Vol. **3**, **4**, 435–512 (1996)

Archives of Computational Methods in Engineering
State of the art reviews

Neural Networks in Computational Mechanics

G. Yagawa
Department of Quantum Engineering and Systems Science
University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113, Japan

H. Okuda
Department of Mechanical Engineering
Yokohama National University
156 Tokiwadai, Hodogaya-ku, Yokohama 240, Japan

Summary

In this paper, recent neural network applications, epecially to the fields related with the computational mechanics, were surveyed. The most outstanding characteristics of the neural network aided computation is that neither complicated programmings nor rigid algorithms are needed. Another important point is that the neural network's inherent parallelism, that is, concurrent signal transmissions over numerous information processing elements suits the massively parallel computer architectures. First, we briefly review the neural network applications to the computational mechanics fields from recent publications, and describe the mathematical basis of the neural network. Next, the following topics are detailed: quantitative nondestructive evaluation, structural identification, modeling of viscoplastic material behaviors, crack growth analysis of welded specimens, structural design, parameter estimation for nonlinear finite element analyses, and equation solver.

1. INTRODUCTION

Progress in the computational mechanics has been dramatic in the last few decades. Nevertheless, practical engineers and designers claim more talented simulation environments, i.e., the CAE (Computer Aided Engineering) systems to be developed, which would satisfy such requirements as (a) high accuracy, (b) faster computation than real time, (c) user-friendliness even for beginners and (d) portability among various computer facilities. To meet these purposes, evolutional computer technologies like the fuzzy theory and the neural network seem to play key roles [1],[2]. This paper focuses on the artificial neural network, and reviews its applications in the field of the computational mechanics. At various scenes of design, the inverse problem and/or the optimization problem need to be solved. The neural network aided simulation methodologies merged with the computational mechanics e.g. the finite element method are expected to be efficient tools for such problems.

Figure 1.1 schematically depicts the biological neuron. Each neuron receives signals, which are either positive or negative impulses, from many other neurons. When the accumulated signals reached a certain amount, the signal is transfered to other neurons through the axon. Thus, signals are successively transfered over the biological neural network, which are constructed by numerous neurons. For example, a human being is estimated to have around 10^{10} brain cells. The artificial neural network (simply called 'neural network' in the following descriptions) was developed to mimic the biological brain neural network [3]-[8]. The neural network is characterized as the information processing structure consisting of highly parallel distributed elements. Each element possesses a local memory and can carry out localized information processing operations. Research on the information processing using the neural network started in 1940's.

 ISSN: 1134–3060 Received: September 1995

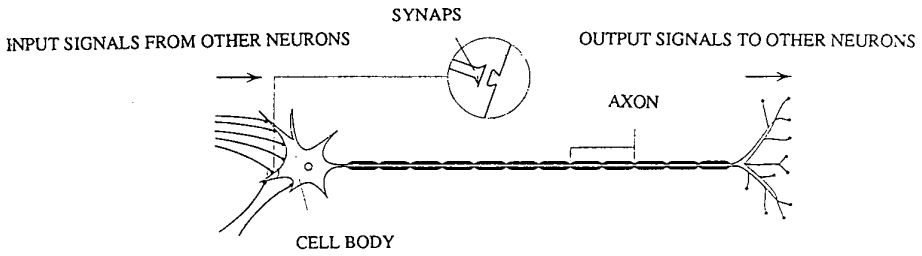
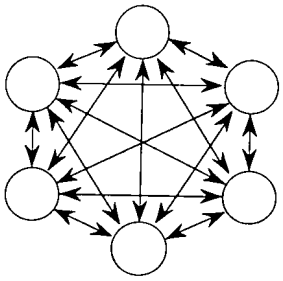
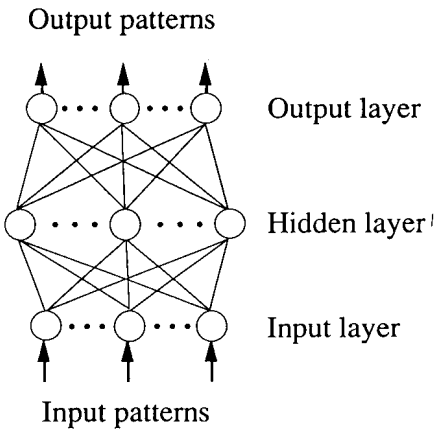


Figure 1.1. Biological neuron



(a) Interconnected neural network



(b) Hierarchal neural network

Figure 1.2. Network topology

The neural network model can be classified in to the interconnected neural network and the hierarchal neural network. Figure 1.2 schematically depicts these network structures. The terminologies came from the connection style of neurons. They are also called the recurrent neural network and the feed-forward neural network, respectively, which stand for the way of data transmission. Basis of the neuron and these two types of neural network will be briefly described in the next section.

Starting from the Hopfield's network [9][47], the interconnected neural network has been applied to a wide variety of optimization problems. On the other hand, the hierarchal neural network [10], which transmitts signals in one direction between perceptron layers, has become a powerful tool in the pattern recognition, control, diagnosis and so on.The ability of parameter identification of the neural network has also been utilized to solve the inverse problems.

There have been published many literatures about the neural network [3]-[8]. The applications of neural networks in a control field were surveyed by Fukuda and Shibata (1992)[11]. In this paper, recent neural network applications, epecially to the fields related with the computational mechanics, were surveyed. The most outstanding characteristics of the neural network aided computation is that neither complicated programmings nor rigid algorithms are needed. Given the teaching data repeatedly, solutions of the system are pursued iteratively until a certain tolelance would be satisfied. Another important point is that the neural network's inherent parallelism, that is, concurrent signal transmissions over numerous information processing elements suits the massively parallel computer architectures, which is a leading faction of the computer development.

The computational mechanics such as the finite element method has widely prevailed as the indispensable tool in the field of structural mechanics [32]-[48]. The neurological computations in structural mechanics have also been exploited in many applications such as structural design optimization, coupled analysis, structural identification, modeling of materials, reasoning system of the stresses.

The neural network's identification ability is effectively used in the nondestructive evaluation (NDE) [49]-[59], which detects the defects hidden in solids. Many sample patterns between the defect parameters and the sensing values of eddy current, ultrasonic wave, electric potential etc. are trained into the hierarchal network. Sensing values are sometimes substituted by numerical simulations such as the finite element method and the boundary element method. The interconnected neural network is also employed in NDE.

Many of industrial systems have no first-principles to reproduce their dynamic behaviors or mapping patterns, which often makes it impossible to utilize the computational mechanics methodologies. The neural networks are employed to model such nonlinear systems instead of, or otherwise hybridized with the numerical simulations [60]-[66]. To improve the efficiency of numerical simulations, the neural networks are also considered as promising tools.

Not only the structural optimization problems but also a wide variety of constrained or unconstrained optimization problems have been solved by the neural networks [67]-[73]. An iterative equation solver formulated as an optimization problem is also constructed on the neural network.

Trends in the supercomputer development is towards the distributed-memory type massively parallel architecture, and full potential of the neural networks would be disclosed if they are implemented on (massively) parallel computers, which would realize their inherent parallelism. The neural networks have been implemented on general purpose parallel machines [74]-[83]. Also, special purpose parallel neuro computers and a wafer scale LSI for neurocomputers, i.e., a digital neurochip have been exploited [84]-[89].

This paper is organized as follows. In Chapter 2, we briefly describe the mathematical basis of the neural network, and mention some recent progress in the neural network. From Chapter 3 to 9, neural network applications to the computational mechanics fields are reviewed from recent publications. The following topics are detailed: quantitative nondestructive evaluation, structural identification, modeling of viscoplastic material behaviors, crack growth analysis of welded specimens, structural design, parameter estimation for nonlinear finite element analyses, and equation solver.

2. BASIS OF THE NEURAL NETWORK

2.1 Neuron Model [3]-[14]

An artificial model of neuron is shown in Figure 1.3. The neurons are connected with neighboring ones by nerve fibers depicted by the solid lines so that the network system is organized as shown in Figure 1.2. In general, a neuron has multiple inputs and a single

output. The inputs are operated and transformed into the output by the state transition rule as

$$t_i = \sum_{\substack{j \\ (i \neq j)}} w_{ij} u_j + \theta_i \tag{1.1}$$

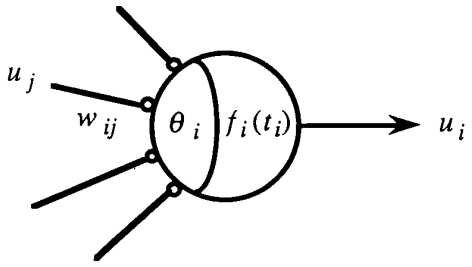
$$u_i = f(t_i) \tag{1.2}$$

where u_j in Eq.(1.1) and u_i in Eq.(1.2) are the input from a neuron j and the output from a neuron i, respectively. w_{ij} is the connection weight, θ_i the offset, t_i the state variable and $f(t_i)$ the activation function. Input signals sent from other neurons are multiplied by the connection weights, which imply the connection strength between the neurons.The weighted signals are then summed up and transformed into the output signal through an activation function. Since activations of neurons are expected to occur independently and randomly, the state transition of a neural network is an attractive computational model for a massively parallel platform.

The sigmoid function, which is a continuous and nonlinear function taking the value of 0 through 1 (see Figure 1.4) is often used as the activation function. A typical sigmoid function is expressed as

$$f(t_i) = 1/\{1 + \exp(-t_i/T)\} \tag{1.3}$$

where T is called a temperature of sigmoid function.



Unit i

u_i	Output from unit i
u_j	Input from unit j
w_{ij}	Weight
θ_i	Offset
t_i	State variable
$f(\,)$	Activation function

Figure 1.3. Neuron model

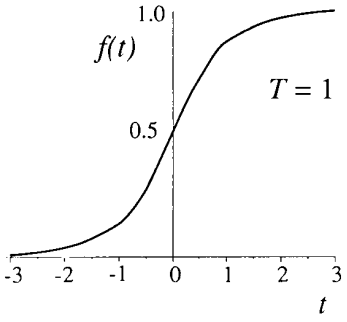


Figure 1.4. Sigmoid function

2.2 Interconnected or Recurrent Neural Network [9]

As is shown in Figure 1.2(a), the interconnected neural network is composed of neurons, which allow mutual communications among themselves. Repetitious state transitions according to Eqs.(1.1) and (1.2) minimize the energy of network, which is defined as:

$$E(u) = -\frac{1}{2}\sum_{\substack{i,j \\ (i\neq j)}} w_{ij}u_iu_j - \sum_i \theta_i u_i \tag{1.4}$$

Therefore, considering the analogy of the equilibrium of the network energy to the minimization of a cost function, the state transition procedure of the interconnected neural network can be utilized for solving many kinds of optimization problems.

2.3 Hierarchical Feed-Forward Neural Network [10]

A three-layered hierarchical neural network is schematically shown in Figure 1.2(b).The whole units are formed into multiple layers, i.e. an input layer, a hidden (or intermediate) layer and an output layer. No connections exist among units in the same layer, while every two units in the neighboring layers have a connection. The attractive features of the hierarchical neural networks can be summarized as follows :

1) One can automatically construct a nonlinear mapping from multiple input data to multiple output data in the network through a learning process of some or many sample input vs. output relations.
2) The network has a capability of the so-called "generalization", i.e. a kind of interpolation, such that the trained network estimates appropriate output data even for unlearned input data.
3) The trained network operates quickly in an application phase. The CPU power required to operate it may be equivalent only to that of a personal computer.

Although the number of hidden layers can be more than two, a three-layered neural network having sufficient number of units is known to map nearly all kinds of relationship. As a teaching method, the error-back-propagation is discussed here.

The connection weights and the offset values are iteratively adjusted to minimize the square error between the output and the teaching data, which is defined by:

$$E^p = \sum_j \frac{1}{2}(T_j^p - u_j^p)^2 \tag{1.5}$$

where E^p, u_j^p and T_j^p are respectively the square error, the output from a unit j and the target value for a unit j as to the learning data p. During the teaching process, modifications of the connection weight and the offset values are repeated according to the follwing formulae until the convergence is attained:

$$\Delta w_{ji} = \alpha(T_j^p - u_j^p) f'(t_j) u_i \tag{1.6}$$

$$\Delta \theta_j = \beta(T_j^p - u_j^p) f'(t_j) \tag{1.7}$$

where Δw_{ji} and $\Delta \theta_j$ are modifications of the connection weight and the offset, respectively. f' is the derivation of f. α and β are positive parameters, which control the convergence characteristic. As the teaching proceeds, the error defined by Eq.(1.5) usually decreases. However, when the teaching is carried out excessively, a gradual increase of the error due to an excessive teaching might occur. Therefore, the error with respect to the unlearning data, which is not used for training the network, is monitored during the teaching. The teaching is terminated when this error is about to rebound.

2.4 Recent Advances in Neural Network [15]-[31]

In Sections 2.1 through 2.3, we briefly described the mathematical basis of the neuron model, the interconnected neural network and the hierarchal neural network. More details could be obtained in the literatures [3]-[8]. In this section, some of recent advances in the neural network will be discussed such as learning algorithm, network size, network topology, new concept of neural network etc. Progress in fundamentals of the neural network would strengthen the credibility of the neural network aided computation tools, and also would widen the area of application fields.

Learning performances with respect to speed, accuracy and robustness are very important issue when considering the feasibility of the neurological computation applied to the engineering practical problems. Various types of learninig algorithms [15]-[18], many of which are for multi-layered error-back-propagation networks, have been developed to improve the learning performance of the neural network.

Since the optimum size and topology of the neural network are usually unknown, users of the neural network would have to resort to a time-consuming heuristic choice before constructing application phase neural networks. As for the hierarchal neural network, an a priori estimation method of optimum network size and topology are discussed [19]-[21].

As is stated above, the generalization is an important and attractive feature of the hierarchal neural network. Improvement of the generalization performance is investigated in [22]-[25]. Other fundamental features of the neural network, which may affect the computational performance and/or the robustness of the neurological computation strategy, are also discussed in [26]-[28]. In [29]-[31] , new notions of the neural network have been proposed.

3. NEURAL-NETWORK-BASED INVERSE ANALYSIS FOR QUANTITATIVE NONDESTRUCTIVE EVALUATION [54],[55]

3.1 Quantitative Nondestructive Evaluation [49]–[59]

Nondestructive evaluation (NDE) techniques to detect cracks and defects hidden in solid are very important to assure the structural integrity of operating plants and structures, and to evaluate their residual life time. Various NDE techniques using ultrasonic wave, X-ray, magnetic powder, eddy current and so on have been studied so far. In most cases, they detect only the existence of cracks and defects. Then, quantitative nondestructive evaluation

(QNDE) techniques of sizes, shapes and locations of cracks and defects are strongly desired.

In the following sections, described are the applications of the neural network to QNDE, which is characterized as an inverse problem. First, in Section 3.2, the principle of the neural-network-based inverse analysis is described for the defect identification problem by the ultrasonic wave. It should be noted that the inverse analysis approach is applicable not only to QNDE, but also to other various engineering fields such as the structural identification, the material identification and so on. These applications will be detailed later. Next, the ultrasonic-based identification and the electric-potential-based defect identification problems are described in Section 3.3 and 3.4, respectively.

3.2 Principle of Neural-Network-Based Inverse Analysis

In the ultrasonic-based defect identification problems, defect parameters such as the size and the location of a defect are determined from the dynamic responses of solid measured at several points on the solid surface. This is one of the inverse problems. On the other hand, if the defect parameters are given, the dynamic responses of solid can be computed through the computational mechanics, e.g. the finite element simulations. This is a direct problem.

Among various researches, much attention has been paid to the inverse analysis approaches based on the computational mechanics combined with kinds of optimization techniques. These approaches have the following advantages and disadvantages: (a) These are good at quantitative evaluation. (b) These are hardly applied to multiple local minima problems. (c) These require a number of computational mechanics simulations in application processes of identification. The item (b) is inevitable in conventional optimization methods.

These direct and inverse problems can be simply regarded as the nonlinear mappings from the multiple data, i.e. the dynamic responses, to the multiple data, i.e. the defect sizes and the locations. Thus, the inverse analysis procedure using the hierarchical neural network can be constructed. As shown in Figure 3.1, the procedure consists of the following three phases:

Phase 1: The dynamic finite element simulations (i.e. direct analyses) of solid containing a defect are performed repeatedly with parametrically changing defect parameters. Data pairs of the defect parameters vs. the dynamic responses of solid are collected.

Phase 2: The neural network is trained with the data pairs collected in Phase 1. The defect parameters and the dynamic responses of solid are given to the network as the teaching data and the input data, respectively.

Phase 3: The trained network is utilized to identify defect parameters from the measured dynamic responses of solid in an application process.

This inverse analysis approach can be applied to any inverse problem by replacing the computational mechanics simulations. Thus, the present inverse analysis approach using the neural networks and the computational mechanics are widely applicable to various engineering fields.

3.3 Defect Identification Based on Ultrasonic Method [54]

3.3.1 Problem definition

A rectangular plate containing a surface defect on its bottom surface is assumed as an analysis domain as shown in Figure 3.2. An ultrasonic wave is given to the midst of upper

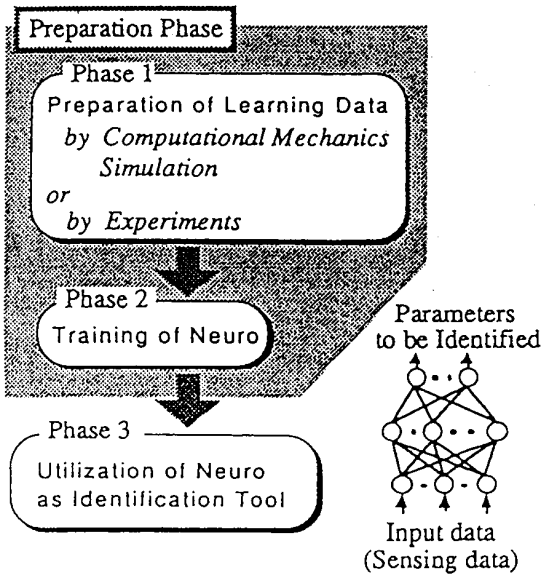


Figure 3.1. Inverse analysis procedure based on hierarchical neural network

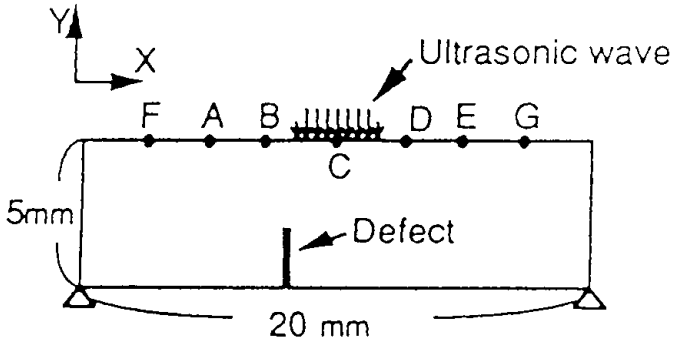


Figure 3.2. 2D analysis models con taining a surface defect

surface of the plate. The v ertical and horizontal locations of the tip of the surface defect are identified by monitoring the w av reflected by the defect at several points on the plate surface. The analysis domain is 5 mm × 20 mm large. The inciden t ultrasonic w ave introduced in the vertical direction is 4.2 mm in width on the plate surface, 10 MPa in amplitude, 2.5 MHz in frequency and one cycle in duration.

Dynamic responses of displacements at 7 monitoring poin ts (A to G) on the surface are computed b y the dynamic finite elemen t method. The material properties assumed are those of steel, i.e. the Y oung's modulus = 205 GPa, the Poisson's ratio = 0.3 and the mass density = 7.84×10^3 kg/m^3. The whole analysis domain is equally divided into 24 × 96 four-noded quadrilateral isoparametric elements. The size of eac h elemen t is 0.21 mm × 0.21 mm, which is almost a ten th of the length of the longitudinal ultrasonic w ave of 2.5 MHz. The explicit time integration scheme, i.e. the cen tral difference method, is emplo yed. A time increment is chosen to be 1.0×10^{-8} sec. Dynamic responses at the monitoring points are recorded in every 3.0×10^{-6} sec, whic h is sufficien tly small compared with the duration such that the longitudinal ultrasonic w av may arrive at the monitoring poin ts after reflected by the defect.

The defect is 0.21 mm wide, and its length is chosen among nine values ranging from 0.42 mm to 3.78 mm. The location of the defect is also varied in the horizontal direction. Figure 3.3 illustrates 99 possible locations of the tip of defect. Fifty open circles designate locations of the tip of defect utilized for network training as the learned data, while 49 closed circles indicate those utilized to examine a capability of generalization as the unlearned data.

In Phase 1 of the inverse analysis procedure, the learning data can also be collected through the experiments. However, the computational mechanics simulations seem superior to the experiments in flexibility, and are expected to improve in accuracy as well as speed in accordance with a drastic progress of computers.

3.3.2 Preparation of input data

Dynamic responses of displacements monitored at the seven points during a 3.0 msec period are sampled at every 10 msecs. Some characteristic values which are more sensitive to defect parameters are extracted from the original dynamic response curves (see Figure 3.4). First, the dynamic responses of solid containing a defect is subtracted by those of solid without any defect. Then, the first peak of the obtained waveform is picked up, and its height, i.e. PeakHeight, and its time, i.e. PeakTimeStep are employed as the characteristic values.

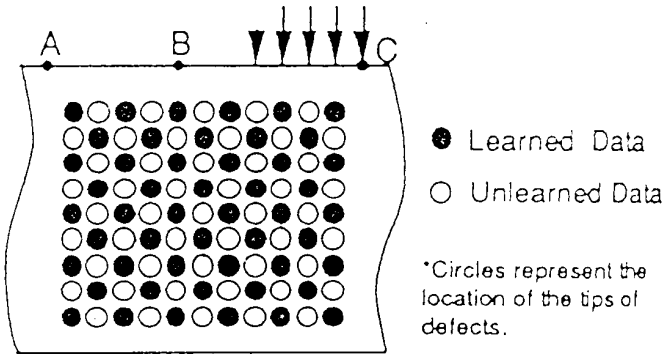


Figure 3.3. Schematic distributions of learned and unlearned data

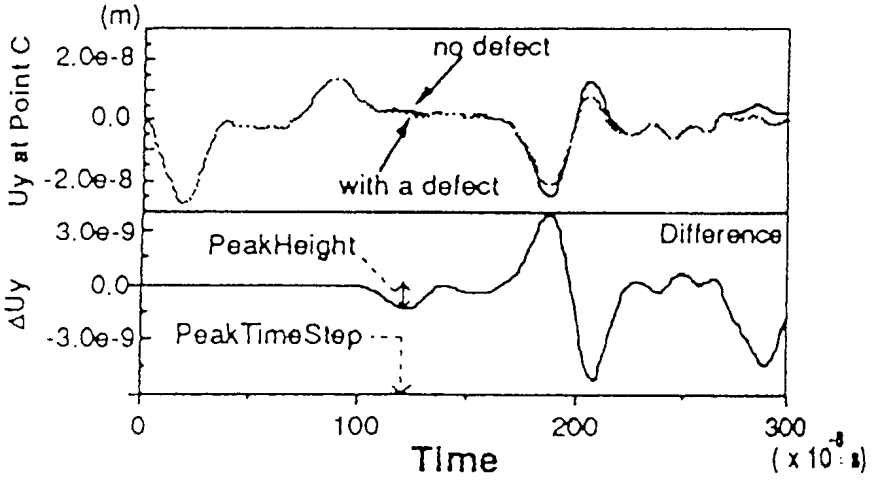


Figure 3.4. Extraction of some characteristic values from dynamic responses

The characteristic values are evaluated only for a displacement in the horizontal direction u_y at each monitoring point. Both PeakHeight and PeakTimeStep are normalized into a unit range of (0.0 - 1.0). The defect parameters, i.e. the horizontal and vertical locations of tip of a defect are normalized into a reduced range of (0.15 - 0.85), in which the sigmoid function is regarded quasi-linear. Such normalization of defect parameters enables the neural network to output smooth analog values.

The following two cases are tested:

Case 1: Four characteristic values of u_y monitored at the two points A and C are adopted.

Case 2 Fourteen characteristic values of u_y monitored at the seven points A to G are adopted.

3.3.3 Learning process

The three-layered network is adopted, and the network is trained by the EBP algorithm combined with the moment method, which accelerates a learning speed. The output layer possesses two units, which output the horizontal and vertical locations of tip of a defect. The numbers of units in the input and output layers are automatically determined, referring to the problem defined. However, the number of units in the hidden layer is still left undetermined. Here 8, 12 and 16 units are employed in the hidden layer through trial and error. The learning rates α and β of Eqs.(6) and (7) are taken to be 0.3 and 0.7, respectively. The number of learning iterations is limited to be less than 50,000. Since the EBP learning algorithm is based on the steepest gradient method, initial values of the connection weights and the offset values significantly influence the learning as well as the estimation processes. In order to minimize such influence of the initial values, twenty sets of initial states of the network are examined for a given learning condition, and the mean value of the ten best results is employed as a final one.

Usually the estimation error for the learning data decreases monotonously as the learning iterations proceed. However, that for unlearning data sometimes stop decreasing, and start increasing. This phenomenon is called "overlearning". Both estimation errors for learning as well as unlearning data are monitored during a learning process.

Number of monitoring points	Number of hidden units	Learning factor $\alpha(=\beta)$
2	8	0.7
7	12	0.7

Table 3.1. Learning conditions

3.3.4 Estimation of tip of defect

Table 3.1 tabulates the learning conditions in which the minimum error for unlearned data is attained. Figure 3.5 illustrates the locations of a tip of defect estimated by the trained network corresponding to the minimum identification error. In the figure, exact solutions are also shown for the sake of comparison. It can be seen from Figure 3.5(a) that, in the case of two monitoring points (Case 1), the discrepancy between the estimated and the exact results is more significant in the upper-left region of the figure, i.e. when a defect is very deep. Figure 3.6 shows the distributions of characteristic values, i.e. PeakHeight and PeakTimeStep monitored at the points A and C with respect to vertical and horizontal locations of a tip of defect. The figure shows that the PeakHeight monitored at point A has

steep gradient when a defect is v ery deep. This ma y be the reason of suh less accuracy in Case 1. How ever, the estimation accuracy is impro ved by emplo ying more monitoring points (Case 2). This ma y be because additional w ell-qualified information monitored at other points are available.

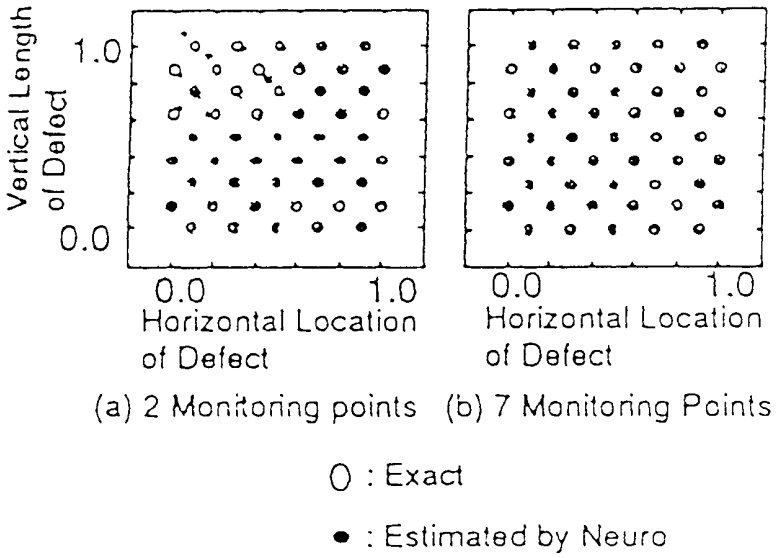


Figure 3.5. Estimation b y network for unlearned data

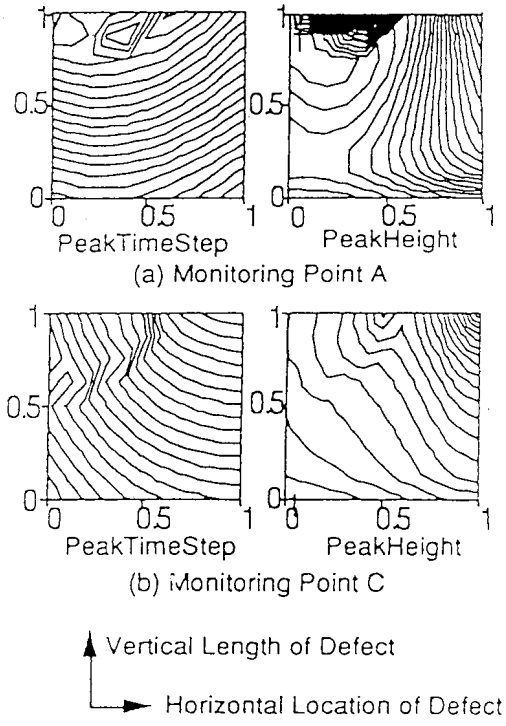


Figure 3.6. Distributions of c haracteristic values

3.4 Defect Identification Based on Electric Potential Drop Method [55]

3.4.1 Procedure of inverse analysis

The procedure described in Section 3.2 is straightforwardly applied to the defect identification problem using the electric potential drop method. In the electric potential drop method, the crack parameters such as the size and the location of a crack are determined from the electric potential values measured at various points. This is one of the inverse problems. On the other hand, if the crack parameters are given, an electric potential distribution can be easily computed using computational mechanics simulations. This is a direct problem. Simply regarding these direct and inverse problems to be the nonlinear mapping problems from the multiple data, i.e. the electric potential values, to the multiple data, i.e. the crack sizes and the locations, the same inverse analysis procedure as described in Section 3.2 can be applied. The procedure consists of the following three phases:

Phase 1: Sample data of crack parameters, i.e. crack size, location and rotational angle, vs. electric potential values are computed through computational mechanics simulations for many combinations of the crack parameters. Each piecewise relation between the crack parameters and the electric potential values is called here "learned pattern".

Phase 2: An error-back-propagation neural network is trained using a number of learned patterns. Here, the crack parameters are given to the network as the teaching data, while the electric potential values are given to it as the input data.

Phase 3: The trained network can identify crack parameters from measured electric potential values.

3.4.2 Problem definitions

2D crack and 3D crack problems are considered. Figure 3.7 shows a plate with a through-wall crack (2D crack problem) whose top and bottom edges are subjected to the fixed potential values of +V and -V, respectively. By measuring the electric potential values around the crack, the following four crack parameters are identified: X, Y (crack location), L (a half of crack length), θ (rotational angle). Figure 3.8 shows a 3D semi-elliptical surface crack in plate. Electric potential values are measured at 11×11 (= 121) grid points on the backside of the plate. The aspect ratio of the surface crack L/D is fixed to be 2.0. Then, the four crack parameters of (X, Y, L, θ) are identified.

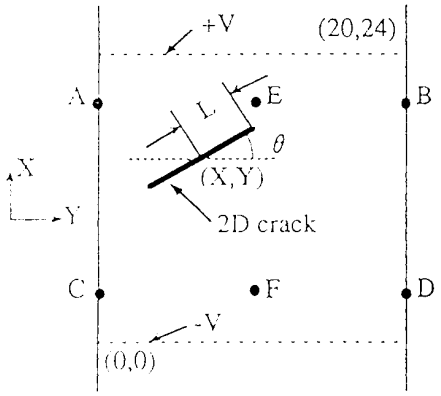


Figure 3.7. Plate with a through-wall crack subjected to prescribed electric potential conditions

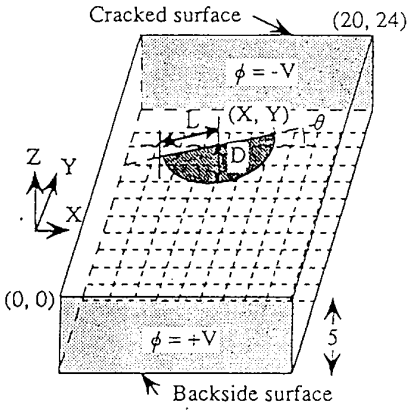


Figure 3.8. Plate with a semi-elliptic surface crack subject to prescribed electric potential conditions

3.4.3 2D crack identification

6-point measurement case

At first, the four crack parameters (X, Y, L, θ) are identified by using the electric potential values measured at six points of A to F as shown in Figure 3.7. Table 3.2 shows a range of each parameter and the number of its samples chosen for the network training. For example, the parameter X is varied from 6 to 10, and 3 samples, that is, 6, 8 and 10 are chosen from the range. As a result, 98 learned patterns are utilized to train the network. The neural network used is of three-layered type, and its input and output layers have six and four units, respectively.

Parameter	Range	No. of samples
X	$6.0 \sim 10.0$	3
Y	$12.0 \sim 16.0$	3
L	$2.0 \sim 5.0$	3
θ(deg)	$0.0 \sim 45.0$	4

Table 3.2. Number of samples and ranges of crack parameters chosen for network training

Table 3.3 shows the mean errors of estimation for the learned patterns and the unlearned patterns when the network with 30 units in its hidden layer is trained through 30,000 iterations. The definition of the mean error of estimation is as follows:

$$\text{Mean Error} = \frac{1}{r}\sum_{p=1}^{r}\left|O^p_{\text{neuro}} - O^p_{\text{true}}\right| \tag{3.1}$$

where O^p_{neuro} is the output of the network for the p-th pattern, O^p_{true} is a true value for the p-th pattern and r is the number of learned patterns or that of unlearned patterns. The input and output data of the neural network are normalized into a unit range of (0.0 - 1.0). The estimation errors for the unlearned patterns are examined to check a feature of generalization of the network. Figure 3.9 shows some comparisons between the actual crack shapes and the estimated ones. Table 3.3 shows that the estimation errors in X, Y (crack location) and L (a half of crack length) are smaller than that in θ (rotational angle).

However, it can be confirmed from Figure 3.9 that the trained network can estimate the crack shapes with sufficient accuracy. It should be noted that the estimation accuracy is different parameter by parameter.

Parameter	Mean error of estimation	
	Learned patterns	Unlearned patterns
X	0.044	0.035
Y	0.017	0.028
L	0.010	0.040
θ	0.043	0.082

Table 3.3. Mean errors of estimation (after 30,000 iterations) (6-point measurement, 30 units in a hidden layer)

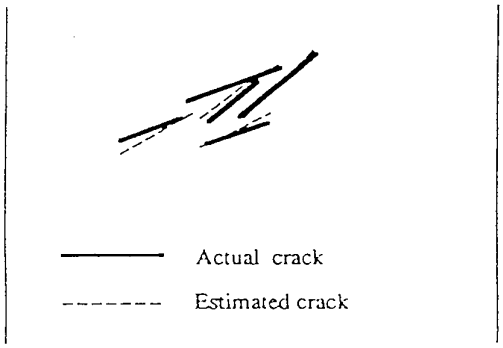


Figure 3.9. Comparison between actual and estimated cracks (2D crack)

Table 3.4 shows the mean errors of estimation for the learned patterns when four kinds of networks with 20, 30, 40 and 50 units in a hidden layer are trained through 30,000 iterations. When the number of units in a hidden layer increases from 20 to 30, estimation accuracy is improved for all the parameters. When increasing from 30 to 40 or 50, this is not true. It should be also noted here that each parameter might have the different optimal number of units in a hidden layer.

Parameter	Mean error of estimation for learned patterns			
	Units of hidden layer			
	20	30	40	50
X	0.083	0.044	0.026	0.033
Y	0.038	0.017	0.013	0.015
L	0.018	0.010	0.007	0.007
θ	0.116	0.043	0.060	0.071

Table 3.4. Effects of the number of units in a hidden layer to estimation accuracy (after 30,000 iterations) (2D crack)

40-point measurement case

Next, the four crack parameters are identified from the electric potential values measured at 40 points along the lines AB, BD, DC and CA (see Figure 3.7). The number of samples and ranges of the four parameters employed for the network training are the same as in Table 3.2. Units in a hidden layer are set to be either 20 or 30. Table 3.5 shows the mean errors of estimation for the learned patterns after training the network through 4,000 iterations. For the purpose of comparison, the table also shows the estimation errors in the case of 6-point measurement. The table clearly shows that estimation accuracy is improved by increasing the number of electric potential values.

Parameter	Mean error of estimation for learned patterns		
	40-point measurement (Iterations = 4,000)		6-point measurement (Iterations = 30,000)
	Units of hidden layer		Units of hidden layer
	20	30	30
X	0.019	0.014	0.044
Y	0.013	0.010	0.017
L	0.004	0.009	0.010
θ	0.018	0.015	0.043

Table 3.5. Comparison of estimation accuracy between 6-point measurement and 40-point measurement (2D crack)

To compare the efficiency of the two cases, the following ratio is examined:

$$\text{Ratio of total numbers of renewals of } W \text{ and } \theta = \frac{\text{Network scale used for 40-point measurement} \times \text{No. of iterations}}{\text{Network scale used for 6-point measurement} \times \text{No. of iterations}} = 0.582 \qquad (3.2)$$

where the number of units in a hidden layer for both cases is 30. The "network scale" which is directly related to the CPU time consumed in one renewal of the network is defined as follows :

$$\begin{aligned} \text{Network scale} = {} & \text{No. of connections between output and hidden layers} \\ & + \text{No. of connections between hidden and input layers} \\ & + \text{No. of output units} \\ & + \text{No. of hidden units} \end{aligned} \qquad (3.3)$$

This result implies that the increase of electric potential values leads to the increase of the network scale, but the number of training iterations reduces, and eventually the training time reduces. This may be because the poseness of the present inverse problem is improved due to the increase of measurement points.

3.4.4 3D crack identification

The crack parameters for the learned patterns are chosen as the same as in the previous 2D crack problem, i.e. Table 3.2. Table 3.6 shows the estimation errors for the four crack parameters when using the four-layered network with the linear activation function. Figure 3.10 shows some comparisons between the actual crack shapes and the estimated shapes. Compared with the results of 2D crack (see Table 3.5), the estimation errors increase in the present 3D problem. Nevertheless, the accuracy of estimation is satisfactory in an engineering sense.

Parameter	Mean error of estimation	
	Learned patterns	Unlearned patterns
X	0.033	0.036
Y	0.021	0.081
L	0.011	0.046
θ	0.031	0.039

Table 3.6. Mean errors of estimation in 3D crack identification when four-layered network is utilized (after 2,000 iterations)

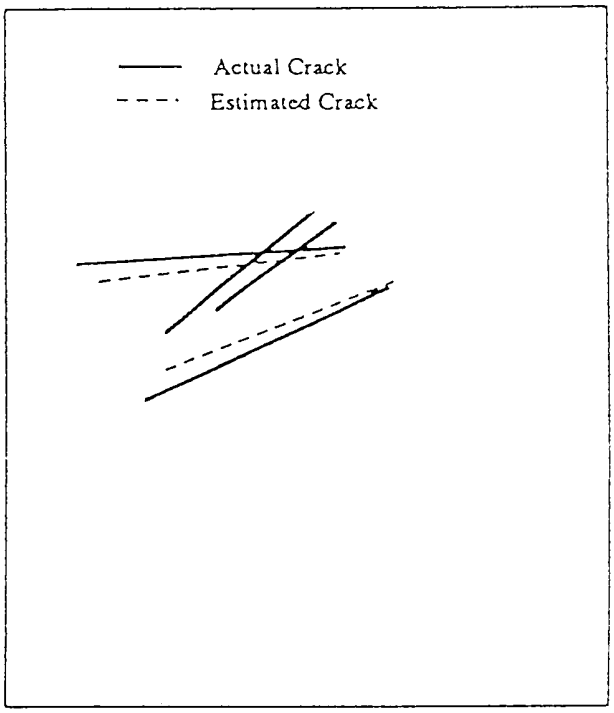


Figure 3.10. Comparison between actual and estimated cracks (3D crack)

4. NEURAL-NETWORK-BASED INVERSE ANALYSIS FOR STRUCTURAL IDENTIFICATION [40]

4.1 Structural Identification

Structural identification is defined here as a process to build a mathematical model, which describes the vibration characteristics of real mechanical and structural components. Various structural identification methods have been proposed so far, which are roughly classified into three categories; (a) the theoretical method, (b) the experimental method and (c) the hybrid theoretical and experimental method. However, none of them have been satisfactory yet. The inverse analysis approach based on the hierarchical neural network and the computational mechanics is applied to the structural identification of a vibrating plate with non-uniform thickness.

4.2 Procedure of Inverse Analysis

The vibration characteristics such as the eigen frequencies and the eigen vectors can be easily computed for given structural parameters such as the plate thickness and the material properties through the computational mechanics simulations as a direct problem. On the other hand, its inverse problem is to determine the structural parameters from the vibration characteristics. The nonlinear mapping problems from the multiple data, i.e. the vibration characteristics, to the multiple data, i.e. the plate thickness and the material properties are treated by the following inverse analysis approach using the hierarchical neural network. Three phases of the approach are as follows (see also Figure 3.1):

Phase 1: Sample data of the structural parameters, i.e. the plate thickness vs. vibration characteristics are computed by the finite element simulations for many combinations of the structural parameters.

Phase 2: An error-back-propagation neural network is trained using a number of learning data, i.e. relations between a set of the structural parameters and a set of the vibration characteristics. The structural parameters are given to the network as the teaching data, while the vibration characteristics are given to it as the input data.

Phase 3: The trained network can identify a set of structural parameters even for a set of unlearned vibration characteristics.

4.3 Structural Identification of Vibrating Plate

4.3.1 Problem definition

A plate with non-uniform thickness, which consists of nine equal subsections as shown in Figure 4.1, is considered. The thickness of each subsection may vary ranging from 5 to 20 mm. Material properties of the plate are the Young's modulus = 68.6 GPa, the Poisson's ratio = 0.33 and the mass density = 27 kg/cm^2. The problem is to determine the plate thickness of each subsection from the vibration characteristics of the plate, i.e. eleven eigen frequencies and eigen modes.

4.3.2 Preparation of learning data

The eigen analyses of the plate are first performed by varying the plate thicknesses of the nine subsections as shown in Figure 4.2. Since the plate thickness of each subsection is taken to be either 5 or 20 mm, the complete combinations of $2^9 = 512$ are analyzed. These are direct analyses. The finite element mesh subdivisions are shown in Figure 4.1. The out-of-plane vibration of the plate is analyzed using the six-noded triangular elements.

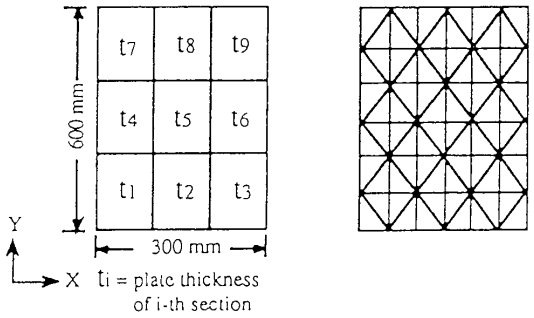


Figure 4.1. Non-uniform plate in thic kness and its finite elemen t model

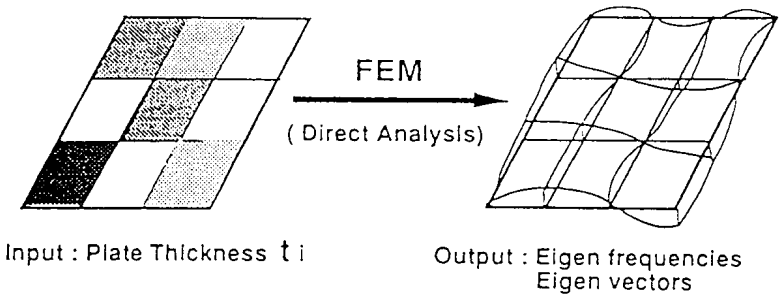


Figure 4.2. Schematic view of direct eigen analyses to prepare learning data

4.3.3 Learning process

Figure 4.3 sho ws the sc hematic view of the emplo yed hierarchical neural net work and its input and output data. The net w ork is a three-layered one, whic h has 9 units in the output layer, 30 units in the hidden la yer and 319 units in the input la yer. The nine units in the output layer output the plate thicknesses of the 9 subsections, $t_1 \sim t_9$. The 1st through 11-th eigen frequencies are given to 11 units in the input la yer. Normalized displacement values at 28 nodes of the 1st through 11-th modes are also giv en to 308 (= 28× 11) units in the input layer. The eigen frequencies and the displacement values are normalized ranging from 0.0 to 1.0.

The neural net w ork is trained until 10,000 iterations at whic h the mean error w as judged to be sufficien tly small. It is confirmed that all the 512 com binations used for the net work training are w ell estimated.

4.3.4 Estimation of plate thickness

The trained net w ork is used to estimate some non-uniform plates with either 5 or 10 mm thickness of subsection. Figure 4.4 sho ws the comparison bet w een the actual distribution of plate thickness and the estimated one. Figure 4.5 shows the comparison betw een the actual eigen frequencies and the estimated ones of the 1st through 11-th modes. Figures 4.6 and 4.7 show the comparisons betw een the actual 2nd eigen v ector and the estimated one and that of the 8-th eigen vector, respectively. These figures sho w that the presen t neural net w ork approach can be applied to this kind of structural iden tification problems. Although some difference is found in the 8-th eigen frequency in Figure 4.5, the actual and the estimated 8-th eigen vectors agree w ell.

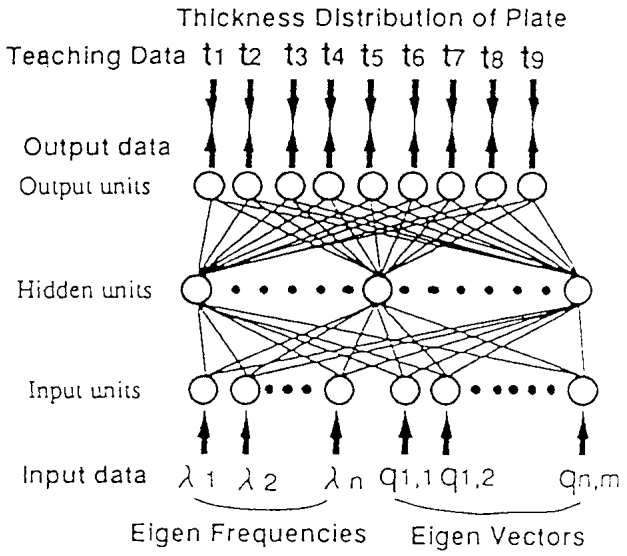


Figure 4.3. Neural network employed and its input and output data

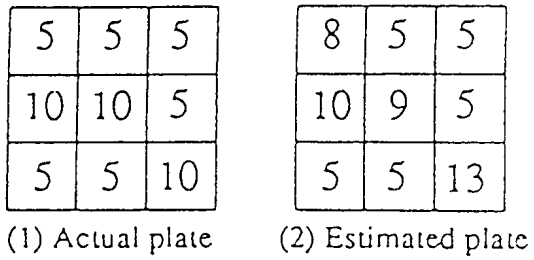


Figure 4.4. Comparison of actual and estimated distributions of plate thickness

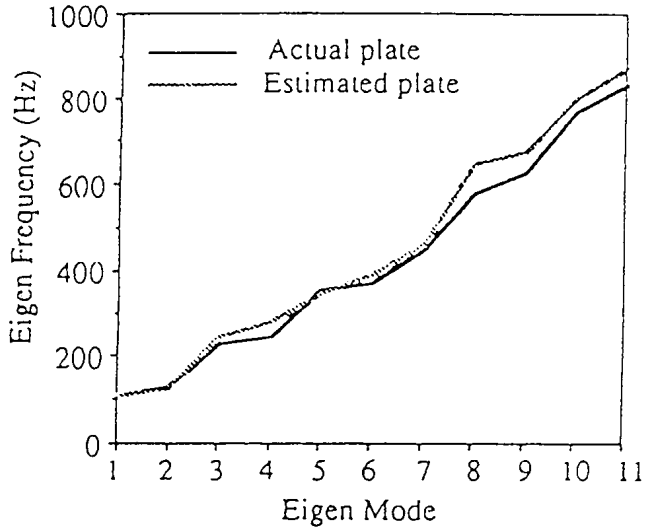


Figure 4.5. Comparison of actual and estimated eigen frequencies of 1st to 11-th modes

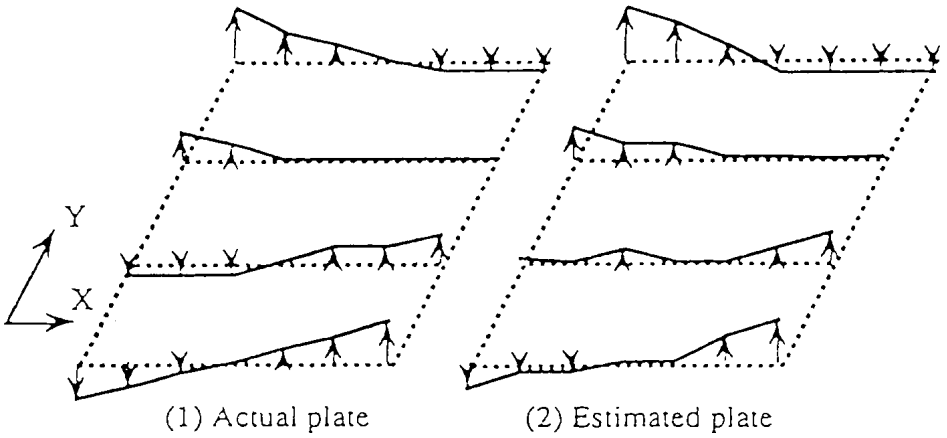


Figure 4.6. Comparison of actual and estimated 2nd eigen v ector

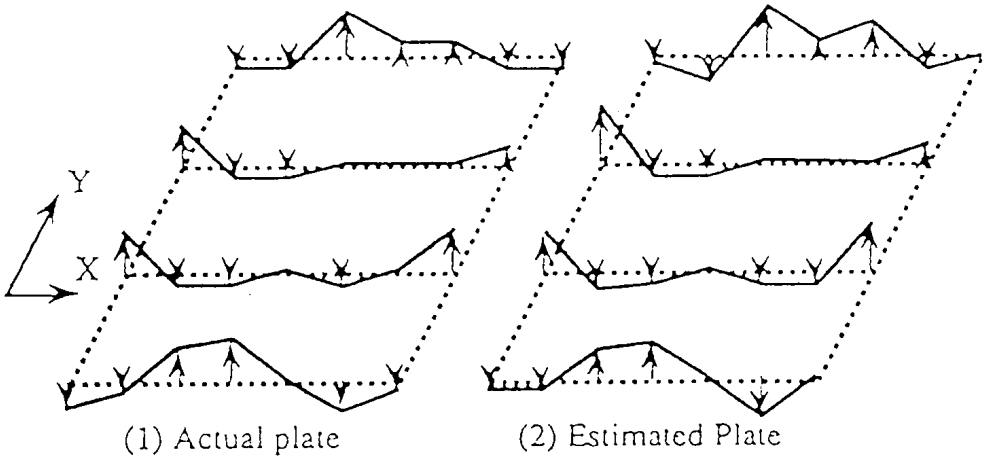


Figure 4.7. Comparison of actual and estimated 8-th eigen v ector

The same net work is used to estimate some non-uniform plates with either 5, 10 or 20 mm thickness of subsection. All the com binations are $3^9 = 19,683$, among which 50 unlearned com binations are examined. Among them, about 50 % of the combinations show satisfactory agreement betw een the true and estimated eigen frequencies and eigen v ectors. It is noted here that the 512 com binations used for the net work training are only about 2.5 % of all the possible combinations of 19,683. Then, it can be said that the present inverse analysis approach possesses a high potential in practical structural identification problems.

In the vibration analyses of practical and complex structures, it is required to have good solution accuracy of a global model ev en with coarser mesh subdivisions in local areas. In such a situation, a key issue is a simplification process of an analysis model. As is described above, the neural net work learned a num ber of vibration characteristics of non-uniform plates in thickness, in whic h nine equal subsections may have different plate thicknesses. If vibration characteristics of a more complex plate are given to the trained net work above, it ma y tell us a nine-subsection plate whic h well approximates those vibration c haracteristics.

5. NEURAL-NETWORK-BASED MODELING OF VISCOPLASTIC MATERIAL BEHAVIORS [45],[46]

5.1 Viscoplastic Constitutive Properties of Materials

In various industrial fields, structural materials are often used under severe operating conditions such as cyclic loading, high temperature, high pressure and high irradiation. For the reliable evaluation of deformation behaviors of these materials, thermo-inelastic analyses are inevitable.

Various theoretical models to describe a wide range of viscoplastic behaviors of metallic materials have been proposed and studied by many researchers. The viscoplastic constitutive equations derived from these theories involve many parameters, which greatly influence behaviors of the constitutive equations. Therefore, these parameters need to be determined appropriately referring real behaviors of materials, i.e. cyclic loading tests, stress relaxation tests, and so on. Every constitutive equation has its own method for the parameter determination, e.g. the algebraic methods, the least-square methods, and so on. However, these methods are not so good at optimizing the parameters referring various material behaviors simultaneously. This situation makes it difficult to fairly compare different inelastic constitutive equations.

This chapter describes two types of neuro approaches to describe a wide range of viscoplastic behaviors of materials. The one is a method for optimizing multiple parameters of a viscoplastic constitutive equation (Section 5.2), and the other is a method for directly modeling the material behaviors without using a constitutive equation (Section 5.3).

5.2 Optimizing Multiple Parameters of Viscoplastic Constitutive Equation [46]

5.2.1 Problem definition

A method for optimizing multiple parameters of a viscoplastic constitutive equation using the hierarchical neural network is described. To simplify the explanation of the present method without any loss of generality, the Chaboche's viscoplastic theory [90] in uniaxial loading and stationary temperature conditions is considered. It should be noted here that the present method is applicable to any viscoplastic model. The Chaboche's viscoplastic model is capable of describing the cyclic hardening and softening behaviors with the yielding surface. Its formulation under uniaxial loading and stationary temperature conditions is given as follows:

$$F(\sigma, Y, R) = |\sigma - Y| - R \tag{5.1a}$$

$$\varepsilon^p = \Lambda(\sigma - Y)/|\sigma - Y| \tag{5.1b}$$

$$\Lambda = (F/K)^n \qquad F > 0$$

$$\Lambda = 0 \qquad F \leq 0 \tag{5.1c}$$

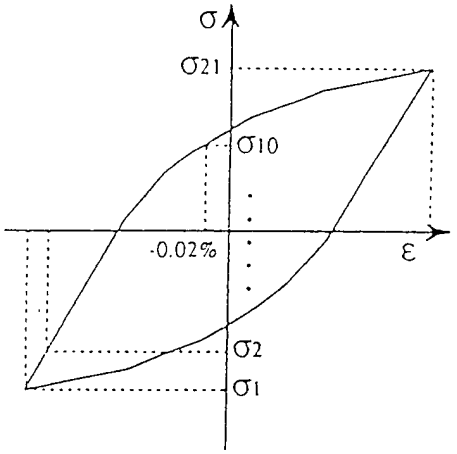
$$Y = H\varepsilon^p - DY|\varepsilon^p| \tag{5.1d}$$

$$R = (h - dR)|\varepsilon^p| \tag{5.1e}$$

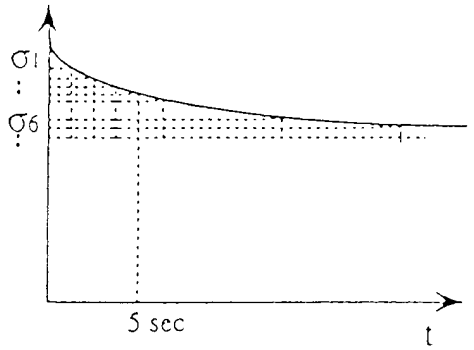
where the state variables σ, ε^p, Y and R are the uniaxial stress, the uniaxial inelastic strain, the uniaxial back stress and the isotropic hardening variable, respectively. The initial value

of R is taken to be k. Equation (5.1a) is the yield function, Eqs.(5.1b) and (5.1c) the flow law, Eq.(5.1d) the cyclic hardening behavior of material, and Eq.(5.1e) the cyclic softening behavior, respectively. Equation (5.1) involves seven parameters, i.e. K, n, H, D, h, d and k, which must be determined so that Eq.(5.1) may approximate well real viscoplastic behaviors of materials. It should be noted here that all the parameters are assumed to be temperature-independent although they aren't in a general case.

The basic principle of the present method is as follows. Utilizing the simulation program of the Chaboche's viscoplastic model, i.e. Eq.(5.1), the material behaviors for a given parameter set of (K, n, H, D, h, d and k) are computed. Then, by changing the combination of the seven parameters, various behaviors of materials under the cyclic loading or creep conditions are described. Figures 5.1(a) and 5.1(b) schematically show the cyclic hysteresis curve and the stress relaxation curve, respectively. The obtained relationship between the parameter set and the calculated material behaviors are called here the "learning pattern". The neural network is trained to learn a number of learning patterns. In the training process, the parameter set is given to the network as teaching data, while the material behaviors are given to it as input values as shown in Figure 5.2. Here each material behavior is discretized into piecewise stress values as shown in Figure 5.1. If real material behaviors such as cyclic hysteresis curve and stress relaxation curve are given to the trained network as input data, the network outputs the corresponding parameter set.



(a) Schematic view of cyclic hysteresis curve



(b) Schematic view of stress relaxation curve

Figure 5.1.

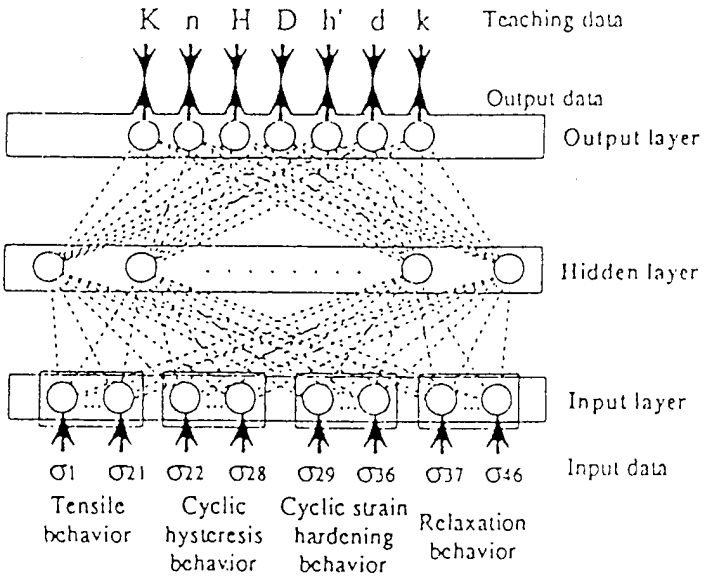


Figure 5.2. Arc hitecture of the neural network used and the input/output data for parameter optimization

The performance of the present method is demonstrated in the parameter optimization of the Chaboc he's viscoplastic model under uniaxial loading and stationary temperature conditions. Experimen tal data referred are cyclic hysteresis curves of about 60 cycles and a stress relaxation curve.

5.2.2 Preparation of learning data

Each material beha vior is discretized into piecewise stress values as shown in Figure 5.1. The input data to the net work are as follows:

1) Stress values at every strain interval of 0.01 % from the yielding point on the tensile curve (8 values).
2) Stress values at every strain interval of 0.02 % from the minimum h ysteresis tip to the maximum h ysteresis tip on the 60-th cyclic curve (21 values).
3) Maximum stress values at the hysteresis tip of the 1st, 2nd, 3rd, 6-th, 10-th, 20-th, 30-th and 60-th cycles on the cyclic strain hardening curve (7 values). Noted that the value of the 60-th cycle is already taken into account in (2).
4) Stress values at 0, 1, 2, 3, 4, 5, 10, 20, 30 and 60 seconds in the stress relaxation curve (10 values).

The original values of the input data are sometimes not so sensitive to the variation of parameters to be optimized. This situation ma y result in less accuracy of parameter optimization. To improve accuracy of the method, the original v alues of the input data are modified so as to be more sensitive to the variation of the parameters. As the input data, taken are the deviation of the original values from those corresponding to the parameter set of ($K = 1.00 \times 10^2$, $n = 3.5$, $H = 1.50 \times 10^5$, $D = 1.50 \times 10^3$, $h' = 1.20 \times 10^2$, $d = 3.5$, $k = 8.00 \times 10$), which is the the center point of the training ranges of parameters. The values of the input data are transformed in to the range from -0.5 to +0.5.

Table 5.1 shows the training range and the number of sampling points of the seven parameters. These ranges are roughly determined based on a conventional curve fitting method. In the table, h' is defined to be $(h/d - k)$ for the purpose of convenience of training. Each training range is set to include possible parameter values.

Parameter	K	n	H	D
Range	0.5～1.5 $\times 10^2$	1.0～6.0	0.5～2.5 $\times 10^5$	0.5～2.5 $\times 10^3$
Samples	7	7	7	7
Parameter	h' (= h/d-k)	d	k	
Range	0.7～1.7 $\times 10^2$	1.0～6.0	0.4～1.2 $\times 10^2$	
Samples	7	7	7	

Table 5.1. Parameter ranges and sampling numbers for network training

Sampling points are usually chosen inside the training ranges at the regular interval. If sampling points are not chosen properly, the neural network would not learn well throughout the training ranges. The more sampling points are chosen, the more accurate results can be obtained, while the longer training time is required. Here are chosen seven sampling points in each training range. The number of all the combinations of the seven parameters with seven sampling points becomes $7^7 = 823,544$. This is too many to prepare learning patterns and to train the network. To shorten these processes, 98 combinations are carefully selected considering the close relations between some parameters, that is, K and n are related to the flow law, H and D are related to the kinematic hardening behavior, and d and h are related to the isotropic hardening. In general, it would take much calculation time for the preparation of the learning patterns for all the combinations of the seven parameters. However, one can reduce significantly the effort on the preparation process because of the feature of generalization of the neural network.

5.2.3 Estimation of parameters in the Chaboche's model

Four cases are examined, in which different material behaviors are referred to optimize the seven parameters of the Chaboche's viscoplastic model (see Table 5.2). For example, Case 1 refers only the cyclic hysteresis behavior, while Case 4 does four kinds of material behaviors.

Table 5.3 tabulates the obtained parameter sets corresponding to Cases 1 - 4. Table 5.4 summarizes the configuration of the network used and the training conditions. In the experiment, the following four kinds of material behaviors are obtained:

1) Tensile behavior
2) Cyclic hysteresis behavior
3) Cyclic strain hardening behavior
4) Stress relaxation behavior

These data are obtained from a cyclic loading test and the subsequent strain-holding test under uniaxial loading and stationary temperature conditions as shown in Figure 5.3. Temperature is set to be 200 °C. In the cyclic test, strain range and strain rate are fixed to be 0.4 % and 2.0×10^{-4} sec^{-1}, respectively, while strain is maintained at 0.2 % in the

Behavior / Case	Tensile	Cyclic hysteresis	Cyclic strain hardening	Stress Relaxation
1	-	○	-	-
2	-	○	○	-
3	○	○	○	-
4	○	○	○	○

Table 5.2. Material beha viors referred for parameter optimization

Parameter / Case	K ($\times 10^{2}$)	n	H ($\times 10^{5}$)	D ($\times 10^{3}$)	h' $(h/d \cdot k)_2$ ($\times 10^{2}$)	d	k ($\times 10^{2}$)
1	1.18	4.61	1.18	1.27	2.60	1.97	1.03
2	1.13	4.63	1.35	1.34	0.94	1.67	1.08
3	0.94	4.18	1.51	1.72	1.52	1.29	1.20
4	1.02	3.76	1.34	1.39	1.85	1.38	1.22

Table 5.3. Optimized parameter sets

Case	No. of learning patterns	No. of units				No. of learning iterations
		Input layer	Hidden layer	Output layer	Total	
1	98	21	60	7	88	4,800
2	98	28	60	7	95	6,200
3	98	36	60	7	103	8,300
4	98	46	60	7	107	12,000

Note 1: The same sigmoid function was emplo yed in all cases
Note 2: Criterion for net work training was: the error$<$ 0.1%.

Table 5.4. Conditions of network training

strain-holding test. The num ber of cycles is above 60, and the hold time is o ver one hour. These conditions are c hosen to achieve saturation of strain hardening and stress relaxation behaviors. These tests are performed on three specimens made of Type 316L stainless steel.

Figure 5.4(a) shows the comparison of experimental and estimated curves of cyclic hysteresis behaviors of Case 1, while Figure 5.4(b) does that of Case 4. It is clearly shown from the figures that Case 1 and Case 4 give almost similar estimation curves as to the cyclic hysteresis curve. Figures 5.5, 5.6 and 5.7 show the similar comparison of tensile curves, that of cyclic strain hardening curves, and that of stress relaxation curves, respectively. It can be observed that Case 4 gives much better estimation for these three behaviors than Case 1. Superior results of Case 4 are due to simultaneous reference of the four kinds of material behaviors.

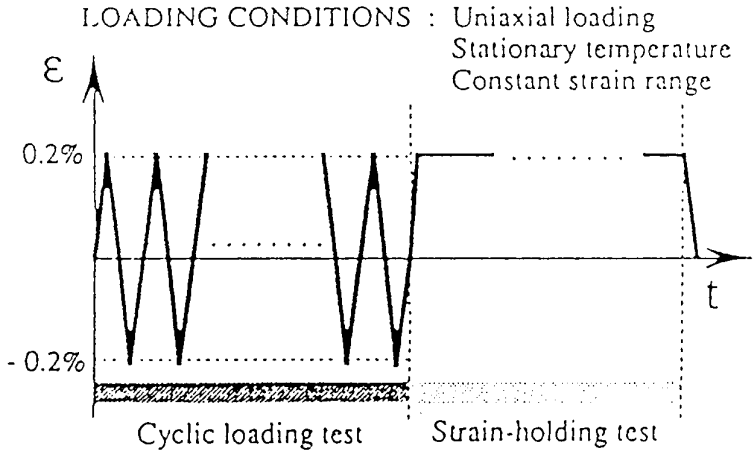


Figure 5.3. Schematic loading history in cyclic loading test and subsequent strain-holding test

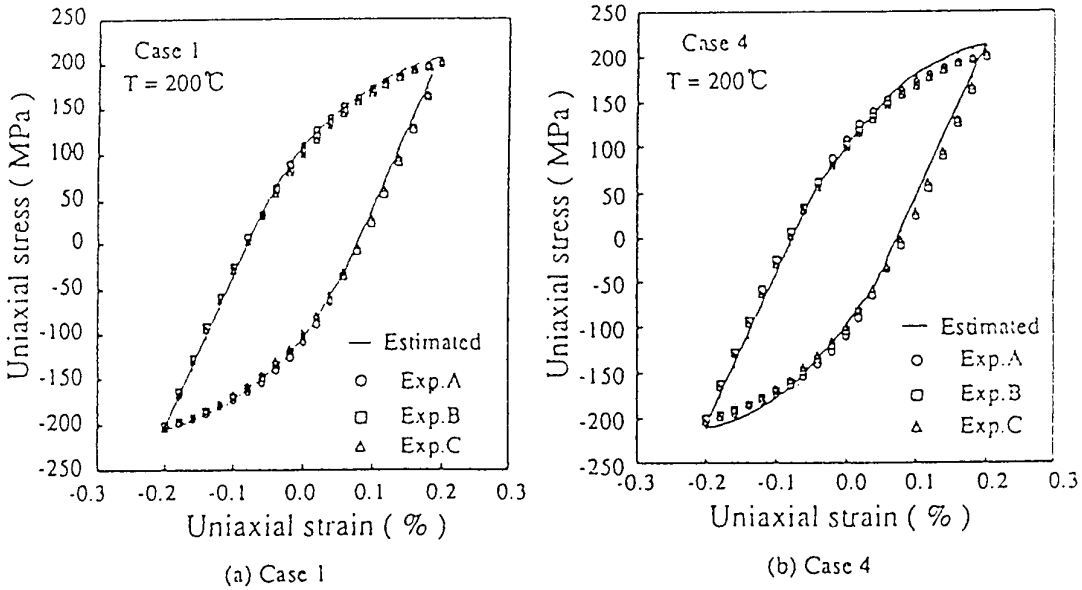


Figure 5.4. Comparison between experimental and estimated cyclic hysteresis behaviours

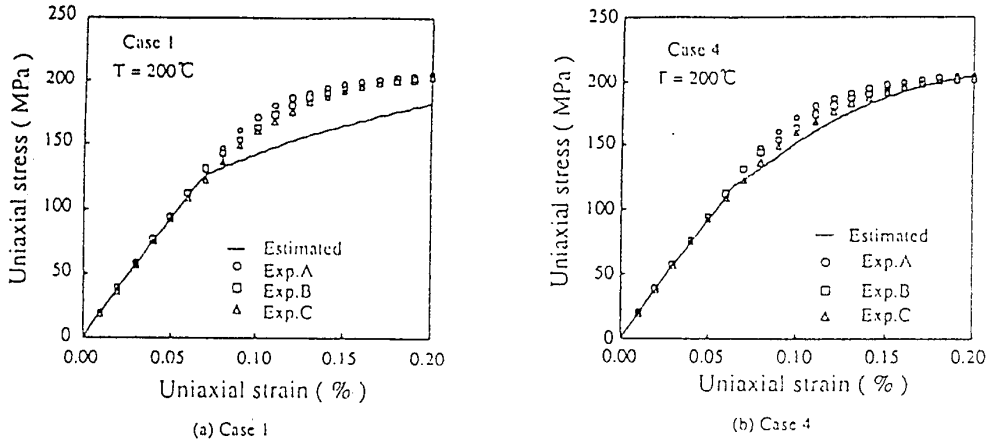


Figure 5.5. Comparison bet ween experimen tal and estimated tensile beha viours

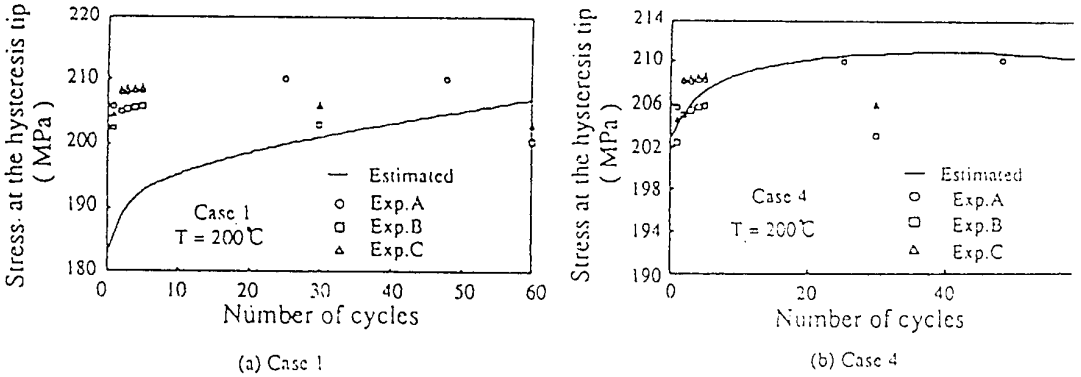


Figure 5.6. Comparison bet ween experimen tal and estimated strain hardening beha viours

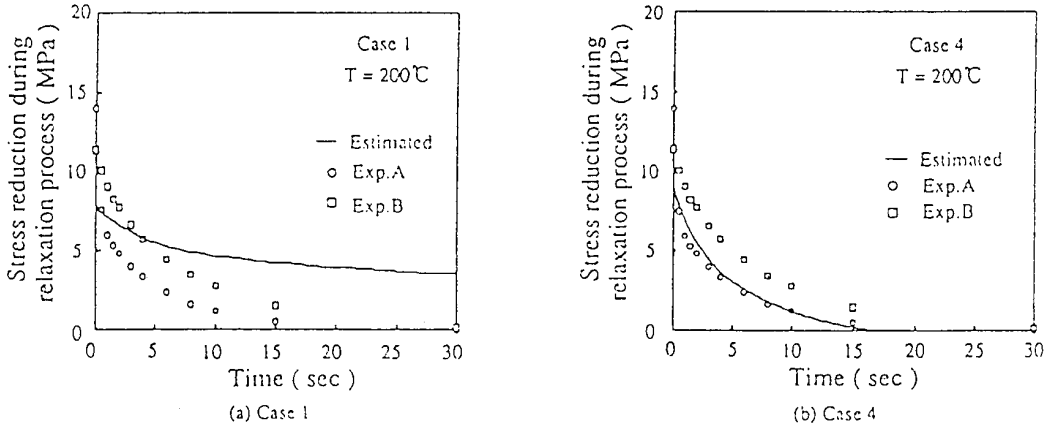


Figure 5.7. Comparison bet ween experimen tal and estimated relaxation beha viours

5.3 Neural Network Modeling of Constitutive Properties [45]

5.3.1 Problem definition

In the previous section, a method for optimizing multiple parameters included in a viscoplastic constitutive equation using the hierarchical neural network was described. This method can be applied to various types of inelastic constitutive equations. However, since the stress-strain response is much affected by the loading history, the constitutive equations still include some limitations.

Here is described another approach, that is, the neural networks are used directly to model the material behaviors in place of the constitutive equations, which does not require to determine the constants explicitly. Approaches to use neural networks in place of the constitutive equations have also been reported by some researchers. Ghaboussi *et al.*, for example, attempted to mimic the whole cyclic material behavior by one neural network. The estimation ability of the neural networks were however not sufficiently accurate. Yamamoto, on the other hand, used the neural network and Ramberg-Osgood model to describe material hysteresis curves. In this approach, the actual material behaviors may not be however expressed because the Ramberg-Osgood curve inherently contains the model error.

In the present approach, for describing the material behavior using the neural network, the idea of the Chaboche's internal stress concept is employed in such a way that the stress is expressed as the sum of the internal stresses, i.e. the back stress and the yield stress. Their behaviors are taught into different networks separately. By doing so, the transient behavior from the initial state to the steady cyclic condition can be modeled better than referring to the observable variables (e.g. total strain).

5.3.2 Learning process

According to the theory of isotropic hardening under the uniaxial loading, fixed temperature conditions and stationary inelastic strain speed, inelastic material behaviors can be represented by the sum of the following two stress curves:

$$\sigma = Y(\varepsilon^p) + R(\int |\delta\varepsilon^p|) \tag{5.2}$$

where ε^p, Y and R are inelastic strain, the back stress and the yield stress, respectively. The stresses Y and R correspond to the strain after the material has yielded and the yield surface as a function of the inelastic strain history, respectively.

The neuro-emulator consists of two neural networks, each of which is used to learn the back stress or the yield stress. The techniques for determining the back and yield stresses from the experimental stress-strain data are schematically depicted in Figure 5.8. Figure 5.9 illustrates the basic strategy for each neural network to learn the stress curves, which are decomposed from the original stress-strain curve. In the present approach, three historical points are used to estimate the extrapolative gradient of the curve. Both the neural networks have the identical architecture. Figure 5.10 shows the structure of the neural networks for modeling the above inelastic laws. As is depicted in the figure, there is one hidden layer composed of 14 units between the input and output layers. The numbers of units in the input layer and in the output layer are six and unity, respectively. Table 5.5 lists the input and output variables used in the network. The input variables to the network are $(u_{n-2},\ u_{n-1},\ u_n,\ \nu_{n-2},\ \nu_{n-1},\ \nu_n)$, while the output data is $d\nu/du$. Here v stands for either the back stress or the yield stress, while u indicate $\int |\delta\varepsilon^p|_{n-1}$ for yield stress curve and ε^p for back stress curve. Once the teaching process is completed, the history curve is successively predicted by giving the previous three points to the trained network. The computational burden for obtaining the whole curve is trivial.

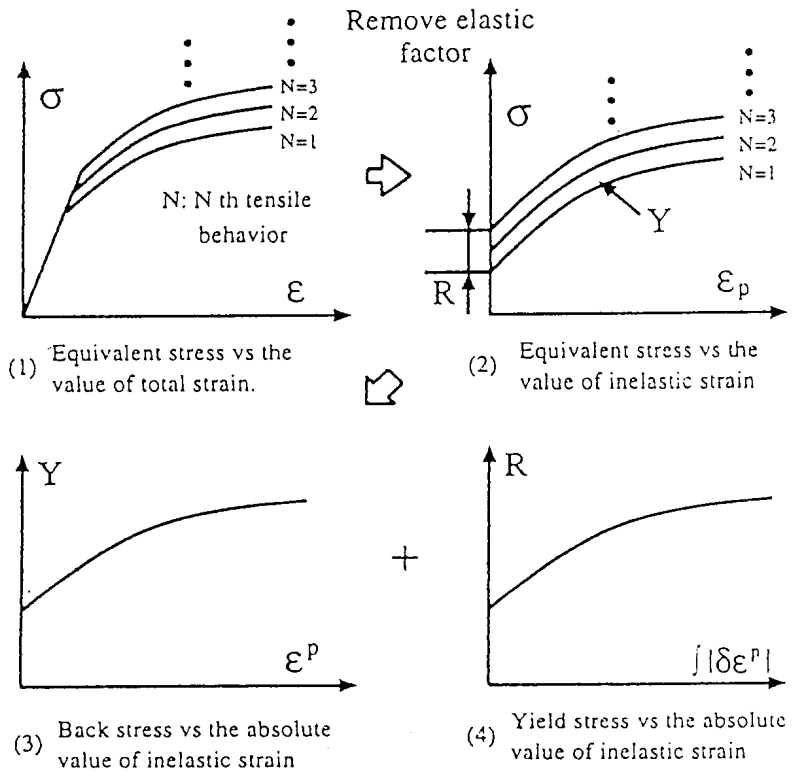


Figure 5.8. Data decomposition from experiment

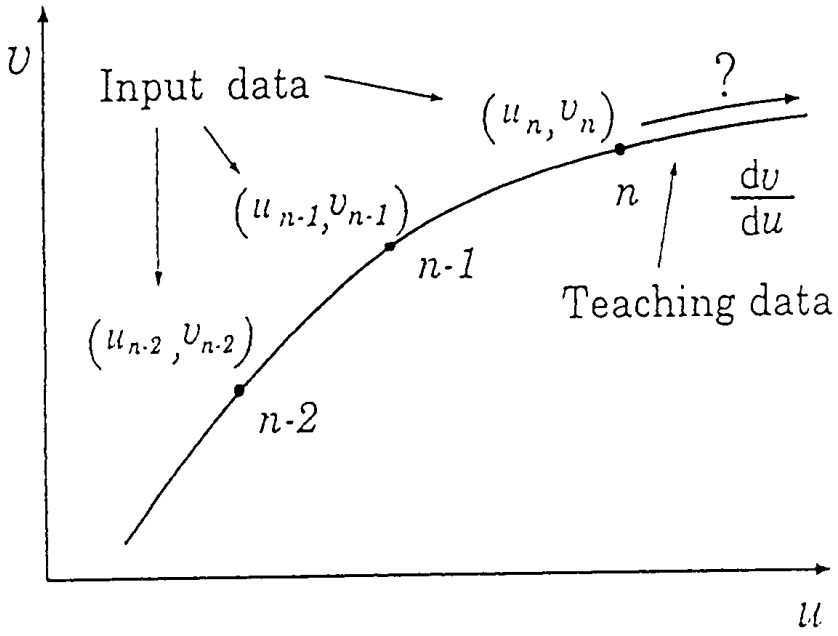


Figure 5.9. Sampling of stress beha viour

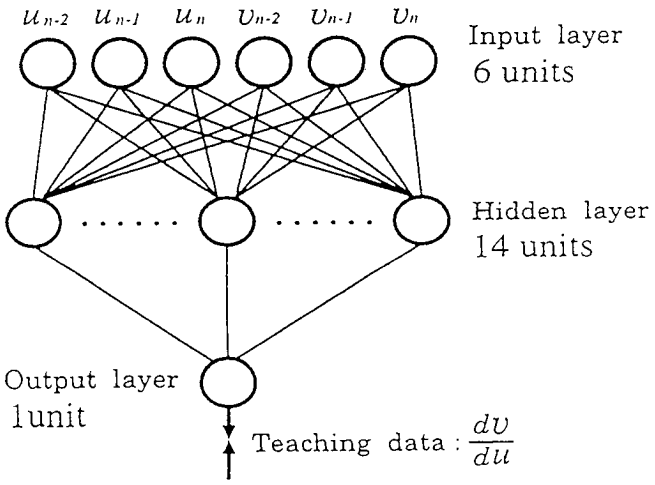


Figure 5.10. Structures of neural network

	Network 1	Network 2
u	ϵ^p	$\int \lvert\delta\epsilon^p\rvert$
v	Y	R

Table 5.5. Input variables used in the Network

5.3.3 Estimating stress-strain curves

The experimental data used to test the performance of the developed methods were obtained by the uniaxial cyclic test under the strain control. The specimen used in the experiment was 2 1/4Cr-1Mo at temperature 873 K. The stress-strain data of the test were referred to by "Bench Mark Project on Inelastic Deformation and Life Prediction of 2 1/4Cr-1Mo Steel at High Temperature (Subcommittee on Inelastic Analysis and Life Prediction of High Temperature Materials, The Society of Materials Science, Japan [91]). The experimental data are shown in Figure 5.11. These data were obtained within a strain range of 1.0 % under a constant strain rate of 0.5 %/s. The data are available at 1, 2, 5, 10 and 50 cyclic loops. Table 5.6 lists the parameters used for the neural networks. The teaching data of the back stress were converted to a logarithmic scale due to its characteristics. The training of each neural network was terminated when the mean square error of unlearned data began to increase.

	T	α	β
Back stress	0.8	0.001	0.001
Yield stress	0.8	0.001	0.001

Table 5.6. Parameters used for the Neural Networks

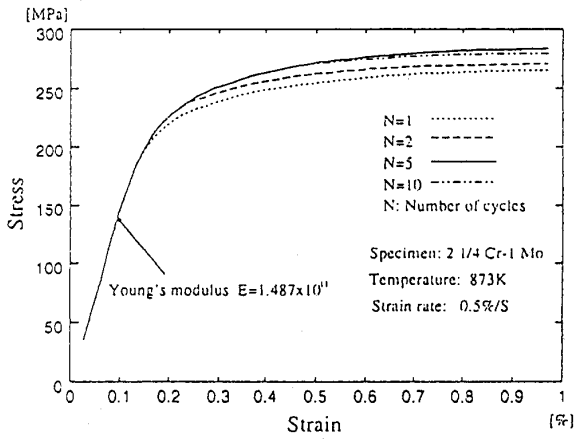
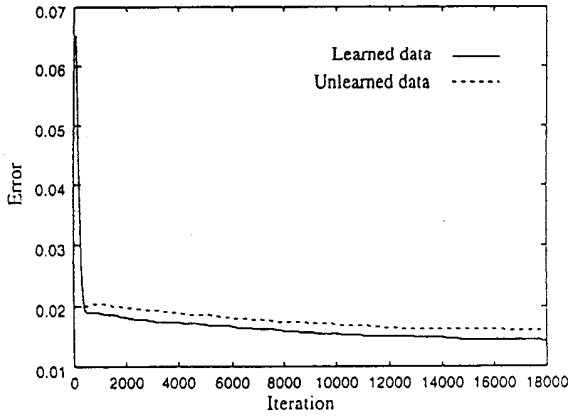
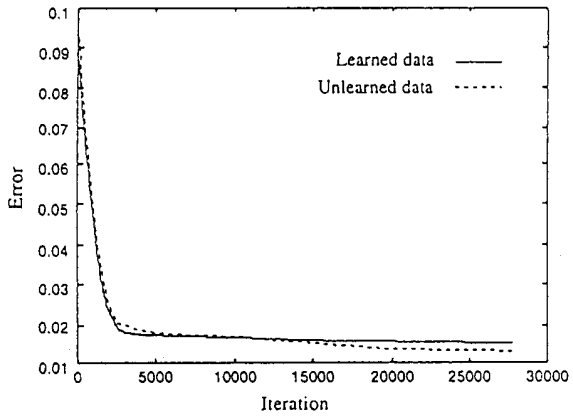


Figure 5.11. Experimen tal stress-strain data



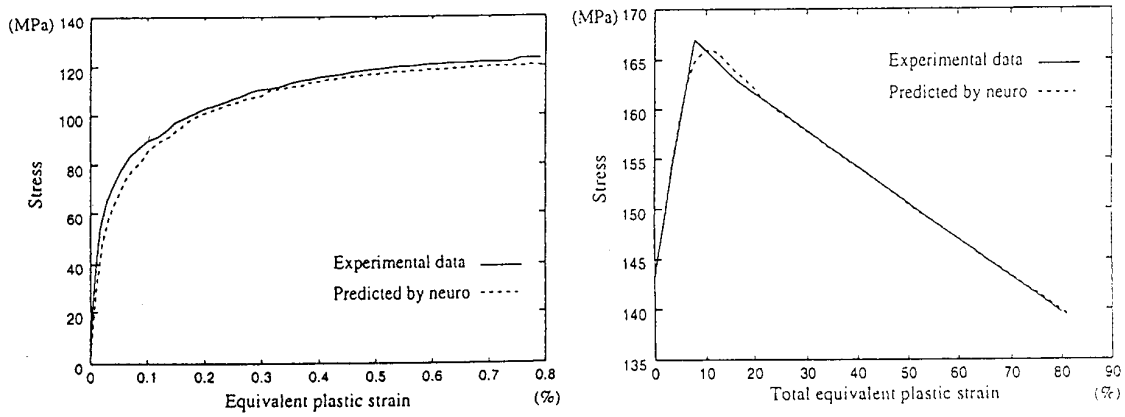
(a) Convergence behaviour of back-stress network



(b) Convergence behaviour of yield-stress network

Figure 5.12.

Figures 5.12(a) and 5.12(b) show the convergence behaviors of the neural networks for the back and yield stresses, respectively. The back and yield stresses, which were decomposed from the original stress-strain curve, were estimated by the trained networks. These results are shown in Figure 5.13. Figure 5.14 shows the cyclic hysteresis behavior retrieved from Eq.(5.2). The present neuro emulator is also implemented in an available FEM code as its user subroutine as shown in Figure 5.15.



(a) Back-stress curve

(b) Yield-stress curve

Figure 5.13.

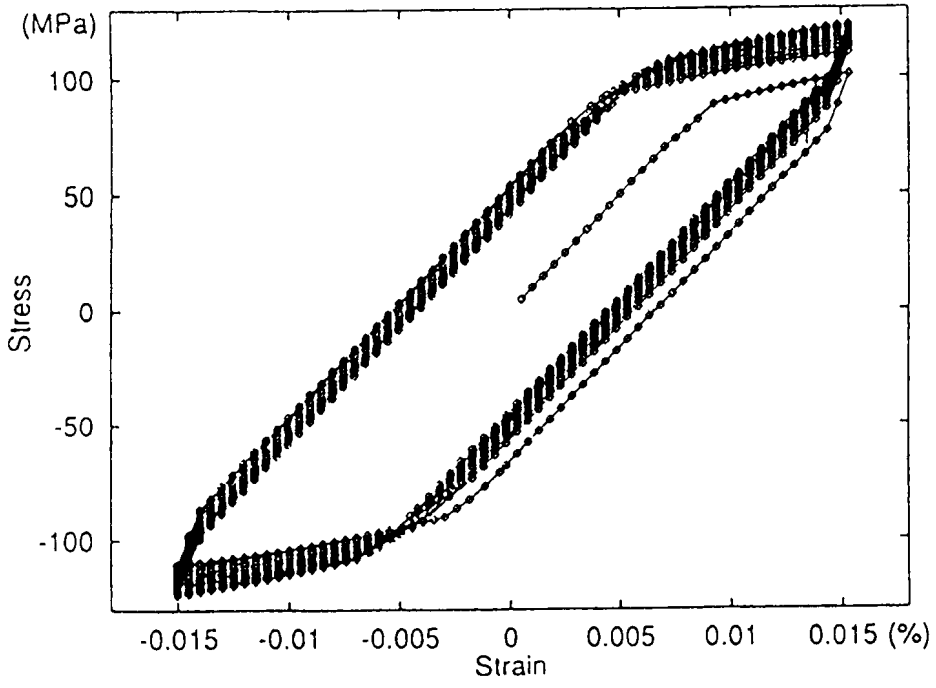


Figure 5.14. Prediction of total stress

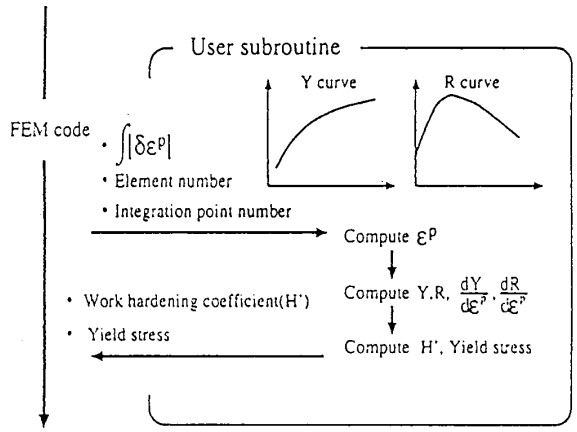


Figure 5.15. Input and output of the user subroutine

6. NEURAL-NETWORK-BASED CRACK GROWTH ANALYSIS OF WELDED SPECIMENS [41],[48]

6.1 Fracture Mechanics Based on J-Integral

Nonlinear fracture mechanics based on the J-integral concept [92] has widely been utilized in the assessment of structural integrity of ductile materials [93, 94]. The J-integral was originally derived on the assumption that materials and structures be homogeneous. This is not always the case in practical situations. For example, the elastic-plastic fracture phenomena of inhomogeneous materials and structures is one of the critical issues concerning the structural integrity of nuclear structural components such as irradiated, welded and cladded pressure vessels and welded piping.

Described here is an application of the hierarchical neural network to the crack growth analyses of two kinds of welded CT specimens using the GE/EPRI method. The GE/EPRI method is one of the simplified schemes to estimate the J-integral. In the present neuro strategy, the effect of inhomogeneousness is considered in the GE/EPRI method by introducing a mixture ratio of material constants w, and the best ratio of mixture is identified using the neural-network-based inverse analysis approach.

6.2 Problem Definition

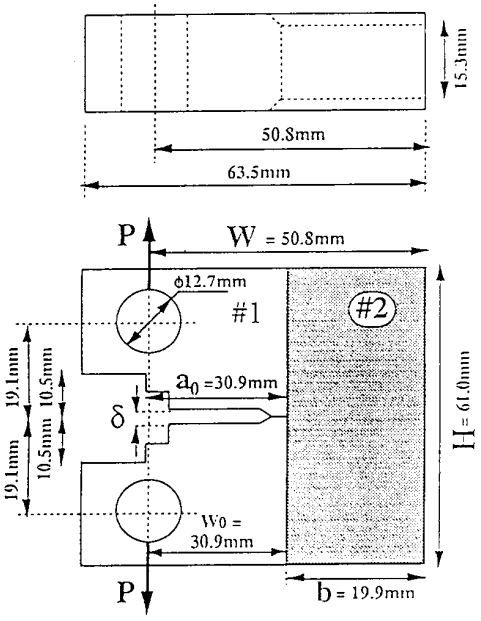
6.2.1 Welded CT specimens

Two kinds of welded CT specimens are considered. The one of the welded specimens is composed of a base metal and a weld metal of A533B class 1 steel, while the other is a base metal of A533B class 1 steel and high strength HT80 steel. Table 6.1 summarizes all the specimens considered here.

"A" series in Table 6.1 are the three welded CT specimens composed of a base metal and a weld metal of A533B. Figure 6.1(a) illustrates the configuration and dimensions of the H5G specimen, where an initial crack-tip was placed about 3 mm behind the phase boundary, i.e. in a heat-affected zone (HAZ). The materials #1 and #2 in the figure denote the base metal (BM) and the weld metal (WM) of A533B, respectively.

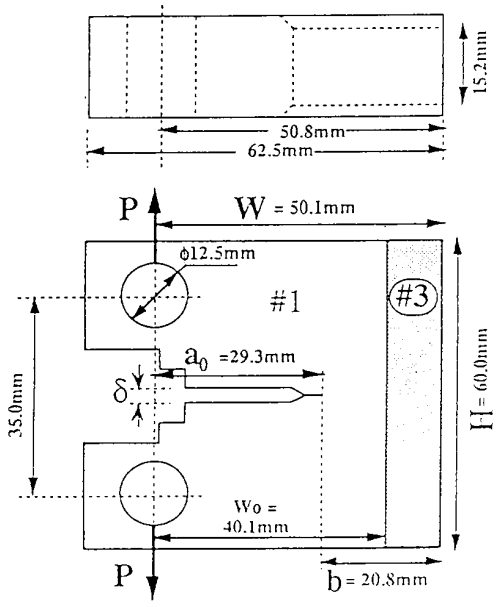
"AH" or "HA" series in Table 6.1 are the bimaterial specimens consisting of the A533B class 1 steel and the high strength HT80 steel. As for the AH type specimens, the holes for

loading are mac hined in the portion of the A533B class 1 steel, while vice versa in the HA type ones. Figures 6.1(b) and 6.1(c) illustrate the configurations and dimensions of the AH8 and HA8 specimens, respectively. The material #3 in the figures denotes the high strength HT80 steel.



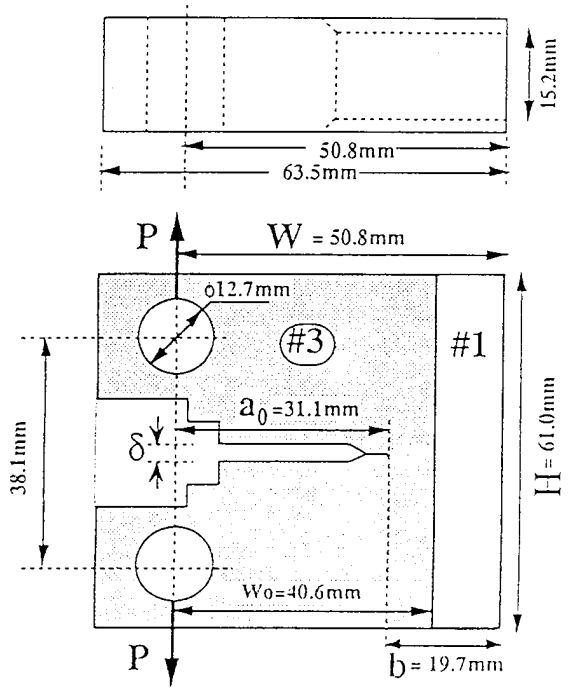
(a) F5G Specimen.

(a) Examples of welded CT specimens



(b) AH8 specimen

Figure 6.1.



(c) HA8 specimen

Figure 6.1. (*cont.*)

Specimen			Materials	Volume Ratio (Wo/W)	S.G. (%)	ao/W	Initial Crack-tip Location
Series		Name					
Homogeneous		M5G	A533B	--	20	0.59	A533B
		HT	HT80	--	0	0.60	HT80
Nonhomogeneous	A series	H5G	A533B's	0.63	20	0.59	HAZ
		F5G	BM&WM	0.61	20	0.61	FL
		D5G		0.48	20	0.60	WM
	AH series	AH3	A533B	0.30	20	0.59	HT80
		AH6	&HT80	0.54	0	0.60	HT80
		AH8		0.80	20	0.59	A533B
	HA series	HA3	HT80 &	0.32	20	0.60	A533B
		HA6	A533B	0.59	0	0.57	FL
		HA8		0.80	20	0.61	HT80

Notes ao: Initial crack length
W: Specimen width, Wo: Distance between load-line and fusion line
A533B: A533B class 1 steel HT80: HT80 steel
BM: Base metal of A533B cl.1, WM: Weld metal of A533B cl. 1
HAZ: Heat affected zone of A533B cl.1, FL: Fusion line

Table 6.1. Summary of tested specimens

The material constants of the metals are given in Table 6.2, where the following Ramberg-Osgood type stress-strain relation is assumed:

$$\left(\frac{\bar{\varepsilon}}{\varepsilon_0}\right)\left(\frac{\bar{\alpha}}{\alpha_0}\right)+\alpha\left(\frac{\bar{\alpha}}{\alpha_0}\right)^n \tag{6.1}$$

where α, ε_0, σ_0 and n are the material constants, while $\bar{\varepsilon}$ and $\bar{\alpha}$ are the Von Mises type equivalent strain and stress, respectively. It is noted that the ratio of yield stresses of the weld metal and the base metal of the A533B class 1 steel is about 1.15, while that of the HT80 steel and the A533B class 1 steel is about 1.4.

6.2.2 Empirical J-integral

The empirical J-integral is available. In crack growth experiments, an applied load (P), a load-line displacement δ and a crack extension amount (Δa) were measured. Using these values, the empirical J-integral, i.e. the deformed J-integral of Ernst *et al.* [95] can be evaluated.

6.2.3 GE/EPRI method

The GE/EPRI estimation scheme [96] is based on the J_2-deformation theory of plasticity and the power-law hardening constitutive relationship. In this scheme, the J-integral (J) and the load-line displacement δ are defined as follows

$$J = J_e + J_p \tag{6.2a}$$

$$\delta = \delta_e + \delta_p \tag{6.2b}$$

where J_e and J_p are the elastic and the plastic components of J, and δ_e and δ_p are the elastic and the fully plastic solutions of δ, respectively. J_p and δ_p are defined as follows

$$J_p = \alpha \times \sigma_0 \times \varepsilon_0 \times (W-a) \times h_1(a/W, n) \times (P/P_0)^{n+1} \tag{6.3a}$$

$$\delta_p = \alpha \times \varepsilon_0 \times a \times h_3(a/W, n) \times (P/P_0)^{n} \tag{6.3b}$$

where

α, n	:	Constants of the Ramberg-Osgood type relation
σ_0	:	Proportional limit stress of material (= yield stress)
ε_0	:	Proportional limit strain of material (= yield strain)
h_1, h_3	:	Fully plastic solutions of J and δ, respectively
P	:	Applied load per unit thickness
P_0	:	Limit load per unit thickness

Here, h_1 and h_3 are given functions. The generation phase crack growth simulations based on the GE/EPRI method are performed as illustrated in Figure 6.2. Using the measured δ vs Δa curve as input data, the applied load P is iteratively calculated from Eq.(6.3b), and then the J value is calculated by substituting the applied load P into Eq.(6.3a).

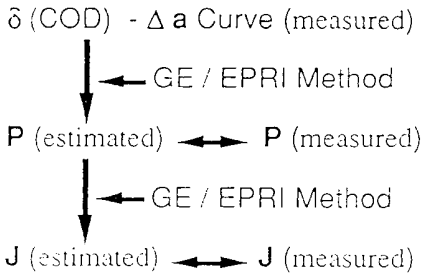


Figure 6.2. Flow of simplified generation phase crack analyses based on GE/EPRI method

If a specimen is homogeneous, the material constants in Table 6.2 are utilized. To analyze the welded specimens as shown in Figures 6.1(a) to 6.1(c) by the GE/EPRI method, the mixed values of material constants of the two component materials are considered. Introducing a ratio of mixture ω, the following hypothetical material constant is derived

$$C = (1 - \omega)C_1 + \omega C_2 \tag{6.4}$$

where C_1 and C_2 denote any material constant, such as α, n, σ_0 and ε_0, of a softer material, eg. the material #1 and that of a harder material, eg. the material #2, repectively.

	A533B cl. 1		HT80
	Base metal (# 1)	Weld metal (# 2)	(# 3)
σ_0, σ_Y (MPa)	550	630	760
σ_u (MPa)	654	722	853
σ_f (MPa)	602	676	807
ε_0 (%)	0.267	0.360	0.37
α	6.48	0.892	2.70
n	10.0	24.5	22.4
E (GPa)	206	175	206

Table 6.2. Material constants of base and weld metals of A533B class 1 steel and those of HT80 steel

6.3 Procedure of Inverse Analyses

In the analyes of welded specimens, a proper combination of the material constants of the two component materials, i.e.a proper ratio of mixture ω is unknown. Thus, the neural-network-based inverse analysis approach is applied to the determination of the best ratio of mixture. The procedure consists of the following three phases:

Phase 1: Assuming some ratio of mixture ω, a set of $P-\delta$ and $J-\Delta a$ curves are calculated using both the mixed material constants and the measured $\delta-\Delta a$ curve as input data. This process is a direct analysis. The obtained data set between the crack growth behaviors, i.e. the $P-\delta$ and $J-\Delta a$ curves and the ratio of mixture ω is called here the "learning data sets". Various $P-\delta$ and $J-\Delta a$ curves are calculated by parametrically varying ω. This phase is a preparation process of learning data sets.

Phase 2: The hierarchical neural network is trained using a number of learning data prepared in Phase 1. Here the $P-\delta$ and $J-\Delta a$ curves are given to the input units of the network, while the ratio of mixture ω to the output units of the network as teaching data. The error back propagation algorithm combined with the moment method is employed to attain stable convergence in the training process.

Phase 3: By providing the measured $P-\delta$ and $J-\Delta a$ curves to the input units of the trained network, the best ratio of mixture can be determined. This process is an inverse analysis.

6.4 Learning Process

For determining the best ratio of mixture, the following three methods are examined.

Method 1: A single ratio of mixture ω is introduced to simultaneously estimate the $P-\delta$ and $J-\Delta a$ curves. The network architecture employed is shown in Figure 6.3. Here, 16 values of load are sampled from the $P-\delta$ curve, and 16 values of J are sampled from the $J-\Delta a$ curve. The 32 sampled values are then given to the input units of the network. One best ratio of mixture ω comes out from the network. The number of units in the hidden layer is determined through trial and error. Nine sets of learning data ranging from $\omega = 0.0$ to 1.6 with an interval of 0.2 are employed. The network stops training when an error becomes less than 0.01.

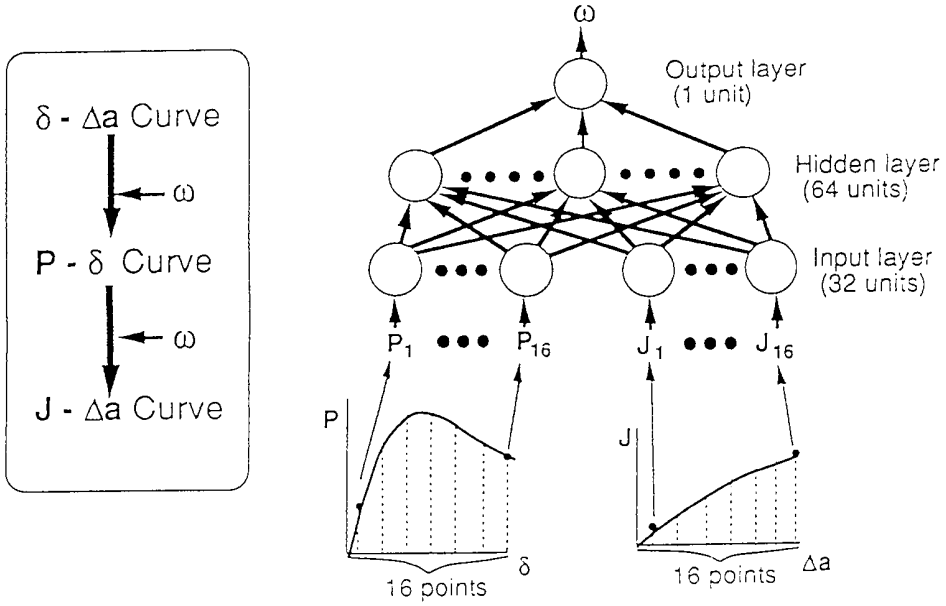


Figure 6.3. Hierarchical neural network and its input and output data (Method 1: single ratio of mixture, single network

Method 2: Two ratios of mixture ω_P and ω_J are introduced. The ratio of mixture ω_P is first utilized to estimate the $P-\delta$ curve. ω_J is then utilized to estimate the $J-\Delta a$ curve. The network architecture employed is shown in Figure 6.4.81 sets of learning data, i.e. $(\omega_P = 0.0 \sim 1.6) \times (\omega_J = 0.0 \sim 1.6)$ are utilized. The network stops training when an error becomes less than 0.01.

Method 3: Two ratios of mixture ω_P and ω_J are also utilized. The $P-\delta$ curve is estimated using ω_P. The $J-\Delta a$ curve is estimated using ω_J. Then, t w o independent networks are emplo yed as shown in Figure 6.5. 18 sets of learning data, i.e. $(\omega_P = 0.0 \sim 1.6) + (\omega_J = 0.0 \sim 1.6)$ are utilized. The criterion of training is the same as above.

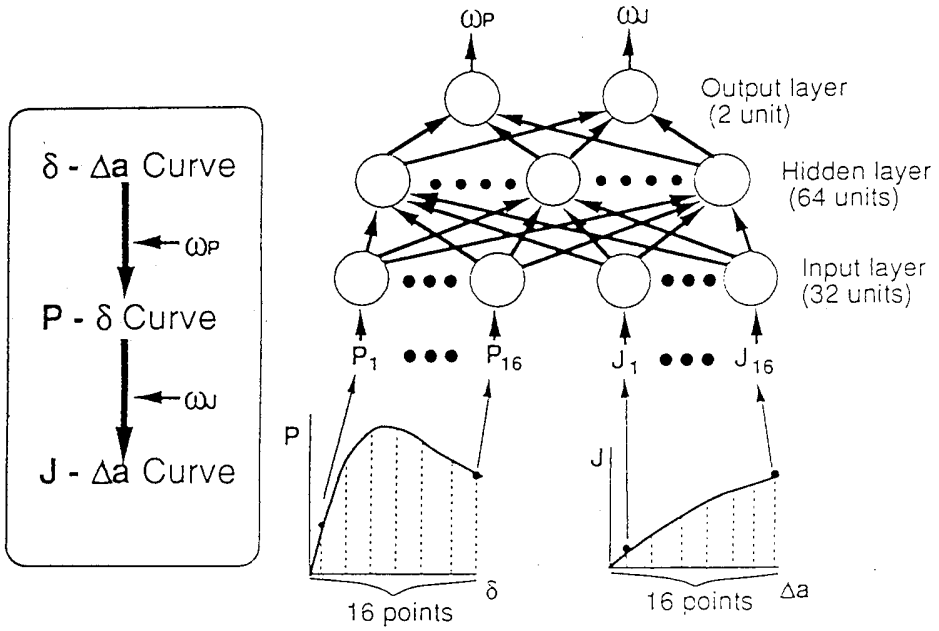


Figure 6.4. Hierarc hical neural network and its input and output data (Method 2: two ratios of mixture, single net work

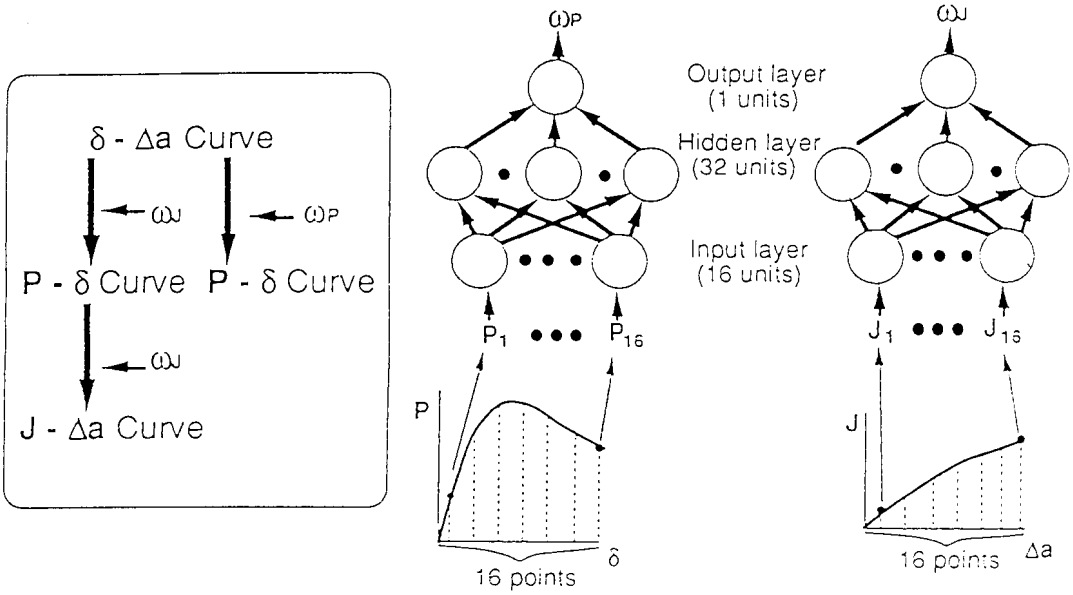


Figure 6.5. Hierarc hical neural network and its input and output data (Method 3: two ratios of mixture, t w o independent networks

6.5 Estimation of Ratio of Mixture

Table 6.3 summarizes the obtained "best ratios of mixture" for all the CT specimens. The following observations are derived from the table.

Specimen		Method 1	Method 2		Method 3	
Series	Name	ω	ω_P	ω_J	ω_P	ω_J
A Series	H5G	0.77	0.95	0.50	0.91	0.46
	F5G	1.00	0.93	0.25	0.95	0.52
	D5G	0.97	0.86	0.44	0.86	0.36
AH Series	AH3	1.42	1.59	1.04	1.49	1.20
	AH6	0.33	0.53	0.22	0.51	0.28
	AH8	0.63	1.08	0.55	1.00	0.67
HA Series	HA3	0.09	0.10	-0.08	0.13	-0.06
	HA6	0.36	0.59	0.49	0.45	0.54
	HA8	0.49	0.83	0.43	0.86	0.52

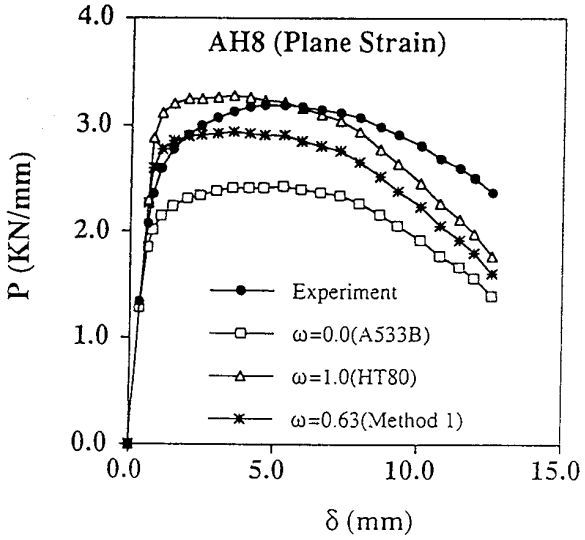
Table 6.3. Best ratios of mixture

a) A most portion of the HA3 specimen is homogeneous, i.e. made of the A533B class 1 steel except a crack mouth area. In such a case, the utilization of the material constants of the A533B steel, i.e. $\omega = 0$ gives the best estimation.
b) As for the AH3 specimen whose most portion is made of the HT80 steel, the utilization of the HT80 steel may give better results although $\omega = 1.2 \sim 1.5$ gives the best estimate. The reason why the best ratio of mixture exceeds 1 so much is not well understood.
c) As for the other seven specimens in which a near-crack-tip area or ligament area is inhomogeneous, the best ratio of mixture lies in between 0.0 and 1.0. The averaged best ratios of mixture ω_J and ω_P are 0.48 and 0.81, respectively.

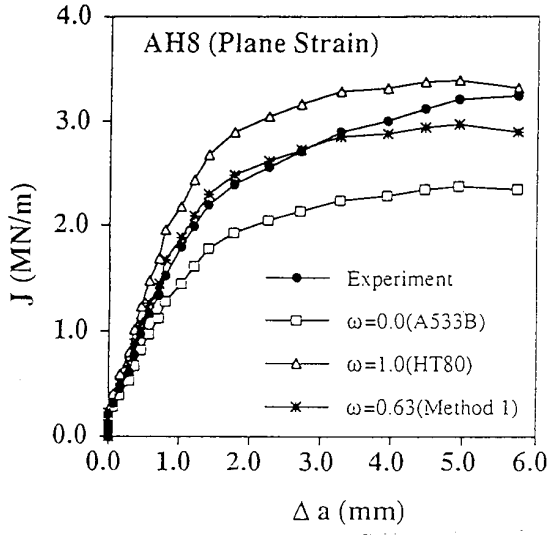
Figure 6.6(a) shows the comparison of the measured and estimated $P - \delta$ curves of AH8 specimen. Figure 6.6(b) shows the similar comparison of the $J - \Delta a$ curves. The following three kinds of estimated results are included in the figures. Open squares indicate the results estimated using the material constants of the softer material, the A533B class 1 steel, i.e. $\omega = 0.0$. Open triangles indicate the results estimated using the material constants of the harder material, the HT80 steel, i.e. $\omega = 1.0$. Crosses indicate the results estimated using the best ratio of mixture obtained by Method 1. Figures 6.7 and 6.8 show the comparisons of the measured and estimated results using the best ratios of mixture obtained by Method 2 and Method 3, respectively. The similar results are obtained for the other welded specimens.

The following observations are derived from the comparisons.

a) The introduction of the ratio of mixture is capable of well estimating the measured $J - \Delta a$ curve of the present welded specimens. As for the $P - \delta$ curve, it is, however, still insufficient to approximate the shape of measured $P - \delta$ curve of the welded specimens.
b) As for Method 1, where a single ratio of mixture ω is utilized, it is a little difficult to find the best ratio of mixture such that both measured $P - \delta$ and $J - \Delta a$ curves are well estimated.
c) The introduction of two ratios of mixture ω_P and ω_J such as Method 2 and Method 3 improves estimation accuracy.



(a) Comparison of experimental and estimated $P - d$ curves of AH8 specimen (Method 1)

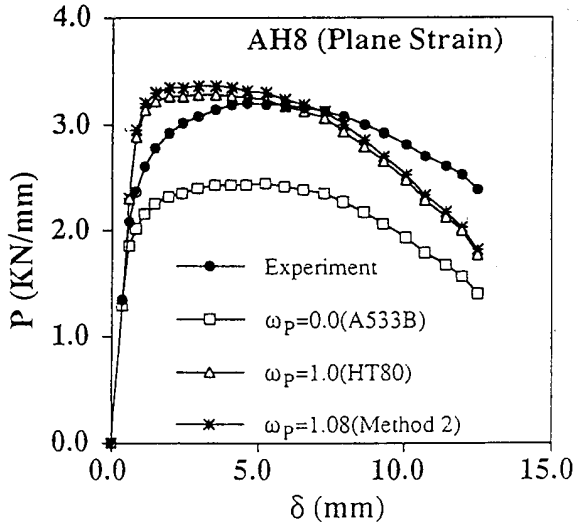


(b) Comparison of experimental and estimated $J - Da$ curves of AH8 specimen (Method 1)

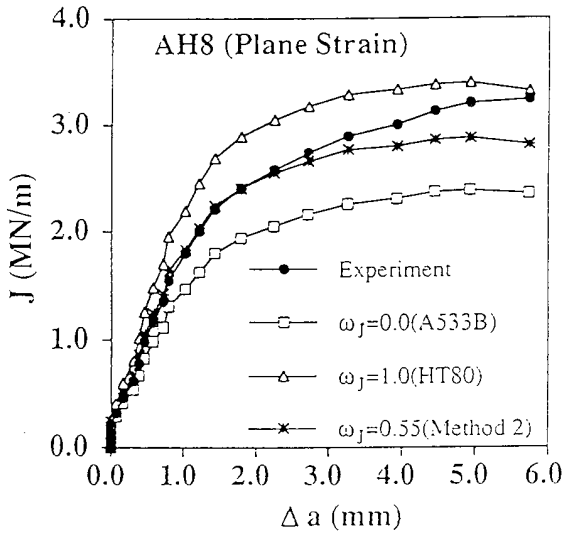
Figure 6.6.

d) Compared with Method 2, Method 3 where two ratios of mixture ω_P and ω_J are determined independen tly from the $P - \delta$ and $J - \Delta a$ curves requires less learning data sets and gives better estimation.

The above results show that when utilizing a proper ratio of mixture considering material arrangement and volume fraction of component materials, the con ventional GE/EPRI method can giv e better estimates ev en in inhomogeneous material regimes.



(a) Comparison of experimental and estimated $P - d$ curves of AH8 specimen (Method 2)



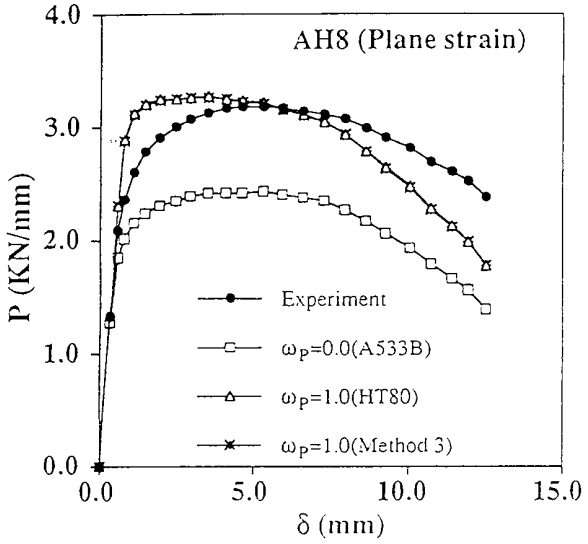
(b) Comparison of experimental and estimated $J - Da$ curves of AH8 specimen (Method 2)

Figure 6.7.

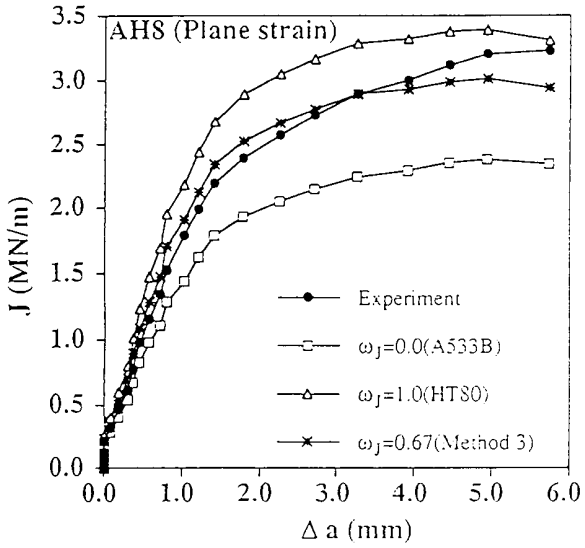
7. NEURAL-NETWORK-BASED STRUCTURAL DESIGN [38]

7.1 Design Window Searc h Approach

In the field of structural design, the structural optimization problems have been solved by the mathematical methods combined with the computational mechanics [97]. Ho w ever, such optimized solutions that are often deriv ed under some assumed analysis conditions are not so useful in practical situations. In other w ords, the conventional optimization tec hniques do not allow us to perform robust designs. In this respect, it seems more meaningful to obtain



(a) Comparison of experimental and estimated $P - d$ curves of AH8 specimen (Method 3)



(b) Comparison of experimental and estimated $J - Da$ curves of AH8 specimen (Method 3)

Figure 6.8.

the design windo w(D W) that sc hematically indicates an area of satisfactory solutions in a permissible design space.

An automated structural design system based on the DW search approach is described. The presen t system consists of three main modules and one sub-engine as shown in Figure 7.1. The three main modules are the "Analyzer", the "DW Searc h Engine" and the "Design Modifier". A multilayer neural network is utilized in the "D W Searc h Engine" as the "Neuro Analyzer". The "Analyzer" performs coupled finite element simulations for various design parameter sets. T o main tain high flexibility and extensibility, the "Analyzer" is

constructed using an object-oriented knowledge representation [98] and a data-flow processing technique [99]. In the "DW Search Engine", some search methods are implemented. By using the "Neuro Analyzer", DWs can be searched very efficiently. The "Design Modifier" has the role of giving an initial satisfactory solution to the "DW Search Engine" so that some DW search methods of design window can start searching. In the "Design Modifier", the empirical design modification approach combined with both the fuzzy control and the moment method is implemented to improve the capability of the empirical approach in quantitative design modification.

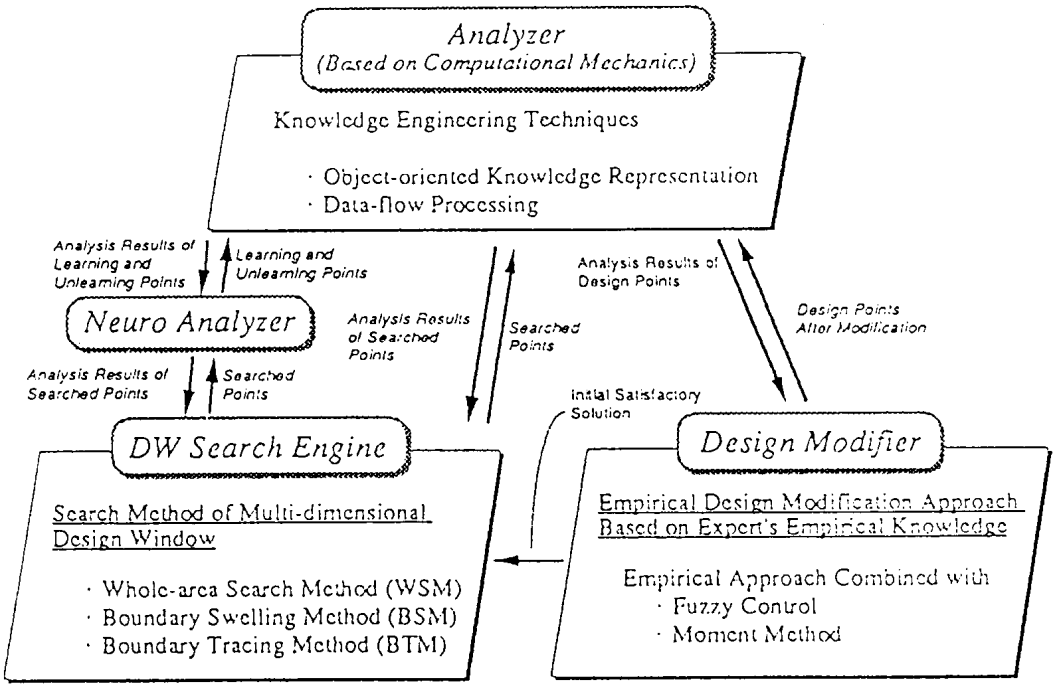


Figure 7.1. Schematic configuration of the present system

As shown in Figure 7.2, several large or small design windows may exist in a design space. The design space is divided into two subspaces, i.e. the "permissible design space" and the "impossible design space". The permissible design space is the subspace where all the geometrical constraints are satisfied, while some of the constraints are violated in the "impossible design space". The design windows are searched only within the permissible design space.

The Whole-area Search Method (WSM) is one of the search methods implemented in the "DW Search Engine". The WSM can search multi- dimensional design windows even if the design windows to be searched are plural or doughnut-shaped, and can obtain quasi-optimum solutions. The algorithm of the WSM is as follows. At first, a multi-dimensional lattice that is determined by engineers empirically is generated in the design space. Then all intersection points of the lattice are examined one by one whether they are inside design windows or not. In the present DW search process, a distribution of an objective function is also obtained if necessary. In the WSM, the number of points to be examined tends to become extremely huge.

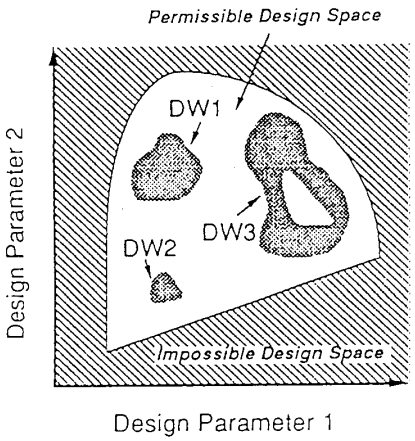


Figure 7.2. Schematic view of design windows in the case of two-dimensional design window

7.2 Principle of Neural-Network-Based DW Search

In searching the design windows, whether a searched point is a satisfactory solution or not is checked through computational mechanics simulations, e.g. FEM. However, the finite element computations at every searched points are very time-consuming especially in the design of high temperature structural components because both thermal and stress analyses have to be executed. Thus, the search method using a neural network that can evaluate nonlinear mapping very fast is proposed here. This method consists of three phases:

Phase 1: Finite element analyses are performed to prepare learning data sets and unlearning ones for the neural network. The unlearning data sets are utilized to prevent the neural network from "overlearning", which is such a phenomenon that the estimation error of unlearning data sets increases largely even though the learning process advances.

Phase 2: An error-back-propagation neural network is trained using a number of the learning data sets. Here, design parameters are given to the network as input data, while physical values such as the maximum temperature and equivalent stress, are given to it as teaching data.

Phase 3: The trained network can imitate a response of the FEM analyzer. Finally multi-dimensional design windows are searched using the trained network.

In the present search method, the finite element analyses are performed at Phase 1. The number of the finite element analyses required is much smaller in this method than in the search method without a neuro. As the trained network calculates physical values very quickly, it can be checked effectively whether a searched point is a satisfactory solution. Thus, by utilizing the trained neural network, even the WSM can be executed within a reasonable processing time.

7.3 Problem Definition

The present system is applied to design the ITER (International Thermonuclear Experimental Reactor) first wall [100]. The first wall of fusion reactors will be required to operate in the severe environment where mechanical, electromagnetic and thermal loading occurs in a complicated manner and various failure phenomena such as melting, yielding and fracturing are competing. In designing such a structure, it seems very important to determine the

design window that schematically indicates an area of satisfactory solutions in a permissible design space.

The ITER first wall currently has two candidate designs regarding to a cooling channel, i.e. rectangular and circular channels. The present system is applied to search the design windows for the two models. The design models are shown in Figures 7.3(a) and 7.3(b). The mother material of the wall is subjected to the membrane tensile loading F caused by the electromagnetic loading and the pressure from the breeder blanket, to the surface heat loading Q_blanket on the blanket-side surface, and to the volumetric heat loading Qv_SUS316 (= 20.0 [MW/m^3]). The armor material is subjected to the surface heat loading Q_plasma (= 0.15 [MW/m^2]) on the plasma-side surface, and to the volumetric heat loading Qv_armor (= 6.0 [MW/m^3]). The temperature and the pressure of the cooling water are 100.0 [°C] and 1.5 [MPa], respectively. The heat transfer coefficient between the armor and the mother materials is 1000.0 [W/m^2 °C], and that between the cooling water and the mother material is 20000.0 [W/m^2 °C]. The type 316 stainless steel and CX-2002U are the mother material and the armor one, respectively. The wall with the armor is modeled with eight-noded isoparametric elements. A static thermal conduction analysis is performed. In an elastic thermal stress analysis, only the mother material is analyzed under the generalized plain strain condition. The finite element analyses are performed using the object-oriented "Analyzer" described above.

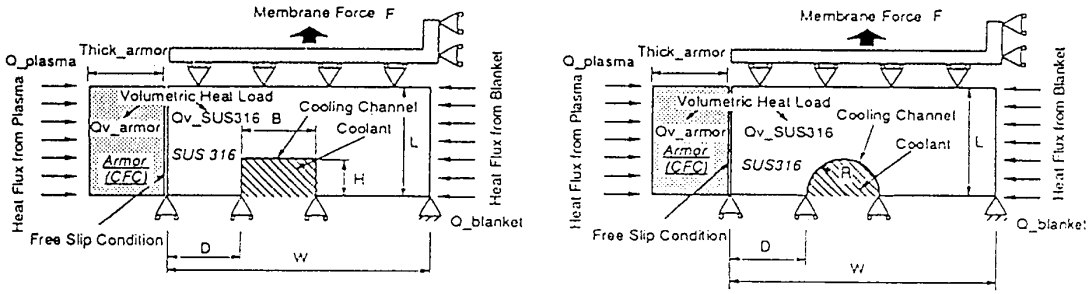


Figure 7.3. Design models of ITER first wall

The design criteria employed here are both geometrical and failure constraints. The geometrical constrains are as follows:

Rectangular cooling channel model

$$D \geq \alpha \tag{7.1}$$
$$L \geq H + \beta \tag{7.2}$$
$$W \geq D + B + \gamma \tag{7.3}$$

Circular cooling channel model

$$D \geq \alpha \tag{7.4}$$
$$L \geq R + \beta \tag{7.5}$$
$$W \geq D + 2R + \gamma \tag{7.6}$$

where D, L, W, H, B and R are the geometrical design parameters as shown in Figures 7.3(a) and 7.3(b), and α, β and g the design margins that are empirically decided by engineers. The failure constraints are as follows:

$$\text{(a) } T_{\max}\_\text{Armor} < T_0\_\text{Armor} \tag{7.7}$$

$$\text{(b) } T_{\max}\_\text{SUS316} < T_0\_\text{SUS316} \tag{7.8}$$

$$\text{(c) } \sigma_{\max}\_\text{SUS316} < s_0\_\text{SUS316} \tag{7.9}$$

where $T_{\max}$_Armor is the maximum temperature occurred in the armor material, T_0_Armor a permissible temperature for the armor material, $T_{\max}$_SUS316 the maximum temperature occurred in the mother material, T_0_SUS316 a permissible temperature for the mother material, $\sigma_{\max}$_SUS316 the maximum equivalent stress occurred in the mother material and σ_0_SUS316 a permissible equivalent stress for the mother material, respectively. For the purpose of simplicity, T_0_Armor, T_0_SUS316 and σ_0_SUS316 are taken to be 1000.0 [°C], 400.0 [° C] and a yielding stress of the type 316 stainless steel, respectively.

7.4 Fundamental Characteristics of Neuro Analyser

First, the fundamental characteristics of the present search method using the neural network is examined through searching $L - W$ design windows of the rectangular cooling channel model. Here, D, H and B are fixed to be 2.0 [mm], 3.5 [mm] and 5.0 [mm], respectively. The design margins for the geometrical constraints, β and γ are taken to be 0.5. As for the boundary conditions, F and Q_blanket are assumed to be 49.0 [N] and 0.0 [MW/m^2], respectively.

The multilayer neural network employed is of three-layered type. The network has two input units, forty hidden units and one output unit. The two input units correspond to the two geometrical design parameters L and W. One output unit is prepared for the maximum equivalent stress $\sigma_{\max}$_SUS316. It should be noted here that the output units for $T_{\max}$_Armor and $T_{\max}$_SUS316 are not prepared because the temperature constraints of Eqs.(7.7) and (7.8), are always satisfied under the given boundary conditions. Figures 7.4 illustrates 16 sets of the input data of learning patterns and 9 sets of the input data of unlearning patterns in the $L - W$ design space. All the input data and output data are normalized to a unit range from 0.01 to 0.99. Figure 7.5 shows the history of learning process in the case that the temperature of sigmoid function T is taken to be 0.6. Here, the following mean error of estimation is employed for both learning and unlearning patterns:

$$\text{Mean Error} = \frac{1}{r}\sum_{p=1}^{r} |T^p - O^p| \tag{7.10}$$

where r, T^p and O^p are the number of learning patterns, the correct solution of p-th learning or unlearning pattern, and the output data of p-th learning or unlearning pattern, respectively.

It can be seen in the figure that the overlearning occurs in the learning process. To examine the correlation between the overlearning and the accuracy of the neural-network-based design windows, the design windows are searched using the following two trained neural networks (see Figure 7.6) :

a) The network adopts the connecting weights at 226 learning iterations when the mean error of estimation for the unlearning patterns becomes the minimum.
b) The network adopts the connecting weights at 5000 learning iterations when the overlearning has fairly advanced.

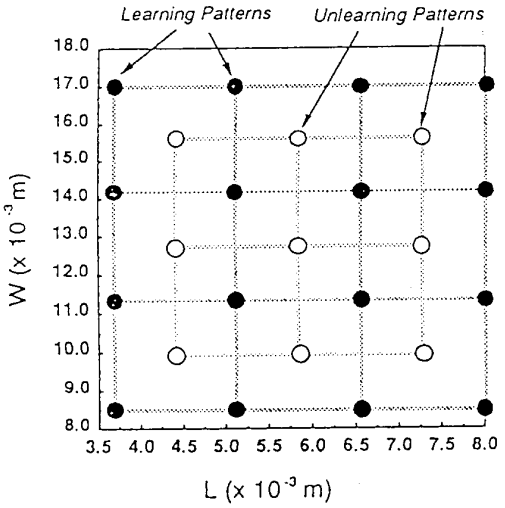


Figure 7.4. Learning patterns and unlearning patterns in L-W space

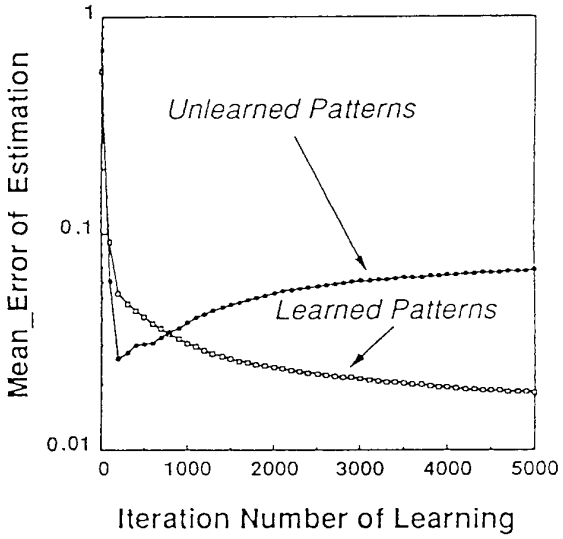
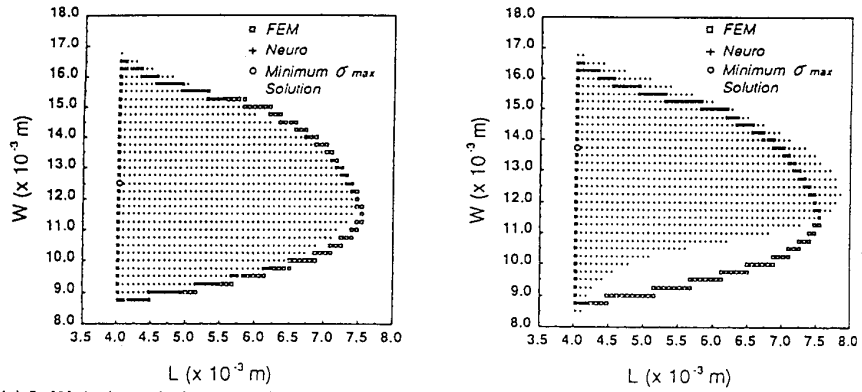


Figure 7.5. Training histories of learning patterns and unlearned patterns

In the figures, the regions surrounded by chained open squares indicate the correct design windows that are searched based on the Boundary Tracing Method without the neural network, and the regions filled with plus-symbols show the design windows that are searched based on the present Whole-area Search Method (WSM) with the neural network. Here, the unit steps for searching, ΔL and ΔW are taken to be 0.075 [mm] and 0.25 [mm], respectively. It can be seen from the figures that the design window obtained with the network (a) agrees well with the correct one, while the design window obtained with the network (b) is rather distorted. Such less accuracy in the latter case may be due to the overlearning. This comparison suggests that it is very important to stop the network learning when the mean error of estimation for the unlearning patterns becomes the minimum.



(a) L-W design window searched by neuro that is trained until the Mean_Error of estimation for unlearned patterns is minimized, i.e. at 226 learning iterations

(b) L-W design window searched by the over-trained neuro until 5000 learning iterations

Figure 7.6. Effects of overlearning of neural network on design windows

The open circles in Figures 7.6(a) and 7.6(b) indicate the quasi-minimum values of $\sigma_{\max}$_SUS316. By using the WSM, the distribution of the objective function over the searched design window and quasi-optimum solutions are simultaneously obtained. Therefore, one can choose a final design solution by considering these design windows and other issues such as the distribution of the objective function, manufacturability, structural integrity and cost.

7.5 D W Search of ITER

The rectangular and the circular cooling channel models in Figure 7.3 are considered taking the three-dimensional design windows of D, L and W into account. Figures 7.7(a) and 7.7(b) show the three-dimensional design windows for both models. The searching unit steps, ΔD, ΔL and ΔW are taken to be 0.15 [mm], 0.15 [mm] and 0.5 [mm], respectively. The surfaces of both cooling channels are chosen to be the same as each other.That is, $A_{\text{rectangular}}$ $(= 2H + B = 2 \times 3.5$ [mm] $+ 5.0$ [mm]) in the case of the rectangular channel model and A_{circular} $(= \pi R = \pi \times 3.82$ [mm]) in the case of the circular one are taken to be the same value of 12.0 [mm]. As to the boundary conditions, F and Q_blanket are 9.8 [N] and 0.15 [MW/m^2], respectively. α, β and γ are taken to be 1.0 [mm].

Both design windows are searched based on the WSM combined with the neural network approach. The multilayer neural networks employed are of three-layered type. The network to design the rectangular channel model has five input units, twenty hidden units and two output units, while that of the circular channel model has four input units, twenty hidden units and two output units. D, L, W, H and B are the input data in the case of the rectangular channel model, while D, L, W and R are those in the case of the circular channel model. Two output units output the maximum equivalent stress and the maximum temperature of the mother material. The output unit for $T_{\max}$_Armor is not considered here because the temperature constraint for the armor material, i.e. Eq.(7.7), is always satisfied under the given boundary conditions. In the case of the rectangular channel model, 460 learning patterns and 120 unlearning ones are adopted, while 168 learning patterns and 56 unlearning ones are utilized in the case of the circular channel model.

As for the rectangular channel model, the input data sets of the learning patterns are prepared as follows:

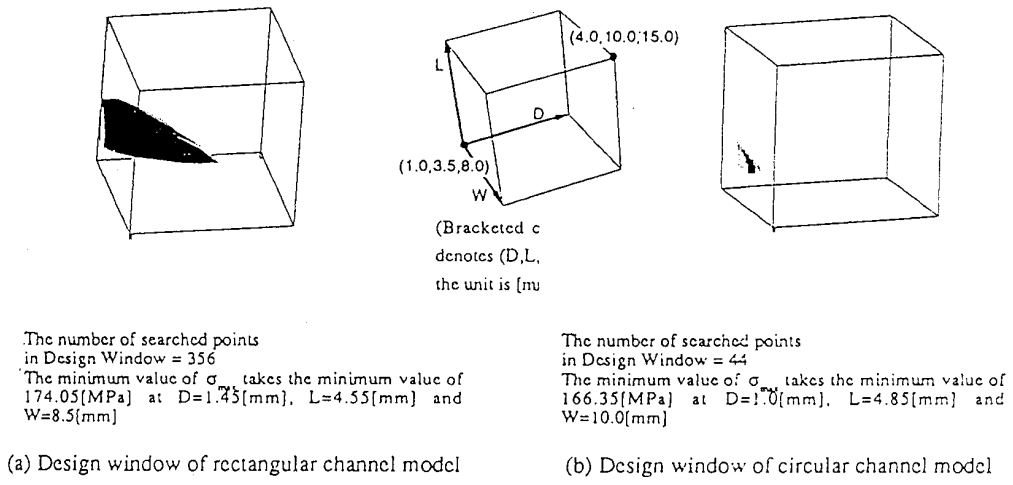


Figure 7.7. Comparison of three-dimensional design windows between rectangular cooling channel model and circular cooling c hannel model

Step 1: Divide the in terval betw een the maximum and the minimum values of each design parameter in to three subsections as:

D	L	W	H	B
$D1$ (D min)	$L1$ (L min)	$W1$ (W min)	$H1$ (H min)	$B1$ (B min)
$D2$	$L2$	$W2$	$H2$	$B2$
$D3$	$L3$	$W3$	$H3$	$B3$
$D4$ (D max)	$L4$ (L max)	$W4$ (W max)	$H4$ (H max)	$B4$ (B max)

Here, D max, L max, W max, H max and B max are 4.0 [mm], 10.0 [mm], 15.0 [mm], 9.5 [mm] and 10.0 [mm], while D min, L min, W min, H min and B min are 1.0 [mm], 3.5 [mm], 8.0 [mm], 1.5 [mm] and 3.0 [mm], respectively.

Step 2: Consider all possible combinations of (D, L, W, H, B).

Step 3: Choose the com binations that satisfy the geometrical constrain ts. These are utilized as the input data sets of the learning patterns.

On the other hand, the unlearning patterns are prepared as follows:

Step 1: Make the following list of eac h design paramater using the learning data abo ev

D	L	W	H	B
$(D1+D2)/2$	$(L1+L2)/2$	$(W1+W2)/2$	$(H1+H2)/2$	$(B1+B2)/2$
$(D2+D3)/2$	$(L2+L3)/2$	$(W2+W3)/2$	$(H2+H3)/2$	$(B2+B3)/2$
$(D3+D4)/2$	$(L3+L4)/2$	$(W3+W4)/2$	$(H3+H4)/2$	$(B3+B4)/2$

Step 2: Make all combinations of (D, L, W, H, B) based on the list.

Step 3: Choose the combinations that satisfy the geometrical constraints. These are utilized as the input data sets of the unlearning patterns.

In making these patterns, the geometrical constraints α, β and γ are taken to be 0.5 [mm] which is 1.0 [mm] less than used for searching design windows. This purpose is to prevent the neural network from extrapolating physical values.

In the case of the circular channel model, the learning and unlearning patterns are generated in the same way. D max, L max, W max and R max are 4.0 [mm], 10.0 [mm], 15.0 [mm] and 5.0 [mm], while D min, L min, W min and R min are 1.0 [mm], 3.5 [mm], 8.0 [mm] and 1.5 [mm], respectively.

As for the learning parameter of the neural networks, the temperature of sigmoid function is taken to be 0.6 for both rectangular and circular channel models. Both neural networks are trained until the mean error of estimation for the unlearned patterns becomes the minimum.

If these design windows are searched based on the WSM without the neural network, 10,651 and 6,233 finite element analyses have to be performed in the rectangular and the circular channel model cases, respectively. On the other hand, when using the WSM with the neural network approach, only 580 and 224 finite element analyses are executed in both cases, respectively. Here, the number of the finite element analyses denotes the sum of the number of the analyses required for the learning and unlearning patterns. Although the neural network approach requires a training time, the searching process itself is much more efficient.

Examined here is a quasi-minimum $\sigma_{\max}$_SUS316 solutions. As for the rectangular channel model, the $\sigma_{\max}$_SUS316 takes the minimum value of 174.05 [MPa] at $D = 1.45$ [mm], $L = 4.55$ [mm] and $W = 8.5$ [mm], while 166.35 [MPa] at $D = 1.0$ [mm], $L = 4.85$ [mm] and $W = 10.0$ [mm] in the case of the circular channel model. Thus, the circular channel model is regarded as a better design from the viewpoint of σ_{max}_SUS316 although the difference between the minimum value of the rectangular channel model and that of the circular one is very small.

Next, the sizes of the design windows of both models are considered. The number of searched points is 356 in the design window of the rectangular channel model, while 44 in the circular channel model. Thus, the rectangular channel model has a larger design window. Besides, as seen in Figure 7.7(a), in the case of the rectangular channel model, satisfactory solutions exist even if D is large. This may be because the cooling capability of the rectangular channel is higher than that of the cirucular one and because the stress concentration around the corners of the rectangular channel is not so severe. Thus, from the viewpoint of manufacturability and structural integrity, the rectangular channel model seems more suitable.

8. NEURAL-NETWORK-BASED PARAMETER ESTIMATION FOR NONLINEAR FINITE ELEMENT ANALYSES [66]

8.1 Parameter Dependency in Nonlinear Analyses

Generally, most of the nonlinear analyses involve several parameters to be tuned, which seriously influence both the numerical results and the stability. These parameters, e.g. the penalty number for the incompressible viscous flow analysis, the time step of the transient analysis need to be determined appropriately as well as dynamically referring to the convergence behaviours. However, since the nonlinear analysis in each field has its own empirical knowledge for choosing optimal parameters, it is difficult to decide a general estimation rule or a procedure for tuning the parameters involved in the nonlinear finite element analysis.

Described here is a new methodology for estimating multiple parameters involved in the nonlinear and the time-dependent finite element analyses using the hierarchical neural

network. The present procedure is applied to the parameter estimation of the augmented lagrangian method for the steady state analysis of the incompressible viscous flow and the time step evaluation of the pseudo time-dependent stress analysis for the incompressible inelastic structure.

8.2 Principle of Neural-Network-Based Parameter Estimation

The method of parameter estimation using the hierarchical neural networks consists of the following three phases:

Phase 1: A number of nonlinear finite element simulations for many combinations of the parameters are carried out to obtain sample data of the parameters vs. the convergence behaviours.

Phase 2: Each piecewise relation between a set of the parameters and a set of the convergence behaviours, which is called here "a learning data set", is used for training the neural network. The convergence behaviours such as computational CPU time are given to the network as the teaching data, while the parameters such as the time step are given as the input data.

Phase 3: The network can readily identify parameters even for unlearned analysis conditions.

8.3 Parameter Estimation of Augmented Lagrangian Method for Steady State Incompressible Viscous Flow Analysis

8.3.1 Augmented lagrangian method

In the incompressible viscous flow analysis, the incompressibility constraint applied to the velocity field causes one of the main difficulties in solving the governing equations numerically [101]. Generally, large scale analysis in dealing with industrial problem can be achieved by using iterative solvers such as the conjugate gradient (CG) method or the conjugate residual (CR) method [102]. However, because of the fact that the matrix equation system has no diagonal component corresponding to the pressure, consequently, iterative solvers commonly result in poor convergence rate. The penalty function method can indeed remove the pressure from the equation system, however, the iterative solvers cannot be suitably applied due to the large condition number of the resulting matrix. On the other hand, the augmented lagrangian (AL) method [103] iteratively controls the degree of the incompressibility of the fluid such that a diagonal dominant matrix system are obtained. In this method an AL functional is an objective function to be minimized. Since the condition number of the resulting matrix is not so large as that of the penalty function method, iterative solvers can be suitably applied.

The AL algorithm for the steady state Navier-Stokes equations is shown in Figure 8.1. This algorithm has a nest of the three loops: [Loop1] Updating the advection nonlinear term, [Loop2] Updating the pressure to satisfy the incompressibility constraint and [Loop3] Solving the kinetic equations by the conjugate residual (CR) solver. Parameters involved in this algorithm are the convergence criterion for the velocity (epsU), and the convergence criterion for the incompressibility (epsP), the penalty number (α), the pressure modification step (β) and the convergence criterion for the CR-iterative solver (epsCR). epsU, epsP and epsCR correspond to the convergence criteria for Loop1, Loop2 and Loop3, respectively. When α is chosen to be large, Loop 2, which is the loop for the incompressibility converges quickly. However, the CR-solver, i.e. Loop3 shows ill-convergence instead because of the large condition number of the matrix system of the Navier-Stokes equations. Furthermore, since the algorithm involves five parameters, it is quite difficult to estimate the optimal set of parameters to attain the well convergence rate.

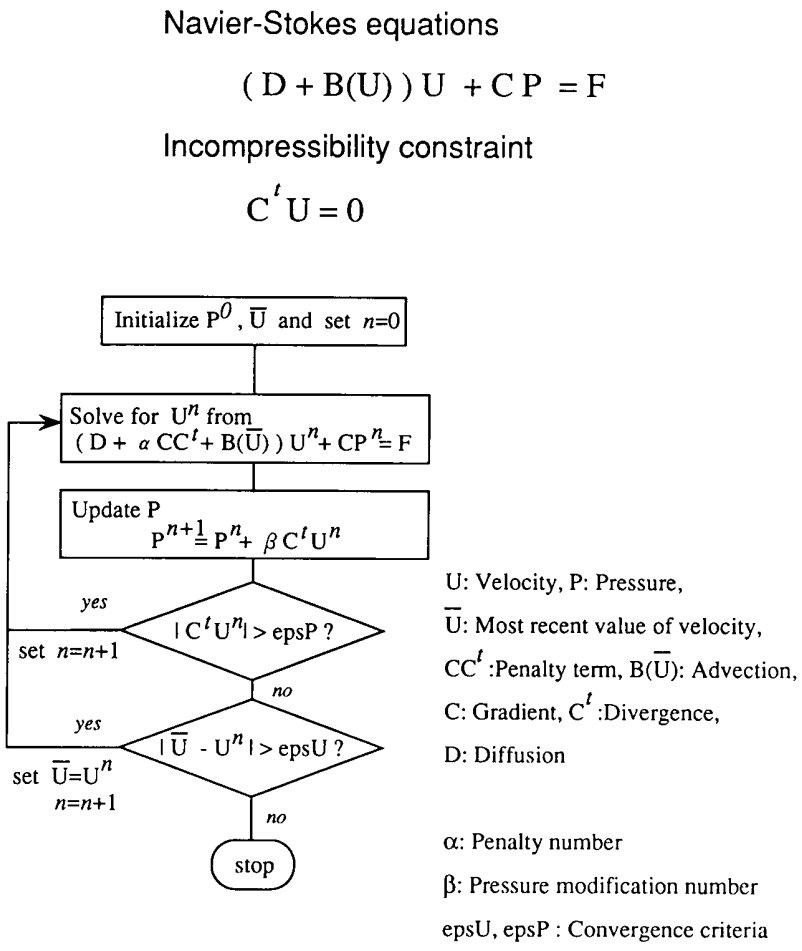


Figure 8.1. Augmen ted lagrangian method for state incompressible viscous flo w analysis

8.3.2 Estimation of parameters in AL algorithm

Theoretical optimal v alue of β can be determined as the function a, i.e. $\beta = \alpha$ when assuming the Stokes flow (linear problem). This value is used here. The con vergence criterion for the incompressibilit y epsP is set to 10^{-7} for all the present computations. Then, estimating parameters to be tuned are only epsU, α and epsCR. In the follo wings, the con vergence efficiency is measured by the necessary amoun t of the CPU time, because this algorithm has complicated nested loops.

First, in vestigated are the relationships bet ween the velocity convergence and the CPU time for 2D ca vity flow analyses. A square domain is discretized with the regular 20×20 elemen ts. P arameters are set to be epsCR= 10^{-6}, $10^{-7}, \ldots, 10^{-14}$ (9 cases), $\alpha = 10^{-2}$, $10^{-1}, \ldots,$ 10^{5} (8 cases). The Reynolds numbers are also varied as $Re = 10$, 25, 50, 75 and 100. After performing the finite element analyses with such various parameters and the Reynolds n um bers, the relationships among the (Re, $epsCR$, α) and (epsU, CPU time) are obtained. Figures 8.2(a) and 8.2(b) sho w the influence of the penalty number onto the CPU time. If epsU is set to 10^{-5}, the most efficient choice is $\alpha = 10^{-2}$ for the flow of $Re = 10$ (Figure 8.2(a)). How ever, as for the $Re = 100$ (Figure 8.2(b)), the most efficient one is $\alpha = 10^{-5}$. These results imply that the conventional w ay of tuning parameters, that

is, parameter optimization carried out in the small scale and/or linear problems, does not work at all. Next, those relationships when under $Re = 10$, 50 and 100 are used as the training data for the three-layered neural network as shown in Figure 8.3. The network has 3-units corresponding to Re, epsCR and α in the input layer, 6-units in the hidden layer and 2-units corresponding to epsU and CPU time in the output layer. The data sets under $Re = 25$ and $Re = 75$ are used to monitor the error of the unlearned data through the training process. The error histories for the learned and the unlearned data sets are shown in Figure 8.4. This figure shows both errors are judged to be sufficiently small. Therefore, this neural network is considered to have an ability to well instruct a set (Re, epsCR, α) from a given set (epsU, CPU time).

8.4 Time Step Evaluation of Pseudo Time-Dependent Stress Analysis for Incompressible Inelastic Structure

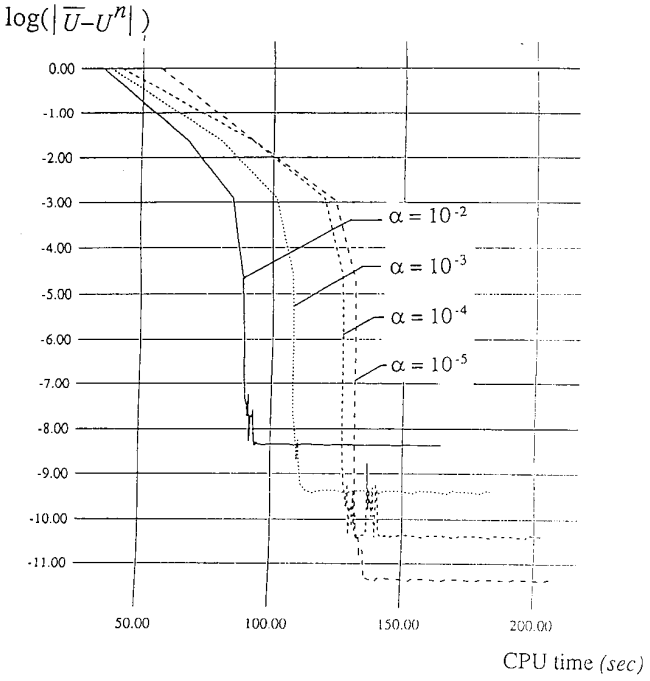
8.4.1 Fractional step method

To evaluate entire elastic-plastic behaviours of cracked bodies, so-called fully plastic solutions are utilized. There was proposed a finite element algorithm to compute the three-dimensional fully plastic solutions [104]. The algorithm employs the mixed formulations, which are commonly used in the incompressible flow analysis. By adding an artificial viscosity term to the governing equations, the static crack problems are treated as the quasi-nonsteady ones. They are solved by the fractional step (FS) method, i.e. one of the solution techniques for the incompressible viscous flow. The present algorithm is illustrated in Figure 8.5. This conversion makes the solution procedure be stable even in the cases of complex crack geometries, though it would consume more computation time. Numerical accuracy of the present algorithm was successfully demonstrated through the analyses of the fully plastic solutions of the center cracked plates [104]. Since this algorithm is based on the fractional step method, the time step Δt has strong influence both on the numerical results and the stability.

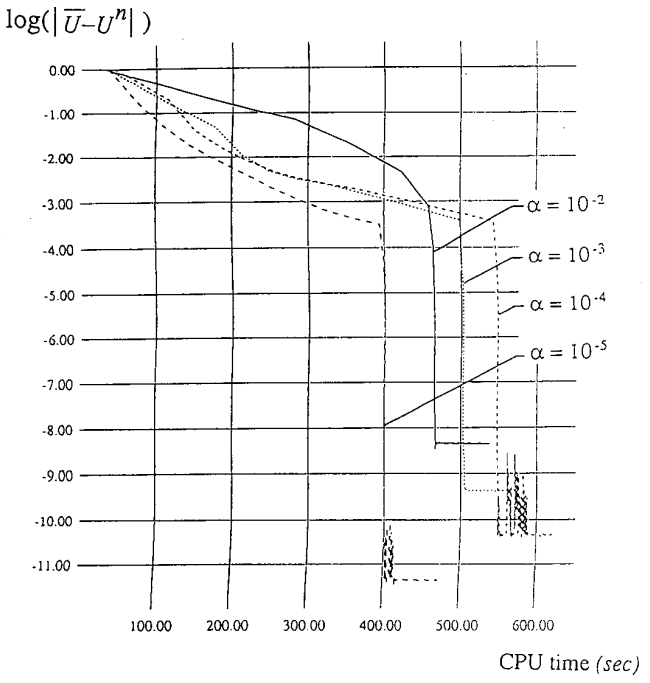
8.4.2 Estimation of time step in FS algorithm

To investigate fundamental performance of the present algorithm, a simple uniform mesh pattern shown in Figure 8.6 of an inelastic cubic material is tested. First, the average strain and the equivalent stress are preliminarily computed by the finite element analyses. These computations have been performed varying the uniform tensile stress $\bar{\sigma} = 0.5 \sim 1.5$ (9 cases), the hardening exponent $n = 1 \sim 10$ (9 cases), the ratio of maximum to minimum element size $R = 1 \sim 200$ (5 cases), the number of elements $N.E. = 6 \sim 600$ (4 cases) and the time step $\Delta t = 10^{-3}, 10^{-2}, \ldots, 10^{1}$ (5 cases). After performing the pseudo time-dependent stress analyses under such various parameters, the relationships among Δt, n and $\bar{\sigma}$ are obtained. Figure 8.7 shows the influence of the time step Δt onto the number of iteration steps. From this comparison among various Δt values, it is concluded that appropriate Δt should be around 0.1 in this problem. This result implies that the conventional way of tuning Δt, that is, the time step optimization carried out in the time increment analysis, does not work at all.

Second, the error-back-propagation neural network is trained using the relationships among the appropriate time increment and the analysis conditions. In a preliminary investigation, convergence behaviours were found to be independent of the values of R and $N.E.$ for this problem. Those relationships when under $n = 1.0$, 3.0, 5.0 7.5 and 10.0, $\bar{\sigma} = 0.5$, 0.75, 1.0, 1.25 and 1.5 are used as the training data for the neural network. The data sets under $n = 2.0$, 4.0, 6.25 and 8.75, $\bar{\sigma} = 0.625$, 0.875, 1.125 and 1.375 are used to monitor the error of the unlearned data through the training process. Thus, the neural network is trained using the relationships among the appropriate value of Δ



(a) Cavity flow of $Re = 10(epsCR = 10^{-8})$



(b) Cavity flow of $Re = 100(epsCR = 10^{-8})$

Figure 8.2.

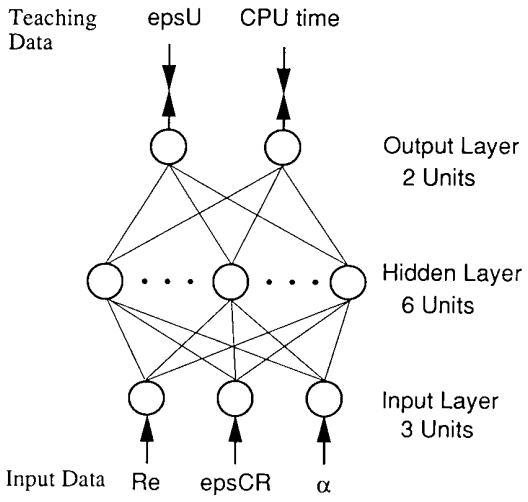


Figure 8.3. Network topology and its input/output for augmented lagrangian method

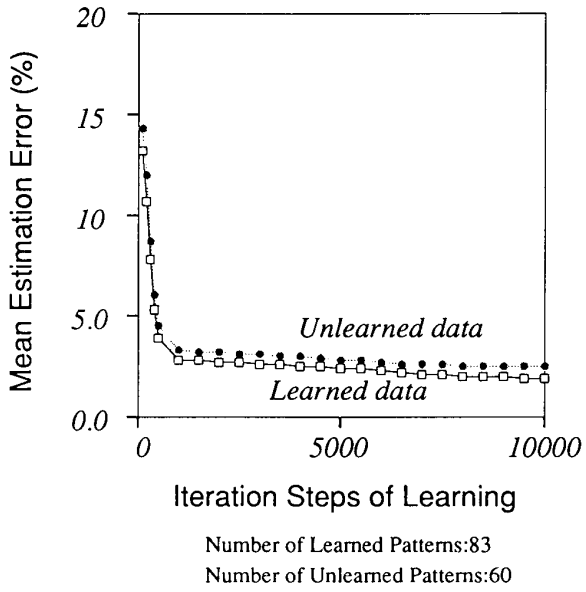


Figure 8.4. Relation between the mean estimation error and iteration steps learning

(or the appropriateness of Δt), the hardening exponent n and the uniform tensile stress $\bar{\sigma}$. The trained network is then used to estimate an appropriate time increment for unlearned analysis conditions.

Six types of network topologies are considered. They are roughly classified into three categories:

1) The appropriate value of Δt, as represented by the digital (Case 1) or the analog value (Case 2), is estimated from the analog values of n and $\bar{\sigma}$ (Figure 8.8(a)).

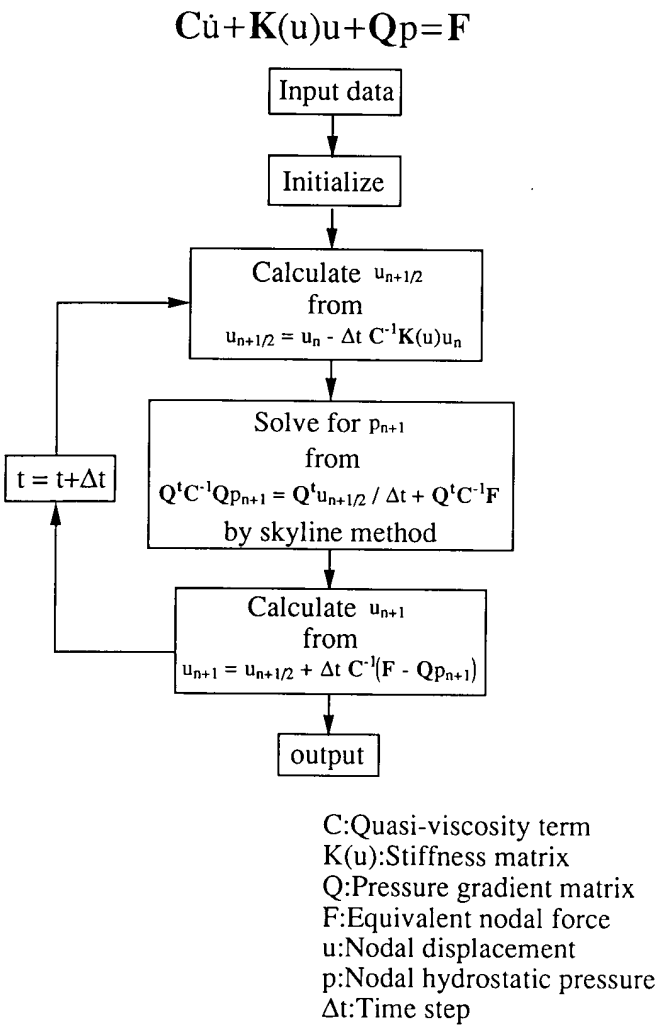
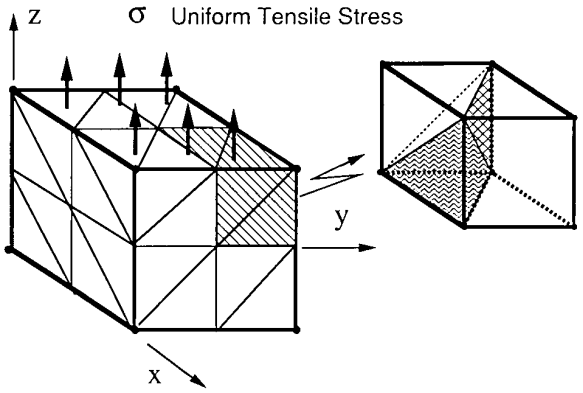


Figure 8.5. Pseudo time-dependen t stress analysis for incompressible inelastic structure

2) The net work gives the judgemen t of appropriateness of the input value of analog Δt with n and $\bar{\sigma}$ (Figure 8.8(b)). The appropriateness of analog Δt is graded into five (Case 3) or three (Case 4) lev els.
3) Input data set consists of the analog values of n and the digital Δt (Figure 8.8(c)). The appropriateness of digital Δt is graded into five (Case 5) or three (Case 6) lev els.

Figure 8.9 sho ws the estimation error for eac h netw ork topology. These estimation errors are judged to be small less than 7 %. As a result the all of the proposed network topologies can be used to evaluate appropriate Δt.

The above results indicate that the present method has an ability to estimate the optimum parameters for simple demonstration problems. It will be important to construct neural netw ork at the early stage for computations. It is also noted that the efficiency of netw ork training process and the accuracy of estimation process depend on the quantity and quality of learning data sets and the net w ork topology.



Tetrahedral element
Pressure (1st order interpolation)
Displacement (2nd order interpolation)

Number of Elements	48
Number of Nodes (Displacement)	125
Number of Nodes (Pressure)	27

Figure 8.6. Inelastic cubic material and its finite elemen t model

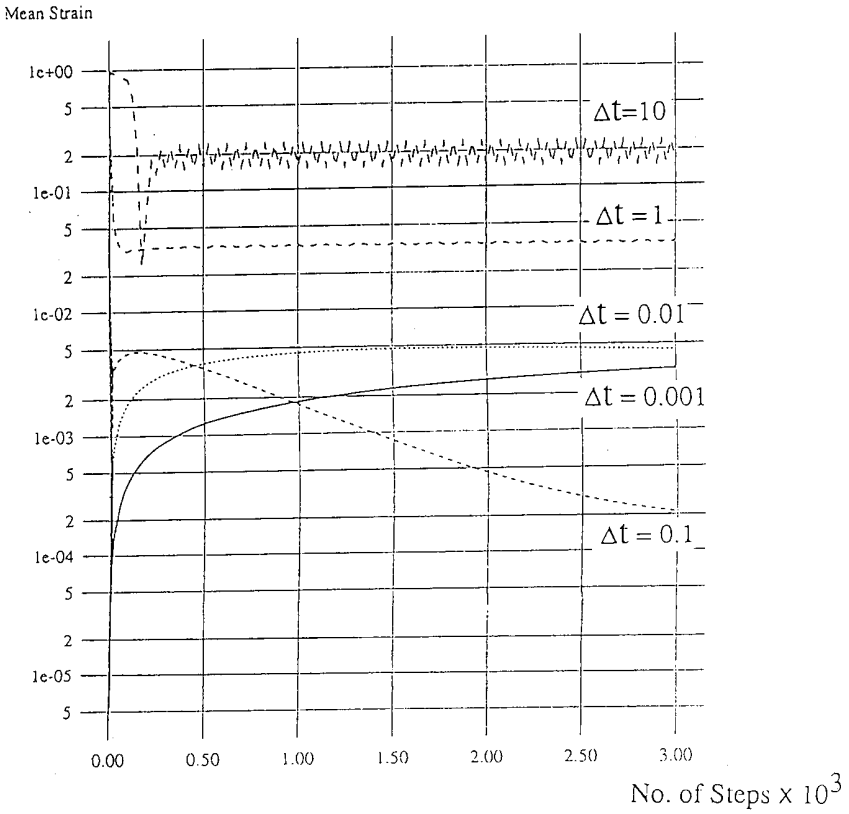
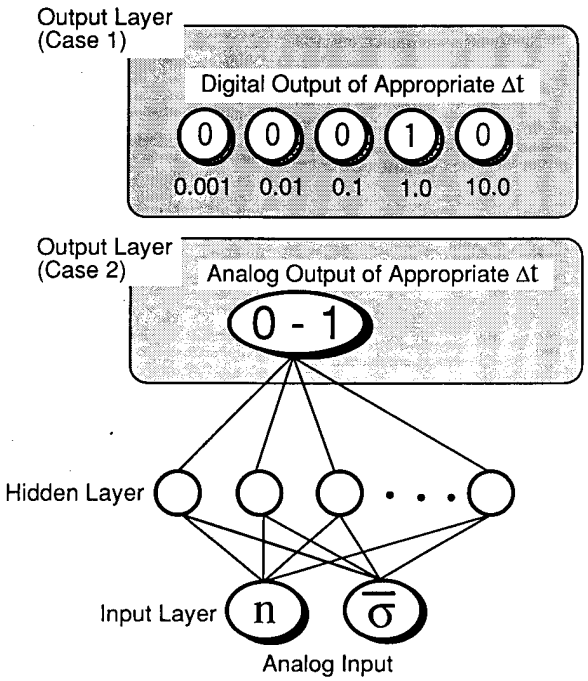
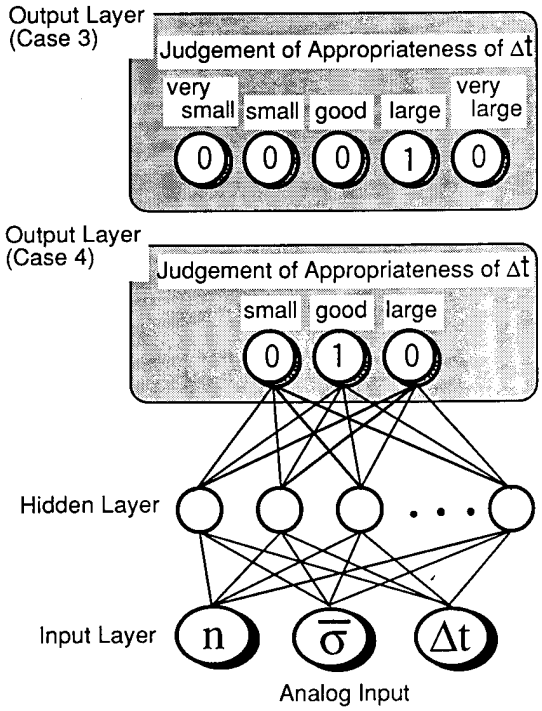


Figure 8.7. Monitored mean strain vs No. of steps

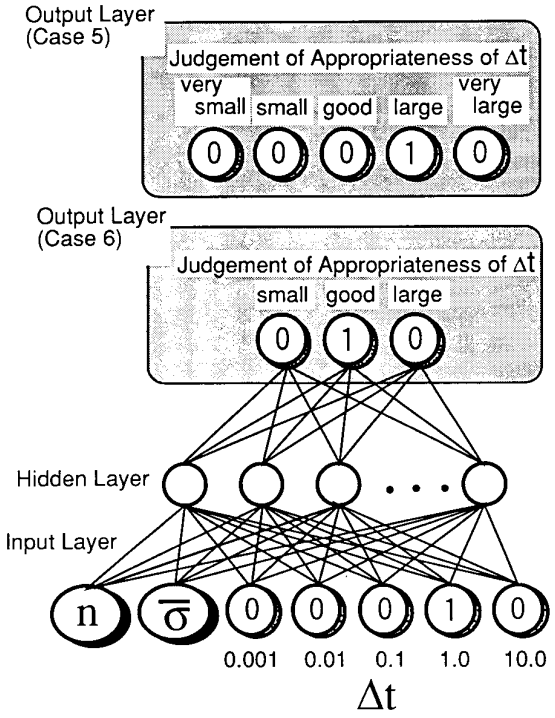


(a) Network topologies (case 1 and case 2)



(b) Network topologies (case 3 and case 4)

Figure 8.8.



(c) Network topologies (case 5 and case 6)

Figure 8.8. (*cont.*)

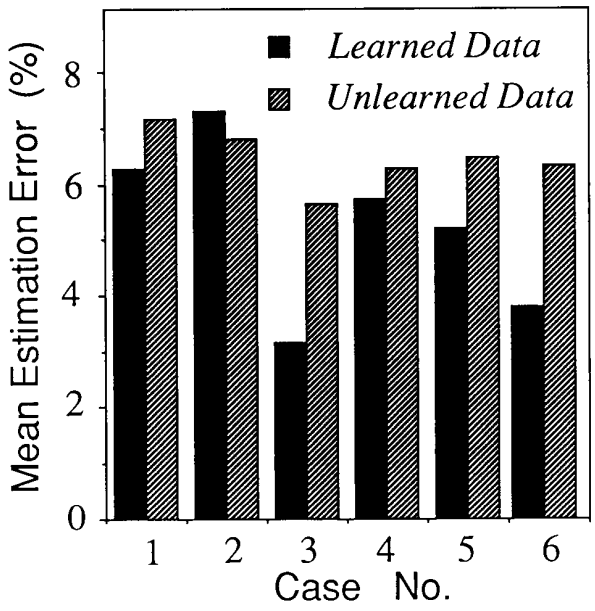


Figure 8.9. Mean estimation error for each network topology

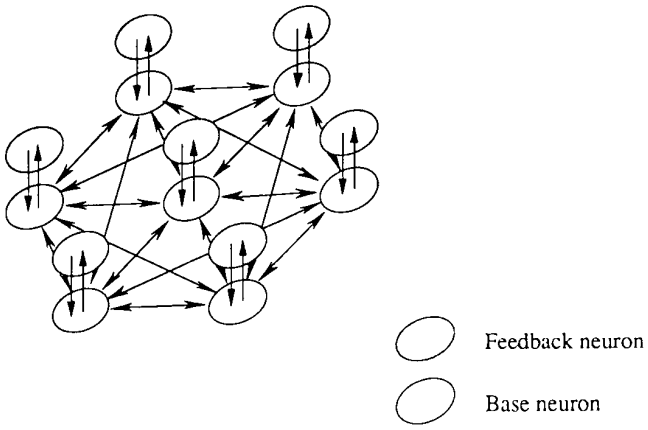


Figure 9.1. Interconnected neural network accompanying feedback neurons

9. NEURAL-NETWORK-BASED EQUATION SOLVER [73],[78]

9.1 Neural Network Solver for the Poisson's Equation

9.1.1 Feedback mechanism to train interconnettd neural network

A concept of neuron called 'feedback neuron' is introduced into the interconnected neural network. To avoid confusion, neurons without the feedback function are called 'base neurons' in the followings. Figure 9.1 schematically shows the structure of the interconnected neural network accompanying feedback neurons. In this model, each base neuron has one feedback neuron. Thus, the number of the feedback neurons is the same as that of the base neurons. The function of the feedback neuron is also formulated by Eqs.(1.1) and (1.2), except that the weights defined between the feedback neuron and its counterpart are not symmetrical. The state transition rules of the present network structure are summarized as follows:

Base neuron:

$$t_i = \sum_{\substack{j \\ (i \neq j)}} w_{ij} u_j + w_{ii'} u_{i'} + \theta_i \tag{9.1}$$

$$u_i = f_i(t_i) \tag{9.2}$$

Feedback neuron:

$$t_{i'} = w_{i'i} u_i + \theta_{i'} \tag{9.3}$$

$$u_{i'} = f_{i'}(t_{i'}) \tag{9.4}$$

where i' indicates the feedback neuron connected to its counterpart base neuron i. $w_{ii'}$ is the weight defined from i' to i, while $w_{i'i}$ is defined in the opposite direction. By virtue of the feedback neuron, the output from the base neuron is mixed with its previous value, which would help accelerating the minimization of the network energy.

According to Eq.(1.4), the net work energy is expressed as:

$$E(u) = -\frac{1}{2} \sum_{\substack{i,j \\ (i \neq j)}} w_{ij} u_i u_j - \sum_i \theta_i u_i - \frac{1}{2} \sum_i w_{ii'} u_i u_{i'} - \frac{1}{2} \sum_{i'} w_{i'i} u_{i'} u_i - \sum_{i'} \theta_{i'} u_{i'} \tag{9.5}$$

It is noted here that the third through fifth terms of the right hand side of Eq.(9.5) are contributions from the feedback neurons. Weights, offsets and activation functions in Eqs.(9.1) through (9.4), are determined such that the network energy of Eq.(9.5) is equated to a cost function of an optimization problem.

9.1.2 State transition rule for the Poisson's equation

We consider the following Poisson's equation:

$$-\Delta u = b \qquad \text{in} \quad \Omega \tag{9.6}$$

where Δ is the Laplace operator, u the unknown variable and b the source term. Both Dirichlet and Neumann type boundary conditions are imposed. The variational principle implies that the solution of Eq.(9.6) minimizes the following functional:

$$J(u) = \int_{\Omega} \left\{ \frac{1}{2} (\nabla u)^2 - bu \right\} d\Omega \tag{9.7}$$

Using an approximation procedures, e.g. the finite element method, Eq.(9.7) is expressed in a descretized form as:

$$J(u) = \frac{1}{2} \sum_{i,j} k_{ij} u_i u_j - \sum b_i u_i \tag{9.8}$$

where u_i and b_i are nodal values of u and b, respectively. k_{ij} is the "stiffness" coefficient, which has meaningful value only if there exists interactions between nodes i and j.

It is found that the functional of Eq.(9.8) can be expressed as the energy of the interconnected neural network with the feedback mechanism (Eq.(9.5)), by assuming weights, offsets and activation functions as:

For base neurons:

$$w_{ij} = w_{ji} = -Ak_{ij} \tag{9.9}$$

$$\theta_i = Ab_i \tag{9.10}$$

$$f_i(t_i) = t_i / k_{ii} \tag{9.11}$$

For feedback neurons:

$$w_{ii'} = (1 - A) k_{ii} \tag{9.12a}$$

$$w_{i'i} = -k_{ii} \tag{9.12b}$$

$$\theta_{i'} = 0 \tag{9.13}$$

$$f_{i'}(t_{i'}) = -t_{i'} / k_{ii} \tag{9.14}$$

where A is a positive parameter, which controls the magnitude of feedback for accelerating the minimization process. $A = 1$ as an extreme case implies the zero-feedback. From Eqs.(9.3), (9.4), (9.12(b)), (9.13) and (9.14), the state transition in the feedback neuron is simply written as:

$$u_{i'} = u_i \tag{9.15}$$

Equation (9.15) implies that the output from the base neuron is copied and memorized as the feedback neuron's output.

Substituting Eqs.(9.9) through (9.14) in to Eq.(9.5):

$$E(u) = \frac{1}{2} \sum_{\substack{i,j \\ (i \neq j)}} A k_{ij} u_i u_j - \sum_i A b_i u_i + \frac{1}{2} \sum_A k_{ii} u_i u_{i'} \tag{9.16}$$

Considering Eq.(9.15), the network energy is finally expressed as:

$$E(u) = A \left[\frac{1}{2} \sum_{i,j} k_{ij} u_i u_j - \sum_i b_i u_i \right] \tag{9.17}$$

It is obvious that search for a minimum point of J (Eq.(9.8)) is equivalent to search for that of E (Eq.(9.17)) if a positive constant value of A is employed. It can be shown that the network energy E decreases monotonously by the state transition if $0 < A < 2$.

As is described above, the memory function in the interconnected neural network is realized by the feedback neurons, where the extent of feedback of the previous state is controled by the parameter $A (0 < A < 2)$. Because of the difficulty in finding the theoretical optimal value of A, the value has to be determined in a heuristic manner. In order to keep the robustness of the scheme, automatic and dymamic control of A is desired. Regarding this matter, "training" of the interconnected neural network, i.e. a dynamic control of the parameter A based on the fuzzy theory will be described later. Here, we should note that the equivalence between E and J is degraded due to the variation of A. However, since the variation of A is gentle enough compared with that of the outputs of neurons, the equivalence may still hold in the practical computations.

Also, it should be mentioned that the state transition rule formulated with Eqs.(9.9) through (9.14) leads to the same operations among the outputs as the successive over-relaxation (SOR) iterations. With other choice of weights, offsets and activation functions, different type of iteration solvers could be derived. It is emphasized here that the feedback mechanism is essential to construct an equation solver by the interconnected neural network.

9.2 Application to Incompressible Flow Analysis [73]

9.2.1 Solution algorithm of incompressible flow

The interconnected neural network with feedback mechanism proposed here can be utilized in wide range of optimization problems including even an iterative equation solver. In this section, the applicability of the present neural-net approach to the incompressible viscous flow analysis is investigated.

The motion of the incompressible visous fluid is governed by the Navier-Stokes equations and the incompressible constrain t equation. Performing the finite element spatial discretizations assuming quadratic and linear profiles for velocity and pressure, respectively, the following matrix forms are obtained:

$$\mathbf{M}\dot{\mathbf{u}} + \mathbf{B}\mathbf{u} - \mathbf{C}\mathbf{p} + \mathbf{D}\mathbf{u} = \mathbf{f} \tag{9.18}$$

$$\mathbf{C}^t\mathbf{u} = \mathbf{0} \tag{9.19}$$

where $\mathbf{u}$, $\mathbf{p}$ and $\mathbf{f}$ are the global vectors consisting of the nodal velocity, the nodal pressure and the nodal external force vectors, respectively. $\mathbf{M}$, $\mathbf{B}$, $\mathbf{C}$, $\mathbf{C}^t$ and $\mathbf{D}$ are the global matrices, which stand for the mass, the advection, the gradient, the divergence and the diffusion, respectively.

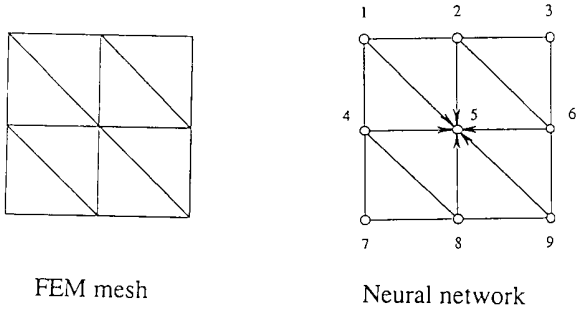


Figure 9.2. Finite elements versus neural network

One of the computational difficulties in solving the above equation system is in dealing with the incompressibility constraint. The incompressibility implies the occurrence of the infinite speed of pressure propagation and hence needs implicit treatment of pressure. In the so-called decomposition algorithms, for instance, the fractional step method [106], this difficulty is overcome by converting the incompressibility constraint equation into the Poisson's equation for pressure, which would yield the pressure field such that the velocity at the next time-step would satisfy the continuity.

Discretizing Eqs.(9.18) and (9.19), respectively, in the direction of time as:

$$\mathbf{u}^{n+1} = \mathbf{n}^n + \Delta t \mathbf{M}^{-1}(\mathbf{C}\mathbf{p}^{n+1} - \mathbf{B}\mathbf{u}^n - \mathbf{D}\mathbf{u}^n) \tag{9.20}$$

$$\mathbf{C}^t\mathbf{u}^{n+1} = 0 \tag{9.21}$$

where n is the time step. Here, external force is omitted for the sake of simplicity. In the fractional step method [106], Eq.(9.20) is decomposed into two stages as follows:

$$\tilde{\mathbf{u}} = \mathbf{u}^n + \Delta t \mathbf{M}^{-1}(-\mathbf{B}\mathbf{u}^n - \mathbf{D}\mathbf{u}^n) \tag{9.22}$$

$$\mathbf{u}^{n+1} = \tilde{\mathbf{u}} + \Delta t \mathbf{M}^{-1}\mathbf{C}\mathbf{p}^{n+1} \tag{9.23}$$

where $\tilde{\mathbf{u}}$ is the intermediate velocity. Introducing Eq.(9.21) into Eq.(9.23), the following Poisson's equation for pressure is obtained.

$$\mathbf{C}^t\mathbf{M}^{-1}\mathbf{C}\mathbf{p}^{n+1} = -\frac{\mathbf{C}^t}{\Delta t}\tilde{\mathbf{u}} \tag{9.24}$$

where $\mathbf{C}^t\mathbf{M}^{-1}\mathbf{C}$ implies the discretized Laplacian operator.

The solution algorithm is summarized as follows:

Step 1 Compute $\tilde{\mathbf{u}}$ from Eq.(9.22).

Step 2 Solve Eq.(9.24) for $\mathbf{p}^{n+1}$.

Step 3 Compute $\mathbf{u}^{n+1}$ from Eq.(9.23).

Step 4 Increment the time step and return to Step 1.

In the above algorithm, the velocity is advanced in time explicitly, while the Poisson's equation for pressure needs to be solved at each time step. Therefore, large part of the computational time is consumed in solving the Poisson's equation. Regarding nodes of pressure as neurons of network (see Figure 9.2), the neural network solver described in the previous section can be straightforwardly applied to the Poisson's equation for pressure.

9.2.2 Verification of neural network solver with feedback mechanism

Comparing with the conventional case that incorporates the conjugated gradient (CG) method as a solver, the efficiency of the present neural-net approach is studied. The minimization procedure is judged to be converged if $|E(t+1) - E(t)| < 1.0 \times 10^{-3}$.

First, a lid-driven cavity flow of $Re = 100$ is taken as an example among flows, which finally reach steady states. A square domain is regularly subdivided with 800 ($20 \times 20 \times 2$) triangular elements; that is, 441 (21×21) neurons for pressure are located at regular intervals as shown in Figure 9.3. The time increment is set to 5.0×10^{-3} sec and the computation is performed until 10 sec (2,000 steps). Figure 9.4 compares the pressure distribution. Agreements between the CG-method and the present method (the neural-net approach) is fairly good. Figure 9.5 shows the CPU-time monitored during the computations, which employ $A = 1.0$ and $A = 1.9$ for the present method. We note here again that $A = 1.0$ means the zero-feedback. On the other hand, $A = 1.9$ was heuristically specified when computing with the feedback mechanism. It is observed that the computational burden required at an early stage, i.e., up to around 500 step, is dramatically reduced by adopting the feedback mechanism.

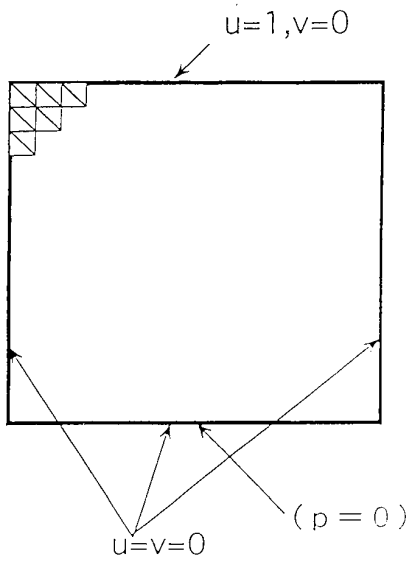
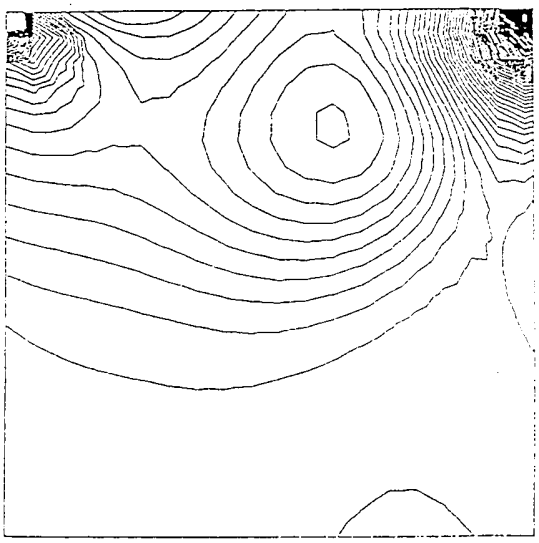


Figure 9.3. Lid-driven cavity flow: computational domain and boundary conditions (441 neurons for pressure calculation

Second example is a flow around a circular cylinder of $Re = 100$. Different from the previous example, a cyclic time-dependent flow will be obtained at a quasi-steady state. Figure 9.6 shows a finite element mesh composed of 3,600 elements. Number of neurons for pressure calculation is 1,878. The time increment is set to be 5.0×10^{-3} sec and the computation is performed until 250 sec or 50,000 steps. The value of A is set to be 1.9. Figure 9.7 shows the pressure distribution obtained by the neural-net approach. Fairly good agreements between the CG-method and the neural-net approach is also observed. The CPU-time is compared in Figure 9.8. It is found from the figure that the present neural network method is more efficient than the CG-method for the time-dependent flow considered.

(a) Present method



(b) CG method

Figure 9.4. Lid-driven cavity flow: pressure contours

9.2.3 Network training based on fuzzy control

Generally speaking, heuristic determination of parameters, which would influence the accuracy and/or the convergence of the computations, should be a voided from the sc heme in order to keep the robustness. In this section, a dynamic con trol of A based on the fuzzy theory is investigated [105]. Every five state transitions, the stability and the gradient of E's variations are judged by the membership functions depicted in Figure 9.9. Here the stability is defined as a deviation from the smoothed variation curve of E. On the other hand, the gradient is defined as an averaged decrease of E per each state transition. Increase or

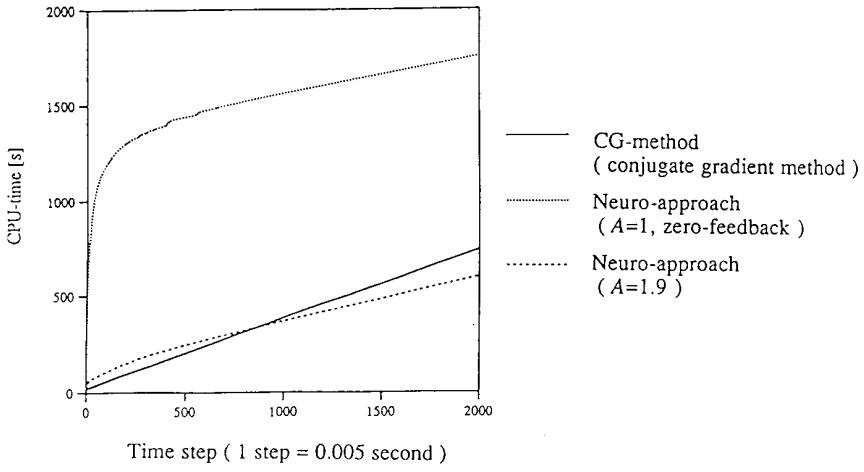


Figure 9.5. Lid-driven cavity flow: CPU-time v ersus time step

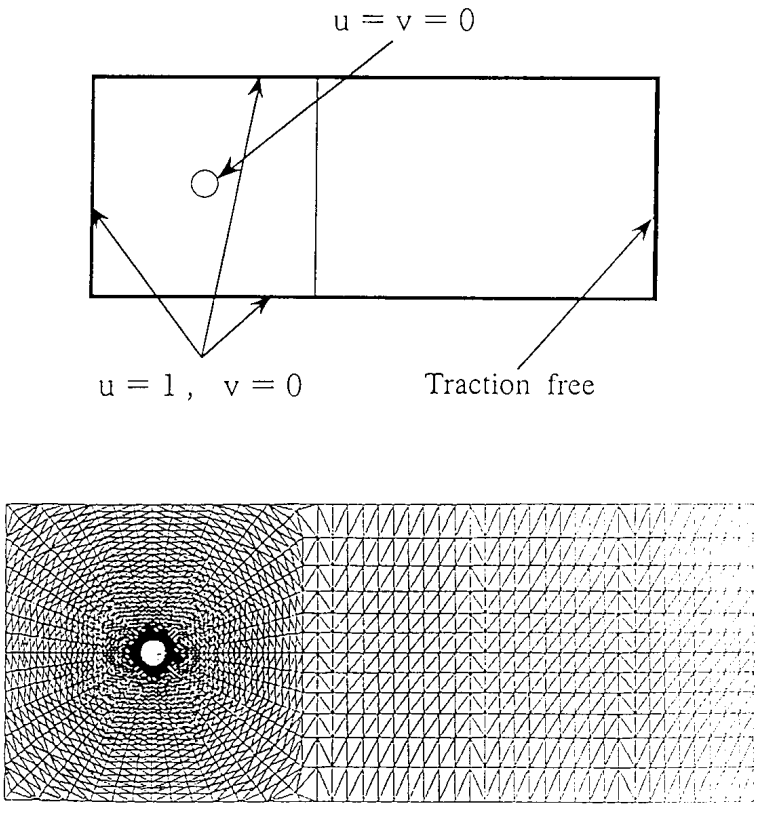
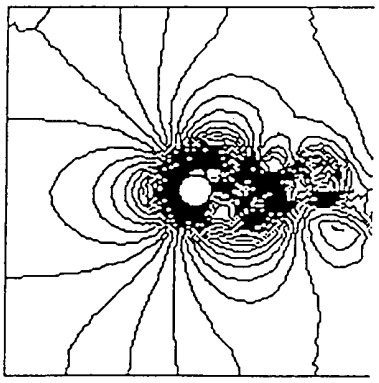
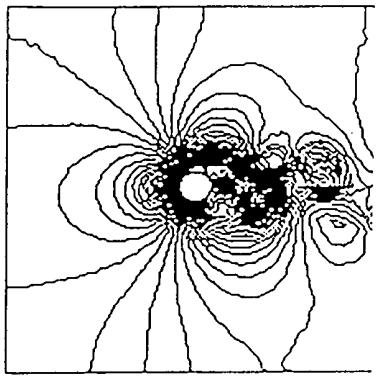


Figure 9.6. Flow around a circular cylinder: computational domain and boundary conditions (1,878 neurons for pressure calculation

decrease of A is defined according to the com binations of the smoothness and the gradien t level as shown in Table 9.1.



(a) Present method



(b) CG method

Figure 9.7. Flow around a circular cylinder: pressure contours

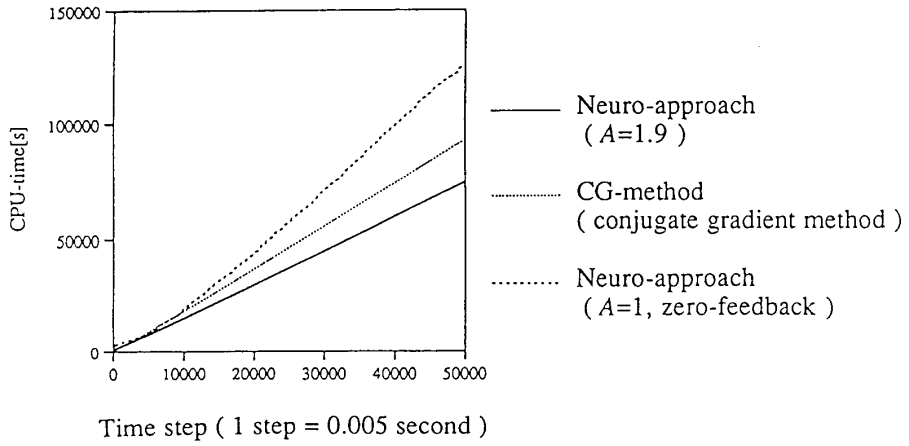


Figure 9.8. Flow around a circular cylinder: CPU-time v ersus time step

		g (Gradient)		
		steep	normal	gentle
s (Stability)	smooth	0	0.05	0.15
	normal	0	0	0.025
	rough	0	-0.025	-0.05

Table 9.1. Rule of modification of A

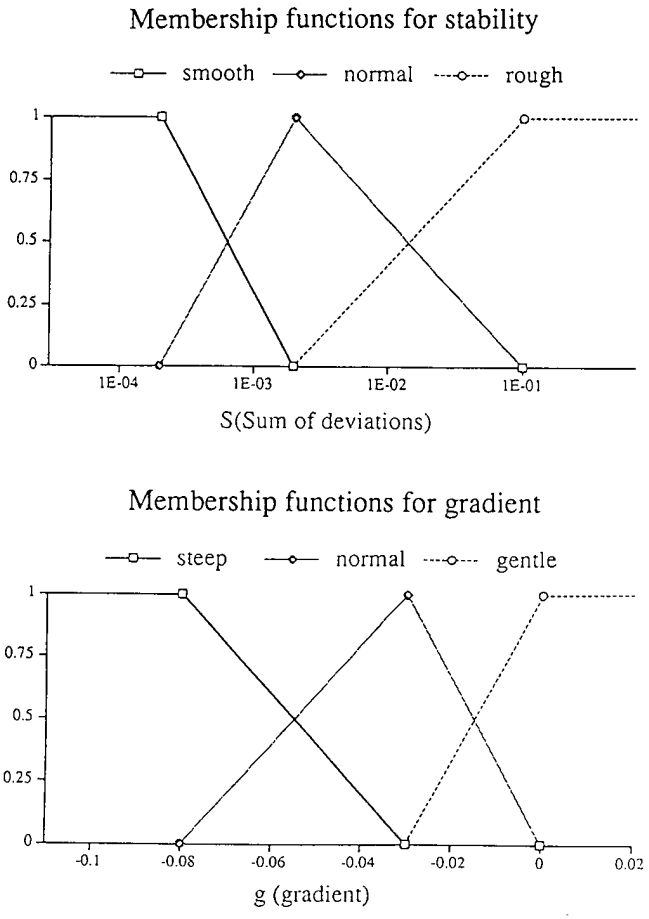


Figure 9.9. Membership functions for stability and gradient

During the minimization process, the present interconnected neural network is "trained" in a sense that the degree of memorizing the previous state is dynamically optimized by varying the weight and the offset. On the other hand, the conventional interconnected neural network is not trained, because the weight and the offset are fixed during the minimization process.

This methodology is applied to a lid-driven cavity flow of $Re = 100$. Figure 9.10 shows the CPU-time monitored during the computations. It is observed that the case with the dynamic fuzzy control of A is faster than the CG-method.

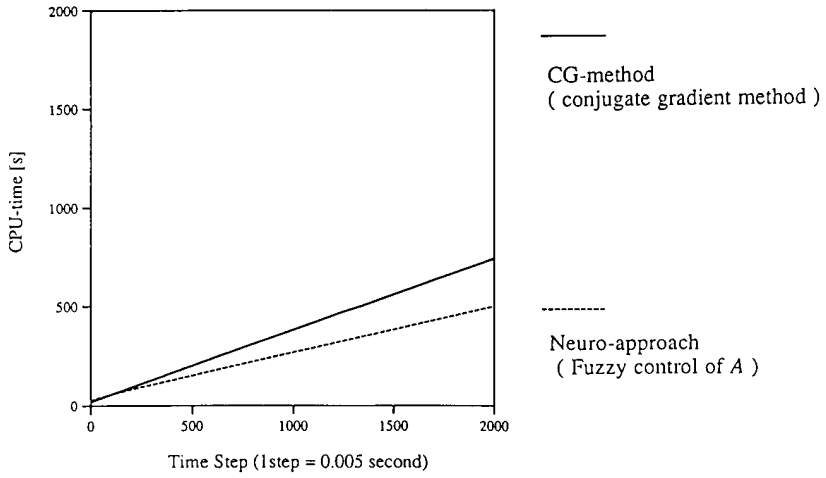


Figure 9.10. CPU-time versus time step: lid-driven cavity flow

9.3 Efficiency on a Massively Parallel Platform [78]

Independent activations of neurons are suitable for a parallel platform. Based on a domain decomposition, the present neural-net solver is executed here on a massively parallel computer AP1000, using up to 256 processors. Considered is a steady-state heat conduction problem. As shown in Figure 9.11, a square domain is descretized using triangular linear elements, where number of nodes denoting neurons for temperature, is 66,049 (257 × 257). A parallel concept based on the domain decomposition is schematically explained in Figure 9.12. Computational load of each subdomain is statically assigned to each processor. In other words, number of subdomains is equal to that of the processors employed. State transitions of subdomain are executed in parallel by communicating data of state variables between the neurons located on inter sub-domain boundaries.

Table 9.2 shows the speedup and the parallel efficiency when employing N processors, which are defined respectively as

$$\text{Speedup} = \frac{\text{CPU time using a single processor}}{\text{CPU time using } N \text{ processors}} \tag{9.25}$$

$$\text{Parallel Efficiency } (\%) = \frac{\text{Speedup}}{N} \times 100 \tag{9.26}$$

Over 90 % parallel efficiency is attained when up to 128 processors are employed. When using 256 processors, the parallel efficiency is deteriorated, because the communication overhead becomes dominant. However, improvement would be expected as the size of problem to be solved becomes larger.

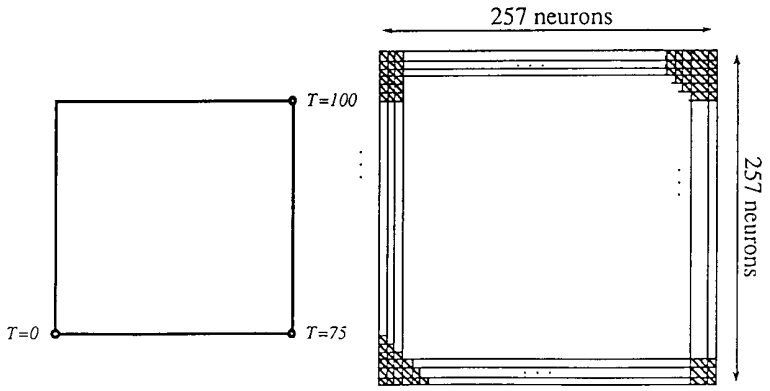


Figure 9.11. Heat condition in a square region: computational domain and boundary conditions (66 049 neurons for temperature calculation

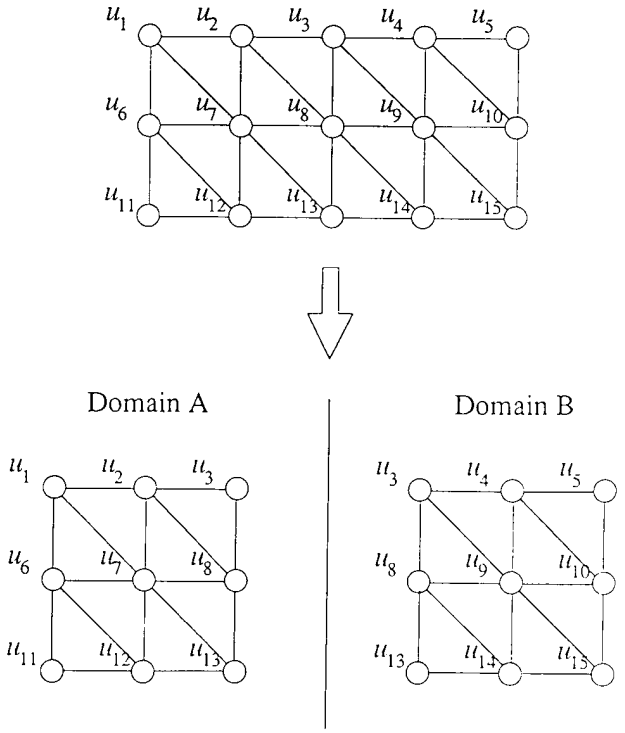


Figure 9.12. Domain decomposition

Num ber of processors N (=n um ber of domains)	4	16	32	64	128	256
Speedup	3.37	14.8	30.7	58.0	116.	179.
Parallel efficiency	0.843	0.923	0.958	0.906	0.904	0.699

Table 9.2. Speedup and parallel efficiency

10. CONCLUDING REMARKS

This paper described the neural net work applications especially to the computational mechanics related fields, i.e., structural mec hanics, nondestructive evaluation, modeling of nonlinear systems, optimization and parallel implementation. The neurological research as well as its applications to the computational mechanics, applied mec hanics and the CAE fields are remark ably developing with the adv ances of the massiv ely parallel computers. These neural-net approaches would become more promising by virtue of future hardw are development like optical nero-computers. References described in this paper do not cover all such activities, but the author w ould like to hope they would be of any use for the readers.

REFERENCES

1 Bennet, J.S. and Englemore, R.S. (1979), "SACON : A Knowledge-Based Consultan t for Structural Analysis", *Proc. 6th Int. Conf. A rtificial Intelligence*, pp. 47–49.

2 Taig, I.C. (1986), "Experts Aids to Finite Element System Applications", *Proc. 1st. Int. Conf. Applic ations of A rtificial Intelligenc e in Engineering Problems*, II-2, pp. 795–770.

3 Minsky , M.L. and P apert, S.A. (1988), *Perceptrons*, MIT Press, Cambridge.

4 Hecht-Nielsen, R. (1989), *Neur ocomputing*, Addison-W esley.

5 Dayhoff, J.E. (1990), *Neur al Network A rchitectures-A Introduction*, Van Nostrand Reinhold, New Y ork.

6 Nilsson, N. (1990), *Learning Machines*, Morgan Kaufmann Publishers, California.

7 Wasserman, P .D. (1993), *A dvanced Methods in Neural Computing*, Van Nostrand Reinhold.

8 Fausett, L. (1994), *Fundamentals of Neur al Networks: A rchitecture, A lgorithms and Applic ations*, Pretice-Hall, Inc., New Jersey .

9 Hopfield, J.J. and T ank, D.W. (1985), "Neural Computation of Decisions in Optimization Problems", *Biol. Cybern.*, **52**, pp. 141–152.

10 Rumelhalt, D.E., Hinton, G.E. and Williams, R.J. (1986), "Learning Representations by Back-Propagating Errors", *Nature*, **329**, 9, pp. 533–536.

11 Fukuda, T. and Shibata, T. (1992), "Theory and Applications of Neural Networks for Industrial Con trol Systems", *IEEE T rans Ind Electron*, **39**, pp. 472–489.

12 Fukuda, T. and Shibata, T. (1991), "Research Trends in Neuromorphic Con trol", *J. Robotics Mechatron.*, **2**, 4, pp. 4–18.

13 Widro w, B. and Lehr, M.A. (1990), "Thirty Years of Adaptive Neural Net works:Perceptron, Madaline, and Bac k-Propagation", *Proc. IEEE*, **78**, 9, pp. 1415–1441.

14 Lippmann, R.P . (1987), "An Introduction to Computing with Neural Nets", *IEEE, ASSP Mag.*, **4** , pp. 4–22.

15 Amari, S. (1993), "A Universal Theorem on Learning Curv es", *Neur al Networks*, **6**, pp. 161–166.

16 Biegler-Koenig, F. and Baermann, F. (1993), "Learning Algorithm for Multilayered Neural Net works Based on Linear Least Squares Problems", *Neur al Networks*, **6**, pp. 127–131.

17 Kara yiannis, N.B. and V enetsanopoulos, A.N. (1992), "F ast learning algorithms for neural networks", *IEEE T ans Circuits Syst. II A nalog Digital Signal Pr ocess*, **39**, pp. 453–474.

18 Du, L.M., Hou, Z.Q. and Li, Q.H. (1992), "Optimum Block-Adaptive Learning Algorithm for Error-Back-Propagation Networks" *IEEE Trans Signal Process*, **40**, pp. 3032–3042.

19 Weymaere, N. and Martens, J.P (1994), "On the Initialization and Optimization of Multilayer Perceptrons", *IEEE Transactions on Neural Networks*, **5**, pp. 738–751.

20 Yu, X., Loh, N.K. and Miller, W.C., (1993), "New Acceleration Technique for the Backpropagation Algorithm", *IEEE International Conference on Neural Networks IEEE Service Center*, San Francisco, California, USA, pp. 1157–1161.

21 Mehrotra, K.G. (1991), "Bounds on the Number of Samples Needed for Neural Learning", *IEEE Transactions on Neural Networks*, **2**, pp. 548–558.

22 Wedd, A.R. (1994), "Functional Approximation by Feed-Forward Networks: A Least-Squares Approach to Generalization", *IEEE Transactions on Neural Networks*, **5**, pp. 363–371.

23 Jean, J.S.N. and Wang, J. (1994), "Weight Smoothing to Improve Network Generalization", *IEEE Transactions on Neural Network*, **5** , pp. 752–763.

24 Cun, Y.L. (1992), "Improving Generalization Performance Using Double Backpropagation", Harris Drucker, *IEEE Transactions on Neural Networks*, **3**, pp. 991–997.

25 Yoshimura, S., Matsuda, A. and Yagawa, G. (1994), "A New Regularization Method for Neural-Network-Based Inverse Analyses and its Application to Structural Identification", Inverse Problems in Engineering Mechanics, *Proc. 2nd ISIP*, pp. 461–466.

26 Mukhopadhyay, S. and Narendra, K.S. (1993), "Disturbance Rejection in Nonlinear Systems Using Neural Networks" *IEEE Transactions on Neural Networks*, **4**, pp. 63–72.

27 Feuer, A. and Cristi, R. (1993), "On the Optimal Weight Vector of a Perceptron with Gaussian Data and Arbitrary Nonlinearity", *IEEE Trans Signal Process*, **41**, pp. 2257–2259.

28 Lee, S. and Kil, R.M (1994), "Inverse Mapping of Continuous Functions Using Local and Global Information", *IEEE Transactions on Neural Network*, **5**, pp. 409–423.

29 Levy, H.J. and McGill, T.C. (1993), "Feedforward Artificial Neural Network Based on Quantum Effect Vector-Matrix Multipliers", *IEEE Transactions on Neural Networks*, **4**, pp. 427–433.

30 Zhang, Q. and Benveniste, A. (1992), "Wavelet Networks" *IEEE Transactions on Neural Networks*, **3**, pp. 889–898.

31 Manoel, F. and Lee, W.T. (1990), "Self-Organizing Network for Optimum Supervised Learning", *IEEE Transactions on Neural Networks*, **1**, pp. 100–110.

32 Berke, L. and Hajela, P. (1992), "Applications of Artificial Neural Nets in Structural Mechanics", *Structural Optimization*, **4**, pp. 90–98.

33 Tanaka, M. and Hanahara, K. (1992), "Criterion Acquisition for the Motion Planning of Adaptive Truss (Implementation by Modular Neural Network)", (in Japanese), *Trans. JSME*, 58-550 (C), pp. 1735–1741.

34 Chen, R. M. M. and Chan, W. W. (1993), "An Efficient Tolerance Design Procedure for Yield Maximization Using Optimization Techniques and Neural Network", *Proc. IEEE Int. Symp. Circuits Syst.*, **3**, pp. 1793–1796.

35 Tsutsumi, K., Tani, A., Kawamura, H., Matsubayashi, T., Kataoka, H., Komono, K. and Fukushima, Y, (1993), "Study of a Computer System for Evaluation of Bearing Wall Stresses in Structural Planning Reasoning by Neural Network, Computing in Civil and Building Engineering", *Proc 5 Int. Conf. Comput. Civ. Build. Eng. V ICCCBE*, pp. 817–824.

36 Oda, J., Mizuk ami, T. and Hattori, M. (1992), "Technique for Estimation of Elastic Con tact Stress Distributions by Neural Net work", (in Japanese), *Trans. JSME*, *58-552(A)*, pp. 1524–1529.

37 Ben (O-il By on), G. and Nishi, Y. (1992), "An Application of a Neural Network to CFRP Elastic Coefficien t Design", (in Japanese), T rans. JSME, *58-548 (A)*, pp. 539-543.

38 Mochizuki, Y., Y oshim ura, S. and Yagawa, G. (1993), "Automated Structural Design System Based on Design Window Search Approach: Its Application to ITER First Wall Design", *Proc. of SMiR T-12 Post Conf. Seminar* **13**, pp. 465–474.

39 Yuuki, R. and T amaki, M. (1993), "Learning and Generation of BEM Adaptive Meshes b y Using Neural Net work", (in Japanese), *Trans. JSME*, **59-558 (A)**, pp. 489–495.

40 Yoshim ura, S. and Yagawa, G. (1993), "In verse Analysis by Means of Neural Network and Computational Mec hanics: Its Application to Structural Iden tification of Vibrating Plate", In verse Problems, S. Kubo (Eds.) Atlanta Technology Publications, pp. 184–193.

41 Yagawa, G., Matsuda, A., Kawate, H. and Y oshim ura, S. (in Print), "Neural Net work Approach to Estimate Stable Crack Growth in W elded Specimens", *International Journal of Pressure Vessels and Piping.*

42 Ghaboussi, J., Garrett J.H. Jr. and W u, X. (1991), "Kno wledge-Based Modeling of Material Behavior with Neural Net works", *Journal of Engineering Me chanics*, **117**, pp. 132–153.

43 W u, X., Ghaboussi, J. and Garrett, J.H. (1992), "User of Neural Networks in Detection of Structural Damage", *Computers and Structur es*, **42**, pp. 649–659.

44 Yamamoto, K. (1991), "Study on Applications of Neural Networks to Non-Linear Structural Analysis (P art-1) Modeling of Hysteretic Behavior and its Application to Non-Linear Dynamic Response Analysis", (in Japanese), *CRIEPI R eport* **U91046**.

45 Okuda, H., Miyazaki, H. and Yagawa, G. (1994), "A Neural Network Approach for Modelling of Viscoplastic Material Behaviors, Adv anced Computer Applications", *ASME/PVP*, Vol. **274**, pp. 141–145.

46 Yoshim ura, S., Hishida, H. and Y agawa, G. (1992), "P arameter Optimization of Viscoplastic Constitutive Equation Using Hierarc hical Neural Net work", *VII International Congr ess of Experimental Me chanics*, **1**, pp. 296–301.

47 Tank, D.W. and Hopfield, J.J. (1986), "Simple Neural Optimization Networks: An A/D Converter, Signal Decision Circuit, and a Linear Programming Circuit", *IEEE T rans. Circuits Syst.* Vol. **CAS-33**, pp. 533–541.

48 Yagawa, G. and Y oshim ura, S. (1993), "Neural Net work Approach to Estimate Crack Growth in W elded Specimens", *Proc. Seminar Post-SMiRT, No. 5 Inelastic A nalysis, Fatigue, Fracture and Life Prediction*, pp. 22–35.

49 Upda, L. and Upda, S.S. (1990), "Eddy Current Defect Characterization Using Neural Networks", *Materials Evaluation*, **48**, pp. 342–347.

50 Kitahara, M., Achenbach, J.D., Guo, Q.C., Peterson, M.L., Notak e, M. and T adokoya, M. (1992), "Neural Net work for Crack-Depth Determination from Ultrasonic Backscattering Data", *Review of Progress in Quantitative Nondestructive Evaluation*, **11**, pp. 701–708.

51 Ogi, T., Notake, M. Yabe, Y. and Kitahara, M. (1991), "Application of Neural Network to Classification of Defects: Basic Study of Weight", *Review of Progress in Quantitative Nondestructive Evaluation*, **10**, pp. 683–688.

52 Kitahara, M., Achenbach, J.D., Guo, Q.C., Peterson, M.L., Ogi, T. and Notake, M. (1991), "Depth Determination of Surface-Breaking Cracks by a Neural Network", *Review of Progress in Quantitative Nondestructive Evaluation*, **10**, pp. 689–696.

53 Brown, L.M., Newman, R.W., DeNale, R., Lebowitz, C.A. and Arcella, F.G. (1992), "Graphite Epoxy Defect Classification of Ultrasonic Signatures Using Statistical and Neural Network Techniques", *Review of Progress in Quantitative Nondestructive Evaluation* **11 A**, pp. 677–684.

54 Yoshimura, S., Yagawa, G., Oishi, A. and Yamada, K. (1993), "Quantitative Defect Identification By Means of Neural Network and Computational Mechanics", *3rd Japan International SAMPE Symposium*, pp. 2263–2268.

55 Yagawa, G., Yoshimura, S., Mochizuki, Y. and Oichi, T. (1992), "Identification of Crack Shape Hidden in Solid by Means of Neural Network and Computational Mechanics", Inverse Problems in Engineering Mechanics, *Proc. IUTAM Symp. Tokyo*, pp. 213–222.

56 Song, S.J. and Schmerr, L.W. Jr., (1991), "Ultrasonic Flaw Classification in Weldments Using Neural Networks", *Review of Progress in Quantitative Nondestructive Evaluation*, **10**, pp. 697–704.

57 Brown, L.M. and DeNale, R. (1991), "Classification of Ultrasonic Defect Signatures Using an Artificial Neural Network", *Review of Progress in Quantitative Nondestructive Evaluation*, **10**, pp. 705–712.

58 Birx, D. and Pipenberg, S. (1991), "A Complex Multi-Layer Neural Network for Defect Detection and Characterization", *Review of Progress in Quantitative Nondestructive Evaluation*, **10**, pp. 713–718.

59 Elshafiey, I., Upda, L. and Upda, S.S. (1992), "A Neural Network Approach for Solving Inverse Problems in NDE", *Review of Progress in Quantitative Nondestructive Evaluation*, **11 A**, pp. 709–716.

60 Ramamurthy, A.C. (1993), "Stone Impact Damage to Automotive Paint Finishes A Neural Net Analysis of Electro-Chemical Impedance Data", *1993 IEEE International Conference on Neural Networks IEEE Service Center*, San Francisco, California, USA, pp. 1708–1712.

61 Shahla, K. (1993), "Evaluation of the Performance of Various Artificial Neural Networks to the Signal Fault Diagnosis in Nuclear Reactor Systems", *1993 IEEE International Conference on Neural Networks IEEE Service Center*, San Francisco, California, USA, pp. 1719–1723.

62 Chang, G.W. and Chang, P.R. (1993), "Neural Plant Inverse Control Approach to Color Error Reduction for Scanner and Printer", *1993 IEEE International Conference on Neural Networks IEEE Service Center*, San Francisco, California, USA, pp. 1979–1983.

63 Kim, S.W. and Lee, J.J. (1993), "Unknown Parameter Identification of Parameterized System Using Multi-layered Neural Networks", *1993 IEEE International Conference on Neural Networks IEEE Service Center*, San Francisco, California, USA, pp. 438–443.

64 Saravanan, N., Duyar, A., Guo, T.H. and Merrill, W.C. (1993), "Modeling of the Space Shuttle Main Engine Using Feed-Forward Neural Networks", American Control conference, *Proc. 1993 Am Contr ol Conf 1993*, pp. 2897–2899.

65 Rico-Martinez, R. (1993), "Continuous Time Modeling of Nonlinear Systems: A Neural Network-Base Approach", *1993 IEEE International Conference on Neural Networks IEEE Service Center*, San Francisco, California, USA, pp. 1522-1525.

66 Yagawa, G. Yoshimura, S. and Okuda, H. (1994), "Neural Network Based Parameter Optimization for Nonlinear Finite Element Analyses", *Extended Abstracts, 2nd Japan-US Symposium on Finite Element Methods in Large-Scale Computational Fluid Dynamics*, pp. 83–86.

67 Maa, C.Y. and Schanblatt, M.A. (1992), "A Two-Phase Optimization Neural Network", *IEEE Transactions on Neural Networks*, **3**, pp. 1003–1009.

68 Zhang, S., Zhu, X. and Zou, L.H. (1992), "Second-Order Neural Nets for Constrained Optimization", *IEEE Transactions on Neural Networks*, **3**, pp. 1021–1024.

69 Sun, K.T. and Fu, H.C. (1993), "Hybrid Neural Network Model for Solving Optimization Problems", *IEEE Trans. Comput.*, **42**, pp. 218–227.

70 Francelin, R.A. and Gomide, F.A. (1993), "A Neural Network to Solve Discrete Dynamic Programming Problems", *1993 IEEE International Conference on Neural Networks IEEE Service Center*, San Francisco, California, USA, pp. 1433–1525.

71 Bouzerdoum, A. and Pattison, T.R. (1993), "Neural Network for Quadratic Optimization with Bound Constraints", *IEEE Transactions on Neural Networks*, **4**, pp. 293–304.

72 Ling, P. and Jin, K. (1993), "Solving Search Problems with Subgoals Using an Artificial Neural Network", *1993 IEEE International Conference on Neural Networks IEEE Service Center*, San Francisco, California, USA, pp. 81–86.

73 Yagawa, G. and Okuda, H. (1996), "Finite Element Solutions with Feedback Network Mechanism Through Direct Minimization of Energy Fuctionals", *Int. J. Num. Meth. in Eng.*, Vol. **39**, pp. 867–883.

74 Kung, S.Y. and Chou, W.H. (1993), "*Mapping Neural Networks onto VLSI Array Processors*", Parallel Digital Implementation of Neural Networks, PTR Prentice Hall, pp. 3–49.

75 Zell, A., Mache, N., Vogt, M. and Huttel, M. (1993), "Problems of Massive Parallelism in Neural Network Simulation", *1993 IEEE International Conference on Neural Networks IEEE Service Center*, San Francisco, California, USA, pp. 1890–1895.

76 Grajski, K.A. (1993), "*Neurocomputing Using the Maspar MP-1*", Parallel Digital Implementation of Neural Networks, PTR Prentice Hall, pp. 51–76.

77 Majumder, A. and Dandapani, R. (1993), "Neural Networks as a Massively Parallel Automatic Test Pattern Generators", *1993 IEEE International Conference on Neural Networks IEEE Service Center*, San Francisco, California, USA, pp. 1724–1730.

78 Yagawa, G. and Aoki, O. (1993), "Neural Network-Based Direct FEM on a Massively Parallel Computer AP1000", *Proc. of the Second Parallel Computing Workshop*, Nov. 11, P1-K-1 K-11.

79 Botros, N.M. and Abdul-Azia, M. (1993), "Hardware Implementation of an Artificial Neural Network", *1993 IEEE International Conference on Neural Networks IEEE Service Center*, San Francisco, California, USA, pp. 1252–1257.

80 Okawa, Y. (1993), "Parallel Computation of Neural Networks in a Processor Pipeline with Partially Shared Memory", *1993 IEEE International Conference on Neural Networks IEEE Service Center*, San Francisco, California, USA, pp. 1638–1643.

81 Fujimoto, Y., Fukuda, N. and Akabane, T. (1992), "Massively Parallel Architectures for Large Scale Neural Network Simulations" *IEEE Transactions on Neural Networks*, **3**, pp. 876–888.

82 Misra, M. and Prasanna, V.K. (1992), "Implementation of Neural Networks on Massive Memory Organizations", *IEEE Trans. Circuits Syst. II Analog digital Signal Process.*, **39**, pp. 476–480.

83 Witbrock, M. and Zagha, M. (1993), "*Back-Propagation Learning on the IBM GF11*", Parallel Digital Implementation of Neural Networks, PTR Prentice Hall, pp. 77–104.

84 Hammerstrom, D., Henry, W. and Kuhn, M., (1993), "*A Neurocomputer System for Neural-Network Applications*", Parallel Digital Implementation of Neural Networks, PTR Prentice Hall, pp. 107–138.

85 Iwata, A. (1993), "*An Artificial Neural-Network Accelerator Using DSP Chips*", Parallel Digital Implementation of Neural Networks, PTR Prentice Hall, pp. 139–173.

86 Means, R.W. and Lisenbee, L. (1993), "*Floating-Point SIMD Neurocomputer Array Processor*", Parallel Digital Implementation of Neural Networks, PTR Prentice Hall, pp. 175–195.

87 Morgan, N., Beck, J., Kohn, P. and Bilmes, J. (1993), "*Neurocomputing on the RAP*", Parallel Digital Implementation of Neural Networks, PTR Prentice Hall, pp. 197–219.

88 Burr, J.B. (1993), "*Digital Neurochip Design*", Parallel Digital Implementation of Neural Networks, PTR Prentice Hall, pp. 223–282.

89 Hirai, Y. (1993), "*A PDM Digital Neural-Network System*", Parallel Digital Implementation of Neural Networks, PTR Prentice Hall, pp. 283–311.

90 Chaboche, J.L. and Rousselier, G. (1983), "On the Plastic and Viscoplastic Equations", *ASME, Journal of Pressure Vessel Technology*, **105**, p. 153.

91 Subcommittee on Inelastic Analysis and Life Prediction of High Temperature Materials, The Society of Materials Science, (in Japanese), (1988), "Bench Mark Project on Inelastic Deformation and Life Prediction of 2 1/4Cr-1Mo Steel at High Temperature", Japan.

92 Rice, J. R., (1968), "A Path-Independent Integral and the Approximate Analysis of Strain Concentration by Notches and Cracks", *Trans. ASME, Journal of Applied Mechanics*, **35**, pp. 376-386.

93 Yagawa, G., Takahashi, Y. and Ueda, H., (1985), "Three-Dimensional Fully Plastic Solutions for Plates and Cylinders with Through-Wall Cracks", Transactions of the *ASME, Journal of Applied Mechanics*, **52**, pp. 319–325.

94 Yagawa, G. and Ishihara, K. (1989), "Cleavage and Ductile Thermal Shock Fractures of Corner-Cracked Nozzles", Transactions of the *ASME, Journal of Pressure Vessel Technology*, **111**, pp. 241–246.

95 Ernst, H.A., Paris, P.C. and Landes, J.D. (1981), "Estimations on J-integral and Tearing Modulus T from a Single Specimen Test Tecord", *ASTM STP* **743**, 476–502.

96 Kumar, V., German, M.D. and Shih, C.F. (1981), "An Engineering Approach for Elastic-Plastic Fracture Analysis", NP-1391, Project 1237-1. Topical Report, EPRI.

97 Bennett, J.A. and Botkin, M.E. (eds.), (1986), "The Optimum Shape (Automated Structural Design)", Plenum Press.

98 Smith, R.G. (1983), "STROBE: Supported or Structural Object Knowledge Representation", *Proc. 8th Int. Joint Conf. on Artificial Intelligence*, pp. 855–858.

99 Dennis, J.B. (1974), "First Version of a Data Flow Procedure Language", *Lecture Notes in Computer Science*, **19**, pp. 362–376.

100 IAEA (1991), "ITER Conceptual Design Report", *ITER Documentation Series*, **18**.

101 Pelletier, D., Fortin, A. and Camarero, R. (1989), "Are FEM Solutions of Incompressible Flows Really Incompressible? (or how simple flows can cause headaches!)", *Int. J. Numer. Meth. in Fluids*, **9**, pp. 99–112.

102 Eguchi, Y., Yagawa, G. and Fuchs, L. (1988), "A Conjugate-Residual-FEM for Incompressible Viscous Flow Analysis", *Compt. Mech*, **3**, pp. 59–72.

103 Fortin, M. and Glowinski, R. (1993), *Augmented Lagrangian Methods: Applications to Numerical Solution of Boundary-Value Problems*, North-Holland, New York.

104 Pyo, C.R., Yagawa, G., Kawai, H. and Yoshimura, S. (1993), "Finite Element Analysis of Three Dimensional Fully Plastic Solutions by Means of Quasi-Nonsteady Algorithm", *Proc. 1993 ASME-PVP Conf.*, **265**, pp. 277–282.

105 Takahashi, R. (1991), "*Thermal Hydraulic Computation by Fuzzy Reasoning, in Fuzzy Theory and Applied Mechanics*", (in Japanese), (Ed. G. Yagawa), Baifukan, 9/52.

106 Donea, J., Giuliani, S., Laval, H. and Quartapelle, L. (1982), "Finite Element Solution of the Unsteady Navier-Stokes Equations by a Fractional Step Method", *Computer Methods in Applied Mechanics and Engineering* **30**, pp. 53–73.

Please address your comments or questions on this paper to:
International Center for Numerical Methods in Engineering
Edificio C-1, Campus Norte UPC
Gran Capitán s/n
08034 Barcelona, Spain
Phone: 34-3-4016035; Fax: 34-3-4016517