SUPPLEMENT TO THE *WELDING JOURNAL,* NOVEMBER 2006
Sponsored by the American Welding Society and the Welding Research Council

A Genetic Algorithm and Gradient-Descent-Based Neural Network with the Predictive Power of a Heat and Fluid Flow Model for Welding

Six neural networks were developed for gas tungsten arc welding of low-carbon steel, with each network providing one of the six output parameters of GTA welds

BY S. MISHRA AND T. DEBROY

ABSTRACT. In recent years, numerical heat and fluid flow models have provided significant insight into welding processes and welded materials that could not have been achieved otherwise. However, these calculations are complex and time consuming, and are unsuitable in situations where rapid calculations are desired. A practical solution to this problem is to develop a neural network that is trained with the data generated by a numerical heat and fluid flow model. Apart from providing high computational speed, the results of this neural network conform to the basic laws of conservation of mass, momentum, and energy.

In the present study, six feed-forward neural networks have been developed for the gas tungsten arc (GTA) welding of low-carbon steel. Each network provides one of the six output parameters of GTA welds, i.e., depth, width, and length of the weld pool, peak temperature, cooling time from 800° to 500°C, and maximum liquid velocity. The networks require values of 17 input parameters including the welding variables like current, voltage, welding speed, arc efficiency, arc radius, and power distribution factor, and material properties like thermal conductivity and specific heat. The weights of the neural networks were calculated using two optimization schemes, first using the gradient descent (GD) method with various sets of randomized initial weights, and then applying a hybrid optimization scheme where a genetic algorithm (GA) is used in combination with the GD method. The neural networks produced by the hybrid optimization approach gave better results than all the networks based on only the GD method. Unlike the GD method alone, the hybrid optimization scheme could find the significantly better weights, which is illustrated by the good agreement between all the outputs from the neural networks and the corresponding results from the heat and fluid flow model.

S. MISHRA is now with Department of Metallurgical Engineering and Materials Science, Indian Institute of Technology, Bombay, India. Previously he was with Department of Materials Science and Engineering, The Pennsylvania State University, University Park, Pa, as is T. Debroy.

Introduction

In recent decades, systematic correlations between welding variables and weld characteristics have been attempted by numerical modeling of heat and fluid flow (Refs. 1–19) and artificial neural networks (Refs. 20–42). Numerical models of heat and fluid flow have provided significant quantitative insights into welding processes and welded materials. These models have accurately predicted temperature and velocity fields, weld pool geometry, cooling rate, peak temperature, phase transformations (Ref. 20), grain structure (Refs. 6, 7), inclusion structure (Refs. 43, 44), and weld metal composition change owing to both the evaporation of alloying elements and the dissolution of gases (Ref. 8). Although these models are recognized as powerful tools for research, they are not extensively used in the welding industry because these are complex, require specialized training to develop and test, and consume a large amount of computer time to run.

KEYWORDS

Neural Networks
Gas Tungsten Arc Welding (GTAW)
Heat Transfer and Fluid Flow Model
Low-Carbon Steel

Neural network models are powerful nonlinear regression analysis methods (Refs. 20, 21, 45–47) that can relate input variables like welding process parameters and material properties with weld characteristics such as weld pool geometry. The previous efforts to model the GTAW process using a neural network were based on training the network with experimental data (Refs. 27, 32, 38). Since the volume of experimental data required to train a neural network depends on the number of input and output variables, most previous efforts considered only a few input parameters to keep the necessary volume of experimental data tractable (Refs. 27, 32, 38). For example, Tarng et al. (Ref. 27), Andersen et al. (Ref. 32), and Juang et al. (Ref. 38) developed neural network models of the GTA welding process, which considered the effects of process parameters like welding speed, arc current, and voltage as inputs. These neural network models were developed using a limited volume of experimental data, and they could not determine the effect of material properties like thermal conductivity, specific heat, etc., on weld pool geometry. Furthermore, the output variables considered in these neural networks were also limited. For example, the existing neural network models do not provide any information about some of the important parameters such as the cooling rate and peak temperature. A review of previous work indicates that what is needed is a framework for rapid calculation of weld pool

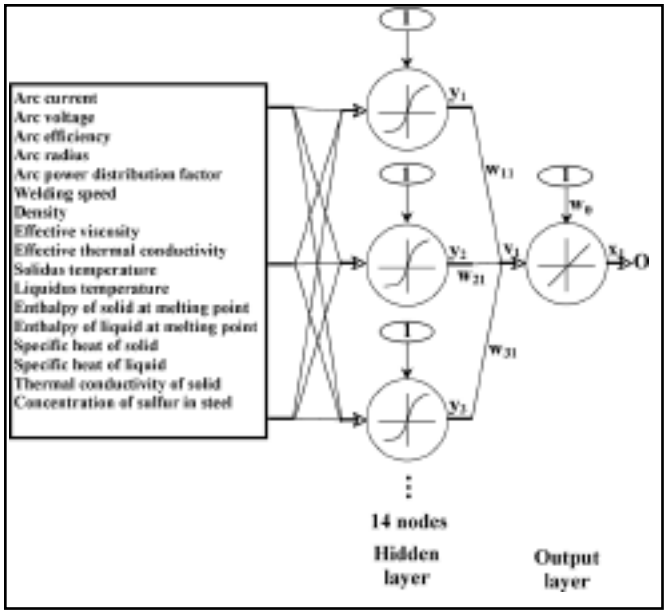

Fig. 1 — Neural network architecture used in this study. The input layer has 17 input variables and the output layer has one of the six output variables, i.e., weld pool depth, weld pool width, weld pool length, peak temperature, cooling time from 800° to 500°C, or maximum liquid velocity in the weld pool.

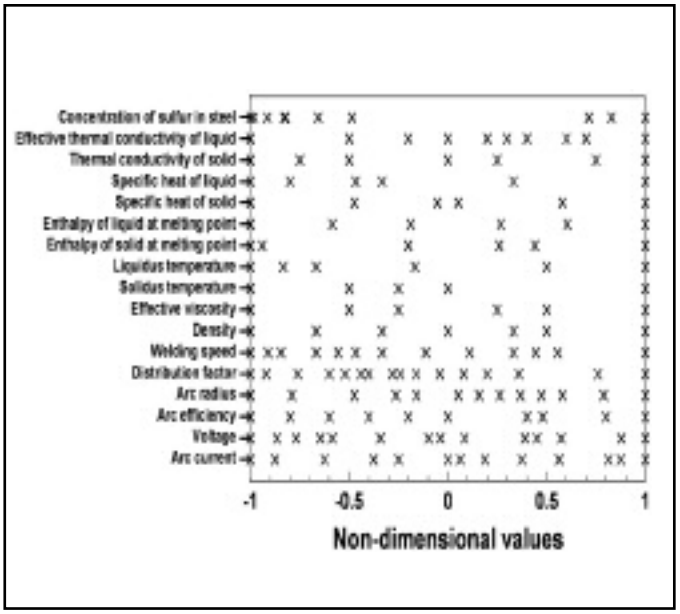

Fig. 2 — The ranges of values of the input variables in the training and testing databases. The normalized values of the variables were obtained using Equation 10 and corresponding minimum and maximum values listed in Table 1.

geometry, cooling rate, and peak temperature for the GTA welding of various materials.

In the present work, a neural network has been trained with the results of a well-tested numerical heat and fluid flow model (Refs. 2, 8, 12). The neural network correlates various output parameters such as weld pool geometry, cooling rate, peak temperature, and maximum liquid velocity with all the major welding variables and material properties. Of these variables, the geometry and cooling rates affect the weld properties. The peak temperature and the velocities are important in understanding the role of convective heat transfer and mixing in the weld pool. Since the training data are made up of results from a reliable numerical heat transfer and fluid flow model, the output of the trained neural network will comply with the basic phenomenological laws of welding physics. This paper seeks to document the problems, issues, and lessons learned in the development of a neural network model from the results of a heat transfer and fluid flow model. The neural network is validated by checking its performance for different sets of material properties and welding conditions, which were not a part of the training data.

The Mathematical Model

Neural Network Model

In the present study, six feed-forward neural networks have been developed for the gas tungsten arc (GTA) welding of low-carbon steel with no filler metal. Partial joint penetration welds with a flat top surface are assumed. Each neural network takes 17 input variables that include various welding variables such as arc current, voltage, welding speed, and material properties such as thermal conductivity, specific heat, and provides a single output, which can be one of the six output parameters, i.e., depth, width, and length of the weld pool, peak temperature, cooling time between 800°C and 500°C, and maximum liquid velocity in the weld pool. The cooling time was calculated on the workpiece surface along the welding direction. The input variables such as arc efficiency, arc power distribution factor, and arc radius determine how heat is absorbed at various locations in the workpiece (Ref. 10). Since temperature-independent thermophysical properties of the solid alloy are used in the model, a question arises as to how to select their values. Since the heat flow in the solid region near the weld pool affects both the size and shape of the weld pool as well as the temperature field in the entire workpiece, it is appropriate to use thermophysical properties at a temperature closer to the melting point than to the ambient temperature. Effective thermal conductivity and effective viscosity are used as input variables because they allow accurate modeling of the turbulence effect in the weld pool. These two variables are system properties, and their values are obtained by enhancing the molecular values of liquid thermal conductivity and viscosity, respectively. Appropriate values of these two variables for GTA welding of low-carbon steel have been determined in the literature (Ref. 5) through reverse modeling.

The structure of each neural network, along with all the input and output variables, is shown in Fig. 1. Each neural network contains an input layer, a hidden layer, and an output layer. The input layer contains all the 17 input variables, which are connected to nodes in the hidden layer, represented by circles in Fig. 1, through the weights assigned for each link. The number of nodes in the hidden layer is found by optimizing the network. Each connection to a node, j, from a node in the previous layer, i, has an adjustable weight, w_{ij}, associated with it. The weights, w_{ij}, embody the nonlinear relationship between the input and the output variables. Also, each node in the hidden and the output layers is given an extra input, which always has a value of 1. The weight of this extra input is called the bias. The net input, v_j, for a node, j, is given as

$$v_j = \sum_i w_{ij} y_i + w_\theta \qquad (1)$$

where i is a node in the previous layer, w_{ij} is the weight of the connection between nodes j and i, y_i is the output of node i, and w_θ is the bias weight. Equation 1 is calculated for all the nodes in the hidden layers as well as the output layer.

Now, the output of node j is calculated by using a transfer function. A hyperbolic tangent function (a symmetric sigmoid function), which is a nonlinear function producing output between –1 and 1, is used as transfer function for the nodes in

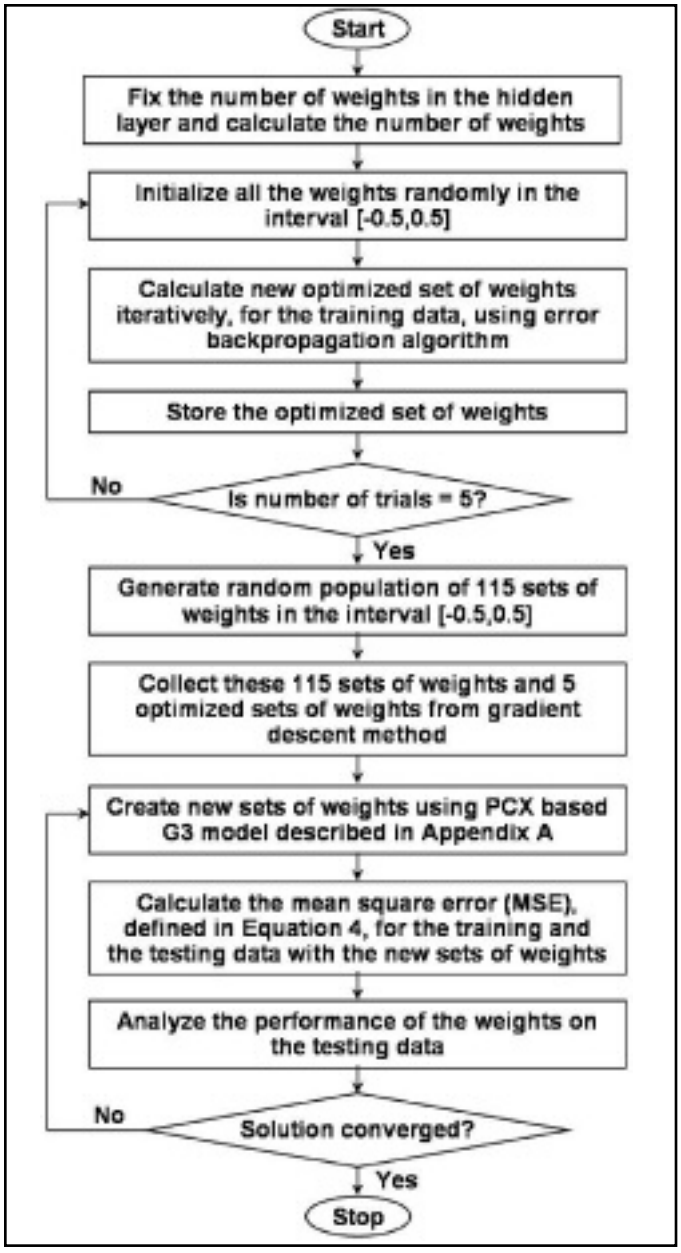

Fig. 3 — Flowchart for training the neural network.

the hidden layer, and a linear transfer function is used for the nodes in the output layer. The use of a nonlinear transfer function in the hidden layer allows the network to learn nonlinear and linear relationships between input and output vectors (Ref. 37), while the use of a linear transfer function in the output layer allows the network to produce values outside the range of –1 to 1. Thus, the output, x_j, of a node j in the hidden layer is given by

(2)

where 'a' is the slope of the sigmoid function. By varying the parameter 'a', sigmoid functions of different slopes can be obtained (Ref. 48). An increase in the value of 'a' increases the slope of the activation function and vice versa. A very high value of the slope makes the curve close to a step function while a low value retards the convergence rate. Based on the findings of previous works, a value of 1.5 was used to achieve rapid convergence (Refs. 49, 50). Furthermore, the use of the tanh function in Equation 2 as the activation function helps in keeping the problem reasonably well conditioned. An attractive feature of the hyperbolic tangent function is that its derivative does not increase computational volume significantly (Ref. 48). The output, x_j, of a node j in the output layer is given by the following:

$$x_j = v_j \qquad (3)$$

The training of a neural network implies finding a set of weights that minimize error between the desired output and the output calculated by the neural network. A back-propagation algorithm (Ref. 45) is used for the training of the neural networks. This algorithm tries to minimize the objective function, i.e., the mean square error (MSE) between the desired output and the neural network output. The MSE is defined as (Ref. 45) the following:

$$MSE = \frac{\sum_p \sum_k \left(d_{pk} - o_{pk}\right)^2}{p \times k} \qquad (4)$$

where p is the number of training datasets; k represents the number of output nodes, which is one in this work; d_{pk} is the desired output; and o_{pk} is the output produced by the neural network. The desired outputs of the neural network such as weld pool depth, width, and length, cooling rate, peak temperature, and maximum liquid velocity are dependent on input welding conditions, material properties, and the network parameters such as the weights.

The back-propagation algorithm adjusts the weights in the steepest descent direction (negative of the gradient) (Ref. 45). This is the direction in which the error in the value of the output variable, E, decreases most rapidly. For a given set of input-output training data, the partial derivatives of the error with respect to each weight, $\partial E/\partial w$, are calculated in two passes (Ref. 45). The forward pass calculates the output of each node in the hidden layers and the output layer, based on the inputs from the previous layers, as described by Equations 1–3. The backward pass propagates the derivatives from the output layer back to the input layer (Ref. 45). The backward pass is well documented in the literature (Refs. 45, 51) and is not described here. Once $\partial E/\partial w$ are calculated, the weights are changed by an amount proportional to $\partial E/\partial w$ as

$$\Delta w = -\varepsilon \frac{\partial E}{\partial w} \qquad (5)$$

where ε is called the learning rate. A large learning rate enables quick convergence, but it can also lead to overstepping of the solution and oscillation of the error (Ref. 24). On the other hand, small learning rate may prevent oscillation of the error, but it requires much more time to reach the solution (Ref. 24). Therefore, in the present study ε was taken as 0.1 during initial iterations, and reduced to 0.01 once the error became very small. A simple method for increasing the rate of learning without oscillation is to include a momentum term in Equation 5 as follows (Ref. 45):

$$\Delta w\left(n\right) = -\varepsilon \frac{\partial E}{\partial w} + \alpha \Delta w\left(n-1\right) \qquad (6)$$

where n is the number of iterations, an iteration being defined as a single sweep through all the input-output pairs in the training dataset, and α is an exponential decay factor between 0 and 1 that determines the relative contribution of the current gradient, $\partial E/\partial w$, and the earlier gradients, $\Delta w(n–1)$, to the weight change. The value of α is set to 0.9 in the present study, based on guidance from previous research (Refs. 37, 51).

The training of the neural network was started with random small weights. It was observed that after the initial rapid decrease in error, further descent became

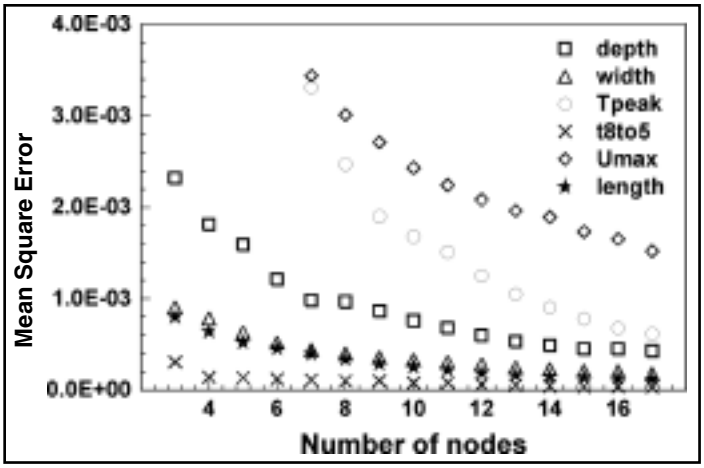

Fig. 4 — Comparison of mean square error (MSE), defined by Equation 4, for different number of nodes in the hidden layer. For each output variable the results are an average of five runs with different sets of initial random weights using the gradient-descent training method.

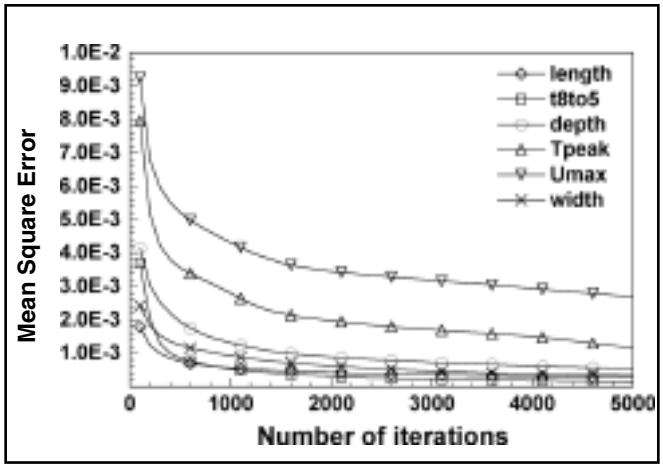

Fig. 5 — Mean square error (MSE), defined by Equation 4, vs. number of iterations for the gradient-descent training method. For each output variable, the results are an average of five runs with different sets of initial random weights.

very sluggish, and the result depended on the initial random weights, which are common problems with gradient-descent algorithms (Refs. 46, 52, 53). Furthermore, for simple two-layer networks (without a hidden layer), the error surface is bowl shaped and using gradient-descent techniques to minimize objective function is not a problem. However, the addition of a hidden layer used to solve more difficult problems like the GTA welding process increases the possibility for complex error surfaces that contain many minima. The gradient-based methods, which are used in the back-propagation algorithm described above, can easily get trapped in such local minima. Stochastic optimization techniques are capable of finding the global minima and avoiding local minima (Refs. 54–56). Therefore, a genetic algorithm (Refs. 54–56) is used along with the gradient-descent method to find the global set of weights in the present work.

The genetic algorithm (GA) used in the present study is a parent centric recombination (PCX) operator-based generalized generation gap (G3) model (Refs. 54–56). This model was chosen because it has been shown to have a faster convergence rate on standard test functions as compared to other evolutionary algorithms and classical optimization algorithms (Ref. 55). Detailed description of this model is available in the literature (Refs. 54–59) and is not repeated here. To start with, many initial sets of randomly chosen values of weights were created. Five of these initial sets of weights were made equal to five different sets of weights calculated by the gradient-descent algorithm. A systematic global search was next undertaken to find the most optimum set of weights that leads to the least mean square error (MSE), given in Equation 4.

Table 1 — The Input Variables and Their Ranges of Values Considered in the Training Dataset

Variables	Minimum value	Maximum value
Arc current, I (A)	9.00E+01	2.50E+02
Voltage, V (V)	9.60E+00	2.60E+01
Arc efficiency, η	3.50E-01	8.50E-01
Arc radius, r (m)	1.00E-03	2.90E-03
Arc power distribution factor, d	5.00E-01	3.00E+00
Welding speed, U (m/s)	1.00E-03	1.00E-02
Density of liquid, ρ (kg/m^3)	6.60E+03	7.80E+03
Effective viscosity of liquid, μ (kg/m-s)	6.00E-02	1.00E-01
Liquidus temperature, T_l (K)	1.730E+03	1.770E+03
Solidus temperature, T_s (K)	1.785E+03	1.845E+03
Enthalpy of solid at melting point, H_s (J/kg)	1.05E+06	1.15E+06
Enthalpy of liquid at melting point, H_l (J/kg)	1.32E+06	1.42E+06
Specific heat of solid, C_{ps} (J/kg-K)	6.27E+02	7.86E+02
Specific heat of liquid, C_{pl}, (J/kg-K)	7.74E+02	8.99E+02
Thermal conductivity of solid, k_s (J/m-s-K)	2.30E+01	3.97E+01
Effective thermal conductivity of liquid, k_l (J/m-s-K)	8.36E+01	5.02E+02
Concentration of surface active species, C_s (wt-%)	0.00E+00	3.50E-01

The mean square error depends on the values of weights:

$$MSE(w) = MSE(w_1, w_2, \ldots, w_q) \qquad (7)$$

where q is the number of weights in the network. The GA produced new individuals, or sets of weights, with iterations based on evolutionary principles (Refs. 20, 55, 56). The specific application of G3-PCX model for obtaining the optimum set of weights is described in Appendix A.

Numerical Heat and Fluid Flow Model for Gas Tungsten Arc (GTA) Welding

The data for the training and testing of the neural network were generated from an extensively tested three-dimensional (3D) numerical heat and fluid flow model for GTA welding (Refs. 1–18). In this model, the transient nature of the problem is transformed to steady-state mode by using a coordinate system moving with the heat source (Refs. 12, 17). The governing equations of conservation of mass, momentum, and energy in three dimensions (3D) are discretized using the power law scheme (Ref. 60). The computational domain is divided into small rectangular control volumes. Discretized equations for the variables are formulated by integrating the corresponding governing equations over the control volumes. The detailed method of discretizing the governing equations is available in the literature (Refs. 12, 17). The discretized equations are solved using the SIMPLE algorithm (Ref. 60) to obtain temperature and velocity fields. The calculated temperature and velocity fields provide the output variables, i.e., weld pool geometry, peak temperature, cooling time between 800° and 500°C, and maximum liquid velocity in the weld pool.

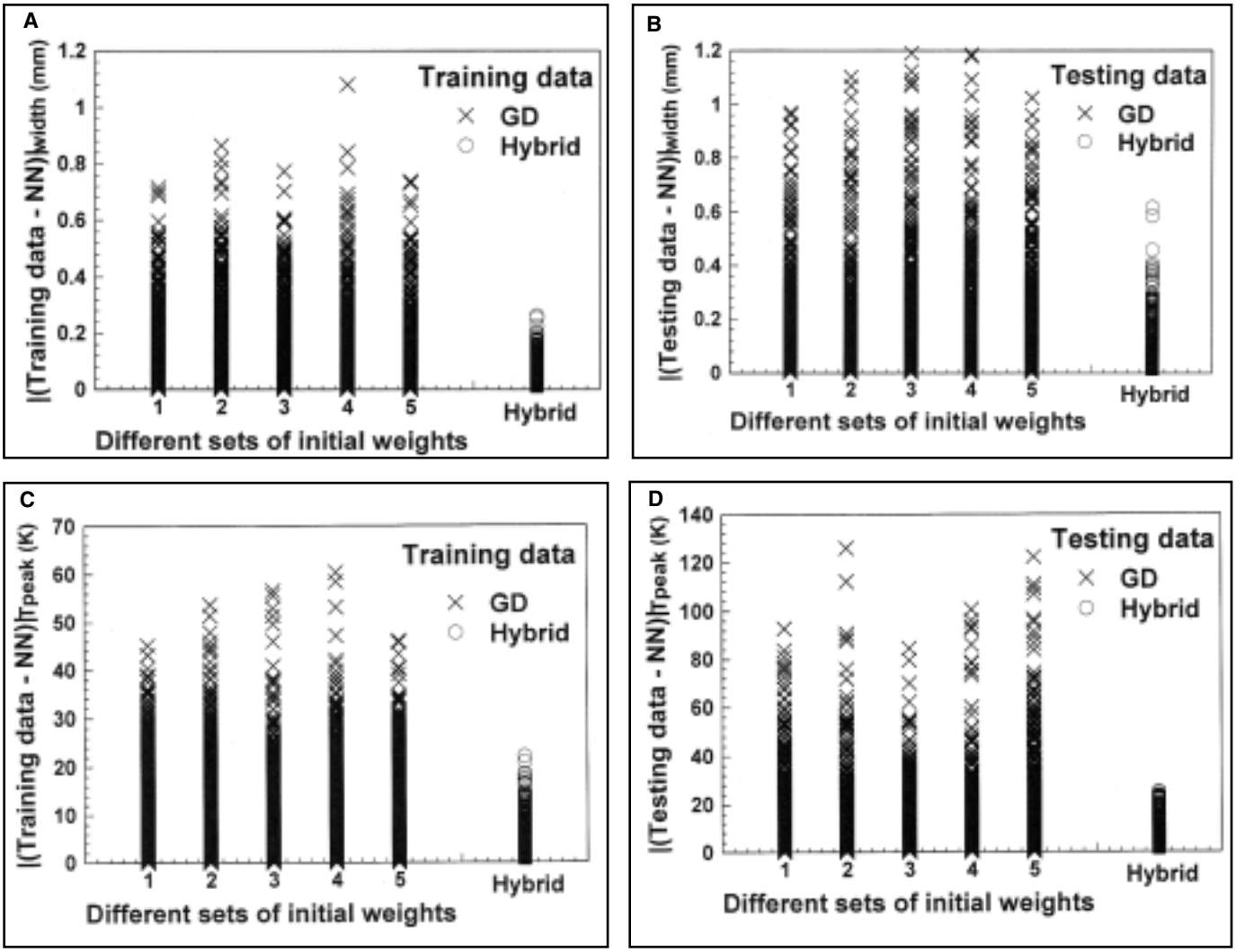

Fig. 6 — Absolute value of the difference between the following: A — The weld pool width in the training data and that calculated from neural network (NN); B — the weld pool width in the testing data and that calculated from neural network; C — the peak temperature in the training data and that calculated from the neural network; and D — the peak temperature in the testing data and that calculated from the neural network. The results are for the five sets of weights calculated by the gradient-descent (GD) method, and the global set of weights calculated by the hybrid approach.

Calculation Procedure

Neural networks require a large database for training and testing. The number of training datasets should be more than the number of weights connecting different nodes. For a single hidden layer network, the number of weights, q, is given as:

$$q = (n_i + 1) \times n_h + (n_h + 1) \times n_o \quad (8)$$

where n_i is the number of input variables, i.e., 17 in the present work, n_h is the number of nodes in the hidden layer, and n_o is the number of output variables, i.e., 1. For a double hidden layer network,

$$q = (n_i + 1) \times n_h + (n_{h1} + 1) \times n_{h2} + (n_{h2}+1) \times n_o \quad (9)$$

where n_{h1} and n_{h2} are the number of nodes in hidden layers 1 and 2, respectively. Since the number of weights increases with the increase in the number of hidden layers, an optimal number of hidden layers are needed.

Number of Hidden Layers in the Network

The number of hidden layers in a neural network depends on the type of problem and the relationships between the input and the output variables represented through the objective function. Theoretically, any continuous variation of output with respect to input can be represented by a single hidden layer (Refs. 61, 62). Two hidden layers are needed when the relationship between the input and the output variables is discontinuous (Refs. 61, 62). The use of more than the optimal number of hidden layers in the network may result in undesirable overfitting of the data (Refs. 48, 61, 62). A single hidden layer was used since the outputs are continuous in nature in GTAW.

Database Generation

A database for training of the neural networks was generated to capture the effects of all the welding parameters and material properties. A well-tested numerical heat transfer and fluid flow model for GTA welding was used to generate the database where the values of various thermophysical properties were assumed to be temperature independent. Out of the 17 input variables in GTA welding, the ten most important variables that affect the output significantly include current, voltage, welding speed, arc radius, arc power distribution factor, arc efficiency, effective

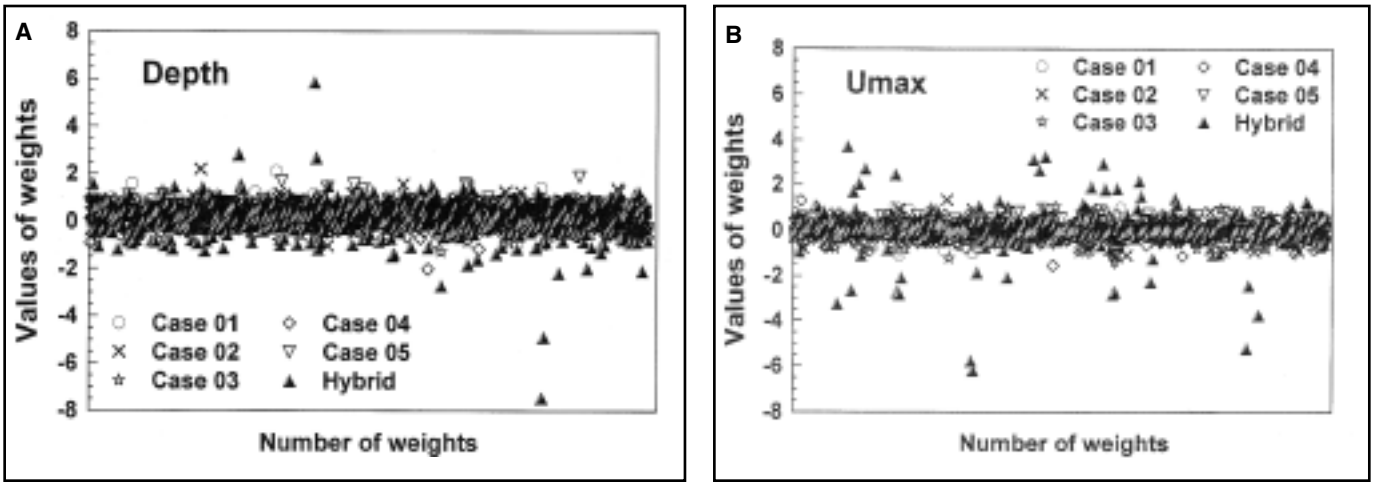

Fig. 7 — Comparison of the values of weights for the gradient-descent and hybrid training approaches for the following: A — Weld pool depth; B — maximum liquid velocity in the weld pool. Cases 01 to 05 are for the five sets of weights obtained from the gradient-descent method when using five different sets of initial random weights.

Table 2 — Comparison of the Mean Square Error (MSE), Defined in Equation 4, for the Gradient-Descent Method and the Hybrid Training Approach. The Results for the Gradient-Descent Method Were Taken after 5000 Iterations

Output variable	MSE Gradient Descent	MSE Hybrid Approach
Weld pool depth (mm)	5.37×10^{-4}	1.50×10^{-4}
Weld pool width (mm)	3.39×10^{-4}	6.77×10^{-5}
Weld pool length (mm)	2.80×10^{-4}	4.00×10^{-5}
Peak temperature (K)	1.17×10^{-3}	1.87×10^{-4}
Cooling time 800° to 500°C (s)	1.50×10^{-4}	3.89×10^{-6}
Maximum velocity (mm/s)	2.69×10^{-3}	9.99×10^{-4}

thermal conductivity of liquid, effective viscosity of liquid, thermal conductivity of solid, and the concentration of sulfur. The complex interactions between these ten variables were captured by making 1250 different runs of the three-dimensional numerical heat and fluid flow model. The effect of the remaining variables such as density, specific heat of the solid, specific heat of the liquid, etc., was captured by making 500 different runs of the numerical heat and fluid flow model. The relative importance of the variables was decided based on their sensitivity on the weld geometry. As described above, more runs of the numerical heat transfer and fluid flow model were made for the variables that have a major influence on the weld geometry. A sufficient number of different values of the variables were considered in order to increase the degrees of freedom and to adequately capture the effect of the variables that have a large influence on the weld geometry and cooling rate. Out of the 1750 total runs conducted, 1250 datasets were chosen randomly and included in the training dataset, and the remaining 500 datasets formed the testing dataset for the validation of the neural network. Thus, the combinations of the values of variables in the testing datasets were completely different from those in the training datasets. The 17 input variables and their ranges of values used for the generation of datasets are shown in Table 1. The ranges of values of the input variables correspond to the GTA welding of low carbon steel (Refs. 12, 54). Furthermore, Fig. 2 shows that the different values considered for each variable are well distributed about the mean value for the variable, which ensures that the effect of almost the entire range of values for each variable can be taken into account in the databases.

Normalizing Inputs and Outputs

There is significant variation in the scales of values of the input and output variables. The vastly different scales of inputs and bias values lead to ill conditioning of the problem (Refs. 48, 49). While large inputs cause ill conditioning by leading to very small weights, large outputs do so by leading to very large weights (Refs. 48, 49). To eliminate the ill-conditioning problem, the data were normalized using the following formula (Ref. 49):

$$x' = 2 \times \left(\frac{x - x_{min}}{x_{max} - x_{min}} \right) - 1 \qquad (10)$$

where x is the original value of the variable and x´ is the normalized value, while x_{min} and x_{max} represent the minimum and maximum values of the variable in the entire dataset. Equation 10 normalizes the data in the range of –1 to 1. The range of values of all input and output parameters from –1 to 1 implies that the standard deviation cannot exceed 1, while its symmetry about zero means that the mean will typically be relatively small (Ref. 49).

Selection of Initial Weights

In the back-propagation algorithm, the magnitude of the error propagated backward through the network is proportional to the value of the weights. If all the weights are the same, the back-propagated errors will be the same, and consequently all of the weights will be updated by the same amount (Refs. 48, 49). To avoid this symmetry problem, the initial weights of the network were selected randomly. Furthermore, to avoid the premature saturation of the network, the initial values of the weights were distributed inside a small range of values, i.e., in the interval [–0.5, 0.5]. When the weights are small, the units operate in the linear regions of the transfer function and consequently the transfer function does not saturate.

The calculation starts with the selection of the number of nodes in the hidden layer. The total number of weights in the network depends upon the number of nodes in the hidden layer. The weights are then initialized randomly in the interval [–0.5, 0.5]. In the next step, a back-propagation algorithm is used to minimize the error on the training dataset. The op-

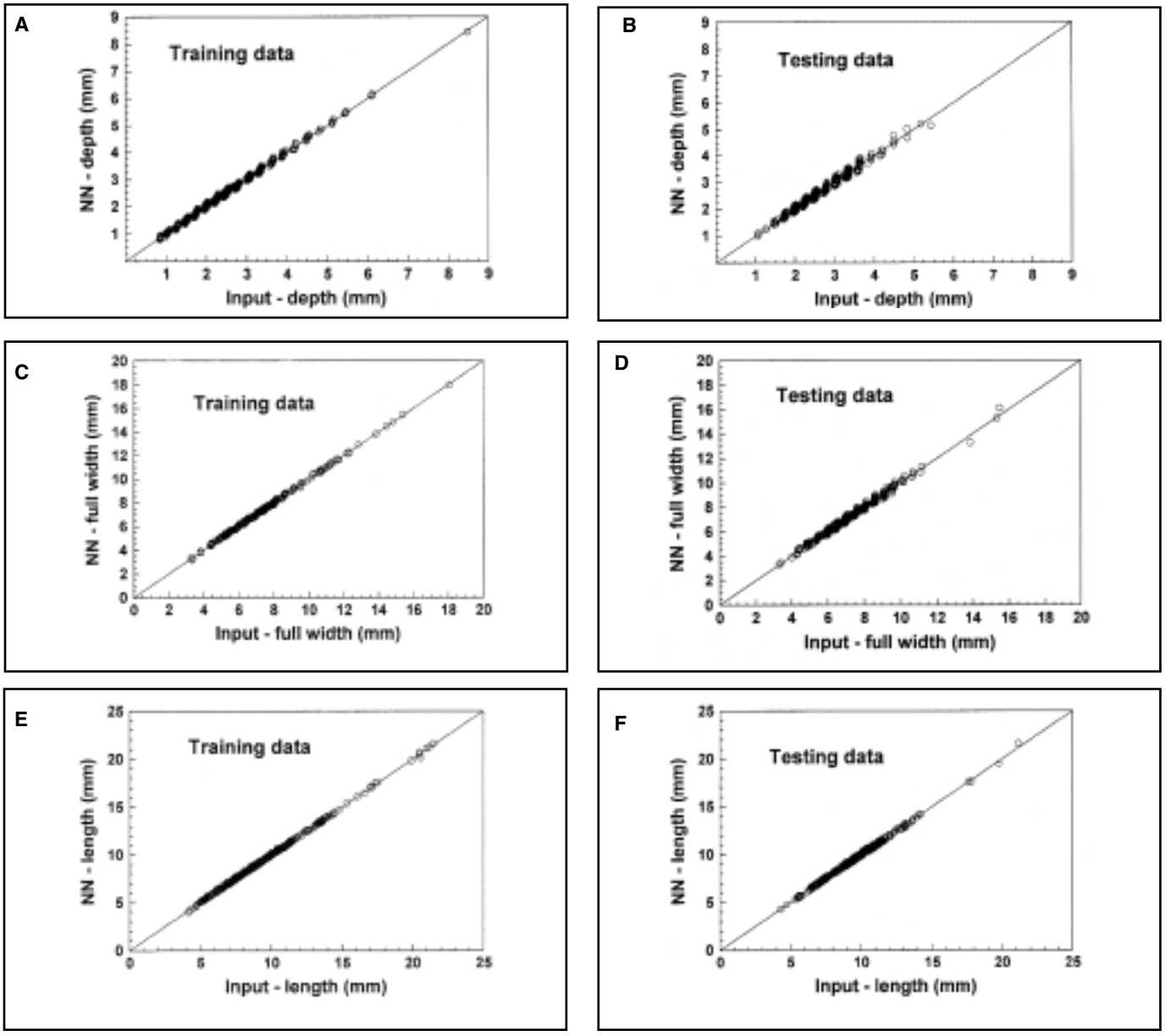

Fig. 8 — Comparison of weld pool dimensions calculated by the respective neural networks and those given in the training and the testing datasets. Top — Depth; Middle — width; Bottom — length.

Table 3 — The Mean Square Error (MSE), Defined in Equation 4, and Mean Error (ME), Defined in Equation 11, for the Results of the Neural Network for the Weld Pool Geometry, Peak Temperature, Cooling Time between 800° and 500°C, and Maximum Liquid Velocity in the Weld Pool. The Results Were Obtained Using the Hybrid Training Approach

Output variable	MSE Training Data	MSI Testing Data	ME Training Data	ME Testing Data	Typical value in the data
Weld pool depth (mm)	1.50×10^{-4}	5.00×10^{-4}	0.04	0.07	2.3
Weld pool width (mm)	6.77×10^{-5}	3.24×10^{-4}	0.05	0.09	6.8
Weld pool length (mm)	4.00×10^{-5}	1.00×10^{-4}	0.05	0.09	8.7
Peak temperature (K)	1.87×10^{-4}	6.90×10^{-4}	4	8	2237
Cooling time 800° to 500°C (s)	3.89×10^{-6}	1.57×10^{-5}	0.01	0.01	2.0
Maximum velocity (mm/s)	9.99×10^{-4}	3.72×10^{-3}	5	10	190

timized weights calculated by the gradient descent method are stored as one possible set of weights. This process is repeated five times with different randomly selected initial weights for fixed values of nodes in the hidden layer. All of these five optimized sets of weight are provided as input to the GA. The final aim of the GA is to find the weights in the network through a systematic global search that will give the least error between the neural network prediction and numerical heat and fluid flow calculations. The flowchart of the calculation scheme is presented in Fig. 3. The convergence is based on the error in training and testing data. When the error during testing starts increasing, the calculation is stopped to avoid overfitting even if the error with

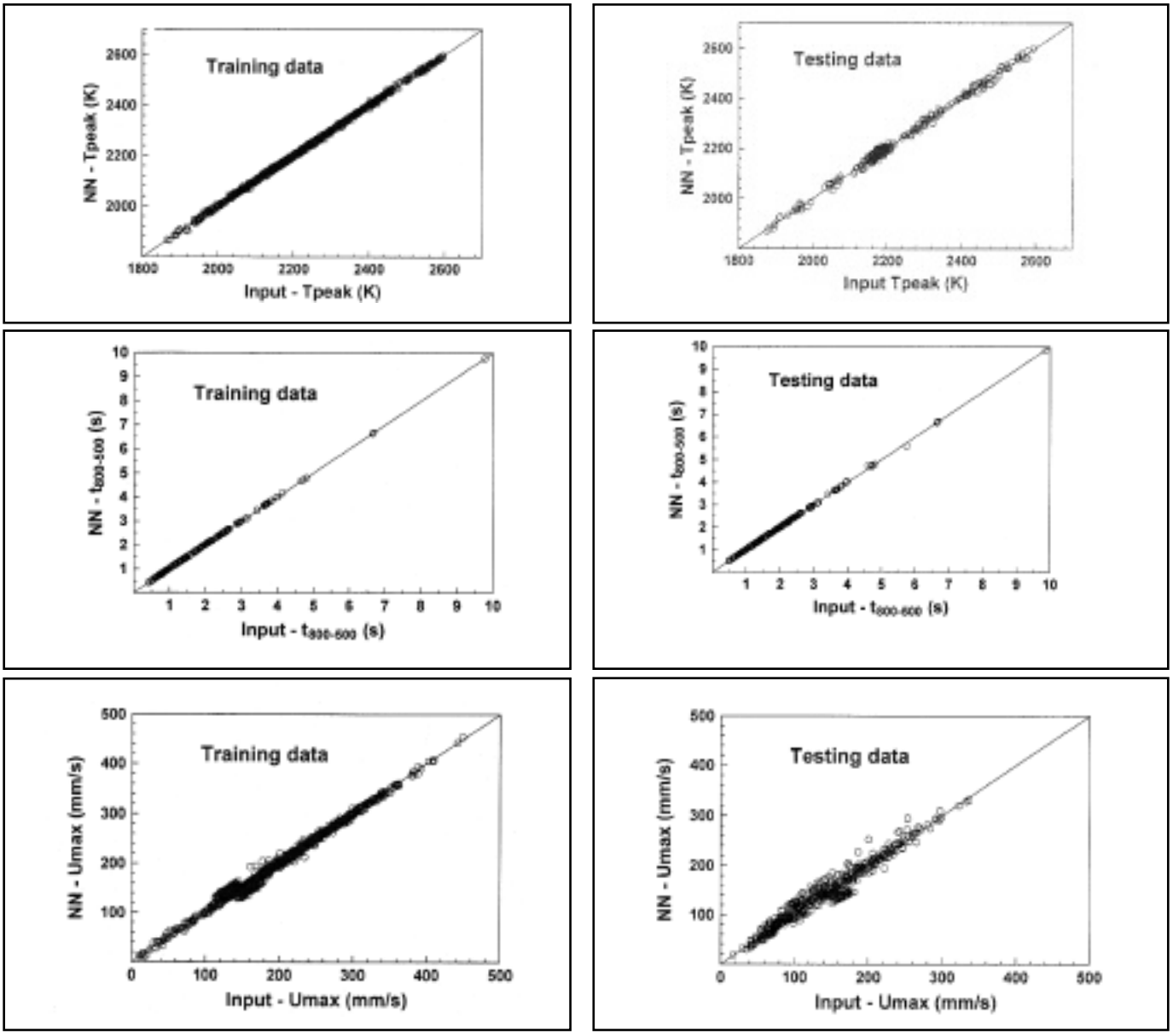

Fig. 9 — Comparison of the following calculated by the respective neural networks and those given in the training and the testing datasets: Top — Peak temperature; Middle — cooling time between 800° and 500°C; Bottom — maximum liquid velocity in the weld pool.

training dataset decreases with further iterations.

Results and Discussion

Structure Chosen for the Neural Networks

The selection of suitable network architecture is important as it affects the network's convergence as well as the accuracy of predictions (Ref. 22). The number of nodes in the hidden layer were varied to get an optimum number of nodes that resulted in minimum mean square error (MSE) defined in Equation 4. The average MSE of five runs made with different initial random weights is plotted in Fig. 4 for different number of nodes in the hidden layer. The results are for all the six neural networks with different output variables, and the runs were conducted using the gradient-descent method. Figure 4 shows that for all the six neural networks, the average MSE decreases with increase in the number of hidden nodes. For the neural networks with depth, width, length, and cooling time from 800° to 500°C, the MSE becomes almost constant for more than 14 hidden nodes in the network. Even for the neural networks with peak temperature and maximum velocity as the output variables, the decrease in MSE on increasing the number of hidden nodes above 14 is rather insignificant. Therefore, networks with 14 nodes in the hidden layer were used in the present study.

Gradient Descent vs. Hybrid Training Approach

As described in the section on mathematical modeling, the neural network model developed in the present study uses a combined gradient descent and genetic algorithm (hybrid) training approach. The initial guidance is provided by a gradient-descent algorithm and then the GA finds the global minimum of the mean square error (MSE) to provide a well-trained network. The neural networks were first trained on the training dataset of 1250 input-output pairs using the gradient-descent method. Since the output of this method depends on the initial set of guessed weights, five sets of initial random

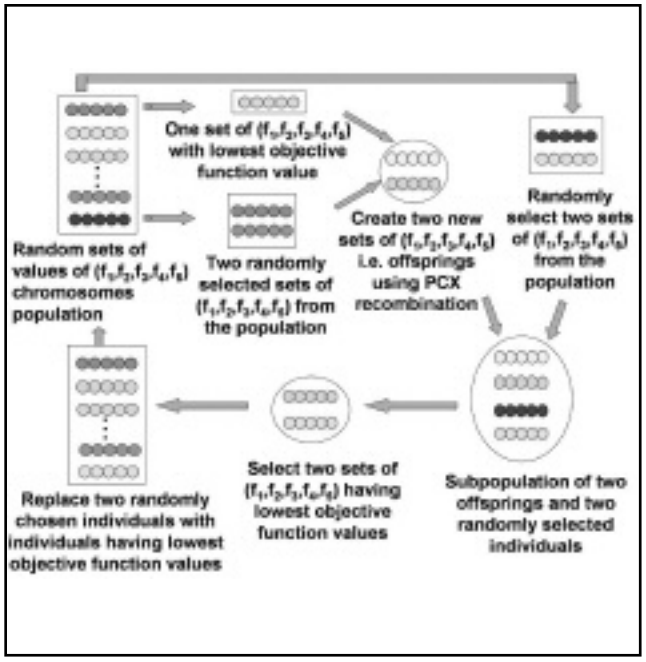

Fig. 10 — Generalized Generation Gap (G3) model using PCX operator.

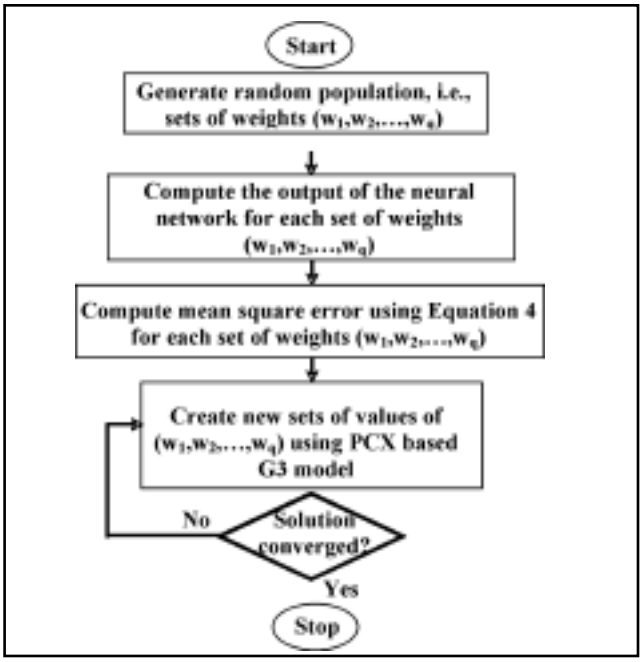

Fig. 11 — Flow chart of the Generalized Generation Gap (G3) model.

weights were used. In order to assess the performance of the gradient-descent method, the decrease of MSE with iterations is plotted for all the six neural networks in Fig. 5. Each plot represents an average of five runs with different random initial weights in order to avoid any local optimal solution. It can be seen that in all the cases, the error decreases very rapidly for the first 2000 iterations, but after that, further decrease is very slow. Since further improvement in error was very slow and insignificant, the training of the network using the gradient-descent method was stopped after 5000 iterations. The MSE values for the six neural networks, after training by the gradient-descent method for 5000 iterations, have been listed in Table 2.

Now, to test the performance of the hybrid approach, the five sets of weights calculated after 5000 iterations of the gradient-descent method, were given as inputs to the genetic algorithm (GA). Adding these sets of weights to the initial population of GA ensures the presence of five sets of near-optimum weights within the starting population, and helps in finding the global solutions more efficiently. Two convergence criteria were used. The first criterion required the training to stop if the mean square error became less than 1.0×10^{-6}, which is a low enough acceptable error for the problem at hand. The second criterion was based on the concern of overfitting of the neural network (Refs. 37, 39, 46–48). During training of the neural network, the error on the training data is driven to a very small value, but when new data are presented to the network, the error is large. This means that the network has memorized the training examples, but it has not learned to generalize to new situations (Ref. 37). A common method of avoiding this problem is that while training the network, its performance is simultaneously tested on a set of testing data. The training of the network is stopped as soon as the mean square error in the testing data starts increasing. This was set as the second convergence criterion in the present study in order to avoid overfitting of the neural network. The testing dataset consists of 500 input-output pairs, and the ranges of values of the input and output parameters lie within their ranges of values given in Table 1. The GA found the global set of weights giving much improved mean square errors as listed under the hybrid approach in Table 2. It can be seen that the MSEs for all the neural networks using the hybrid approach are much better than those obtained using the gradient-descent method alone.

To further compare the performance of the gradient-descent and hybrid approaches, the five sets of weights provided by the gradient-descent approach and the global set of weights provided by the hybrid approach were used to calculate the respective output variable values from the neural network for the training and the testing datasets. Figure 6A shows the absolute value of the difference between the training data and the corresponding results from the neural network, for weld pool width as the output variable, while Fig. 6B shows a similar plot for testing data. Figure 6C, D shows similar plots for peak temperature as the output variable. These plots contain results for the five sets of weights obtained using the gradient-descent method and the global set of weights obtained using the hybrid approach. It can be seen that in all the cases the weights from the hybrid approach provide results that fit much better to both the training and the testing data. The results obtained by using the weights from the gradient-descent method have much more scatter as compared to those obtained from the hybrid approach. Similar results were obtained for other output variables as well. Thus, the hybrid approach provides a well-trained network with significantly better weights, where the output from the trained neural network can accurately map the inputs to the outputs.

The values of the five sets of weights from the gradient-descent method and those from the hybrid approach have been plotted in Fig. 7A and B for output variables of weld pool depth and maximum liquid velocity, respectively. It can be seen that the weights from the gradient-descent method are mostly confined to a narrow range of [–1.0, 1.0], which is very close to their initial range of [–0.5, 0.5], while the weights from the hybrid approach are scattered in a much wider area. For example, the global set of weights for weld pool depth lie in the range of [–8.0, 6.0] and those for maximum liquid velocity lie in the range of [–7.0, 4.0]. This means that in the hybrid approach, the genetic al-

gorithm explored a much wider search area, irrespective of the initial guessed values, and was able to find significantly better weights. Thus, the hybrid approach conducts a more thorough search of the possible search space and provides a better trained network.

Evaluating the Predicting Capability of the Neural Networks

Six neural networks have been developed, each providing a specific output, i.e., depth, width, and length of the weld pool, peak temperature, cooling time between 800° and 500°C or maximum liquid velocity in the weld pool. The performance of the neural networks is illustrated in Figs. 8 and 9, which compare the output parameters calculated by the model with their corresponding values provided in the training and the testing datasets. Figures 8A and B compare the weld pool depth calculated by the neural network with that provided in the training and testing datasets, respectively. All points lie on or very close to the diagonal line and the results obtained from the neural network agree well with the values calculated using heat and fluid flow model. The MSE for the training dataset was 1.5×10^{-4} mm^2 and that for the testing dataset was 5.0×10^{-4} mm^2, which was the least error that could be obtained on the testing dataset. To further evaluate the error in the values of depth provided by the neural network model, the absolute value of mean error, ME, between the target depth, i.e., the one given in the training and the testing datasets, and the depth calculated by the neural network is calculated as

$$ME = \left| \frac{\sum_{i=1}^{p} \left(d_i - o_i \right)}{p \times k} \right| \qquad (11)$$

where i is the index for the input dataset, d_i is the desired output, and o_i is the output produced by the neural network. The ME for depth in the training dataset was 0.04 mm and that in the testing dataset was 0.07 mm, where a typical depth value for GTA welding of low-carbon steel is 2.3 mm. Thus, the error in depth is well within the error limits for the process being considered.

In Fig. 8A, the depth varies from 0.9 to 8.5 mm, depending on the values of the input process parameters and the material properties. This shows that the process parameters and material properties considered in this study have a significant impact on the weld pool depth. The fact that the neural network model could accurately predict the depth values even for the testing data indicates that it is capable of accurately representing the results of the three-dimensional numerical heat and fluid flow model for GTA welding.

Figure 8C, D shows similar results for weld pool width, and Fig. 8E, F shows similar results for weld pool length. Similarly, the results for peak temperature are presented in Fig. 9A, B, those for cooling time between 800° and 500°C in Fig. 9C, D, and those for maximum liquid velocity in the weld pool in Fig. 9E, F. The corresponding MSEs and MEs are listed in Table 3. The MEs for the training data for weld pool width and length, peak temperature, cooling time between 800° and 500°C, and maximum liquid velocity in the weld pool are 0.05 mm, 0.05 mm, 4 K, 0.01 s, and 5 mm/s, respectively. These MEs are quite small compared to the respective magnitudes of these output variables, which are also listed in Table 3. Though the MEs for the testing data were slightly higher than those for the training data in all the cases, the difference was not very significant as can be seen from Table 3. The low values of ME for the testing data indicate that the neural networks can accurately predict different features of weld pool geometry as well as the peak temperature and cooling time, and hence can be used for simulations with predetermined good accuracy.

Summary and Conclusions

Six neural networks were developed for GTA welding of low-carbon steel. Each of these neural networks takes 17 input variables, which include welding process parameters and important material properties, and provides one output variable. The output variables include depth, width, and length of the weld pool, peak temperature, cooling time from 800° to 500°C, and maximum liquid velocity in the weld pool. The networks were trained using a hybrid optimization scheme including the gradient-descent method and a genetic algorithm. The hybrid approach gave lower errors than only the gradient-descent method on both training and testing datasets, and the results did not depend on the initial choice of weights. The training and testing datasets contained results from a reliable numerical heat and fluid flow model for GTA welding. The accurate prediction of these results by the neural networks ensured that the output of these networks complies with the phenomenological laws of welding physics.

Acknowledgments

This research was supported by a grant from the U.S. Department of Energy, Office of Basic Energy Sciences, Division of Materials Sciences, under grant number DE-FGO2-01ER45900. The authors thank Amit Kumar for his interest in this research.

Appendix A

Parent Centric Recombination (PCX) Based Generalized Generation Gap (G3) Genetic Algorithm (GA)

The genetic algorithm used in this study to calculate the optimized set of weights is a parent centric recombination (PCX) operator-based generalized generation gap (G3) model (Refs. 55, 56). This model was chosen because it has been shown to have a faster convergence rate on standard test functions as compared to other evolutionary algorithms. The algorithm for the model is as follows:

1. A population is a collection of many individuals and each individual represents a set of randomly chosen weights. A parent refers to an individual in the current population. The best parent is the individual that has the best fitness, i.e., gives the minimum value of the mean square error, defined by Equation 4, in the entire population. The best parent and two other randomly selected parents are chosen from the population.

2. From the three chosen parents, two offsprings or new individuals are generated using a recombination scheme. PCX-based G3 models are known to converge rapidly when three parents and two offspring are selected (Ref. 56). A recombination scheme is a process for creating new individuals from the parents.

3. Two new parents are randomly chosen from the current population.

4. A subpopulation of four individuals that includes the two randomly chosen parents in step 3 and two new offspring generated in step 2 is formed.

5. The two best solutions, i.e., the solutions having the least values of the mean square error, are chosen from the subpopulation of four members created in step 4. These two individuals replace the two parents randomly chosen in step 3.

6. The calculations are repeated from step 1 again until convergence is achieved.

The above steps, as applied to this study, are shown in Fig. 10. The process of finding the optimum set of weights by minimizing the mean square error is illustrated in Fig. 11. The recombination scheme (step 2) used in the present model is based on the PCX operator. A brief description of the PCX operator, as applied to the present problem of optimum set of weights, is presented below.

The first three parents, i.e.,

$$\left(w_1^0, w_2^0, \ldots, w_q^0 \right), \left(w_1^1, w_2^1, \ldots, w_q^1 \right), \left(w_1^2, w_2^2, \ldots, w_q^2 \right)$$

are randomly selected from the current population. Here, the subscripts represent the q

weights of the neural network, while the superscripts denote the parent identification number. The mean vector or centroid,

$$\bar{g} = \left(\frac{w_1^0 + w_1^1 + w_1^2}{3}, \frac{w_2^0 + w_2^1 + w_2^2}{3}, \ldots, \frac{w_q^0 + w_q^1 + w_q^2}{3} \right),$$

of the three chosen parents is computed. To create an offspring, one of the parents, say

$$\bar{x}^{(par)} = \left(w_1^0, w_2^0, \ldots, w_q^0 \right)$$

is chosen randomly. The direction vector,

$$\bar{d}^{(par)} = \bar{x}^{(par)} - \bar{g},$$

is next calculated from the selected parent to the mean vector or centroid. Thereafter, from each of the other two parents, i.e.,

$$\left(w_1^1, w_2^1, \ldots, w_q^1 \right) \text{ and } \left(w_1^2, w_2^2, \ldots, w_q^2 \right),$$

perpendicular distances, D_i, to the direction vector, $\bar{d}^{(par)}$, are computed and their average, $\bar{D}$, is found. Finally, the offspring, i.e.,

$$\bar{y} = \left(w_1', w_2', \ldots, w_q' \right),$$

is created as follows:

$$\bar{y} = \bar{x}^{(par)} + v_\xi \left| \bar{d}^{(par)} \right| + \sum_{i=1, i \neq par}^{q} v_\eta \bar{D}\bar{h}^{(i)} \quad \text{(A1)}$$

where $\bar{h}^{(i)}$ are the orthonormal bases that span the subspace perpendicular to $\bar{d}^{(par)}$, and v_ξ and v_η are randomly calculated zero-mean normally distributed variables. The values of the variables that characterize the offspring,

$$\bar{y} = \left(w_1', w_2', \ldots, w_q' \right),$$

are calculated next. As an example, only the calculation of w'_1 and w'_2 will be described here:

$$w_1' = w_1^0 + w_{11} + w_{12} \quad \text{(A2.a)}$$

$$w_2' = w_2^0 + w_{21} + w_{22} \quad \text{(A2.b)}$$

where,

$$w_{11} = v_\xi \left(\frac{2w_1^0 - w_1^1 - w_1^2}{3} \right) \quad \text{(A3.a)}$$

$$w_{21} = v_\xi \left(\frac{2w_2^0 - w_2^1 - w_2^2}{3} \right) \quad \text{(A3.b)}$$

$$w_{12} = v_\eta \left(\frac{a_2 + b_2}{2} \right) \left[1 - \left(\frac{2w_1^0 - w_1^1 - w_1^2}{3d} \right)^2 \right] \quad \text{(A3.c)}$$

$$w_{22} = v_\eta \left(\frac{a_2 + b_2}{2} \right) \left[1 - \left(\frac{2w_2^0 - w_2^1 - w_2^2}{3d} \right)^2 \right] \quad \text{(A3.d)}$$

The expressions for the variables d, a_2, and b_2 used in Equations A3.c and A3.d, are as follows:

$$d = \sqrt{\left(\frac{2w_1^0 - w_1^1 - w_1^2}{3} \right)^2 + \left(\frac{2w_2^0 - w_2^1 - w_2^2}{3} \right)^2 + \ldots + \left(\frac{2w_q^0 - w_q^1 - w_q^2}{3} \right)^2} \quad \text{(A4.a)}$$

$$a_2 = e_1 \times \sqrt{1 - \left(a_1 \right)^2} \quad \text{(A4.b)}$$

$$b_2 = e_2 \times \sqrt{1 - \left(b_1 \right)^2} \quad \text{(A4.c)}$$

$$a_1 = \sum_{i=1}^{q} \frac{\left(w_i^1 - w_i^0 \right) \left(\frac{2w_i^0 - w_i^1 - w_i^2}{3} \right)}{d \times e_1} \quad \text{(A4.d)}$$

$$e_1 = \sqrt{\left(w_1^1 - w_1^0 \right)^2 + \left(w_2^1 - w_2^0 \right)^2 + \ldots + \left(w_q^1 - w_q^0 \right)^2} \quad \text{(A4.e)}$$

$$b_1 = \sum_{i=1}^{q} \frac{\left(w_i^2 - w_i^0 \right) \left(\frac{2w_i^0 - w_i^1 - w_i^2}{3} \right)}{d \times e_2} \quad \text{(A4.f)}$$

$$e_2 = \sqrt{\left(w_1^2 - w_1^0 \right)^2 + \left(w_2^2 - w_2^0 \right)^2 + \ldots + \left(w_q^2 - w_q^0 \right)^2} \quad \text{(A4.g)}$$

References

1. De, A., and DebRoy, T. 2005. Reliable calculations of heat and fluid flow during conduction mode laser welding through optimization of uncertain parameters. *Welding Journal* 84(7): 101-s to 112-s.

2. Zhang, W., Kim, C.-H., and DebRoy, T. 2004. Heat transfer and fluid flow in welds with complex geometry during gas-metal arc fillet welding, Part I: Numerical model. *Journal of Applied Physics* 95: 5210–5219.

3. Zhang, W., Kim, C.-H., and DebRoy, T. 2004. Heat transfer and fluid flow in welds with complex geometry during gas-metal arc fillet welding, Part II: Application to fillet welding of mild steel. *Journal of Applied Physics* 95: 5220–5229.

4. Zhang, W., DebRoy, T., and Elmer, J. W. 2005. Integrated modeling of thermal cycles, austenite formation, grain growth and decomposition in the heat affected zone of carbon steel. *Science and Technology of Welding and Joining* 10(5): 574–582.

5. De, A., and DebRoy, T. 2004. Probing unknown welding parameters from convective heat transfer calculation and multivariable optimization. *Journal of Physics D* 37: 140–150.

6. Mishra, S., and DebRoy, T. 2004. Grain topology in Ti-6Al-4V welds — Monte Carlo simulation and experiments. *Journal of Physics D* 37: 2191–2196.

7. Mishra, S., and DebRoy, T. 2004. Measurements and Monte Carlo simulation of grain structure in the heat-affected zone of Ti-6Al-4V welds. *Acta Materialia* 52(5): 1183–1192.

8. He, X., Fuerschbach, P., and DebRoy, T. 2003. Probing temperature during laser spot welding from vapor composition and modelling. *Journal of Applied Physics* 94(10): 6949–6958.

9. Mishra, S., Chakraborty, S., and DebRoy, T. 2005. Probing liquation cracking and solidification through modeling of momentum, heat and solute transport during welding of aluminum alloys. *Journal of Applied Physics* 97: 94912–94920.

10. Kumar, A., and DebRoy, T. 2003. Calculation of three-dimensional electromagnetic force field during arc welding. *Journal of Applied Physics* 94(2): 1267–1277.

11. De, A., and DebRoy, T. 2004. A smart model to estimate effective thermal conductivity and viscosity in weld pool. *Journal of Applied Physics* 95(9): 5230–5239.

12. Zhang, W., Roy, G. G., Elmer, J. W., and DebRoy, T. 2003. Modeling of heat transfer and fluid flow during gas tungsten arc spot welding of low carbon steel. *Journal of Applied Physics* 93(5): 3022–3033.

13. Kumar, A., Zhang, W., and DebRoy, T. 2005. Improving reliability of modeling heat and fluid flow in complex gas-metal-arc fillet welding — Part I: An engineering physics model. *Journal of Physics D* 38: 118–126.

14. Kumar, A., and DebRoy, T. 2005. Im-

proving reliability of modeling heat and fluid flow in complex gas-metal-arc fillet welding — Part II: Application to welding of mild steel. *Journal of Physics D* 38: 127–134.

15. He, X., Elmer, J. W., and DebRoy, T. 2005. Heat transfer and fluid flow in laser microwelding. *Journal of Applied Physics* 97: 84909–84917.

16. Zhang, W., DebRoy, T., Palmer, T. A., and Elmer, J. W. 2005. Modeling of ferrite formation in a duplex stainless steel weld considering nonuniform starting microstructure. *Acta Materialia* 53(16): 4441–4453.

17. Mundra, K., DebRoy, T., and Kelkar, K. 1996. Numerical prediction of fluid flow and heat transfer in welding with a moving heat source. *Numerical Heat Transfer* 29: 115–129.

18. Kumar, A., and DebRoy, T. 2004. Guaranteed fillet weld geometry from heat transfer model and multivariable optimization. *International Journal of Heat and Mass Transfer* 47(26): 5793–5806.

19. Lancaster, J. F. 1986. *The Physics of Welding*, 2nd Edition, Pergamon: Oxford, UK.

20. Vitek, J. M., Iskander, Y. S., and Oblow, E. M. 2000. Improved ferrite number prediction in stainless steel arc welds using neural networks — Part 1: Neural network development. *Welding Journal* 79(2): 33-s to 40-s.

21. Vitek, J. M., Iskander, Y. S., and Oblow, E. M. 2000. Improved ferrite number prediction in stainless steel arc welds using neural networks — Part 2: Neural network results. *Welding Journal* 79(2): 41-s to 50-s.

22. Kim, I. S., Son, J. S., Lee, S. H., and Yarlagadda, P. K. D. V. 2004. Optimal design of neural networks for control in robotic arc welding. *Robotics and Computer-Integrated Manufacturing* 20: 57–63.

23. Kim, I. S., Son, J. S., and Yarlagadda, P. K. D. V. 2003. A study on the quality improvement of robotic GMA welding process. *Robotics and Computer-Integrated Manufacturing* 19: 567–572.

24. Nagesh, D. S., and Datta, G. L. 2002. Prediction of weld bead geometry and penetration in shielded metal-arc welding using artificial neural networks. *Journal of Materials Processing Technology* 123: 303–312.

25. Bhadeshia, H. K. D. H., Mackay, D. J., and Svensson, L. E. 1995. The impact toughness of C-Mn steel arc-welds — A Bayesian neural network analysis. *Materials Science and Technology* 11: 1046–1051.

26. Li, P., Fang, M. T. C., and Lucas, J. 1997. Modelling of submerged arc weld beads using self-adaptive offset neutral networks. *Journal of Materials Processing Technology* 71: 288–298.

27. Tarng, Y. S., Tsai, H. L., and Yeh, S. S. 1999. Modeling, optimization and classification of weld quality in tungsten inert gas welding. *International Journal of Machine Tools & Manufacture* 39: 1427–1438.

28. Jeng, J.-Y., Mau, T.-F., and Leu, S.-M. 2000. Prediction of laser butt joint welding parameters using back propagation and learning vector quantization networks. *Journal of Materials Processing Technology* 99: 207–218.

29. Srikanthan, L. T., and Chandel, R. S. 1988. Neural network based modeling of GMA welding process using small data sets. *Proceedings of the 5th International Conference on Control, Automation, Robotics and Vision*. Singapore, pp. 474–478.

30. Metzbower, E. A., Deloach, J. J., Lalam, S. H., and Bhadeshia, H. K. D. H. 2001. Analysis of strength and ductility of welds in shipbuilding steels. *Science and Technology of Welding and Joining* 6: 116–124.

31. Kim, I. S., Jeong, Y. J., Lee, C. W., and Yarlagadda, P. K. D. V. 2003. Prediction of welding parameters for pipeline welding using an intelligent system. *International Journal of Advanced Manufacturing Technology* 22: 713–719.

32. Andersen, K., Cook, G. E., Karsai, G., and Ramaswamy, K. 1990. Artificial neural networks applied to arc welding process modeling and control. *IEEE Transactions on Industry Applications* 26(5): 824–830.

33. Cook, G. E., Barnett, R. J., Andersen, K., and Strauss, A. M. 1995. Weld modeling and control using artificial neural networks. *IEEE Transactions on Industry Applications* 31(6): 1484–1491.

34. Tsuei, H., Dunne, D., and Li, H. 2003. Neural network analysis of impact properties of flux cored arc weld metals for structural steels. *Science and Technology of Welding and Joining* 8(3): 205–212.

35. Gao, J., and Wu, C. 2003. Neurofuzzy control of weld penetration in gas tungsten arc welding. *Science and Technology of Welding and Joining* 8(2): 143–148.

36. Kim, I. S., Lee, S. H., and Yarlagadda, P. K. D. V. 2003. Comparison of multiple regression and back propagation neural network approaches in modeling top bead height of multipass gas metal arc welds. *Science and Technology of Welding and Joining* 8(5): 347–352.

37. Demuth, H., and Beale, M. 1998. *Neural network tool box user's guide — Version 3.0*. Mathworks, Inc., Mass.

38. Juang, S. C., Tarng, Y. S., and Lii, H. R. 1998. A comparison between the backpropagation and counter-propagation networks in the modeling of the TIG welding process. *Journal of Materials Processing Technology* 75: 54–62.

39. Vitek, J. M., David, S. A., Richey, M. W., Biffin, J., Blundell, N., and Page, C. J. 2001. Weld pool shape prediction in plasma augmented laser welded steel. *Science and Technology of Welding and Joining* 6: 305–314.

40. Eguchi, K., Yamane, S., Sugi, H., Kubota, T., and Oshima, K. 1999. Application of neural network to arc sensor. *Science and Technology of Welding and Joining* 4(6): 327–334.

41. Li, X., Simpson, S. W., and Rados, M. 2000. Neural networks for online prediction of quality in gas metal arc welding. *Science and Technology of Welding and Joining* 5(2): 71–79.

42. Metzbower, E. A., Deloach, J. J., Lalam, S. H., and Bhadeshia, H. K. D. H. 2001. Analysis of the toughness of welds in shipbuilding steels. *Science and Technology of Welding and Joining* 6: 368–374.

43. Hong, T., Pitscheneder, W., and DebRoy, T. 1998. Quantitative modeling of inclusion growth in the weld pool by considering their motion and temperature gyrations. *Science and Technology of Welding and Joining* 3(1): 33–41.

44. Hong, T., and DebRoy, T. 2001. Effects of time, temperature and steel composition on the growth and dissolution of inclusions in liquid steels. *Ironmaking and Steelmaking* 28(6): 450–454.

45. Rumelhart, D. E., Hinton, G. E., and Williams, R. J. 1986. Learning representations by back-propagating errors. *Nature* 323: 533–536.

46. Burke, L., and Ignizio, J. P. 1997. A practical overview of neural networks. *Journal of Intelligent Manufacturing* 8(3): 157–165.

47. Fausett, L. 1994. *Fundamentals of Neural Networks*. Englewood Cliffs, N.J.: Prentice-Hall.

48. Haykin, S. 2001. *Neural Networks – A Comprehensive Foundation*, 2nd Edition. Singapore: Pearson Education.

49. Masters, T. 1993. *Practical Neural Network Recipes in C++*. Boston, Mass.: Academic Press.

50. Kalman, B. L., and Kwasny, S. C. 1992. Why tanh? Choosing a sigmoidal function. *Proceedings of the International Joint Conference on Neural Networks - Volume 4*. Baltimore, Md. pp. 578–581.

51. Bose, N. K., and Liang, P. 1996. *Neural Network Fundamentals with Graphs, Algorithms, and Applications*. New York, N.Y.: McGraw-Hill.

52. Yarlagadda, P. K. D. V. 2001. Prediction of processing parameters for injection moulding by using a hybrid neural network. *Proceedings of the Institution of Mechanical Engineers B Journal of Engineering Manufacture* 215(10): 1465–1470.

53. Huang, S. H., and Zhang, H. C. 1995. Neural-expert hybrid approach for intelligent manufacturing: A survey. *Computers in Industry* 26: 107–126.

54. Kumar, A., Mishra, S., Elmer, J. W., and DebRoy, T. 2005. Optimization of Johnson Mehl Avarami equation parameters for α-ferrite to γ-austenite transformation in steel welds using a genetic algorithm. *Metallurgical and Materials Transactions A* 36: 15–22.

55. Deb, K., Anand, A., and Joshi, D. 2002. A computationally efficient evolutionary algorithm for real-parameter optimization. *Evolutionary Computation* 10(4): 371–395.

56. Deb, K. 2001. *Multi-Objective Optimization Using Evolutionary Algorithms*, 1st Edition, New York, N.Y.: Wiley.

57. Mishra, S., and DebRoy, T. 2005. A computational procedure for finding multiple solutions of convective heat transfer equations. *Journal of Physics D* 38: 2977–2985.

58. Mishra, S., and DebRoy, T. 2005. A heat transfer and fluid flow based model to obtain a specific weld geometry through multiple paths. *Journal of Applied Physics* 98: 044902-1 to 044902-10.

59. Kumar, A., and DebRoy, T. 2005. Tailoring complex weld geometry through reliable heat transfer and fluid flow calculations and a genetic algorithm. *Metallurgical and Materials Transactions A* 36: 2725–2735.

60. Patankar, S. V. 1980. *Numerical Heat Transfer and Fluid Flow*. New York, N.Y.: Hemisphere.

61. Lippmann, R. P. 1987. An introduction to computing with neural nets. IE*EE Acoustics, Speech, and Signal Processing Magazine* 3(4): 4–22.

62. Makhoul, J., El-Jaroudi, A., and Schwartz, R. 1989. Formation of disconnected decision regions with a single hidden layer. *Proceedings of the International Joint Conference on Neural Networks* — Vol. 1. Washington, D.C., pp. 455–460.