gives the desired output. Training a SOM however, requires no target vector. A SOM learns to classify the training data without any external supervision whatsoever [54]. This approach is markedly different from the supervised training techniques such as back-propagation networks where the training data consists of vector pairs - an input vector and a target vector [54, 55].

The SOM has spread into numerous fields of science and technology as an analysis method. There is a list of over 5000 scientific articles that apply the SOM or otherwise benefit from it [53, 54].

The most promising fields of application of the SOM seem to be:

- Process analysis, diagnostics, monitoring, and control,
- Data mining at large, in particular visualisation of statistical data and document collections
- Biomedical applications, including diagnostic data analysis in bioinformatics &
- Data analysis in commerce, industry, macroeconomics, and finance.

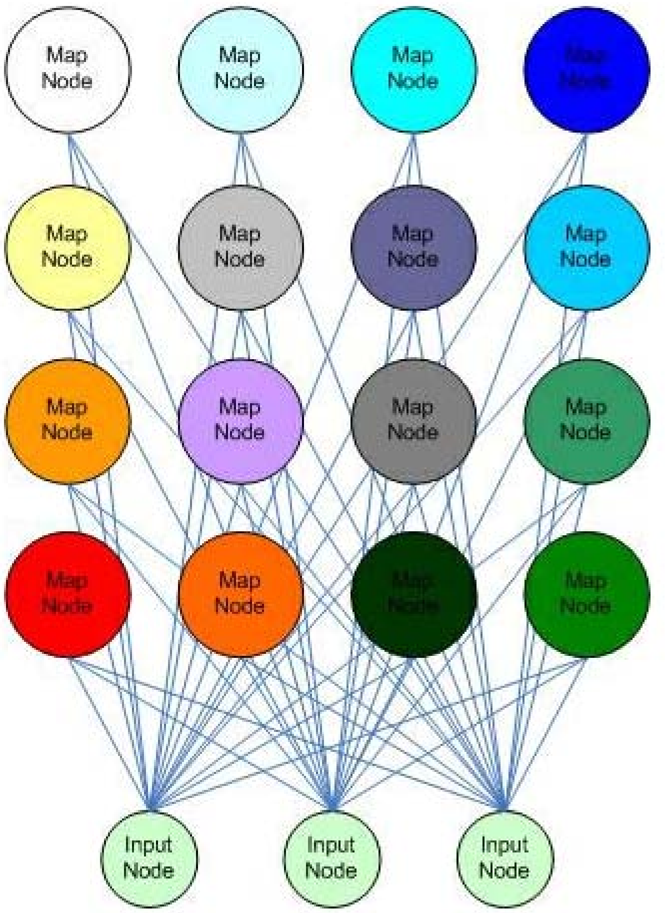


Figure 2-14 Basic Self Organising Map architecture [37]

2.4.1 SOM Algorithm - Overview

A SOM does not need a target output to be specified unlike many other types of network. Instead, where the node weights match the input vector, that area of the lattice is selectively optimized to more closely resemble the data for the class the input vector is a member of [48].

Each node receives all elements of the training set, one at a time, in vector format. For each element, a calculation is made to determine the fit between that element and the “weight" of the node. Often, this calculation is the Euclidian distance between the two vectors, but can be another function of distance. In many cases, this weight is a vector of the same dimension as the input vectors. This will allow us to determine the “winning" node, that is, the node that represents the best the training element.

Once the winning node is found, the neighbours of the winning node are then identified. The winning node and these neighbors are then updated to reflect the new training element. In this way, the map learns from the individual elements.

It appears to be customary that both the neighborhood function and the learning rate are a decreasing function of time. This means that, as more training elements are learned, the neighborhood is smaller, and nodes are less affected by the new elements [51].

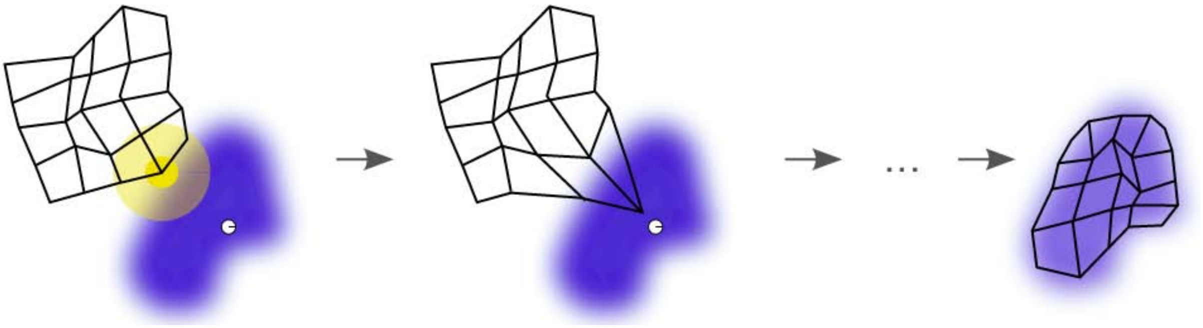


Figure 2-15 An illustration of the training of a self-organising map. The blue blob is the distribution of the training data, and the small white disc is the current training sample drawn from that distribution. At first (left) the SOM nodes are arbitrarily positioned in the data space. The node nearest to the training node (highlighted in yellow) is selected, and is moved towards the training datum, as (to a lesser extent) are its neighbours on the grid. After much iteration the grid tends to approximate the data distribution (right) [37, 51].

We express this change as the following function: for a node x, the update is equal to:

$$x(t + 1) = x(t) + N(x, t)\, \alpha\, (t)\, (\partial(t) - x(t))$$

where:

- $x(t + 1)$ is the next value of the weight vector
- $x(t)$ is the current value of the weight vector
- $N(x, t)$ is the neighbourhood function, which decreases the size of the neighbourhood as a function of time
- $\alpha(t)$ is the learning rate, which decreases as a function of time
- $\partial(t)$ is the vector representing the input document

Based on this information, here is simplified view of the algorithm [37]:

1. Initialize the weights of the nodes, either to random or pre-computed values

2. for all input elements:

(a) Take the input, get its vector

(b) For each node in the map:

i. Compare the node with the input's vector

(c) The node with the vector “closest" to the input vector is the winning node

(d) For the winning node and its neighbours, update them according to the formula above

Another way to view the SOM is expressed here, taken from [51]:

> The other way is to think of neuronal weights as pointers to the input space. [...] More neurons point to regions with high training sample concentrations and fewer where the samples are scarce.

Indeed, the linkage of the neural network forms an elastic fabric, where the input documents pull the covers in their direction. If a large number of inputs “pull the covers" in a certain direction, a greater number of nodes will be used to represent that portion of the input space.

2.4.1.1 Algorithmic complexity

The conversion of the input elements into a vectorial format varies based other type of input, and thus will not be considered here. The factors involved in the map creation are:

- *d*, the dimension of the vectors
- *N*, the number of input samples
- *m*, the number of map elements

Based on the algorithm above, the algorithmic complexity is *O(dNm)*. However, most analyses of the question (in particular, [58] and [64]) postulate as assumption that the number of map elements is chosen proportionally to the number of input samples, and thus the complexity becomes *O*(dN^2)*.*

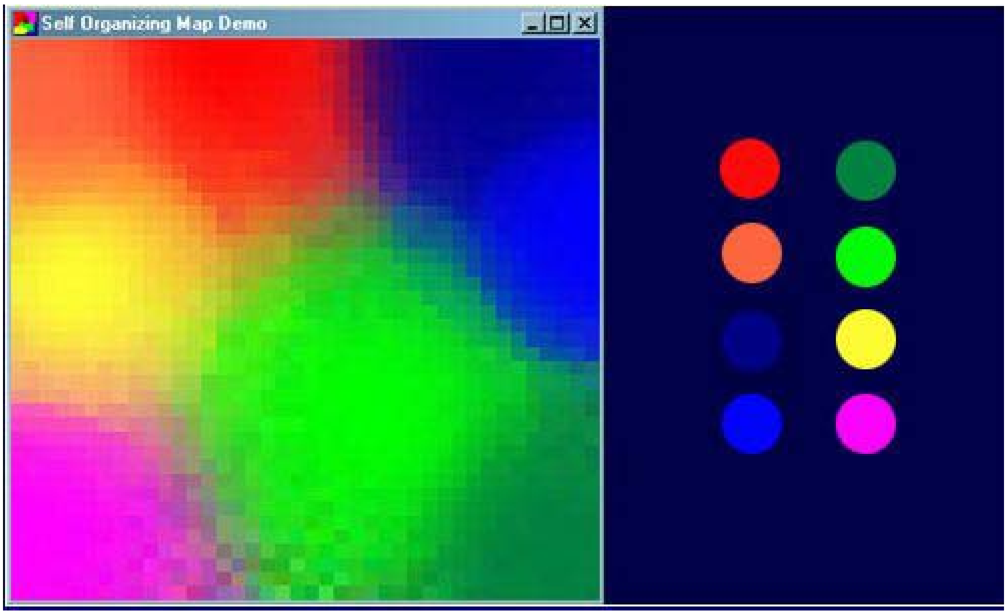


Figure 2-16 Screenshot of a demo program (left) and the colours it has classified (right) [37]

2.4.2 SOM Fault Diagnosis

In engineering, the most straightforward applications of the SOM are in the identification and monitoring of complex machine and process states, otherwise very difficult to perceive and interpret. The SOM has been used for development of new pattern classification and target recognition systems, whereby categorisation of the input signal states is performed by it [52].

As the SOM is a nonlinear projection method, such characteristic states or clusters can often be made visible in the self organised map, without explicit modelling of the system.

An important application of SOM is in fault diagnosis. The SOM can be used in two ways: to detect the fault and to identify it. In practical engineering, we can distinguish two different situations; either there are no prior measurements of the faulty situations, or the faults have been recorded [61].

During Training, the weight vectors of the SOM become adapted to that domain of the system space from which measurements have been taken. Thus the state of space will be divided in to two parts.

1. The orientation space represented by the SOM and
2. It's complimentary space

Fault detection can also be based on the quantisation error: (when the feature vector corresponding to the measurements is compared with the weight vectors of all map units, and the smallest difference exceeds a predetermined threshold, the process is probably in fault situation. This conclusion is based on the assumption that a large quantisation error corresponds to the operation point belonging to the complimentary space not covered by the training data. Therefore the situation is new and something is going wrong [59, 61]

If the fault situations and their reasons are known well enough, simulated data for SOM training can be produced easily. In practical engineering systems, the different types of faults or errors usually encountered are [38, 61];

1. Sudden change in some regulated parameter value indicating fault in the control mechanism
2. Jamming of a measurement value, often caused by a mechanical fault in a measuring instrument
3. Break in signal lines resulting in abrupt drop in signal

4. Slow drift in measured value due to again of device
5. Heavy disturbance with large sporadic change in signals

All the mentioned fault types can be simulated easily and independently as demonstrated by J Huysmans et al [61]. If measurements of most of the typical fault types are available or can be simulated, mechanically or by modelling, the SOM can be used as a monitor of the operating conditions of the system.

Examples of fault detection and identification in an anaesthesia system are described in [61]. The complete system comprises of the anaesthesia machine, anaesthesia personnel and the patient. The purpose of the SOM based monitoring system is to minimise the risks of anaesthesia poisoning by detecting and identifying the faults before they cause damage to the patient. Figure 2-17 describes the SOM computed for an anaesthesia machine, and different areas on it correspond to obstruction of tubes, hoses or pumps, their breakage, wrong gas mixtures, positioning errors of the intubation tube etc.

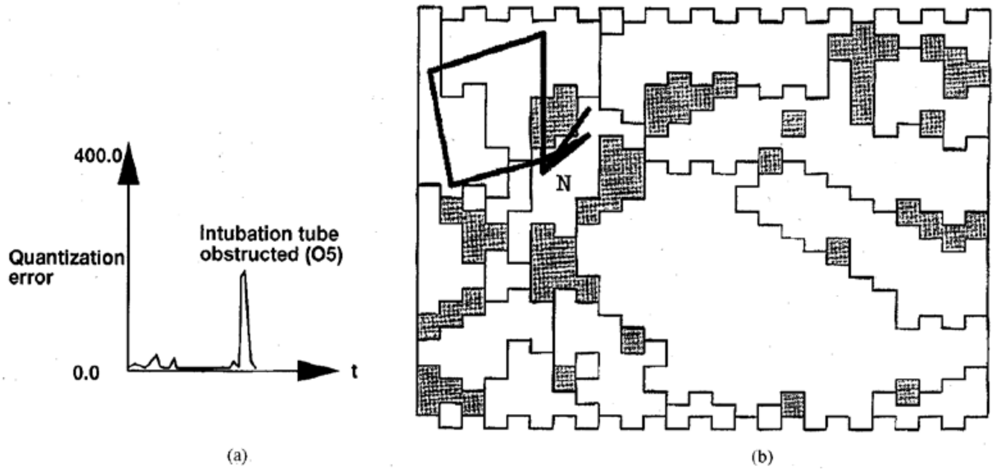


Figure 2-17 Fault identification of an anaesthesia system tested in a true situation N=Normal state. The position of the patient was changed and the intubation tube was obstructed for a short period of time. The increase in the quantisation error shown in (a) indicates that a fault has been detected. The trajectory of the operating point was moving from the area corresponding to the normal situation to the area that corresponded to an obstruction in the specific part of the system. The trajectory is depicted in (b).

2.4.2.1 Data Visualisation

Understanding and modelling complex relationships between multiple variables in large systems is often problematic. Automated measurements produce masses of data that may be very hard or even impossible to interpret. Visualising complex and multi-dimensional data is increasing in relevance in various research and industrial areas. Self organising maps are often proposed for this task since they generate a mapping from a multi dimensional input space to a low-dimensional structure used as a network topology. In many cases this is a rectangular, often square, two dimensional grid of units [51].

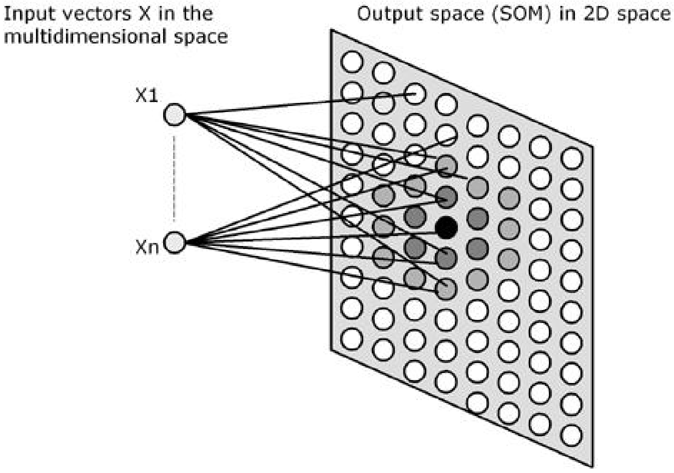

Input vectors X in the multidimensional space
Output space (SOM) in 2D space
X1
Xn

Figure 2-18 Data visualisation in self organising map structure [59]

2.4.3 Why Kohonen's Self Organising map (SOM)?

The ANN model itself provides little information about the design factors and their contribution to the response if further analysis has not been done. The most popular ANNs are learning vector quantization neural networks, back-propagation and counter propagation algorithms. However, for this particular project Self Organising Maps (SOM) will be employed since SOMs are relatively simpler to understand. Plots that are closer and have grey shades connecting them can be bracketed as similar. If there is a black ravine between them, then they are bracketed as dissimilar. Unlike Multidimensional Scaling or N-land, they are more adaptable and network modelers can quickly pick up on how to use them in an effective manner [56].

Furthermore, SOMs classify data well and they are relatively simpler to evaluate with respect to their quality. Hence one can actually calculate how good a map is and how strong the similarities between objects are. The SOM map seeks to preserve the topological properties of the input space. The SOM is the best choice for feature extraction because it eliminates the need to make a difficult choice of the network structure and the need to define a decay schedule for various parameters [56, 57].

Kohonen's model, aims at mapping high dimensional input signals onto an (often two-dimensional) neural sheet of fixed size and the structure in such a way that neighbourhood relations among the input signals are preserved as well as possible [59]. An essential prerequisite for this is a network structure matching the structure of the distribution. If this is approximately the case, Kohonen's model is able to find appropriate mapping .The non- linear relationship between the FSW input and output parameters may be identified well by using a Self Organising Neural Network

Summarising, the key benefits of using SOM are:

- The possibility to use problem dependant error measures to determine where new units are inserted (insertion where necessary) [55].

- The possibility to interrupt the self organisation process or to continue a previously interrupted one. Due to the constant parameters there are no different phases in self-organisation [58].
- Fewer magic numbers to define: the network need not be defined in advance but can be defined indirectly by giving a performance criterion which must be met. For each parameter only its value must be defined and not starting value, end value as well as its function over time as in many other approaches [50, 64].

2.5 Summary

This chapter aims to familiarise the reader with self organising neural networking. We have seen that Artificial Neural Networks offer an alternative way to tackle complex and ill-defined problems. The increased utilization of ANNs is linked to several features they possess, importantly (i) the ability to recognize and learn the underlying relations between input and output without explicit physical consideration, regardless of the problem's dimensionality and the system's nonlinearity, (ii) once trained from examples (sample data) it can perform predictions and generalisations at high speed and (iii) has a high tolerance to data containing noise and measurement errors due to distributed processing within the network.

ANNs also have limitations that should not be overlooked. These include (i) dependance on both the quality and quantity of the data [27], (ii) a lack of clear rules or fixed guidelines for optimal ANN architecture design [10, 27], (iii) a lack of physical concepts and relations [20], and (iv) the inability to explain in a comprehensible form the process through which a given decision (answer) was made by the ANN [20, 24].

ANNs are further classified as supervised or unsupervised systems depending on their learning paradigm. An unsupervised neural network does not receive any feedback from its supervisor; instead it relies on an internal criterion to guide its learning outcomes. The supervised neural network such as the back propagation algorithm requires training before it can be used for mapping or classification purposes. During the training, BP calculates the weights of the ANN to represent the relationship between the inputs and outputs. However, unsupervised neural networks may start classifying the inputs without a separate training session and are adept at dealing with abrupt changes in the characteristics of the input–output relationship of the system.

It is of practical importance to have a sufficient understanding of the FSW process fundamentals. Of the FSW literature and data publicly available, many have explored aspects of various process fundamentals. Each study provides useful information about FSW. However, the wealth of information available and the advances in process equipment has not yielded an agreement on optimal control schemes. For instance, the effect of input parameters on the weld quality is not thoroughly understood. Many studies provided valuable information but did not include quantitative results as part of their discussion.

The survey has also revealed that the SOM algorithm has attracted a great deal of interest among researches and practitioners in a wide variety of fields [54]. The SOM has been analysed extensively, a number of variants have been developed and, perhaps most notably, it has been applied extensively within fields ranging from engineering sciences to medicine, biology, and even economics. Since most manufacturing processes are complex in nature, highly non- linear and there are a large number of input variables, there is no close mathematical model which can describe the behavior of these processes. Self organising maps are up to the task of modeling these complex interactions because they have the ability to learn from examples, do not require lengthy supervised training times, are cost effective and are relatively easier to interpret.

Numerous applications have been found in process modeling for monitoring and control purposes, as forecasters, as intelligent sensors to estimate variable that usually cannot be measured in fault detection and diagnosis systems, dynamic online systems and, finally in, process control.

The aim of this project is demonstrate the possibility of using a self organising map capable of predicting FSW weld quality based on process input parameters. Careful experiments with an emphasis on FSW control must be performed to better understand potential control schemes. This survey also reveals a high level of interest in the adaptation of Artificial Neural Networks to predict response(s) in a wide array of industrial processes.

The proposed methodology, experimental data collection, and the results will be outlined in the following chapters.

3 Methodology and Experimentation

The previous chapters have outlined and elucidated the objectives of this research. This chapter provides the experimental procedures for assessing the aforementioned objectives.

Experimental data was collected by performing a series of FSW trials on Aluminium alloy AA2024 and A253 against a select range of parameters. Kurtosis analysis was initially performed on the raw data collected from the tool-work piece interaction using an external sensory unit. The approach was split into the following phases:

- Setting up experimental equipment and carrying out FSW trials
- Collection of comprehensive data relating to multiple welding parameters
- Pre processing of raw data to reduce dimensionality and integrate vectors from the external sensor while maintaining completeness in information
- Application of the state-of-the-art Self Organising Feature Map and to coalesce integrated sensory data, providing a detailed mapping of the various FSW trials.

These stages are discussed in detail in the following sections.

3.1 Experimental Setup

FSW trials were done by using both a Lagun turret-milling machine and a Tos Olomouc machine at AUT's workshop; these are shown in Figure 3-1. The tool pins made of H13 steel were used to carry out the FSW trials. Heat treatment was applied to ensure that the threads would not deform after an extended period of welding. Suitable hardness of the thread was critical for detecting the contact conditions on the interface.

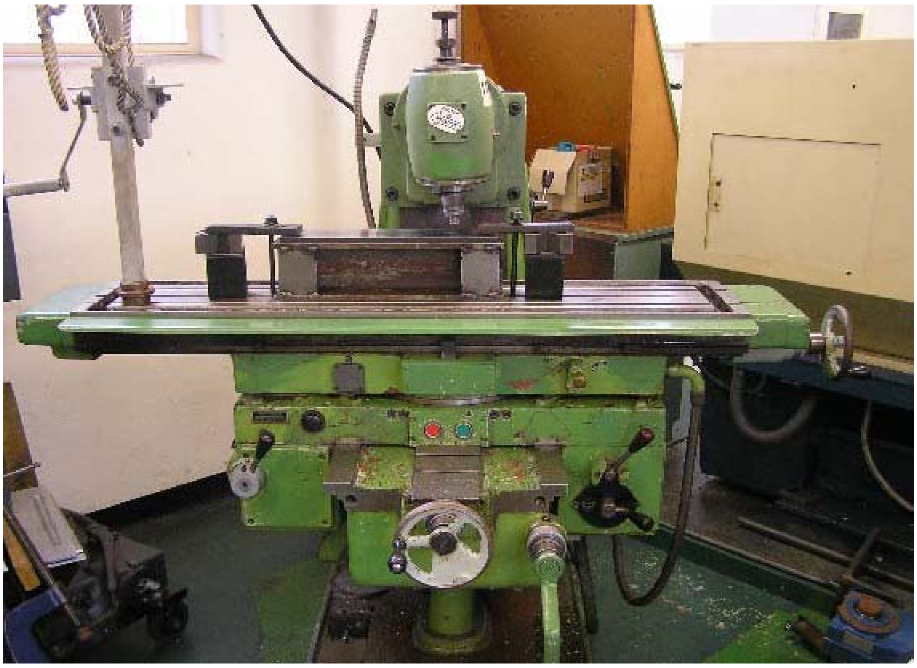


Figure 3-1 TosOlomouc machine

The tool pin was made to rotate at varying speeds in a fixed position while the machine bed moved at varying speeds rather than the tool travelling through the work piece.

3.1.1 Plate Preparation – Aluminium A356

Commercially available aluminium A356 was carefully sectioned into 24 similar sized plates. These plates were then milled from all six sides to form square plates roughly about 8x8cm in dimensions. These plates were then friction stir welded to form about eight plates. The eight resulting plates were once again clamped to the Lagun milling machine to level and remove any unevenness/distortion from the plate's surfaces before performing the FSW trials with LowStir sensor unit. FSW trials were then performed on the plates using the Lagun- Turret milling machine as shown in Figure 3-2. The gaps between some of the plates were variable and significant care was taken to ensure proper clamping of the plates to prevent plate movement of them during welding.

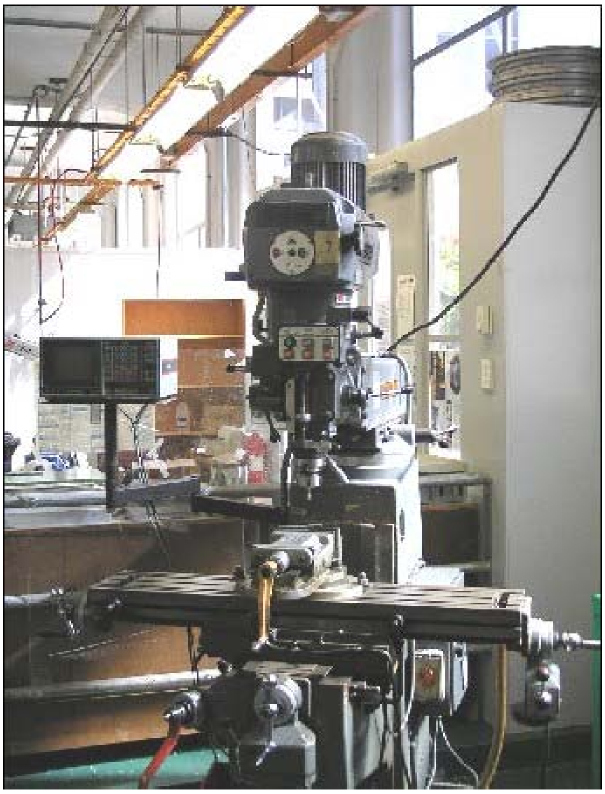


Figure 3-2 Lagun Turret Milling Machine

3.1.2 Tool Design

The tool, having a 25 mm diameter shoulder, was used during the trials. This diameter was chosen because it would fit the tool assembly of the low stir sensor unit Figure 9. The tool has three features which were - the shoulder diameter, the pin diameter and the tool profile. The H13 tool was heat treated after machining to ensure the hardness of the teeth. The final tool design is described in Table 3-1.

Tool type	Tool (H13 tool steel)
Shoulder diameter	25 mm
Pin length	5.70 mm
Screw on the pin	left handed thread Pitch 1.2 mm; 1.6 mm depth

Table 3 Tool design in the research

Figure 3-3 below shows the tool profile. The drawings of the tools are shown in the appendix B.

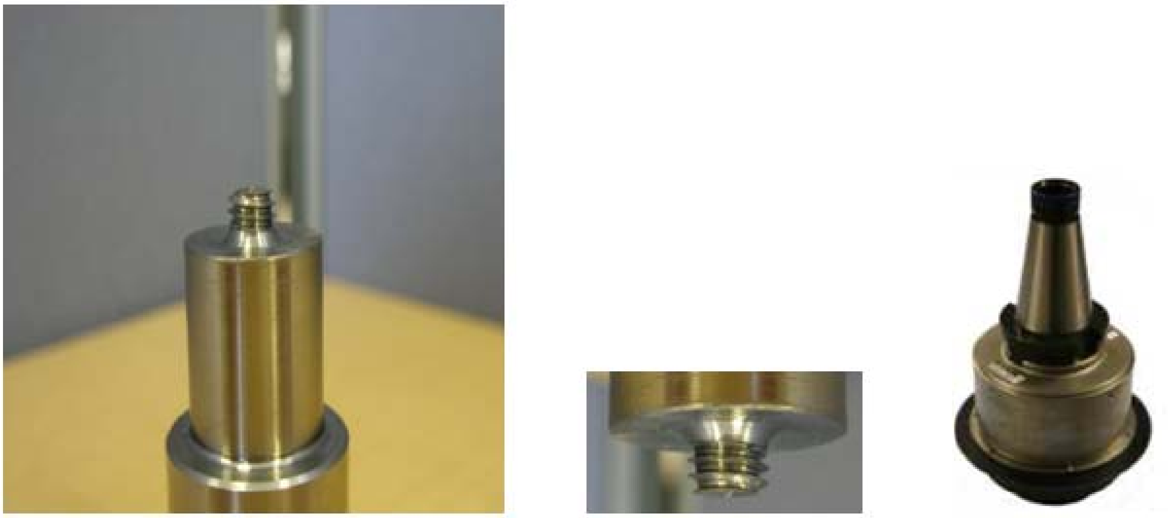


Figure 3-3 FSW tool and the LowStir sensor unit including ISO taper, tool holder and heat shield disc. This is supplied already bolted together as a complete system, ready to use

3.1.3 Welding Machine & Weld Monitoring System Setup

Milling machines are normally targeted at specific applications, which govern machine architecture and the maximum workpiece size. FSW trials were conducted using the Tos Olomouc FA3AV milling machine at AUT's workshop.

The *Tos Olomouc* machine characteristics including forces, torque, spindle speed and traverse speed were examined to determine suitability for friction stir welding. The machine had a 5.8 kW maximum power output, and was able to vary spindle rotation speeds and set a desirable speed value for both vertical and horizontal movement. In addition, the machine had a vertical position setting of 0.02mm per step. Rotation speed varied between 250 rpm and 1400 rpm, and linear speed varied between 28 mm/min and 450 mm/min.

FSW *LowStir* sensor apparatus is a full weld monitoring system that can be retro fitted to standard milling machines such as the *Tos Olomouc* machine at AUT to join aluminium alloys within a thickness range of 2 – 8 mm.

The unit has been tested by TWI, BAE Systems UK, Sapa, Aeronautical Research and Test Institute of the Czech Republic and Instytut Spawalnictwa [26]

The sensor unit was installed onto the milling machine head using the attached ISO taper and care was taken to ensure any of the trailing wires were tied out of the way of any moving parts. Figure 3-4 shows the setup picture.

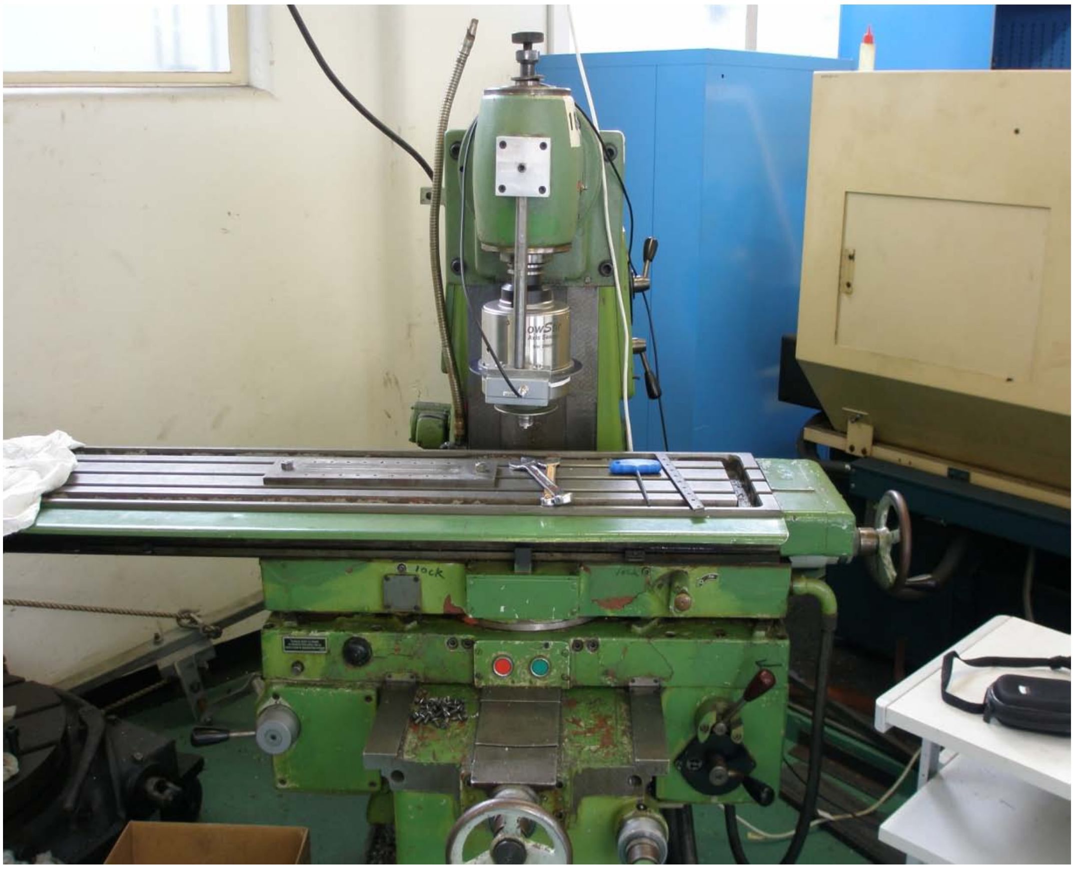
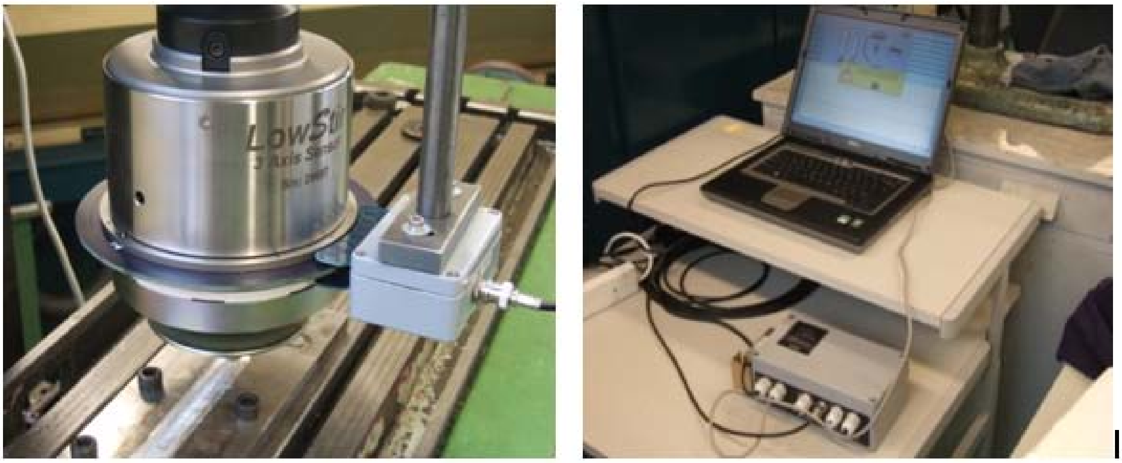


Figure 3-4 LowStir 3 Axis Sensor unit retro fitted on to the Tos Olomouc machine at AUT

3.1.3.1 Weld monitoring software setup

The monitoring software needs to be installed on the computer connected to the LowStir unit when the system is run for the very first time. Additionally, the software can be configured to display real time numerical values of forces, torque and other key weld parameters (as identified by TWI).

The information gathered by the *LowStir* device is displayed to the operator in a clear and straightforward manner using a laptop PC running Labview, a sample display screen format is shown in Fig. 3-5. The instrument panel displays real time numerical values of forces, torque, the temperature adjacent to the system electronics and (if desired) the tool temperature. The system also has the capability to add real-time event markers to allow correlation between process conditions/stages and the recorded data. The main display screen has buttons to start and stop recording of data. Alternatively an automatic trigger facility exists for initiating the recording of data. The display also shows the current captured data values for the weld in progress indicating whether they are within the acceptable range for satisfactory welding. The display also has a multi-graph facility where the user can select which sensor values are displayed.



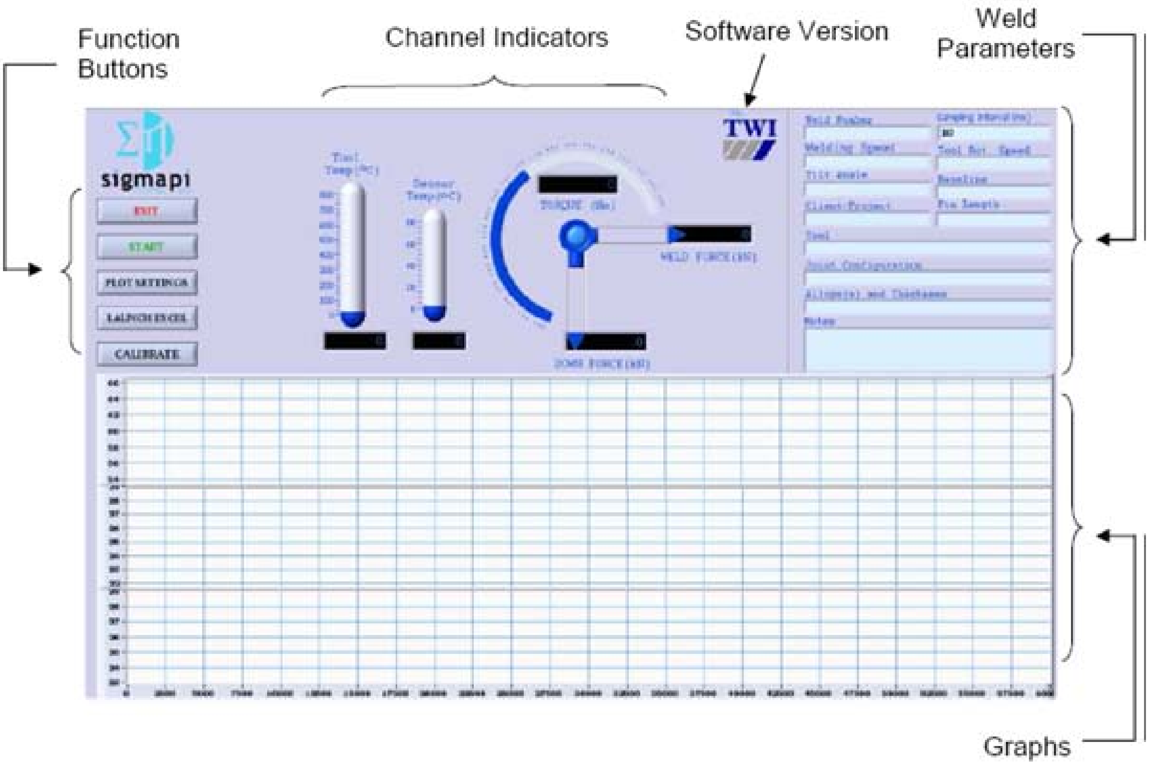


Figure 3-5 LowStir software main screenshot after being sucessfully loaded on to the system

3.1.4 Materials used in Experiments

Welding plates of aluminium alloy A356 and AA2024 were chosen as welding material for experiments which have different strength characteristics particularly at near peak FSW temperatures [35]. “Bead on plate” FSP have been conducted on 290mm long, 85 mm wide, 6.35 mm thick A356 (Al-7Si-0.3Mg) cast aluminium plates. Tools with 18 mm shoulder diameter were cut by CNC machine followed by heat treatment to obtain hardness around 48 HRC. Tool pins are 6 mm in diameter, 5.7 mm in length, with left-handed threads (1.25 mm in pitch). In the experiments, the welding line was made parallel to the extrusion texture direction. Fifteen welds were conducted using different combinations of speeds and a constant tilt angle of 2.5 degrees.

Alloy	Si	Fe	Cu	Mn	Mg	Ti
A356	6.5-7.5	0.2	0.2	0.1	0.25-0.45	0.2
2024	0.5 max	0.5max	3.5*4.9	0.3-0.9	1.2-1.8	0.2max

Table 4 Chemical composition of A356 & 2024 [35]

Alloy	Fluidity	Resistance to hot Cracking	Corrosion Resistance	Machinability	Elevated Temperature Strength
A356	1	1	2	3	3
2024	1	1	1	2	3
Ratings:	1=Excellent	2=Very Good	3=Good	4=Fair	5=Poor

Table 5 A356 & 2024 Physical Characteristics [35]

3.1.5 Parameters considered while making a friction stir weld

A sum total of 22 combinations of parameters were chosen to undertake trials on two aluminium alloys A356 and AA2024. There were 22 combinations possible by changing either the tool speed / welding speed or rotation speed parameter for each of the frition stir weld. By doing this, a complete set of trail data was obtained which enabled better classification of the relationship between each of the parameter (input) and the weld quality (output). In the present FSW experiments, a range of tool speeds were used and the resulting torque was measured. All plates were machined to a standard size to eliminate the effect of size on material flow. Welding force, down force are monitored during welding.

Commonly used parameters were considered before choosing a range of welding parameters.

The intention was to perform a maximum number of trials within the most commonly used range (parameters) – [6], [28] and [33]. The trial range values of v & ω were chosen so that the upper and lower values were taken in to consideration to give a comprehensive picture.

Secondly, the intervals chosen between any two separate values of v (mm/min) & ω (rev/min) were the set values of the *Tos Olomouc* milling machine. The machine works reliably only for the following range of speeds and tool rotations speeds.

weld no.	v (mm/min)	ω (rev/min)
1	28	250
2	28	500
3	28	710
4	28	1000
5	56	250
6	56	710
7	56	1400
8	112	250
9	112	500
10	112	710
11	112	1000
12	112	1400
13	224	250
14	224	500
15	224	710
16	224	1000
17	224	1400
18	315	500
19	315	710
20	450	500
21	450	710
22	450	1000

Table 6 Input parameter combination table

The tilt angle tends to produce a “heel plunge depth”, which is the deepest part the tool shoulder that appears below the work piece surface. Heel plunge is used to consolidate material behind the tool pin to avoid defects. The tilt angle of the tool was set at 2.5 degrees.

Welding defects can also occur if the gap between the plates before welding is significant. It is recommended that the gap be no wider than 10% of the plate thickness [45]. Smaller gaps are closed by the material pressure that develops under the tool shoulder during weld [46]. Clamping is important because plates can tend to separate or lift off the bed during the weld (lift is mainly a problem for thinner plates).

Position-control technique was implemented in all welds in order to make sure the centre of the pin-shoulder intersection is just immersed in the base plates. Cross section samples were taken near the end of the welds where a stable welding condition is achieved. All samples have gone through normal metallurgical preparation steps and were analysed under a stereomicroscope.

3.2 Data Analysis

The data capture was conducted in real time using LowStir sensor software. The system is designed to provide friction stir welds using only a suitable milling machine, computer and the LowStir unit itself. The computer can monitor down force, lateral force, torque, internal temperature and, if required, an external temperature. Once the data is captured, preprocessing allows the data to be ANN ready for subsequent manipulation and analysis.

3.2.1 FSW Sequence/Conditions.

The work piece and the weld parameters used in the experiment are shown below. Trials were carried out showing the influence of multiple FSW input parameters (welding speed, rotational speed, tilt angle) on the weld quality. Two groups of response were measured: the forces involved in the FSW process and the resulting quality of weld.

Work piece	Parameters of FSW

Aluminium A356
(Al-7Si-0.3Mg) 6.35mm Thick
 	
Aluminium AA 2024
(Al-0.5Si-1.5Mg) 6.35mm Thick
	Vtravelling = 28, 56, 112, 224, 448mm/min
	θ =2.5
	ω = 250, 500, 710, 1000 & 1400 rpm
	Dshoulder = 18mm
	Lpin = 5.7mm
	Dpin = 6mm

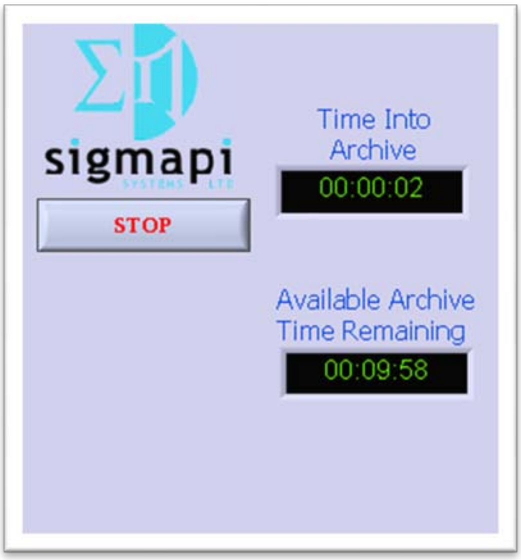
Table 7 FSW parameter window

The parameters in table 3-5 are:

- Dshoulder - the diameter of the tool shoulder;
- Lpin - the length of the tool pin;
- Dpin - the diameter of the tool pin;
- Vtravelling - the tool traverse speed along the joint line;
- ω - the tool rotation speed;
- and θ tilt - the angle between the tool neutral line and the vertical plane

3.2.2 Data Extraction

The software can capture and display data from five transducers (two temperatures, down force, lateral force and torque). Each of the monitored channels (tool temperatures, sensor temperature, torque, weld force and down force) has a corresponding indicator that displays the current values during real time monitoring. The graphs can plot a maximum of three channels at a time. Also, during monitoring, the length of time the monitoring has been running (Time Into Archive) and available monitoring time remaining (Available Archive time Remaining) will be shown as below.



Time Into Archive: 00:00:02

Available Archive Time Remaining: 00:09:58

Figure 3-6 during monitoring this available time is determined by the limitation Microsoft imposes on the number of rows allowed in some versions of Excel

Each channel's value is constantly updated during the monitoring period and also the selected channels are plotted on the graph (refer to "plotting channels" for how to select channels to plot). Each graph's associated channel name is shown on the right hand corner of the graph. See below

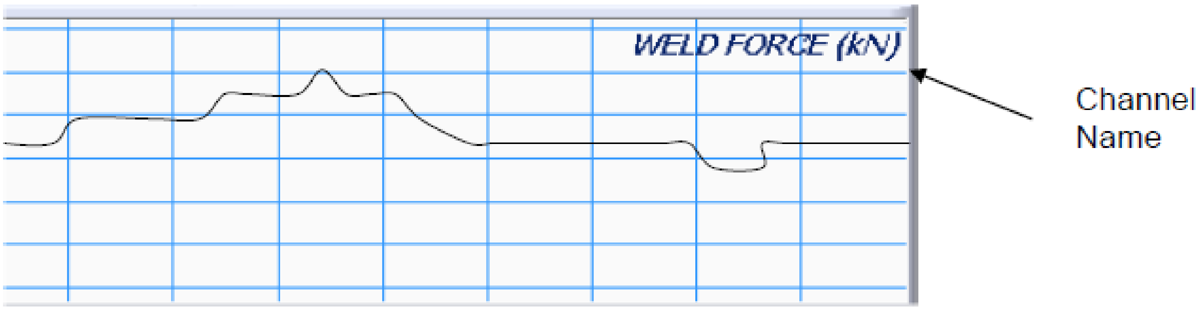


Figure 3-7 Graph's associated channel name is shown on the right hand corner of the graph

.

The "sampling interval (ms)" is one of the weld parameters, and it determines the sample rate and, therefore, the maximum duration for monitoring to one file. Because the maximum number of samples to be stored to file is 6000, the monitoring duration will be adjusted accordingly based on the specified "sampling interval (ms)". When monitoring stops, the captured channel data, weld parameters and calibration information is stored to file in a Microsoft Excel format.

3.2.3 Metallography

The Nikon optical microscope was used to view the microstructure of the samples. It has objective lenses of 5 times to 100 times and images were taken accordingly. The optical microscope was mainly used to identify the weld quality.

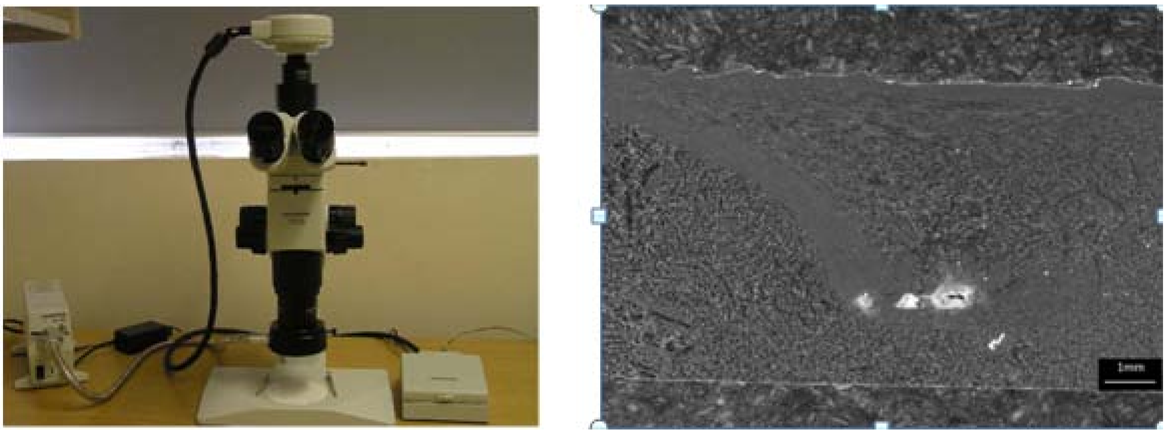


Figure 3-8 Olympus stereomicroscope was used to analyse weld quality of A356.

3.2.4 Data Analysis and Pre processing

The data capture was conducted in real time using the LowStir software along with the sensory unit and signal processing module. This data was then exported as digital data into an excel worksheet.

The digital data was analysed using the DaDiSP/32 software. DaDiSP is a powerful generic tool for data display and analysis. Once the data is captured, DaDiSP allows displaying the data for subsequent manipulation and analysis. The Figure below shows how the data is manipulated using DaDiSP.

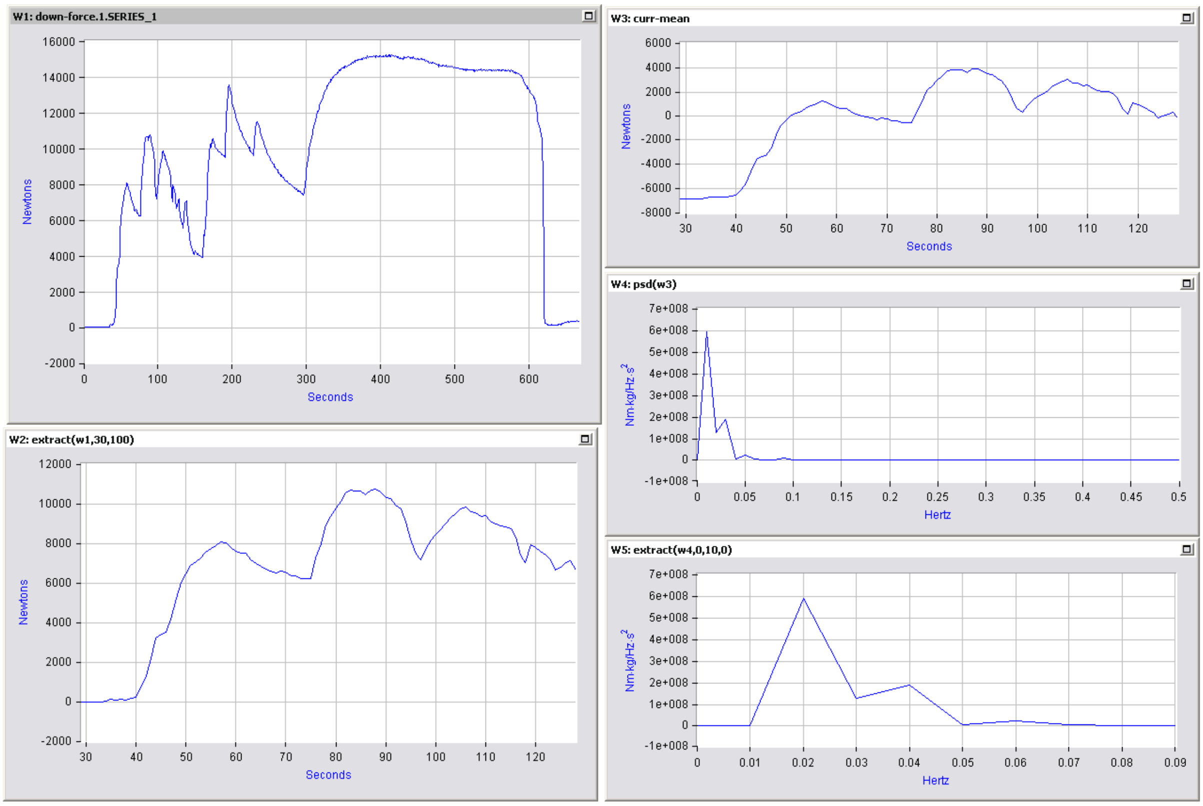


Figure 3-9 Using DaDisp for data display, manipulation and analysis

The exported digital data is first read as a series in DaDiSP. This is shown in Fig. 3-9 in the first window from the top left. This data is then broken up into several parts of dynamic data which are extracted as shown in the bottom-left window in the figure 3-10. These extracted parts of the data have an overlap (of 48 sampling points) with each other to maintain the continuity and completeness in the output results. This extracted data is then analysed by calculating the power spectral density (PSD) of the signal. This is done using the psd function. The dynamic part of the PSD is shown in the bottom-right window in Fig. 3-10. PSD is a very useful tool to identify oscillatory signals in time series data. It also gives the amplitudes of the data set. PSD analysis is especially useful to detect unwanted vibrations that stem from machining operations. To further characterise the data, it was considered to calculate the Kurtosis of the signals.

The kurtosis distributions were calculated and it was clear from the calculated kurtosis distributions that the deviations were relatively high and some of the calculated values were significant. Thus, the kurtosis distributions were not considered to be a fitting indicator of the data distribution. Thus, the power spectral density of the signal was calculated and the dynamic part of the signal was extracted as a pre-processed input for the ANN. Pre-processing the data gives us a fair idea as to whether the signal distribution is normal / peaked etc.

In the Toolbox, a graphical user interface is used for preprocessing data from the function som_normalize. This fuction can be used to perform linear and logarithmic scalings and histogram equalisations of the numerical variables. Scaling of variables is of special importance, since the SOM algorithm uses Euclidean metric to measure distances between vectors. If one variable has values in the range of [0,...,1000] and another in the range of [0,...,1] the former will almost completely dominate the map organization because of its greater impact on the distances measured. Typically, one would want the variables to be equally important. The standard way to achieve this is to linearly scale all variables so that their variances are equal to one [40].

The data does not necessarily have to be preprocessed at all before creating a SOM using it. However, in most real tasks preprocessing is important; perhaps even the most important part of the whole process [40,65].

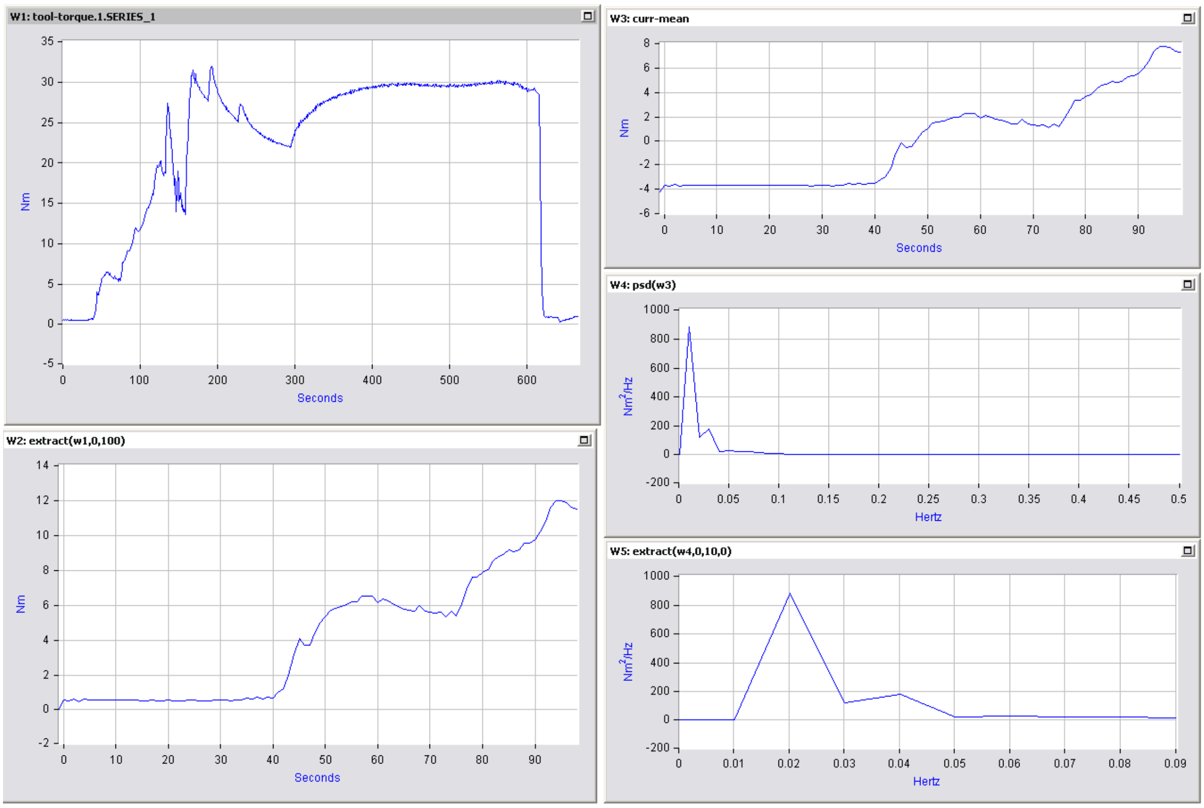


Figure 3-10 Showing the tool torque analysis in DaDisp

3.3 ANN approach and modelling data with Network

The R program package contains all the programs necessary for the correct application of the Self Organising Map algorithm for the visualisation of complex experimental data. The core of R is an interpreted computer language which allows branching, looping and modular programming using functions. Most of the user-visible functions in R are written in R. It is possible for the user to interface to procedures written in the C, C++, or FORTRAN languages for efficiency [40]. The R distribution contains functionality for a large number of statistical procedures. Among these are: linear and generalized linear models, nonlinear regression models, time series analysis, classical parametric and nonparametric tests, clustering and smoothing. There is also a large set of functions which provide a flexible graphical environment for creating various kinds of data presentations. Additional modules ("add-on packages") are available for a variety of specific purposes [37, 40].

3.3.1 Installation of R Program Package

Current binary versions of R run on Windows 2000 or later. A typical install took 50Mb of disk space, a full installation about 65Mb and a minimal one about 29Mb. Since installing to a network share was not supported for users of XP/Vista/Windows7 it had to be installed on to a non-system area (such as C:\R).

Command-line arguments were added at the end of the Target field (after any final double quote, and separated by a space), for example --sdi --max-mem-size=1G. Also, environment variables were set at the end of the Target field, for example R_LIBS=p:/myRlib, and the language settings for the menus and messages were set to (New Zealand) English, SET LANGUAGE=en.

3.3.2 Data Format

The kind of data that can be processed with the R Toolbox is so-called spreadsheet or table data. Each row of the table is one data sample. The columns of the table are the variables of the data set. The variables might be the properties of an object, or a set of measurements measured at a specific time.

Every sample had the same set of variables such as tool rotation speed, weld speed, torque etc. The table representation is a very common data format [40]. Initially, the available data did not conform to the toolbox format. However, it was transformed to the prescribed format using mapping & data manipulation schemes within the R toolbox. The toolbox can handle both numeric and categorical data, but only the former is utilised by the SOM algorithm.

In the Toolbox, categorial data had to be inserted into labels associated with each data sample.

Function som_autolabel was used to handle categorial variables since numeric variables were needed to train the SOM. They were converted into numerical variables using 1-of-n coding.

Note that for a variable to be “numeric”, the numeric representation must be meaningful: values 1, 2 and 4 corresponding to objects A, B and C should really mean that (in terms of this variable) B is between A and C, and that the distance between B and A is smaller than the distance between B and C. Identification numbers, error codes, etc. rarely have such meaning, and as such were handled as categorial data.

3.3.3 Construction of Data Sets

The preprocessed data had to be brought into R using standard programming functions such as *load* and *fscanf*. In addition, the Toolbox has function *som_read_data* which can be used to read ASCII data files:

sD = som_read_data('data.txt');

The data was put into a data struct, which is a struct defined in the Toolbox to categorise information related to a data set. It had fields for numerical data (.data), strings (.labels), as well as for information about data sets and the individual variables.

The Toolbox utilizes many other structs as well, for example a map struct which holds all information related to a SOM. A numerical matrix can be converted into a data struct with: *sD = som_data_struct(D*). If the given data only consisted of numerical values, then it would not have been necessary to use data structs at all. Most functions accept numerical matrices as well. However, in this study, categorial variables were given and thus data structs had to be used. The categorial variables were converted to strings and put into the .labels field of the data struct as a cell array of strings.

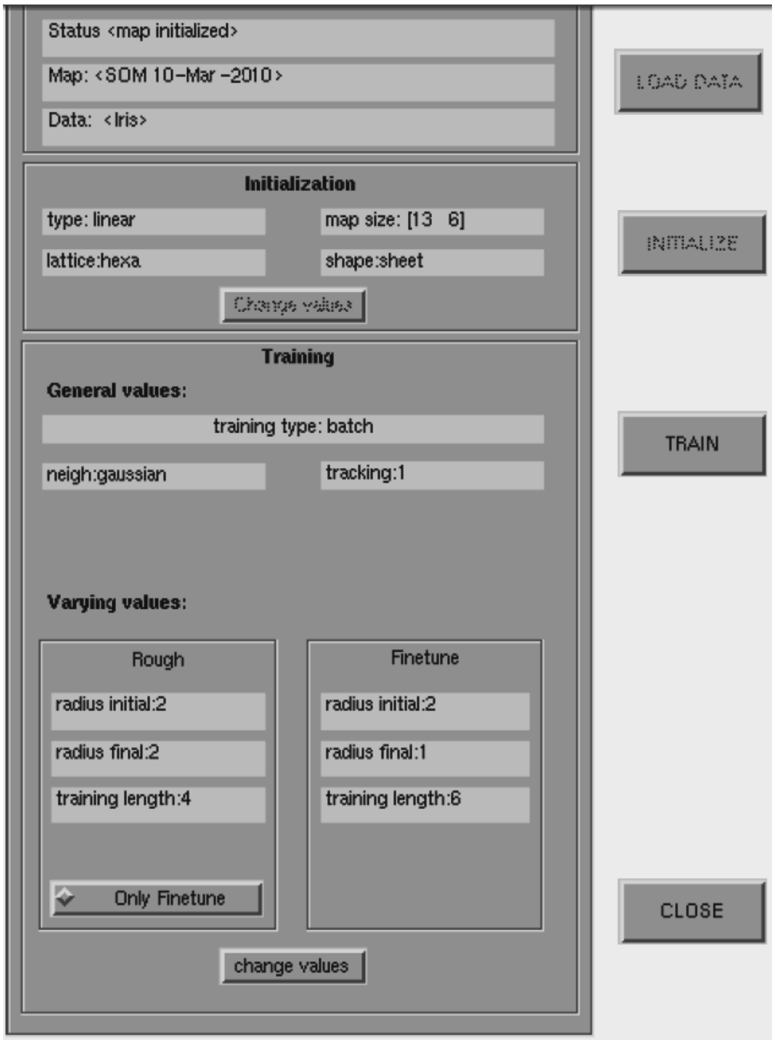
3.3.4 Initialisation

There are two initialisation (random and linear) and two training (sequential and batch) algorithms implemented in the R Package [40]. By default linear initialisation and batch training algorithm were used. The efficient way to initialise and train a SOM was to use the

function *som_make* which did both using automatically selected parameters:

sM = som_make(sD);

The training was done is two phases: rough training with large (initial) neighborhood radius and large (initial) learning rate, and fine tuning with small radius and learning rate. If tighter control over the training parameters was desired, the respective initialisation and training functions, e.g. *som_batchtrain*, was used directly. There is also a graphical user interface tool for initialising and training SOMs, see Figure 3-11



Status <map initialized>
Map: <SOM 10-Mar -2010>
Data: <Iris>

Initialization
type: linear
map size: [13 6]
lattice:hexa
shape:sheet

Training
General values:
training type: batch
neigh:gaussian
tracking:1

Varying values:

Rough
radius initial:2
radius final:2
training length:4

Finetune
radius initial:2
radius final:1
training length:6

Figure 3-11 SOM initialization and training tool

When the entries in the map were trained to their final values, the resulting quantization errors were evaluated. The training file was used for this purpose. The program *qerror* was used to evaluate the average quantization error.

```
>qerror –din ex.dat –cin ex.cod
```

This program computed the quantization error over all the samples in the data file.

3.3.5 Visualisation and Analysis

There are a variety of methods to visualise the SOM. In the Toolbox, the basic tool is the function som_show. It was used to show the U-matrix and the component planes of the SOM:

som_show(sM);

The U-matrix visualised distances between neighboring map units, and thus showed the cluster structure of the map: high values of the U-matrix indicated a cluster border, uniform areas of low values indicated clusters themselves. Each component plane showed the values of one variable in each map unit. On top of these visualisations, additional information was shown: labels, data histograms and trajectories.

With function som_vis much more advanced visualisations were possible. The function is based on the idea that the visualisation of a data set simply consists of a set of objects, each with a unique position, color and shape. In addition, connections between objects, for example neighborhood relations, can be shown using lines. With som_vis function it was possible to assign arbitrary values to each of these properties. For example, x-, y-, and z-coordinates, object size and color can each stand for one variable, thus enabling the simultaneous visualization of five variables. The different options were:

- the position of an object can be 2- or 3-dimensional
- the color of an object can be freely selected from the RGB cube, although typically indexed color is used

- the shape of an object can be any of the R plot markers ('.','+', etc.), a pie chart, a bar chart, a plot or even an arbitrarily shaped polygon, typically a rectangle or hexagon
- lines between objects can have arbitrary color, width and any of the Matlab line modes, e.g. '-'
- in addition to the objects, associated labels can be shown

For quantitative analysis of the SOM there are at the moment only a few tools. The function *som_quality* supplies two quality measures for SOM: average quantization error and topographic error. However, using low level functions, like *som_neighborhood*,

som_bmus and *som_unit_dists*, it was easy to implement new analysis functions. Much research is being done in this area, and many new functions for the analysis will be added to the Toolbox in the future, for example tools for clustering and analysis of the properties of the clusters [40]. Also new visualisation functions for making projections and specific visualization tasks will be added to the package.

4 Results & Discussion

This chapter presents the results obtained by the trials explained in the previous chapter and discussed in order to analyse the results. The results presented in this chapter consist of FSW input parameter values which were given to the SOM for both, learning and subsequent classification. The classification results obtained from the SOM have also been shown - these quantify the classification performance of the SOM under varying operating conditions. Graphical plots have been used to aid in visualising the data sets and the data display is formatted to read as clearly and practicably as possible. The neural network coding and simulation was done using R.

R is a language and environment for statistical computing and graphics. R provides a wide variety of statistical and graphical techniques, and is highly extensible. The program written for the SOM is based on the function provided in the kohonen package of R. Although the basic calculation subroutine is little changed, the data input, handling, execution, storage and output formatting is all original.

4.1 Evaluation of Results

In present work, the following steps were undertaken:

- Data collection.
- Analysis and pre-processing of the data.
- Training of the neural network.
- Testing the trained network.
- Use of the trained SOM for simulation and prediction.

The value of the SOM analysis was to observe interrelationships that exist between multiple FSW variables that were tested and thereby provide a basis for generating hypotheses that can be experimentally examined. The SOM does not replace existing statistical tools, but complements our ability to examine relationships between disparate types of variables in a visual presentation of the data. In this model, the primary concern was the weld quality as a desired output, which was hypothesized to be the strongest determinant of the friction stir welding process.

The input layer of the SOM comprised of tool rotation speed, welding speed, tilt angle, alloy thickness, weld force, down force and tool torque and the weld quality was defined as the output. The experimental data set included 44 patterns, of which 33 patterns were used for training the network and 11 patterns were selected randomly to test the performance of the trained network.

4.2 Post Weld Analysis

FSW trials were performed on two aluminium alloys A356 and AA2024 which have different physical characteristics (particularly strength characteristics at near peak FSW temperatures) under a wide range of rotation and welding speeds. A total of twenty two welds combinations for each material were produced. The resulting torque and forces have been recorded for all welds.

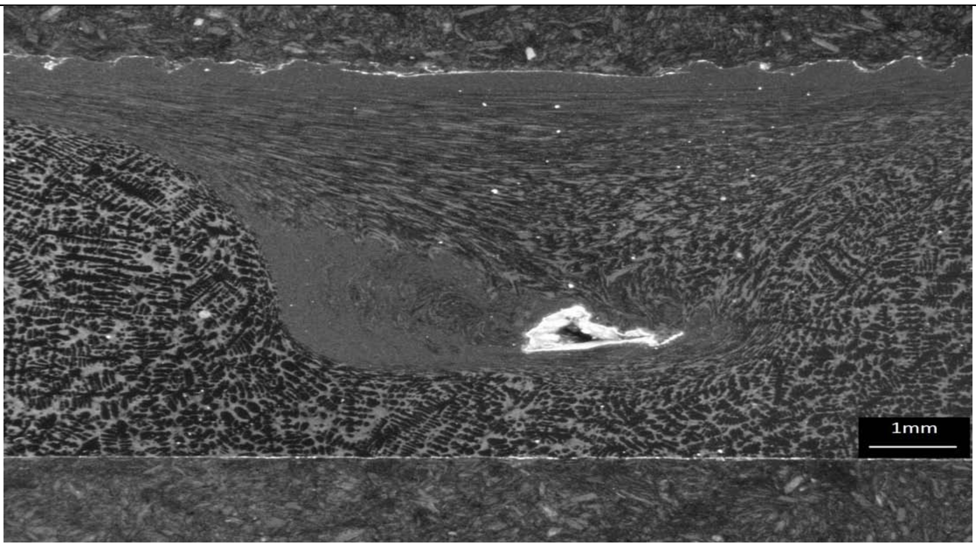
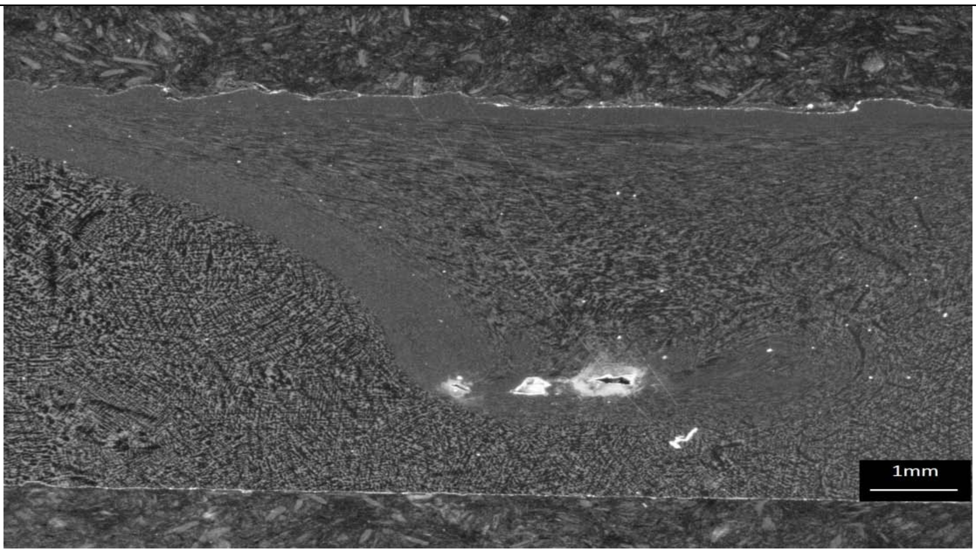
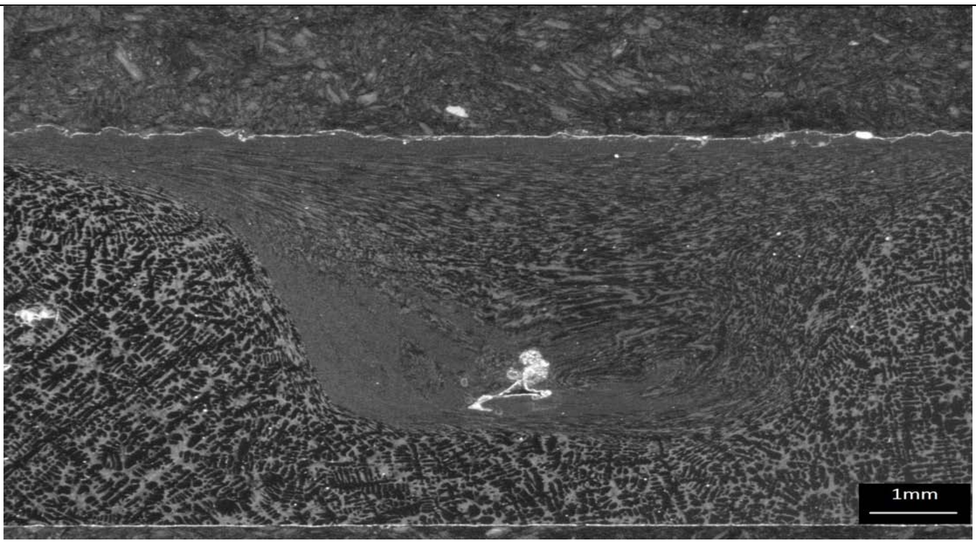
Currently, no national or international standards exist for evaluating the quality of friction stir welds [29]. However, Lloyds Register of Shipping has issued guidance notes for weld qualification [45], which are largely based on requirements for arc welds in British Standard BS EN288 [47] and an AWS standard is in preparation [46]. The former document specifies 100% visual examination, 100% radiographic or ultrasonic inspection and 100% penetrant inspection, together with bend tests, tensile tests and metallography.

In this paper, welding flaws were identified by a combination of metallographic sectioning and visual observation. No attempts were made to determine the limits of detectability, of the flaws by the above mentioned techniques. The experiments for A356 were carried out under the following process windows.

$$\begin{aligned}&\text{Workpiece: } 6.35\,\text{mm thick A356 (Al-7Si-0.3Mg)}\\&\omega = 250, 500, 710, 1000\,\&\,1400\,\text{rpm}\\&v = 28, 56, 112, 224, 448\,\text{mm/min}\\&\theta = 2.5^{\circ}\;\; D_{shoulder} = 18\,\text{mm}\;\; D_{pin} = 6\,\text{mm}\;\; L_{\text{pin}} = 5.7\,\text{mm}\end{aligned}$$

A number of A356 welds contained defects. It was observed that when welding at a higher speed, the material received less work per unit of weld length, i.e fewer tool rotations per mm. Under such conditions, the plasticised material might be cooler, and less easily forged by the shoulder, resulting in unconsolidated voids.

Minor defects were also observed elsewhere in the welds. In some instances they might have been due to inadequate forging. Others were present intermittently. Macrostructures of post-weld transverse cross sections are shown in table 4-1.

Weld No.	Microstructure of Post Welds	Conditions
4.		***V*** = 112 mm/min ***w*** = 250 rpm
5.		***V*** = 224 mm/min ***w*** = 250 rpm
7.		***V*** = 315 mm/min ***w*** = 500 rpm

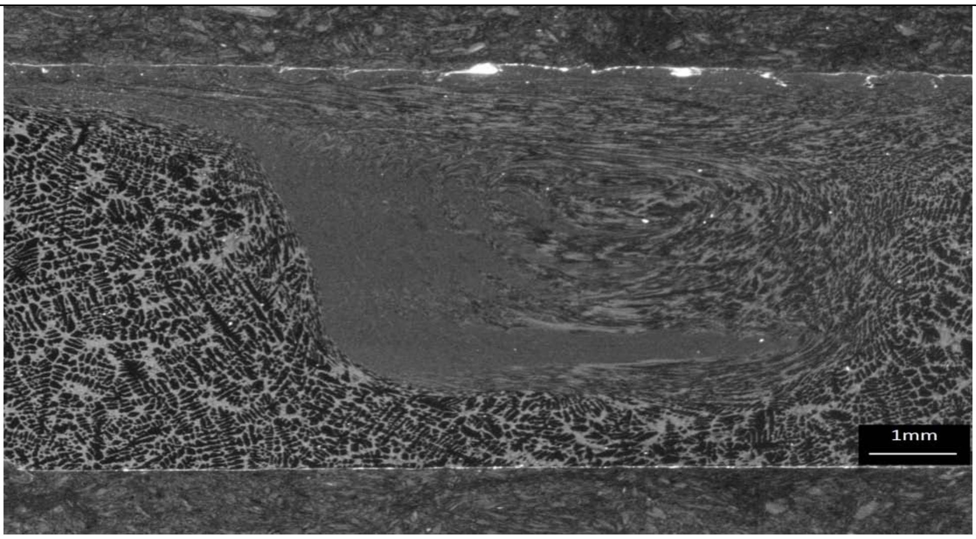
10.		***V*** = 224 mm/min ***w*** = 710 rpm

Table 8 Microstructure of Post Welds

4.3 The Self Organising Map – Training Phase

The neural network model was trained to simulate FSW runs that were done in the labs. The data from these trials was fed to the Self Organising Map and its performance or classification accuracy was tested against the actual data observed from the trials performed in the labs. The neural network learned to organise or tune itself to the fed input and eventually created a “self organised map” of the data it had seen during training.

The figure 4-1 below shows a plot of the SOM after 100 iterations during the training phase. The figure shows the neurons within the SOM output layer and the distribution of data within these neurons. Depending on the input domain signals, the best matching unit from the output layer is chosen to represent the output.

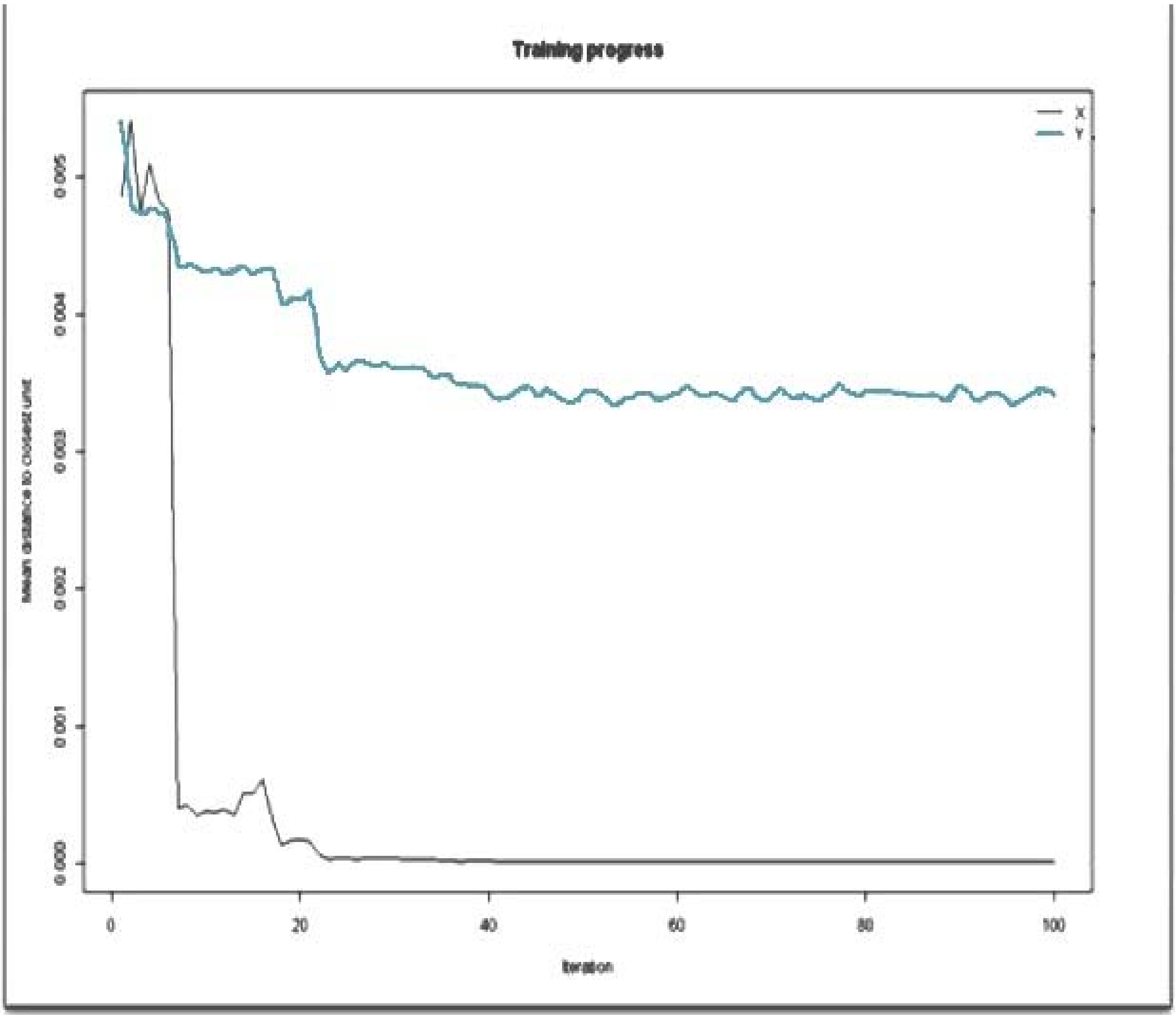


Figure 4-1 Training the SOM - after 100 iterations

In the next figure, the distribution of the data within the neurons in the computational layer of the neural network is shown. The output of the network is classified into acceptable or unacceptable weld quality. The neuron that is stimulated in the output layer will be the neuron whose data distribution matches the closest to the input data distribution. It can be seen from the figure that the neighbourhoods or the sampling regions of the neurons do not overlap with each other and thus information sharing between the neurons is nil or very low.

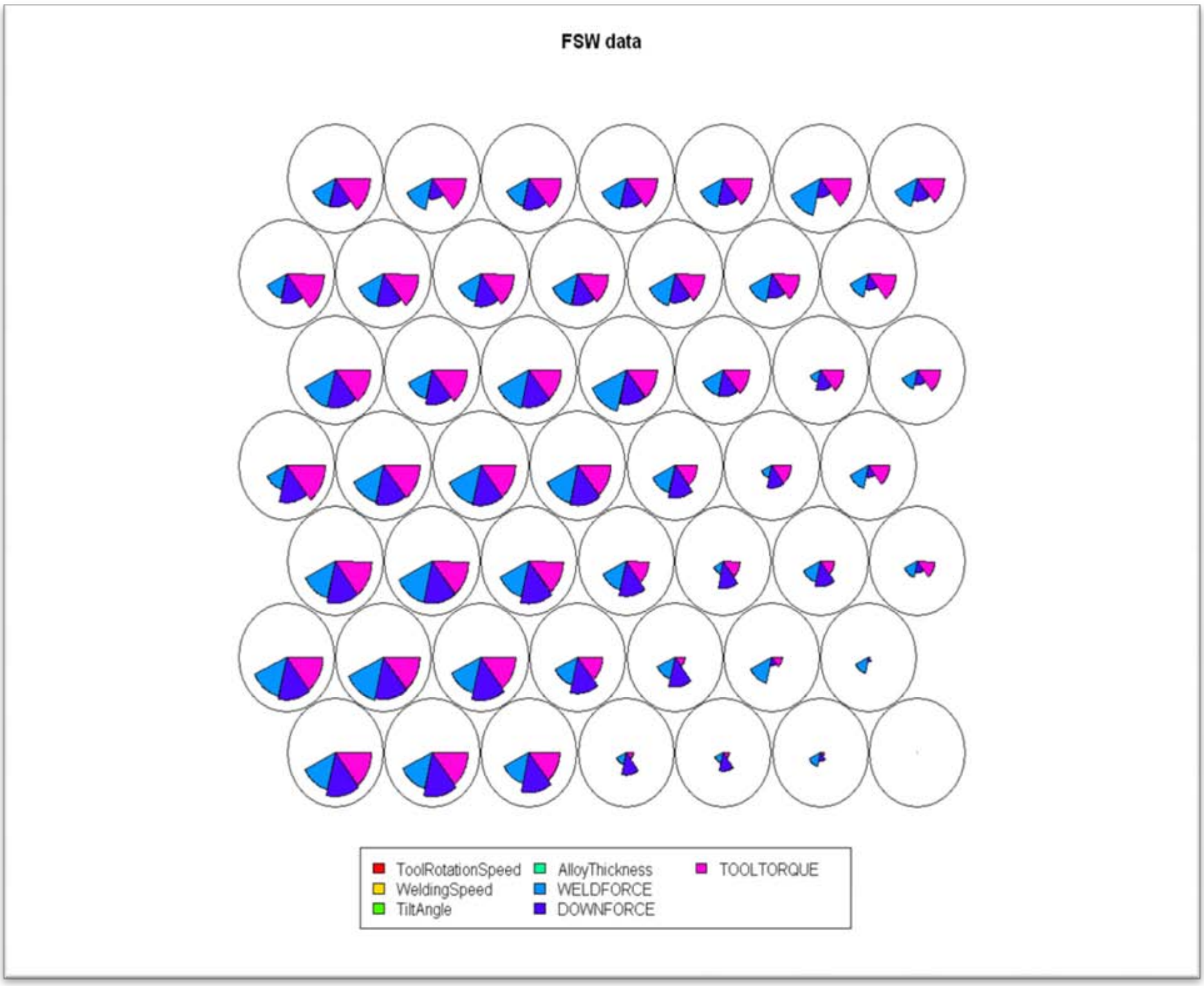


Figure 4-2 Data distribution within neurons during initial training phase

Figure 4-3 shows the neuronal plot of the SOM after 20,000 iterations during the training phase. The difference between Figures 4-2 & 4-3 is apparent. The neurons in Fig. 4-3 are more in number; the data distribution inside the neurons is gradually starting to form clusters or groups of neurons with analogous data distributions. This is the beginning of the classification stage. Also, the neuronal neighbourhoods are coming closer to each other ever so slightly indicating that the data distribution within the neurons is inclining towards being fuzzy or representative of the real world data.

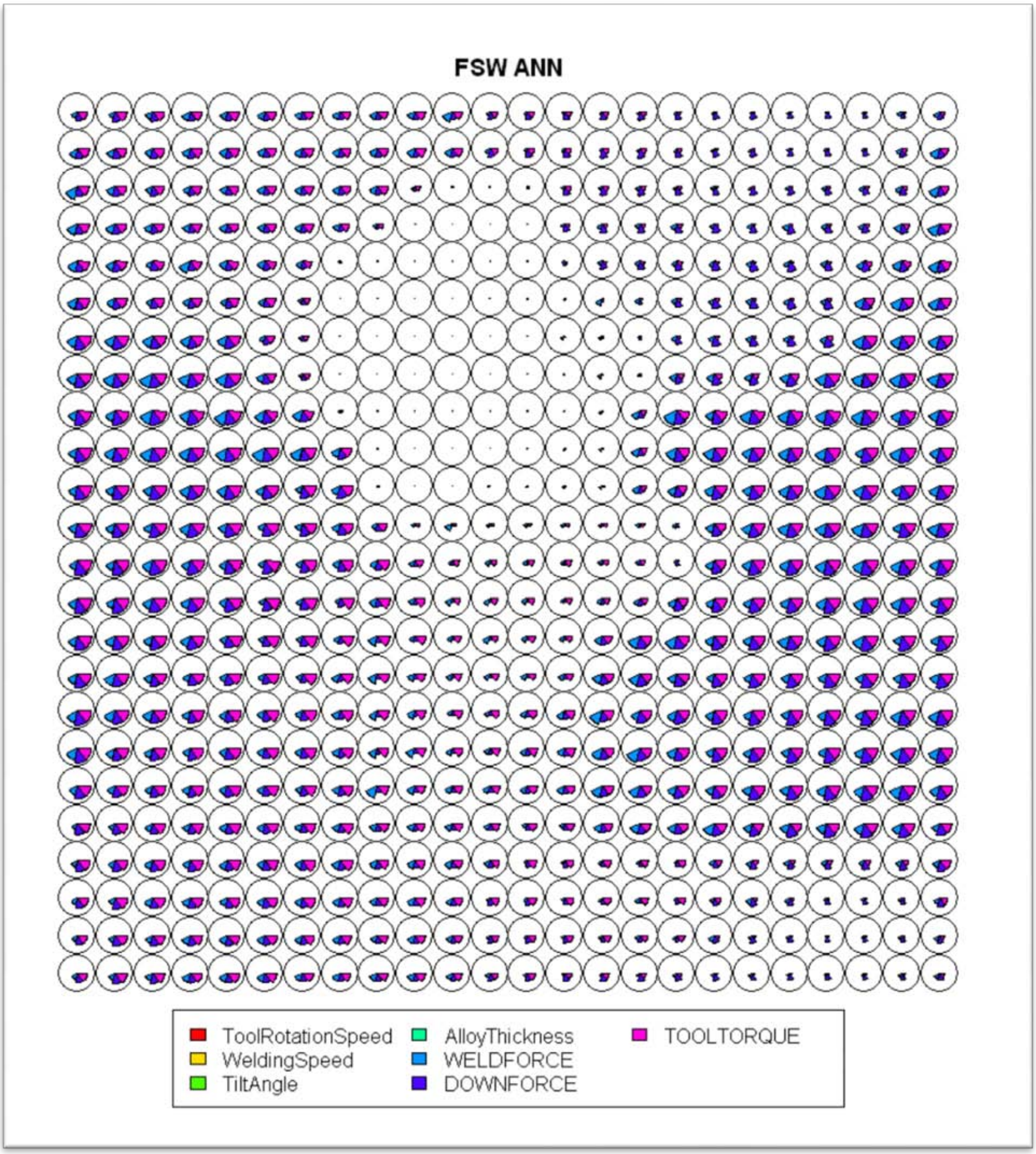


Figure 4-3 Beginnings of classification within neurons after 20,000 iterations

In figure 4-4 it can be see that the neurons have re-organised or classified themselves into patterns that match the input vectors. The neurons in the top left corner of the figure, for example, represent input vectors where all the forces acting on the metal plate are at a minimum. This is only a snapshot of the classified neuronal layer in the SOM and should not be considered to be the actual classification pattern of all the neurons in the SOM.

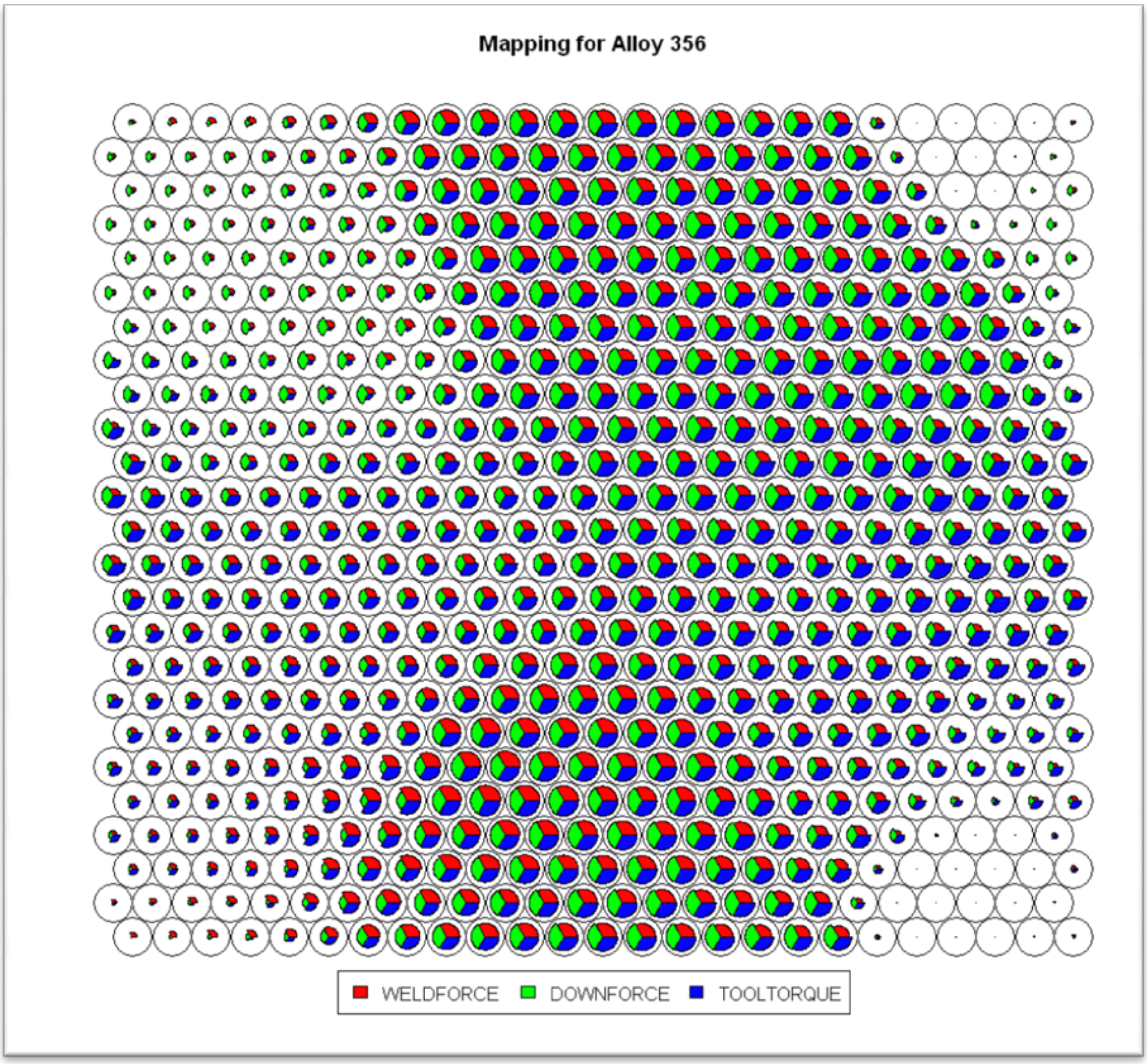


Figure 4-4 Classified neuronal layer representing the welding forces

4.4 Trained SOM

Figure 4-5 shows the neuronal plot of the SOM that has undergone the training or learning process and has re-organised itself to match the patterns in the input domain. This SOM is the result of 100,000 iterations of the program.

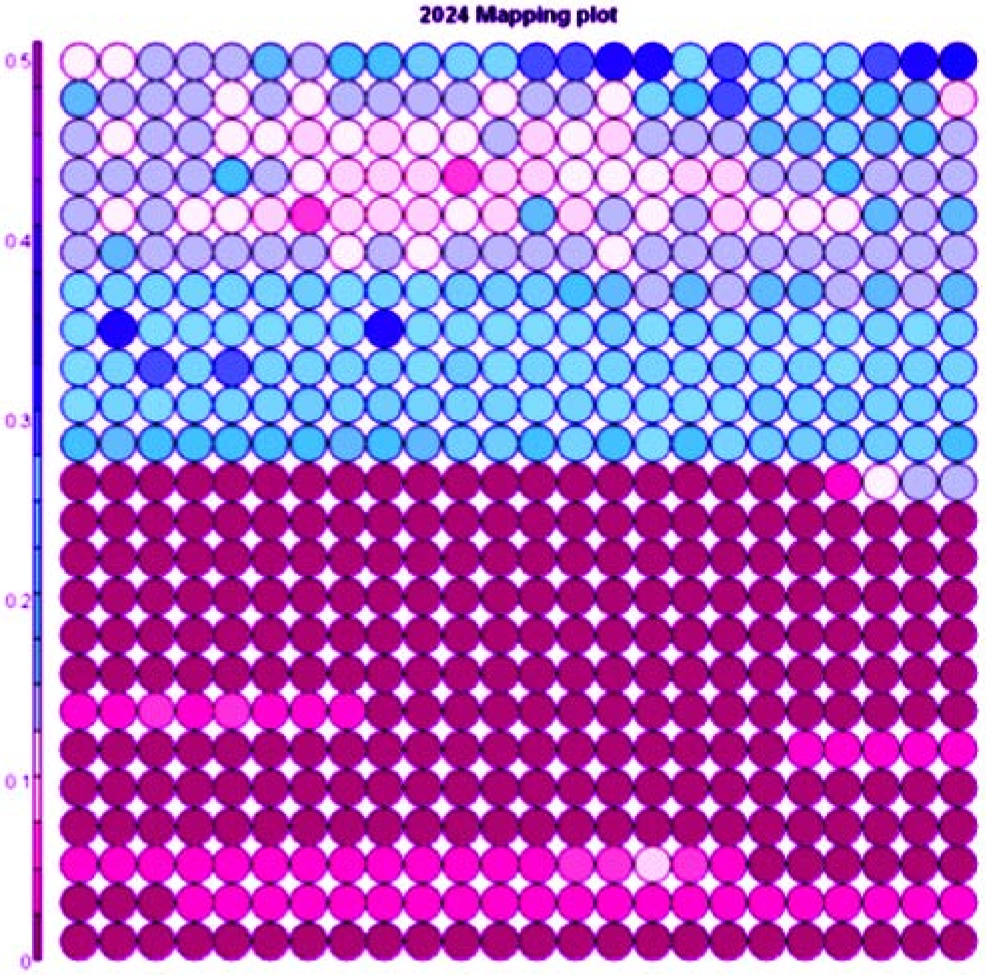


Figure 4-5 Classification after training

In Fig. 4-5, it can be see that the SOM has now re-organised itself to match the input patterns as closely as possible. The different colours of the neurons in the map represent or depict the varying degrees of the signal in the input domain. The neurons with similar data distributions are bunched together and there is significant overlap between the neuronal neighbourhoods – i.e. for a given set of inputs, a bunch of neurons in the output layer are excited as opposed to a single or small number of neurons that are excited in conventional ANNs. These relatively large numbers of excited neurons reach their desired outcome through a "consensus of opinion" and thus correspond to the real-world data distributions in the input domain much more accurately than ANNs that learn through the supervised learning paradigm. This classification and re-organisation of the neurons within the neural network is done autonomously by the SOM during the learning phase and involves no human supervision or input. This independent nature of the SOM serves us well for predicting the outcomes of non-linear processes in stochastic environments.

ANN Structure	Feed (mm / min)	Speed (rpm)	Percentage of correct predictions	
			Good Weld	Bad Weld
Initial network (after training run)	28	250	77	72
	28	500	78	74
	28	710	70	71
	28	1000	82	80
	56	250	85	76
	56	710	88	81
	56	1400	75	82
	112	250	78	87
	112	500	79	72
	112	710	85	73
	112	1000	83	78
	112	1400	72	77
	224	250	71	84
	224	500	80	78
	224	710	75	78
	224	1000	80	75
	224	1400	71	75
	315	500	68	92
	315	710	88	87
	450	500	90	88
	450	710	94	92
	450	1000	89	90

Table 9 Training results

Table 9 represents the percentages of correct predictions averaged over 40 test samples for each weld classification. The training phase helps the ANN model to increase its generalisation accuracy.

4.5 Testing with Unseen Data

Figure 4-6 shows the neuronal plot of the neural network during the prediction phase. The bottom right corner of the plot represents the neurons which correspond to the data distributions showing acceptable weld quality in the input domain. If the multi-sensory input to the SOM were indicative of acceptable levels of weld quality, then the output of the SOM would be generated by the excitatory response generated by these neurons. The neurons in the top-left corner of the map show the neurons that are representative of unacceptable levels of weld quality. Thus, the neural network has learned to distinguish between two different levels of weld quality.

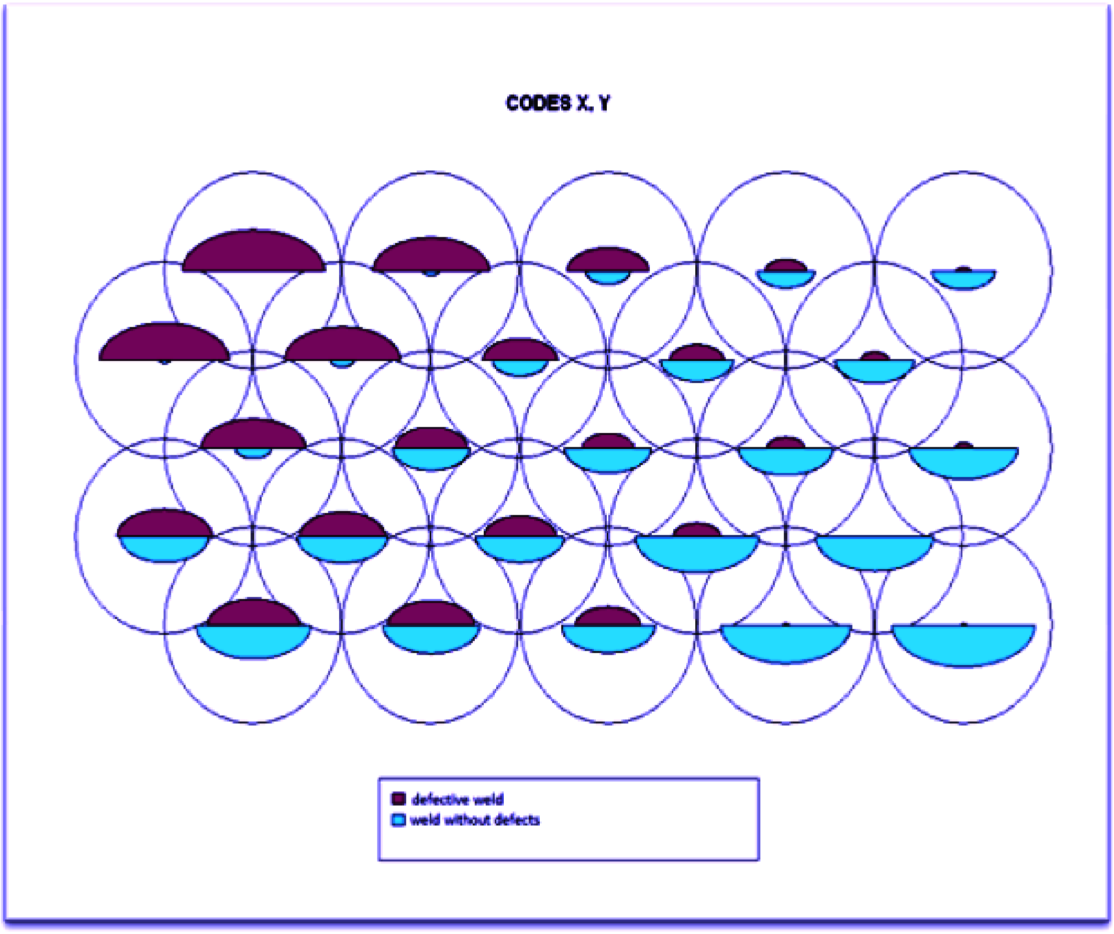


Figure 4-6 Classification results showing the grouping of neurons corresponding to weld qualities

Figure 4-7 shows the plot of the weld quality information within the neurons in the neural network. Here, the information is displayed on the plot by showing where every object is mapped and then by plotting a symbol in the neuron – which is the container circle.

The map was trained to predict weld quality - the position of the symbols (circular spots) within the circle indicates the mapping of the expected weld quality for the given sample, and the measure of weld quality is indicated by the concentration of the circular spots within the circumference of the container circle. In our case this is binary, i.e. good / bad weld quality. The location of the circular spots comprise of two components: first, the position of the unit onto which the sample is mapped and second, a random component within that unit.

The mapping in Figure 4-7 shows that the modelled output parameter (weld quality) indeed has a spatially smooth (or coherent) distribution. Moreover, in the prediction of the quality of the weld, the samples are ordered in such a way that the high quality welds (circular spots) are located in close proximity.

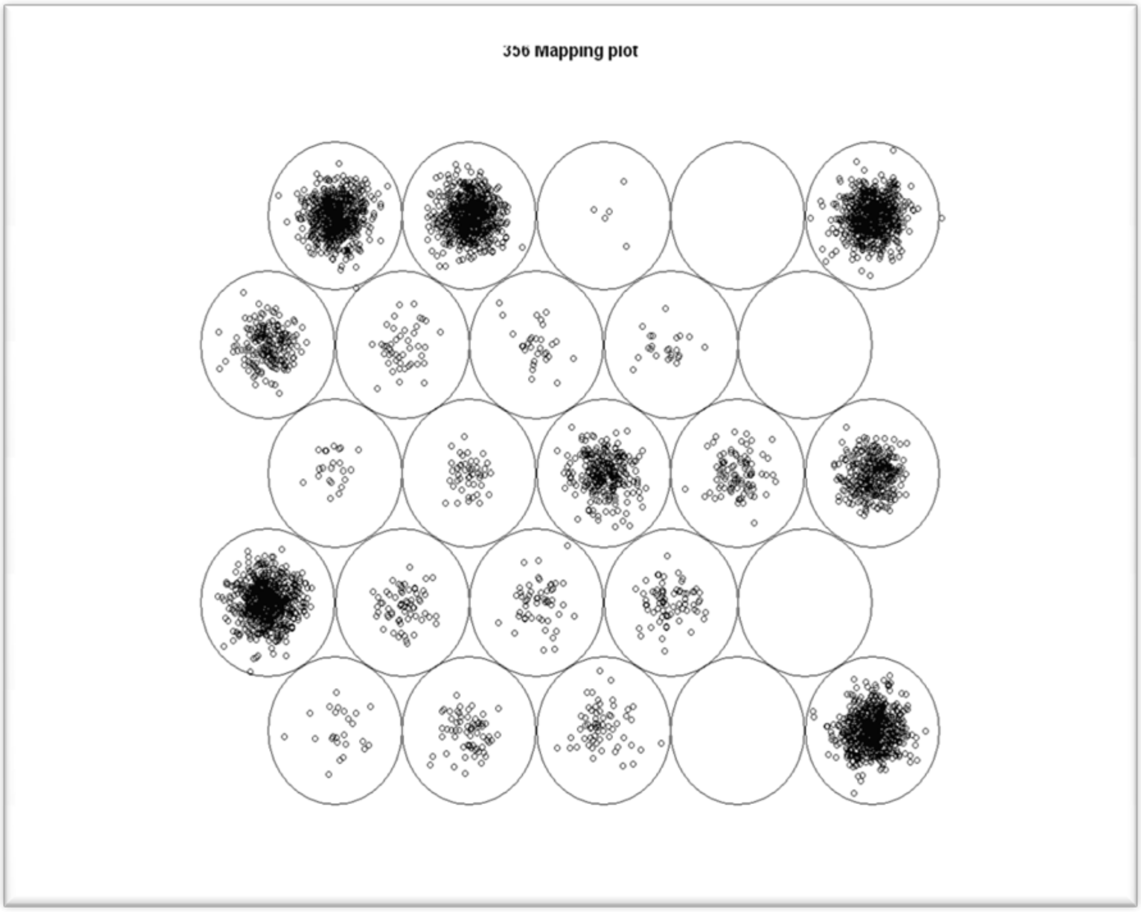


Figure 4-7: Prediction of Weld Quality

Figure 4-8 shows the prediction performed by the SOM in 100,000 iterations of the program.

The input domain is dynamic for the first 40,000 runs of the program and then stabilises. From the graph it is plain that the output patterns closely follow the input patterns and therefore the SOM prediction is accurate. The total number of miscalculations can be considered to be low in relation to the number of runs of the program that were made. This phenomenon would tend to indicate that the SOM has successfully learned the problem of weld quality detection and prediction and thus has good generalisation abilities. The odd misclassification probably suggests that there were local variations in the data caused due to the input domain being dependent on the machine variables or external noise. The misclassifications were also limited to adjacent data sets in the input domain displaying a general trend of classifying to a lower level of weld quality.

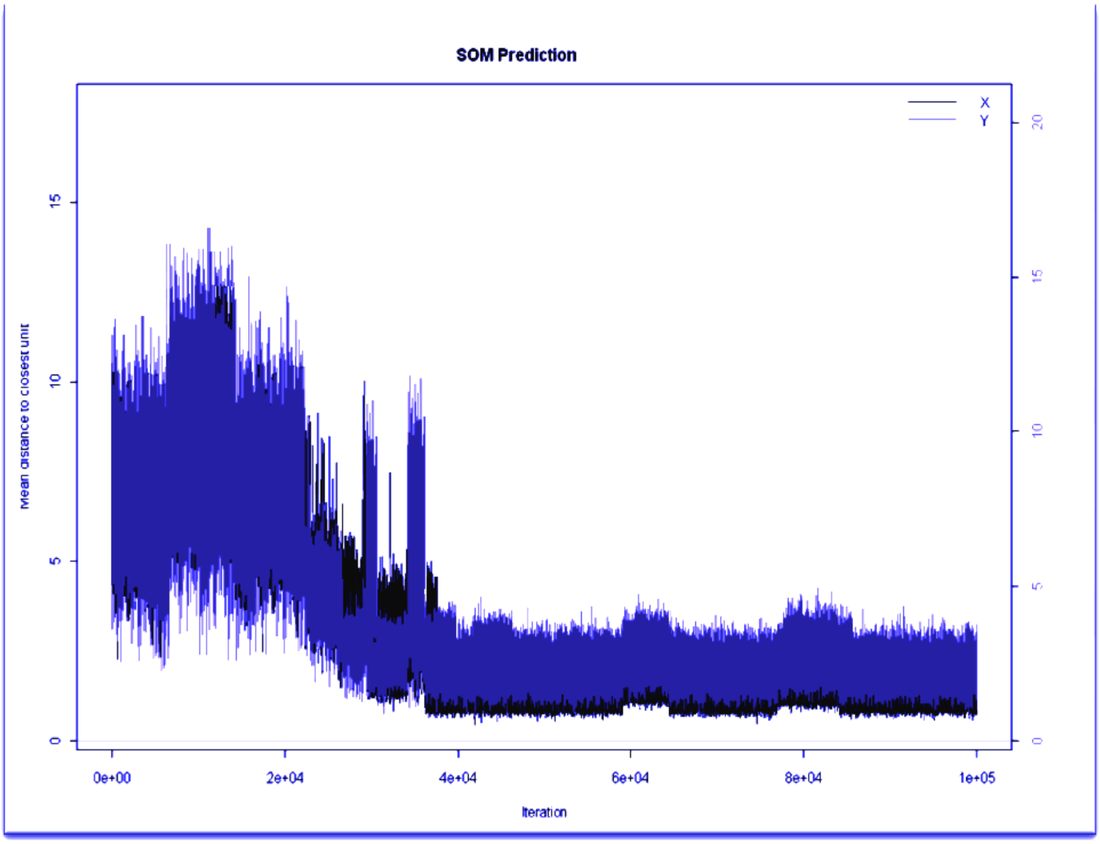


Figure 4-8 Prediction - Black is input signal and Blue is predicted output

ANN Structure	Feed (mm / min)	Speed (rpm)	Percentage of correct predictions	
			Good Weld	Bad Weld
Testing with unseen data (with trained network)	28	250	93	91
	28	500	90	92
	28	710	91	93
	28	1000	90	95
	56	250	92	90
	56	710	93	90
	56	1400	96	92
	112	250	90	94
	112	500	91	93
	112	710	93	90
	112	1000	95	90
	112	1400	94	91
	224	250	94	95
	224	500	96	95
	224	710	93	94
	224	1000	88	86
	224	1400	87	89
	315	500	88	90
	315	710	89	84
	450	500	82	84
	450	710	92	91
	450	1000	89	80

Table 10 Prediction in the trained network

The statistics in Table 10 indicate that the best performance was obtained with feed rate of 224 mm/min with a speed of 500 rpm. More correct classifications were made for the “Good Weld” cases. The ANN model sometimes missed or mixed up the two weld classes (good was classified as bad and vice versa). It was noticed that the down-force and the weld-force shared some similarities in the input domain, which could be a source of prediction difficulty.

5 Conclusions

The results presented in the previous chapter demonstrate the capabilities of the developed SOM system. The process of developing the capability of the neural network was done by evaluating a particularly demanding set of data obtained from different welding configurations under varying parametric conditions. Accurate classification of weld quality provides evidence that the self organising map was able to identify and quantify similarities between the training data sets and the test data sets. Moreover, it indicates that the data was highly process dependant with little machine dependencies. The self-organising map appears, therefore, to be an effective and efficient prediction model with adequate knowledge retained by reorganisation of the neurons constituting the map.

Weld quality classification and prediction is a complex phenomenon. Accurate modelling of the problem requires a highly evolved and comprehensive solution that draws knowledge from a spectrum of variables and learns dynamically. The work done in the field of FSW and ANN so far has mainly focused on the use of neural networks which learn using the supervised learning paradigm. These networks perform well under known conditions, but even a minor deviation from their predefined parameters can cause such networks to fail and become unreliable. The principal aim of this research is to construct a robust and efficient system for predicting weld quality of a friction stir welded plate.

The promising performance of the self organising predictive system presented here, is not merely a reflection of its capabilities - the pre-processing and integration techniques are the information suppliers on the problem, and as such, these have also demonstrated their adeptness allowing for judgemental decision making. Hence, successful classification is an indicator of the “system” performance.

The Self-Organising Map is a neural network that closely resembles how our brains function by mirroring the way in which we decode data from various sources (senses). This cognitive capability of the SOM possesses tremendous merit. To build on this capability, the information that is captured to assess the quality of a weld should contain primary and secondary sources of data which are used to arrive at a consensus of opinion.

The research presented here demonstrates the type of system which can successfully be employed to predict the weld quality of aluminum plates that are welded together using Friction Stir Welding. The system is an artificially intelligent program, providing information on a variety of weld parameters, which can be successfully deployed using a personal computer. Sufficiently long data samples, which ensure accuracy, need not result in prohibitively large computation times thus making the program’s application to online weld quality prediction a real possibility. The true robustness of the system is to be established by the application of the system in other industrial environments. The classification of weld quality using unsupervised neural networks is regarded as a strategic step forward in the progress towards the creation of a truly unmanned manufacturing environment.

The main point of departure in this work is the use of an unsupervised learning algorithm that is used for training the SOM - thus making it independent of human errors and perceptions that are enforced during the SOM training phase. Furthermore, the neural network is able to adapt to changing environments and conditions. This flexibility in adaption integrates well with the stochastic nature of industrial environments

Some of the key findings of the research may be summarized as below:

- There is a need for a reliable and robust weld quality prediction system that is capable of characterising weld quality in process time.
- It appears unlikely that the weld quality of aluminium plates can always accurately be predicted using experiential knowledge and human expertise under ever changing industrial conditions.
- Artificial Neural Networks are the best suited for modelling non-linear processes which make them inherently suitable for problems such as weld quality prediction which itself is a highly non-linear and stochastic process.
- Complex time domain information can be satisfactorily expressed using the power spectral density of the data.
- The unsupervised learning paradigm is proven to be better suited and more robust for the prediction of weld quality as opposed to the supervised learning paradigm used by the majority of research done so far.

- The same basic system, once trained, is capable of accurately classifying weld quality into varying levels of quality and not just good or bad.

- This study was conducted as a proof of principle of the effectiveness of the self organising map for detecting & predicting the presence of weld defect. While these results are promising, further exploration is warranted.

6 Further Work

The research presented in this thesis has led to the identification of a number of areas which are considered worthy of further investigation and development. They may be identified as follows:

- For the developed system to be widely accepted as a prediction tool for use in actual industrial environments, the system has to remain effective in dynamic scenarios with altering weld parameters. Development of a universal prediction system is a particularly active research area with the continuous introduction of more advanced technologies.

- A question that remains unanswered is the impact each machine will have on the output signals generated during the welding process. It is likely that each machine will contribute noise to the weld which may or may not be filtered by the neural network. Additional tests ought to be undertaken to determine the extent of the machine effect, the robustness of the network in the presence of the machine contribution to the signal, and the possibility of pre-filtering the machine dependent component of the output signals prior to classification

- Having established that weld quality prediction is possible, the next step would be to automate the modification of weld parameters for a given value of expected weld quality. This step is becoming considerably simpler to achieve with greater utilisation of micro-chip based machine controllers.

- The software and hardware elements for a comprehensive prediction system must be devised into a dedicated framework for data-capture, pre-processing and prediction. This framework should include systems that have the memory capabilities for simultaneous data capture of various dynamic characteristics and also allow suitably fast classification of weld quality for industry acceptance.

- Embedding the SOM architecture in a computerized FSW system would enable greater levels of un-manned welding operations. The ultimate aim would be to create machines that are capable of performing all the functions which are done by a machine operator in contemporary factory settings.

- The tool rotation and weld speeds are two the most influential weld parameter in terms of the plate weld quality – monitoring small changes in this variable is particularly crucial for the automation of the welding process. The collection of in-process data is undoubtedly the key element to effective weld quality prediction and this can be simply achieved by utilising the relationship of the input parameters to the collected time domain information. The use of relatively short time domain signals in this work is therefore considered to be highly transportable to other machining operations.

7 References

1. Tlusty, Jiri. *Manufacturing Processes and Equipment*: Prentice Hall; 1999.

2. Stephen W Kallee. *Railway Industry Friction Stir Welding Meeting*: 11th April 2002

3. Dr. John Baumann, Kurt Burton, Herb Bommer, Mike Matlack *Friction Stir Process capabilities for highly contoured applications*: 11th October 2006

4. Toshihisa Yamada, Hideshi Ohba. *Latest A-Train Railcar Enhancements*. Hitach

   Review.53: PP 10-14.

5. Thomas, WM; Nicholas, ED; Needham, JC; Murch, MG;Temple-Smith, P;Dawes, CJ.*Friction-stir butt welding*, GB Patent No. 9125978.8, International Patent No. PCT/GB92/02203, (1991)

6. R.S Mishra, Z.Y Ma/ *Materials Science and Engineering* R 50 (2005) 1-78

7. Bar-Yam, Yaneer (2003). *Dynamics of Complex Systems, Chapter 2*

8. X. Yao, "*Evolution of connectionist networks*," in Preprints Int. Symp. AI, Reasoning & Creativity, Queensland, Australia, Griffith Univ., 1991, pp. 49–52

9. Aleksander, I. and Morton, H. 2nd edition. *An introduction to neural computing*. Thomson Computer Press, 1995

10. Christos Stergiou and Dimitrios Siganos; *Neural Networks,* Available from: http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html [Accessed 16th June 2009]

11. L. Dubourg a, P. Dacheux b *Design and properties of FSW tools: a literature review*

12. Technology Division; Available from: http://www.technology-division.com/nn-opt/nn-overview.html [Accessed 27th April 2009];

13. B.H.ChandraShekar, Dr.G.Shoba; *Classification Of Documents Using Kohonen's self organizing map -* International Journal of Computer Theory and Engineering, Vol. 1, No. 5, December, 2009.

14. R Johnson, N L Horrex: *Preliminary examination of forces generated during the friction stir welding process* - TWI Report 696/2000, January 2000

15. W.J. Arbegast, P.J. Hartley, in: *Proceedings of the Fifth International Conference on Trends in Welding Research*, Pine Mountain, GA, USA, June 1–5, 1998, p. 541.

16. T. Hashimoto, S. Jyogan, K. Nakata, Y.G. Kim, M. Ushio, in: Proceedings of the First International Symposium on Friction Stir Welding, Thousand Oaks, CA, USA, June 14–16, 1999.

17. Dr. Tracy W. Nelson *FRICTION STIR WELDING - A brief Review and Perspective for the Future,* Friction Stir Welding and Processing III - TMS 2005, San Francisco, CA.

18. Terry Dickerson1, Hugh R Shercliff1 and Henrik Schmidt2,3 - *A Weld Marker Technique for Flow Visualization in FSW,* 4th International Symposium on Friction Stir Welding. Park City, Utah, USA, 14-16 May 2003.

19. A.P. Reynolds and W. Tang, "*Alloy, Tool Geometry, and Process Parameter Effects of Friction Stir Weld Energies and Resultant FSW Joint*

Properties", Friction Stir Welding and Processing, ed. by K.V. Jata, M.W. Mahoney, R.S. Mishra, S.L. Semiatin, and D.P. Field, (TMS 2001),15-23.

20. Esprit, I.F.Croall, J.P.Mason - *Industrial Applications of Neural Networks* (research reports) – Year of Publication 2002
21. K.Y. Benyounis, A.G. Olabi - *Optimisation of different welding processes* - Advances in Engineering Software 39 (2008) 483–496
22. Andersen K, Cook G, Karsai G, Ramaswamy K. *Artificial neural network applied to arc welding process modelling and control*. IEEE Trans Ind Appl 1990; 26(5):824–30.
23. Cook G, Barnett RJ, Andersen K, Strauss AM. *Weld modelling and control using artificial neural networks*. IEEE Trans Ind Appl1995; 31(6):1484–91.
24. Hasan Okuyucu a, Adem Kurt a, Erol Arcaklioglu -*Artificial neural network application to the friction stir welding of aluminum plates- Materials and Design 28 (2007) 78–84*
25. H.T. Siegelmann; *Neural and Super-Turing Computing*, Philosophy 2002.
26. Kathryn Beamish, Andy Ezeilo, Simon Smith, TWI Ltd UK ; *DEVELOPMENT OF A LOW COST FRICTION STIR WELDING MONITORING SYSTEM*:: 2007
27. Oludele Awodele and Olawale Jegede . *Neural Networks & Application in Engineering*. Proceedings of Informing Science & IT Education Conference (InSITE) 2009
28. Robert, W. and J. Messler, *Principles of welding: processes, physics, chemistry, and metallurgy*. 1st ed. 1999, New York: John Wiley. 3.

29. A J Leonard & S A Lockyer, *Flaws in Friction Stir Welds.* 4th International Symposium on Friction Stir Welding, Park City, Utah, 14-16 May 2003

30. Teuvo Kohonen, *Self-Organizing Maps*, Springer Series in Information Sciences, Vol. 30, Springer, Berlin, Heidelberg, New York, 1995, 1997, 2001, 3rd edition.

31. G. Zhang et al. / International Journal of Forecasting 14 (1998) 35 –62

32. I.A. Basheera , M. Hajmeer *Artificial neural networks: fundamentals, computing, design, and Application*, Journal of Microbiological Methods 43 (2000) 3–31

33. Fujii, H., et al., *Effect of tool shape on mechanical properties and microstructure of friction stir welded aluminum alloys*. Materials Science and Engineering: A, 2006.

34. Kim, Y.G., et al., *Three defect types in friction stir welding of aluminum die casting alloy*. Materials Science and Engineering: A, 2006. 415(1-2): p. 250-254.

35. Kumar, K. and S.V. Kailas, *The role of friction stir welding tool on material flow and weld formation. Materials Science and Engineering*: A, 2007

36. Y. K. Yousif , K. M. Daws , B. I. Kazem, *Prediction of Friction Stir Welding Characteristic Using Neural Networks.,* ISSN 1995-666, Pages 151 – 155

37. Ai –junkie - *Kohonen's Self Organising Feature Maps* – Tutorials http://www.ai-junkie.com/ann/som/som1.html [Accessed 5th Jan 2010]

38. Teuvo Kohonen, Olli Simula, *Engineering Applications of the Self Organising Map,* Proceedings of the IEEE,VOL 84, 1996

39. Teuvo Kohonen and Erkki Oja. *A note on a simple self-organizing process.* Technical Report TKK-F-A474, 1982, Helsinki University of Technology.

40. R FAQ- http://cran.r-project.org/doc/FAQ/R-FAQ.html#Introduction Version 2.10.2010-02-04.ISBN 3-900051-08-9

41. Pyle D. *Data Preparation for Data Mining*. Morgan Kaufman Publishers, San Francisco, 1999.

42. Mario Van Velzen, *Self Organising Map*s, October 2007

43. Gershenfeld, N.A.,Weigend, A.S., 1993 The future of time series

44. Hill, T., Marquez, L., O'Connor, M., Remus, W., 1994. *Artificial neural networks for forecasting and decision making*

45. Pryzdatek J: 'A ship classification view on friction stir welding', Proc 1st International Symposium on friction stir welding, Thousand Oaks, CA, USA

46. AWS D17.3 'Specification for friction stir welding of aluminium alloysfor aerospace applications', Draft.

47. BS EN 288-3:1992. *Specification and approval of welding procedures for metallic materials.* Welding procedure tests for the arc welding of aluminium and it's alloys

48. R. Wehrens, L. M. C. B. (2007). Self- and Super-organising Maps in R: the kohonen package. *Journal of Statistical Software, 21(5).*

49. Vitek JM, Iskander YS, Oblow EM, et al. *Neural network modelling of pulsed-laser weld pool shapes in aluminium alloy welds*. Proceedings of 5th international conference on trends in welding research, 1998.

50. L Fratini and G Buffa- *Continuous dynamic recrystallization phenomena modelling in friction stir welding* Proceedings of the Institution of Mechanical Engineers; May 2007; 221, B5; ProQuest Science Journals pg. 857.

51. Kadim Tas¸demir and Erzsébet Merényi, *Exploiting Data Topology in Visualization and Clustering of Self-Organizing Maps.* IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 20, NO. 4, APRIL 2009

52. Chung-Chian Hsu, *Generalising Self Organising map for Categorical Data,* IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 17, NO. 2, MARCH 2006

53. Teuvo Kohonen, *Self-organizing neural projections,* 2006 – Elsevier.

54. H Tokutaka, M Ohkita, K Fujimura, *Self-Organizing Maps and its Application,* 2007 - Springer, Japan

55. S Furao, T Ogura, O Hasegawa, *An enhanced self-organizing incremental neural network for online unsupervised learning,* Neural Networks, 2007 – Elsevier

56. L Peeters, F Bação, V Lobo, A Dassargues*; Kohonen's Self-Organizing Map;* 2006 - en.scientificcommons.org. Publication View. 42479385

57. Teuvo Kohonen , Merja Oja , Samuel Kaski , Panu Somervuo, *Self-Organizing Maps: introduction;* 2008 - Citeseer

58. Kevin Burn [1] and Geoffrey Home; *Environment classification using Kohonen self-organizing maps;* Expert Systems, 2008 - interscience.wiley.com

59. H Yin; *On multidimensional scaling and the embedding of self-organising maps;* Neural Networks, 2008 – Elsevier

60. G. Simona , A. Lendasse, M. Cottrell, J.-C. Fort and M. Verleysen, *Time Series Forecasting: Obtaining Long Term Trends with SOMs;* arXiv:cs/0701052v1 [cs.LG] 8 Jan 2007

61. J Huysmans, B Baesens, J Vanthienen, T van gestel; *Failure prediction with self organizing maps;* Pergamon-elsevier science ltd Apr 2006

62. J. Asamer, K. Din, and W. Toplak; *Self Organizing Maps for Traffic Prediction;* Artificial Intelligence and Applications, Feb 2007

63. M. Herbst and M. C. Casper; *Towards model evaluation and identification using Self-Organizing Maps;* Hydrology and Earth System Sciences, 2008 - Citeseer

64. Elie Prudhomm, Stephane Lallich; *Optimization of Self-Organizing Maps Ensemble in prediction;* Conference on Data Mining 2008

Appendix – A

Self-Organising Map Algorithm Program Listing

Function SOM:

```
function (data, grid = somgrid(), rlen = 100, alpha = c(0.05,
    0.01), radius = quantile(nhbrdist, 0.67) * c(1, -1), init,
    toroidal = FALSE, n.hood, keep.data = TRUE)
{
    if (!is.numeric(data))
        stop("Argument data should be numeric")
    data <- as.matrix(data)
    nd <- nrow(data)
    ng <- nrow(grid$pts)
    if (missing(init)) {
        init <- data[sample(1:nd, ng, replace = FALSE), , drop = FALSE]
    }
    else {
        init <- as.matrix(init)
        if (nrow(init) != ng | ncol(init) != ncol(data) | !is.numeric(init))
            stop("incorrect init matrix supplied")
    }
    codes <- init
    if (missing(n.hood)) {
        n.hood <- switch(grid$topo, hexagonal = "circular", rectangular = "square")
    }
    else {
        n.hood <- match.arg(n.hood, c("circular", "square"))
    }
    grid$n.hood <- n.hood
    nhbrdist <- unit.distances(grid, toroidal)
    if (length(radius) == 1)
        radius <- sort(radius * c(1, -1), decreasing = TRUE)
    changes <- rep(0, rlen)
    res <- .C("SOM_online", data = as.double(data), codes = as.double(codes),
        nhbrdist = as.double(nhbrdist), alpha = as.double(alpha),
        radii = as.double(radius), changes = as.double(changes),
        n = as.integer(nrow(data)), p = as.integer(ncol(data)),
        ncodes = as.integer(nrow(init)), rlen = as.integer(rlen),
        PACKAGE = "kohonen")
    changes <- matrix(res$changes, ncol = 1)
    codes <- res$codes
    dim(codes) <- dim(init)
    colnames(codes) <- colnames(init)
    if (keep.data) {
        mapping <- map.kohonen(list(codes = codes), newdata = data)
        structure(list(data = data, grid = grid, codes = codes,
            changes = changes, alpha = alpha, radius = radius,
            toroidal = toroidal, unit.classif = mapping$unit.classif,
            distances = mapping$distances, method = "som"), class = "kohonen")
    }
    else {
        structure(list(grid = grid, codes = codes, changes = changes,
            alpha = alpha, radius = radius, toroidal = toroidal,
            method = "som"), class = "kohonen")
    }
}
```

Function to initialise:

```
function (data, Y, grid = somgrid(), rlen = 100, alpha = c(0.05,
    0.01), radius = quantile(nhbrdist, 0.67) * c(1, -1), xweight = 0.75,
    contin = !(all(rowSums(Y) == 1)), toroidal = FALSE, n.hood,
    keep.data = TRUE)
{
    if (!is.numeric(data))
        stop("Argument data should be numeric")
    data <- as.matrix(data)
    nd <- nrow(data)
    nx <- ncol(data)
    if (is.vector(Y))
        Y <- matrix(Y, ncol = 1)
    ny <- ncol(Y)
    ng <- nrow(grid$pts)
    xdists <- ydists <- rep(0, ng)
    starters <- sample(1:nd, ng, replace = FALSE)
    init <- data[starters, , drop = FALSE]
    codes <- init
    if (!contin) {
        codeYs <- 0.5 + 0.5 * (Y[starters, ] - 0.5)
    }
    else {
        codeYs <- Y[starters, ]
    }
    if (missing(n.hood)) {
        n.hood <- switch(grid$topo, hexagonal = "circular", rectangular = "square")
    }
    else {
        n.hood <- match.arg(n.hood, c("circular", "square"))
    }
    grid$n.hood <- n.hood
    nhbrdist <- unit.distances(grid, toroidal)
```

```
if (length(radius) == 1)
    radius <- sort(radius * c(1, -1), decreasing = TRUE)
changes <- rep(0, rlen * 2)
if (contin) {
    res <- .C("BDK_Eucl", data = as.double(data), Ys = as.double(Y),
        codes = as.double(codes), codeYs = as.double(codeYs),
        nhbrdist = as.double(nhbrdist), alpha = as.double(alpha),
        radii = as.double(radius), xweight = as.double(xweight),
        changes = as.double(changes), xdists = as.double(xdists),
        ydists = as.double(ydists), n = as.integer(nd), px = as.integer(nx),
        py = as.integer(ny), ncodes = as.integer(ng), rlen = as.integer(rlen),
        PACKAGE = "kohonen")
}
else {
    res <- .C("BDK_Tani", data = as.double(data), Ys = as.double(Y),
        codes = as.double(codes), codeYs = as.double(codeYs),
        nhbrdist = as.double(nhbrdist), alpha = as.double(alpha),
        radius = as.double(radius), xweight = as.double(xweight),
        changes = as.double(changes), xdists = as.double(xdists),
        ydists = as.double(ydists), n = as.integer(nd), px = as.integer(nx),
        py = as.integer(ny), ncodes = as.integer(ng), rlen = as.integer(rlen),
        PACKAGE = "kohonen")
}
changes <- matrix(res$changes, ncol = 2)
codes <- list(X = matrix(res$codes, nrow(init), ncol(init)),
    Y = matrix(res$codeYs, ng, ny))
colnames(codes$Y) <- colnames(Y)
if (keep.data) {
    mapping <- map.kohonen(list(codes = codes), newdata = data,
        whatmap = 1)
    structure(list(data = data, Y = Y, contin = contin, grid = grid,
            codes = codes, changes = changes, alpha = alpha,
            radius = radius, toroidal = toroidal, unit.classif = mapping$unit.classif,
            distances = mapping$distances, method = "bdk"), class = "kohonen")
    }
    else {
        structure(list(contin = contin, grid = grid, codes = codes,
            changes = changes, alpha = alpha, radius = radius,
            toroidal = toroidal, method = "bdk"), class = "kohonen")
    }
}
```

Function for mapping:

```
function (xdim = 8, ydim = 6, topo = c("rectangular", "hexagonal"))
{
    topo <- match.arg(topo)
    x <- 1:xdim
    y <- 1:ydim
    pts <- as.matrix(expand.grid(x = x, y = y))
    if (topo == "hexagonal") {
        pts[, 1] <- pts[, 1] + 0.5 * (pts[, 2]%%2)
        pts[, 2] <- sqrt(3)/2 * pts[, 2]
    }
    res <- list(pts = pts, xdim = xdim, ydim = ydim, topo = topo)
    class(res) <- "somgrid"
    res
}
<environment: namespace:class>
```

Function for training:

```
function (data, Y, grid = somgrid(), rlen = 100, alpha = c(0.05,
    0.01), radius = quantile(nhbrdist, 0.67) * c(1, -1), xweight = 0.5,
    contin = !(all(rowSums(Y) == 1)), toroidal = FALSE, n.hood,
    keep.data = TRUE)
{
    if (!is.numeric(data))
        stop("Argument data should be numeric")
    data <- as.matrix(data)
    nd <- nrow(data)
    nx <- ncol(data)
    if (is.vector(Y))
        Y <- matrix(Y, ncol = 1)
    ny <- ncol(Y)
    ng <- nrow(grid$pts)
    xdists <- ydists <- rep(0, ng)
    starters <- sample(1:nd, ng, replace = FALSE)
    init <- data[starters, , drop = FALSE]
    codes <- init
    if (!contin) {
        codeYs <- 0.5 + 0.5 * (Y[starters, ] - 0.5)
    }
    else {
        codeYs <- Y[starters, ]
    }
    if (missing(n.hood)) {
        n.hood <- switch(grid$topo, hexagonal = "circular", rectangular = "square")
    }
    else {
        n.hood <- match.arg(n.hood, c("circular", "square"))
    }
    grid$n.hood <- n.hood
    nhbrdist <- unit.distances(grid, toroidal)
```

```
if (length(radius) == 1)
    radius <- sort(radius * c(1, -1), decreasing = TRUE)
changes <- rep(0, rlen * 2)
if (contin) {
    res <- .C("XYF_Eucl", data = as.double(data), Ys = as.double(Y),
        codes = as.double(codes), codeYs = as.double(codeYs),
        nhbrdist = as.double(nhbrdist), alpha = as.double(alpha),
        radii = as.double(radius), xweight = as.double(xweight),
        changes = as.double(changes), xdists = as.double(xdists),
        ydists = as.double(ydists), n = as.integer(nd), px = as.integer(nx),
        py = as.integer(ny), ncodes = as.integer(ng), rlen = as.integer(rlen),
        PACKAGE = "kohonen")
}
else {
    res <- .C("XYF_Tani", data = as.double(data), Ys = as.double(Y),
        codes = as.double(codes), codeYs = as.double(codeYs),
        nhbrdist = as.double(nhbrdist), alpha = as.double(alpha),
        radius = as.double(radius), xweight = as.double(xweight),
        changes = as.double(changes), xdists = as.double(xdists),
        ydists = as.double(ydists), n = as.integer(nd), px = as.integer(nx),
        py = as.integer(ny), ncodes = as.integer(ng), rlen = as.integer(rlen),
        PACKAGE = "kohonen")
}
changes <- matrix(res$changes, ncol = 2)
colnames(changes) <- c("X", "Y")
codes <- list(X = matrix(res$codes, nrow(init), ncol(init)),
    Y = matrix(res$codeYs, ng, ny))
colnames(codes$Y) <- colnames(Y)
if (keep.data) {
    mapping <- map.kohonen(list(codes = codes), newdata = data,
        whatmap = 1)
        structure(list(data = data, Y = Y, contin = contin, grid = grid,
            codes = codes, changes = changes, alpha = alpha,
            radius = radius, toroidal = toroidal, unit.classif = mapping$unit.classif,
            distances = mapping$distances, method = "xyf"), class = "kohonen")
    }
    else {
        structure(list(contin = contin, grid = grid, codes = codes,
            changes = changes, alpha = alpha, radius = radius,
            toroidal = toroidal, method = "xyf"), class = "kohonen")
    }
}
```

Function for predicting:

```
function (object, newdata, trainX, trainY, unit.predictions = NULL,
    threshold = 0, whatmap = NULL, weights = 1, ...)
{
    mapping <- NULL
    if (missing(newdata)) {
        if (!is.null(object$data)) {
            newdata <- object$data
            mapping <- object$unit.classif
        }
        else {
            stop("No data given with which to predict")
        }
    }
    if (is.null(mapping))
        mapping <- map(object, newdata, whatmap, weights)$unit.classif
    if (is.null(unit.predictions)) {
        if (object$method %in% c("xyf", "bdk")) {
```

```
  unit.predictions <- object$codes$Y
}
else {
  if (object$method == "supersom" & !is.null(whatmap)) {
    whatmap <- check.whatmap(object, whatmap)
    if (length(whatmap) < length(object$data))
     unit.predictions <- object$codes[-whatmap]
  }
  else {
    if (missing(trainY))
     stop("For unsupervised forms of mapping, trainY is required")
    if (is.list(trainY))
     stop("Prediction for trainY lists not implemented")
    if (is.vector(trainY))
     trainY <- matrix(trainY, ncol = 1)
    nY <- ncol(trainY)
    trainingMapping <- NULL
    if (missing(trainX) & !is.null(object$data)) {
     trainX <- object$data
     trainingMapping <- object$unit.classif
    }
    nX <- ifelse(is.list(trainX), nrow(trainX[[1]]),
     nrow(trainX))
    if (nX != nrow(trainY))
     stop("Unequal number of rows in trainX and trainY")
    if (is.null(trainingMapping))
     trainingMapping <- map(object, trainX)$unit.classif
    unit.predictions <- matrix(NA, nrow(object$grid$pts),
     nY)
    huhn <- aggregate(trainY, by = list(cl = trainingMapping),
     mean)
```

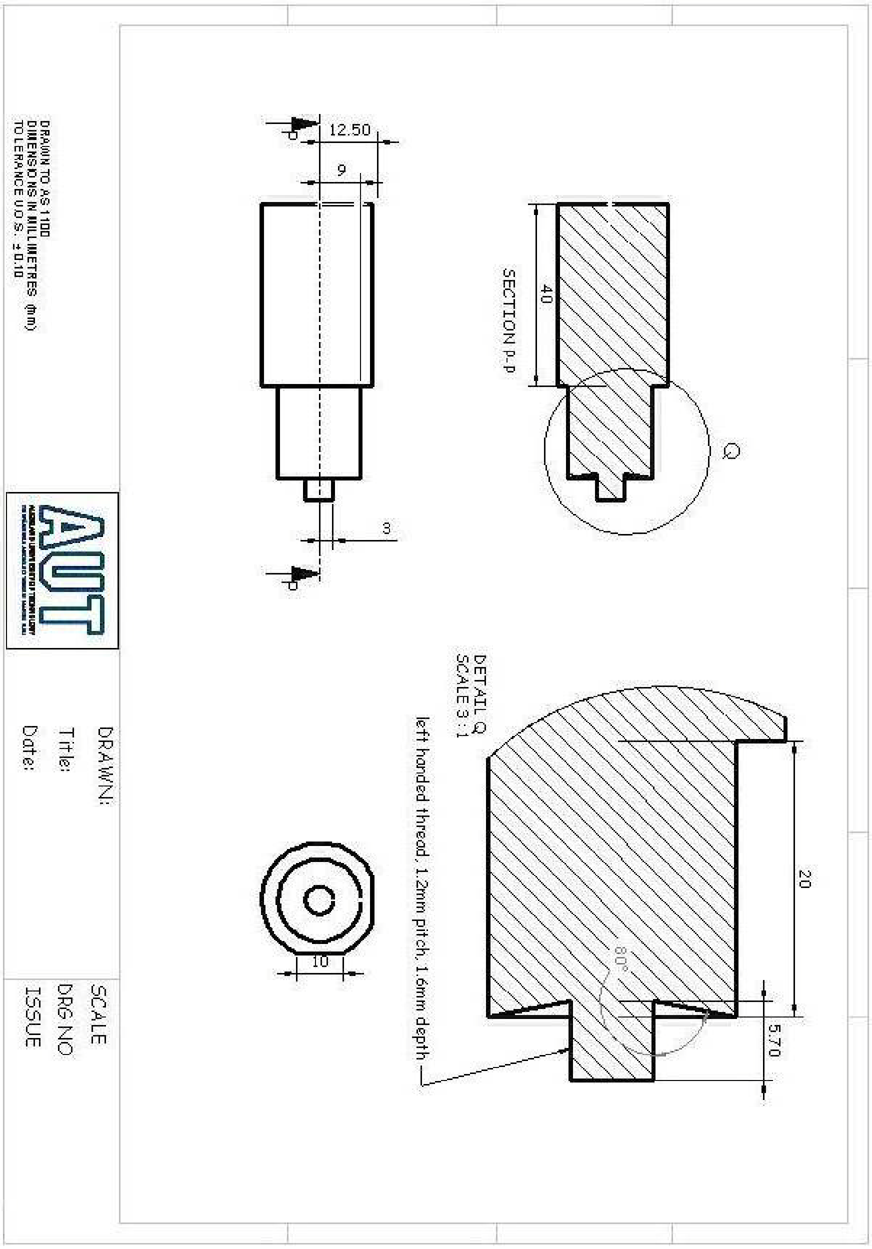
```
                if (R.version$major <= "2" & R.version$minor <
                  "6.0") {
                  unit.predictions[sort(as.numeric(levels(huhn[,
                    1]))), ] <- as.matrix(huhn[, -1])
                }
                else {
                  unit.predictions[huhn[, 1], ] <- as.matrix(huhn[,
                    -1])
                }
                nas <- which(apply(unit.predictions, 1, function(x) all(is.na(x))))
                nhbrdist <- unit.distances(object$grid, object$toroidal)
                for (i in seq(along = nas)) {
                  unit.predictions[nas[i], ] <- colMeans(unit.predictions[nhbrdist[nas[i],
                    ] == 1, , drop = FALSE], na.rm = TRUE)
                }
                colnames(unit.predictions) <- colnames(trainY)
            }
        }
    }
    if (!is.null(object$contin) && !object$contin) {
        prediction <- classmat2classvec(unit.predictions, threshold =
threshold)[mapping]
    }
    else {
        if (is.list(unit.predictions)) {
            prediction <- sapply(unit.predictions, function(x) x[mapping])
        }
        else {
            prediction <- unit.predictions[mapping, ]
        }
    }
    list(prediction = prediction, unit.classif = mapping, unit.predictions =
unit.predictions)
}
```

Function for plotting:

```
function (x, type = c("codes", "changes", "counts", "mapping",
    "property", "quality"), classif = NULL, labels = NULL, pchs = NULL,
    main = NULL, palette.name = heat.colors, ncolors, bgcol = NULL,
    zlim = NULL, heatkey = TRUE, property, contin, whatmap = NULL,
    codeRendering = NULL, keepMargins = FALSE, ...)
{
    type <- match.arg(type)
    switch(type, mapping = plot.kohmapping(x, classif, main,
        labels, pchs, bgcol, keepMargins, ...), property = plot.kohprop(x,
        property, main, palette.name, ncolors, zlim, heatkey,
        contin, keepMargins, ...), codes = plot.kohcodes(x, main,
        bgcol, whatmap, codeRendering, keepMargins, ...), quality =
plot.kohquality(x,
        classif, main, palette.name, ncolors, zlim, heatkey,
        keepMargins, ...), counts = plot.kohcounts(x, classif,
        main, palette.name, ncolors, zlim, heatkey, keepMargins,
        ...), changes = plot.kohchanges(x, main, keepMargins,
        ...))
    invisible()
}
function (x, main, keepMargins, ...)
{
    if (is.null(main))
        main <- "Training progress"
    nmaps <- ncol(x$changes)
    if (nmaps > 1) {
        if (!is.null(colnames(x$changes))) {
            varnames <- colnames(x$changes)
        }
        else {
            varnames <- paste("Matrix", 1:ncol(x$changes))
        }
    }
    if (nmaps == 2) {
        if (!keepMargins) {
            opar <- par("mar")
            on.exit(par(mar = opar))
        }
        par(mar = c(5.1, 4.1, 4.1, 4.1))
        huhn <- x$changes
        huhn[, 2] <- max(x$changes[, 1]) * huhn[, 2]/max(x$changes[,
            2])
        ticks <- pretty(x$changes[, 2], length(axTicks(2)))
    }
    else {
        huhn <- x$changes
    }
    matplot(huhn, type = "l", lty = 1, main = main, ylab = "Mean distance to
closest unit",
        xlab = "Iteration", ...)
    abline(h = 0, col = "gray")
    if (nmaps == 2)
        axis(4, col.axis = 2, at = ticks * max(x$changes[, 1])/max(x$changes[,
            2]), labels = ticks)
    if (nmaps > 1)
        legend("topright", legend = varnames, lty = 1, col = 1:nmaps,
          bty = "n")
}
```

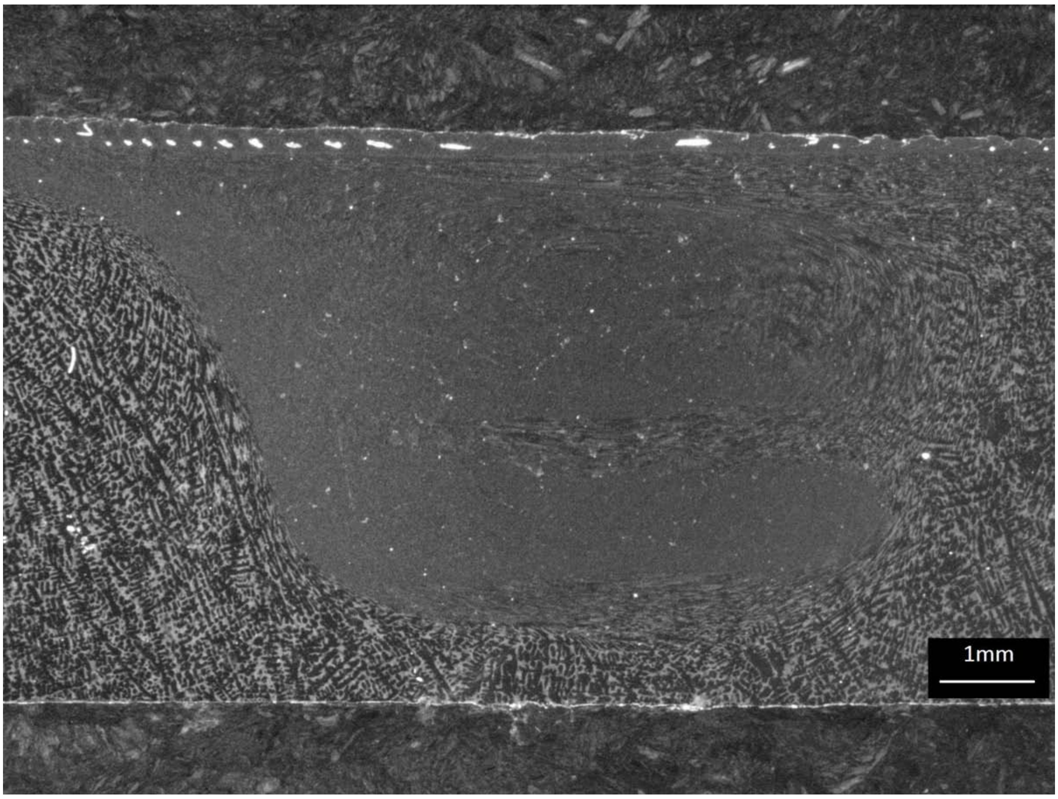
Appendix – B

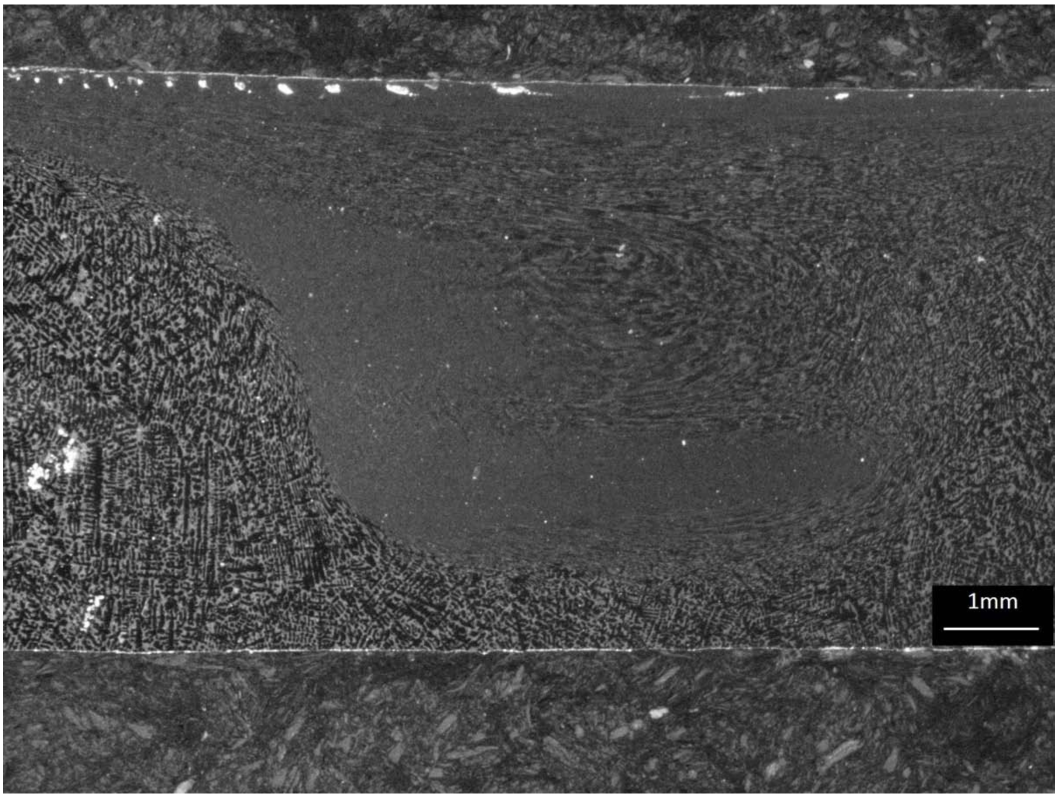
Tool specifications

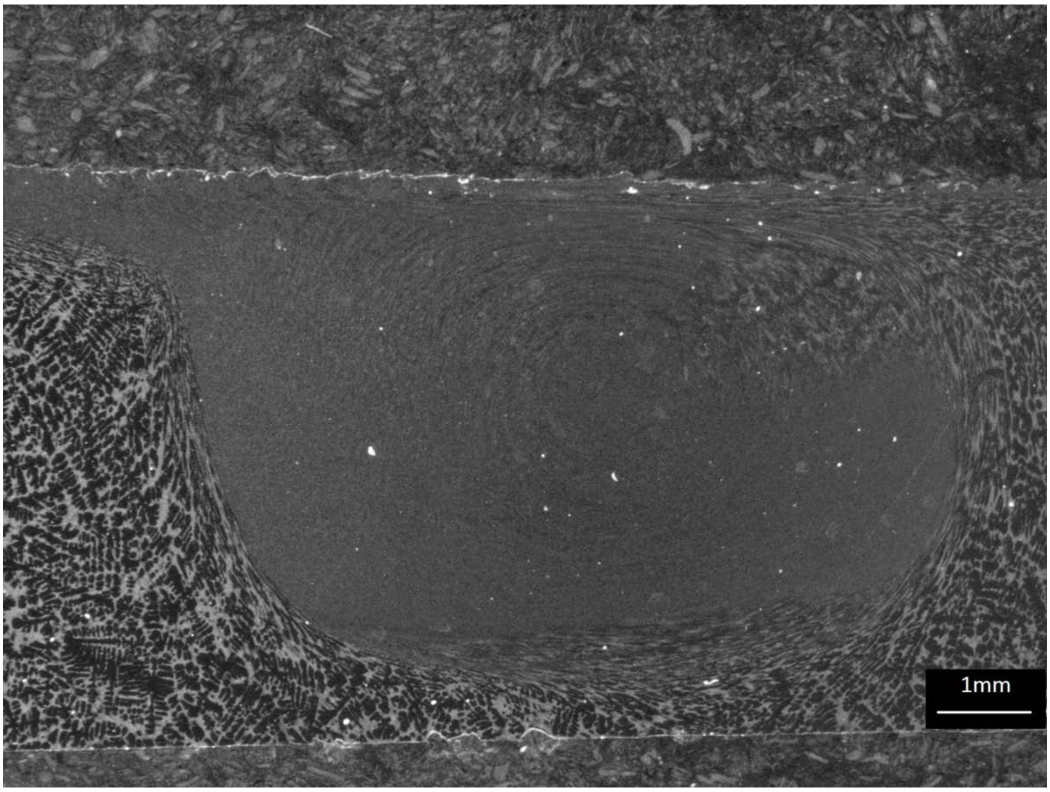


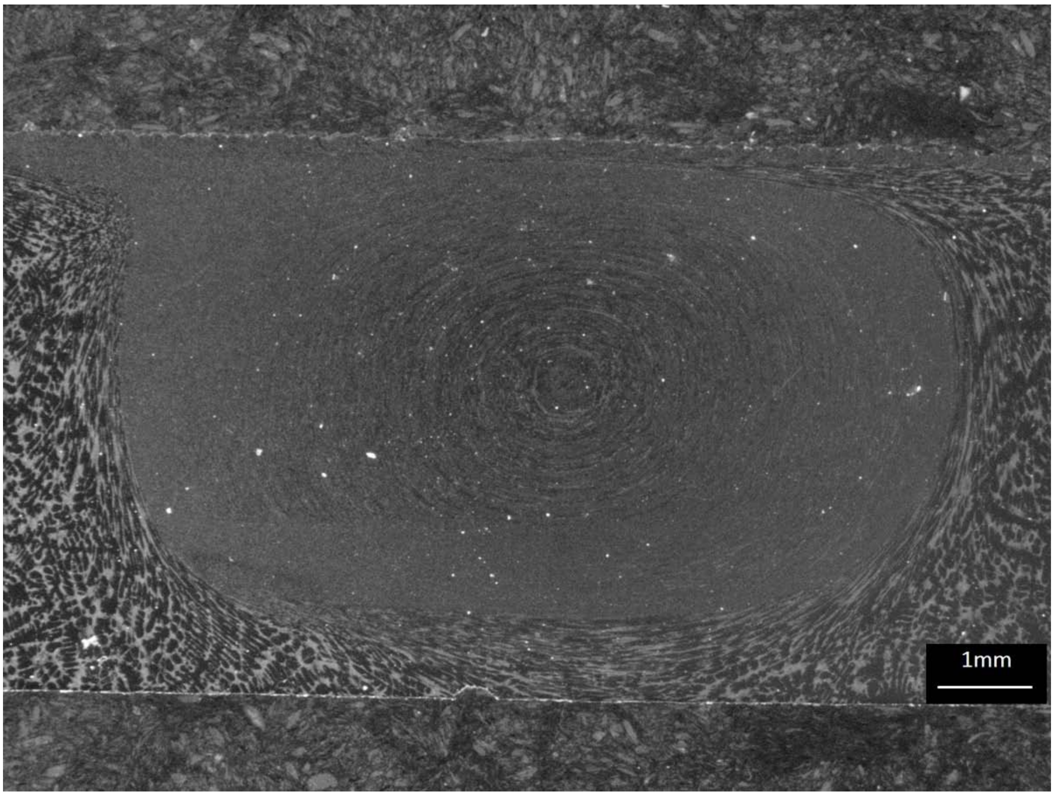
Appendix – C

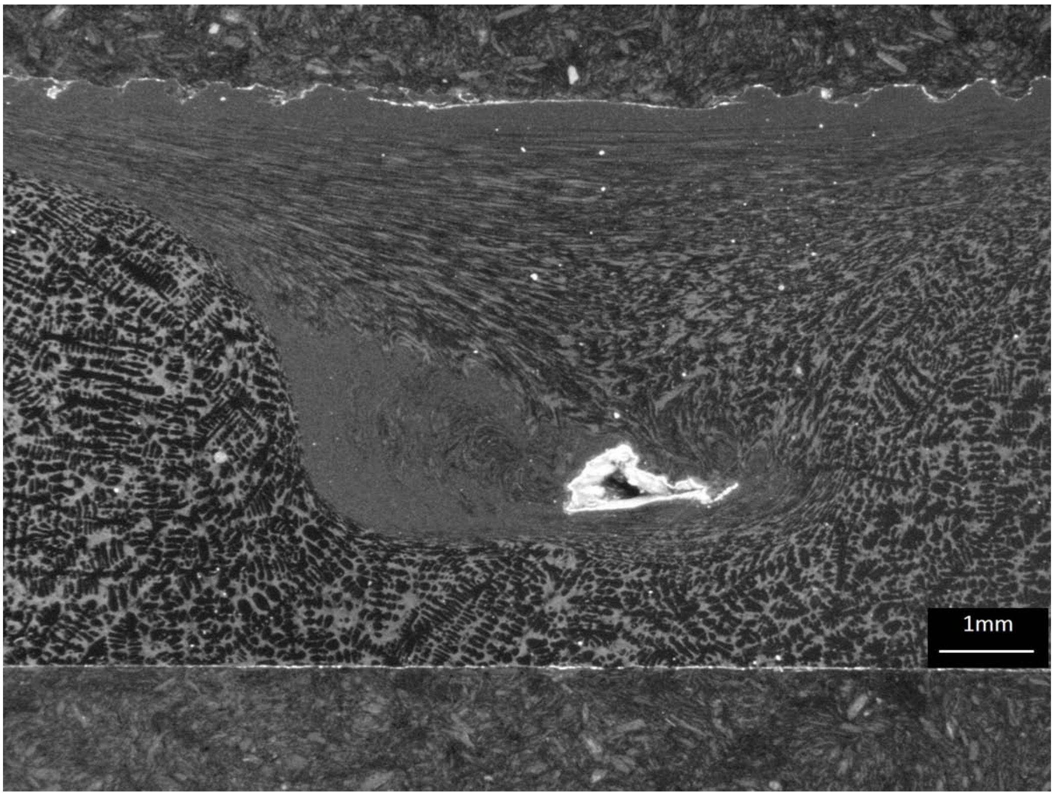
Weld quality microsturcture pictures.

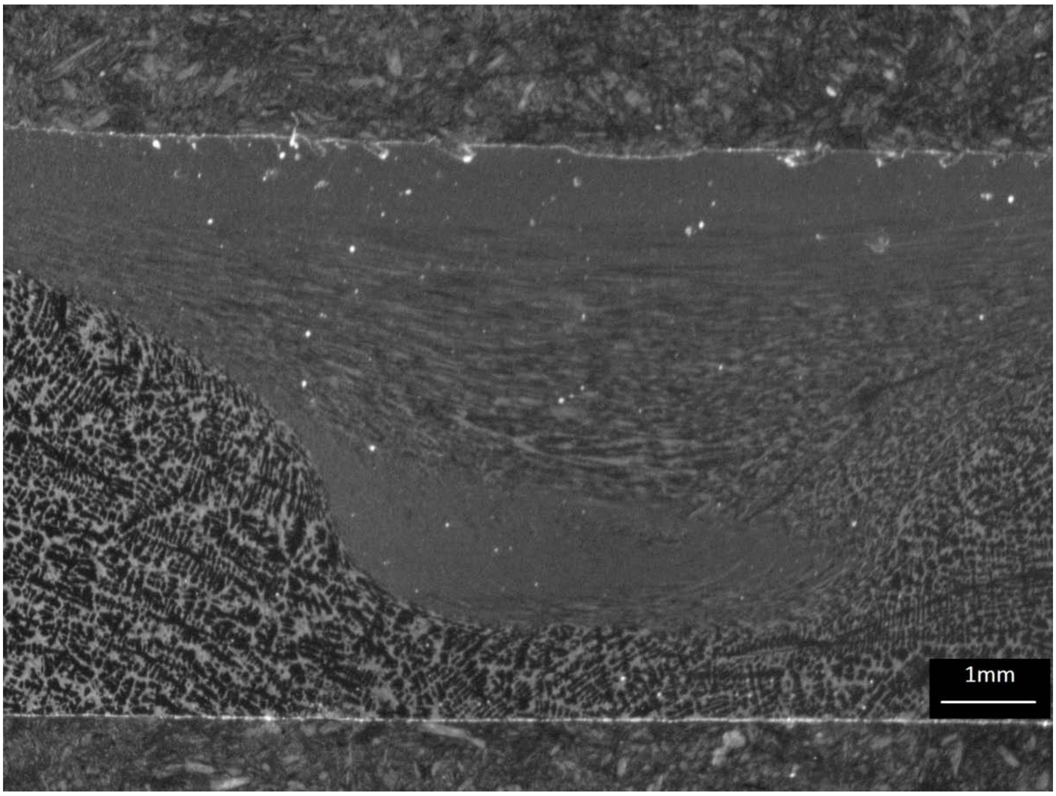


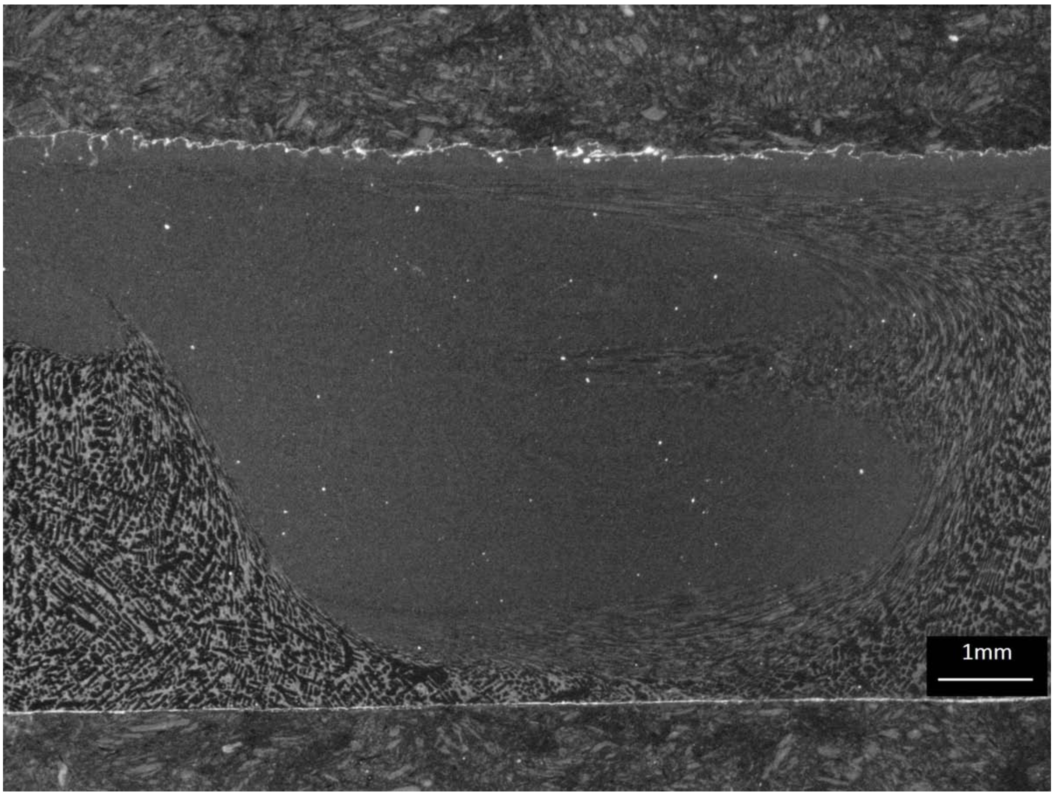


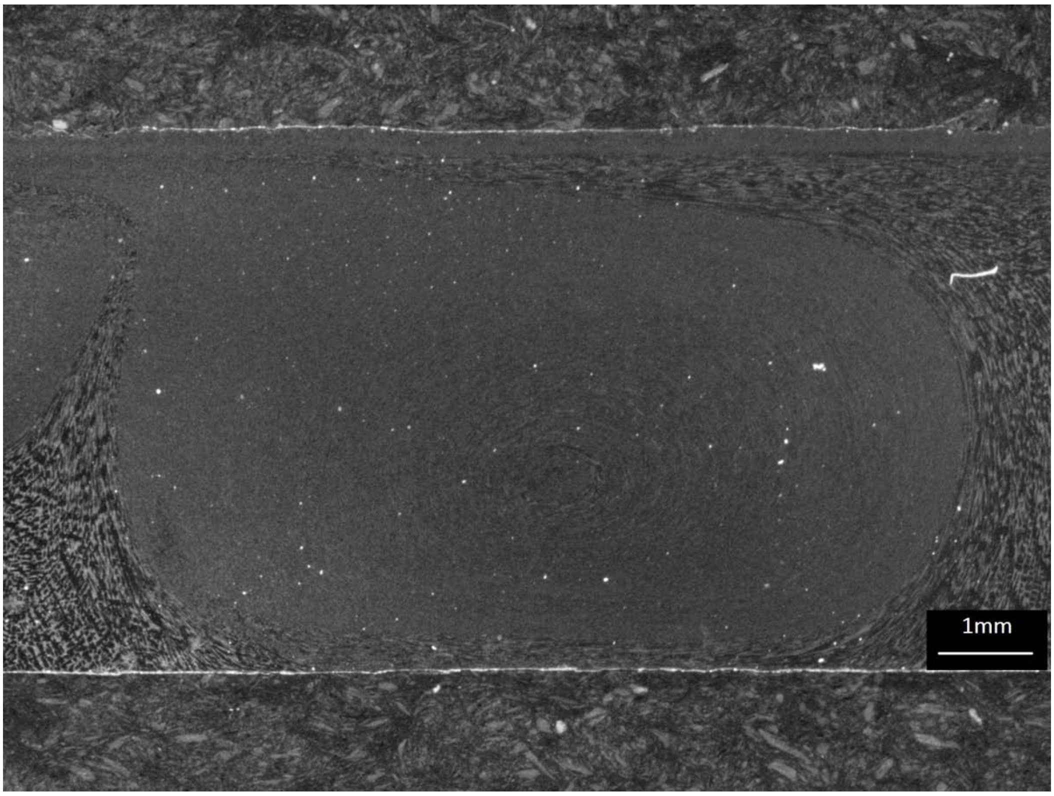

1mm

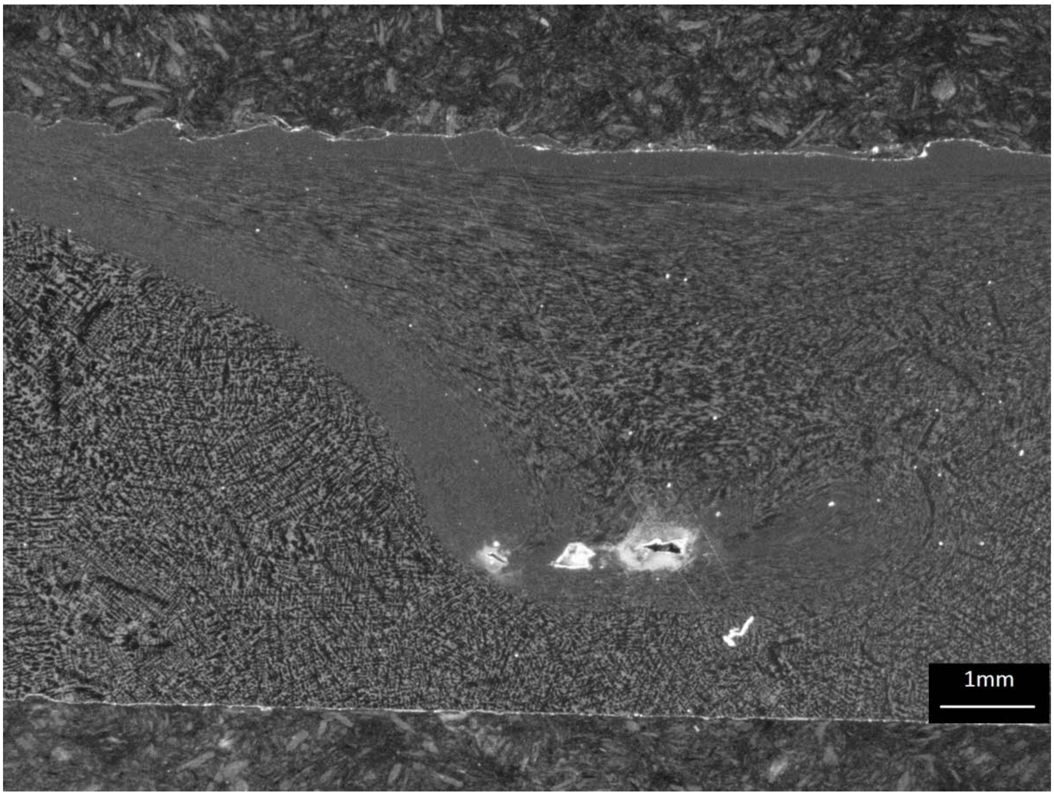

1mm

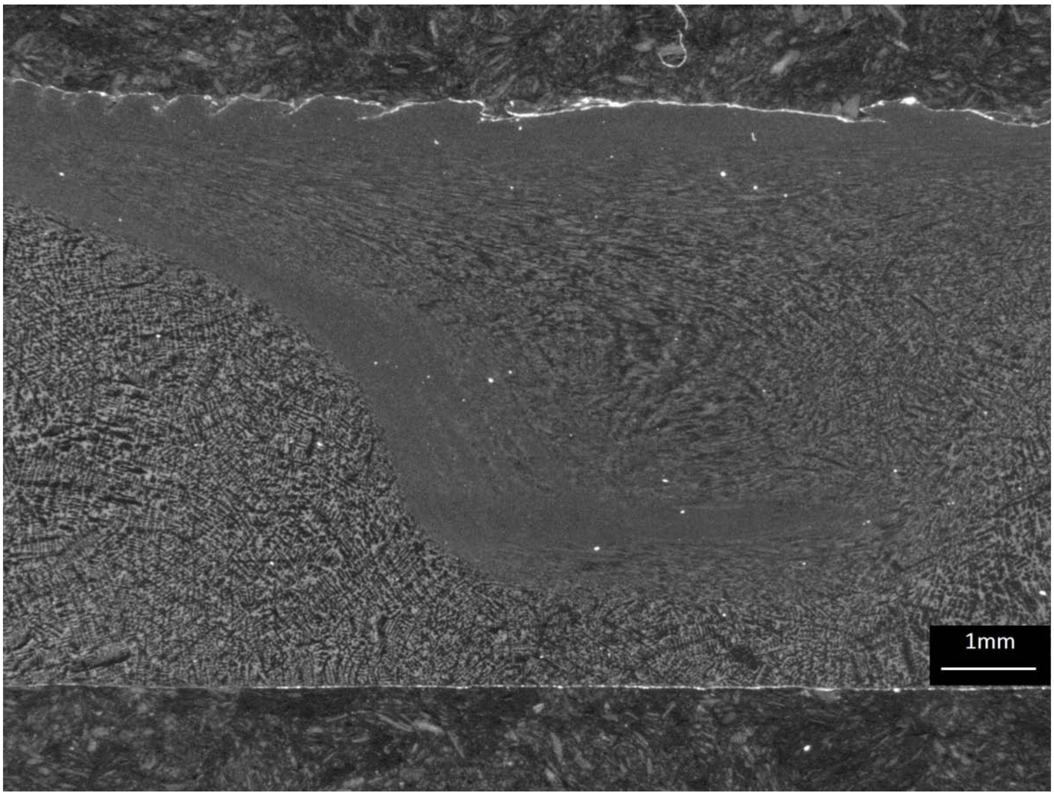

1mm


1mm


1mm


1mm


1mm


1mm