NASA Technical Memorandum 4389

Application of Artificial Neural Networks to the Design Optimization of Aerospace Structural Components

Laszlo Berke
Lewis Research Center
Cleveland, Ohio

Surya N. Patnaik
Ohio Aerospace Institute
Brook Park, Ohio

Pappu L. N. Murthy
Lewis Research Center
Cleveland, Ohio

National Aeronautics and
Space Administration
Office of Management
Scientific and Technical
Information Program
1993

Summary

The application of artificial neural networks to capture structural design expertise is demonstrated. The principal advantage of a trained neural network is that it requires trivial computational effort to produce an acceptable new design. For the class of problems addressed, the development of a conventional expert system would be extremely difficult. In the present effort, a structural optimization code with multiple nonlinear programming algorithms and an artificial neural network code NETS were used. A set of optimum designs for a ring and two aircraft wings for static and dynamic constraints were generated by using the optimization codes. The optimum design data were processed to obtain input and output pairs, which were used to develop a trained artificial neural network with the code NETS. Optimum designs for new design conditions were predicted by using the trained network. Neural net prediction of optimum designs was found to be satisfactory for most of the output design parameters. However, results from the present study indicate that caution must be exercised to ensure that all design variables are within selected error bounds.

Introduction

The nervous system from slugs to humans follows the same basic design: neurons connected to many other neurons forming a biological neural network. The difference between extremes, such as slugs with only dozens of neurons and humans with around billions, is the number and complexity of the connectivities. Their organization and diversity allow for the specialization of the various areas of this massively parallel, information-processing, living tissue. The human brain is the ultimate technology with respect to miniaturization and processing power. The majority of our neurons and their connections reside in the cerebral cortex, the seat of most of our intellectual capabilities. The cerebral cortex, in physical terms, is the size of a six-page newspaper, no more than 1/4 in. thick, and crumpled up for packaging in its protective hard cover. Other parts of the 3-pound wet tissue perform hard-wired life support functions, including quick-response emotions, inherited from our reptilian ancestors. At a critical number of neurons and their connectivities, awareness and cognition emerged, beginning with our human ancestors millions of years ago. All living creatures exhibit instinctive or some level of cognitive reaction to input, responding to feeding opportunity or engaging in threat avoidance. This cognitive tissue has hard-wired, programmable, and self-organizing capabilities and it is trainable. It has been the subject of intense studies on its anatomy and physiology to its capabilities and the way it does what it does. We have learned much about the electrochemical activity that occurs in the nervous system, but the way in which the measurable physical activities acquire meaning for us is not known now, and is not likely to be known in the foreseeable future. The cerebral cortex has evolved to perform certain tasks better than others. Vision, for example, is such that a thousand Cray Y-MP's would have difficulty modeling the same real-time fidelity and perception of meaning. At the same time, this ultimate technology cannot come close to the arithmetic capabilities of a credit card sized calculator.

Biotechnology and rapidly advancing computer science have motivated the introduction of increasingly sophisticated artificial neural network models of intriguing brain functions both in hardware and in software implementations. Vision, perception, natural language understanding, classification, associative memory, learning, and accumulation of expertise are some targets of this activity. The artificial neural network (ANN) research is truly multidisciplinary, encompassing neurobiology, physiology, psychology, medical science, mathematics, computer science, and engineering.

As in computer science, advancements in ANN are progressing at a rapid pace. In current conventional ANN applications, neurons number only in the hundreds with their connectivities limited to a few thousand. These numbers will approach millions or greater in the near future with enhancements in computational technology, promising capabilities approaching lower order living entities. Neural net models of learning and the accumulation of expertise in a narrow domain have found their way into practical applications in many areas. ANN is being attempted for business applications as a profit-making tool to perform jobs of loan officers, tax auditors, and stock market experts. Industrial applications are growing also.

Neural net literature is diverse; only a small sampling can be given here (Garret, J.H., Jr. et al., 1993, J. Intel. Man., to be published and (refs. 1–4)). Rumelhart and NcCleland (ref. 1) provide a fundamental introduction to the theory of ANN. The use of a trained neural network as an expert structural designer was suggested by Berke and Hajela and is illustrated at a "toy" problem level (ref. 4). As in structural optimization, using mathematical programming techniques, current neural net capabilities appear to have major limitations in problem size, especially in the number of variables used in the mathematical model. The objective of the investigation reported here was to further explore the applicability of ANN when the problem size was computationally feasible for conventional structural optimization.

The expert ANN design model considered here is based on feed-forward, supervised learning and an error back-propagation training algorithm. This is the simplest and most popular ANN paradigm. More sophisticated approaches involving clustering and classification of data (ref. 5) or other candidate

approaches, such as functional links (ref. 6) or radial base functions (RBF), are under investigation at this time. An ANN is trained first, by utilizing available information generated from several similar optimum designs of aerospace structural components. The trained artificial neural network, as the expert designer, is then used to predict an optimum design for a new situation. This situation should resemble closely, though not identically, the conditions of the training set, bypassing conventional reanalysis and optimization iterations. The major advantage of a trained neural network as an expert designer over the traditional computational approach is that results can be produced with trivial computational effort. Further, the predictive capability of a trained network is insensitive to numerical instabilities and convergence difficulties typically associated with computational processes (e.g., during reanalysis, direction generations, one-dimensional searches, and design updates of the nonlinear optimization schemes). The disadvantage in generating sufficient design sets to train the artificial neural network is the potential expense.

In this report, the feasibility of ANN as an expert designer is considered for a complex set of engineering problems, representative of the optimum designs of structural components of the aerospace industry. The components are a ring, an intermediate complexity wing, and a forward swept wing. The number of design variables used in the optimization problems range from 60 to 157 and the number of behavior constraints range from 200 to 400. The design load conditions and constraint limitations are selected to ensure that, at optimum, all three types of behavior constraints (i.e., stress, stiffness, and frequency) become active. The design sets required to train the neural networks for the three components are generated with an optimization code CometBoards, which is described later. The neural network training is carried out through the code NETS, developed at NASA Johnson Space Center (ref. 10). The optimization code CometBoards was run on a Convex mainframe computer at NASA Lewis Research Center to generate the training data sets, and NETS was run on a SUN SPARC workstation to train the neural networks.

This report is divided into the following five subject areas: (1) a feed-forward back-propagation artificial neural network, (2) structural optimization, (3) code CometBoards, (4) discussion of neural net results, and (5) conclusions.

A Feed-Forward Back-Propagation Artificial Neural Network

Neural network simulations represent attempts to emulate biological information processing. The fundamental processor is the neuron, or brain cell, which receives input from many sources and processes these to generate a unique output. The output, in turn, can be passed on to other neurons. Learning is accomplished by changing connection strengths as knowledge is accumulated. The term "neural network" refers to a collection of neurons, their connections, and the connection strengths between them. Figure 1 shows an idealized neural network where the artificial neurons are shown as circles, the connections as straight lines, and the connection strengths (or weights) as calculations derived during the learning process for a problem. This network contains three layers—an input layer, an output layer, and a hidden layer—with each layer consisting of several neurons or nodes. The adaptation scheme used is based on the popular delta error back-propagation algorithm. In error back-propagation, the weights are modified to perform a steepest-descent reduction of the sum of the squares of the differences between the generated outputs and the desired outputs as indicated in the training pairs.

The optimal number of nodes in the hidden layer and the optimal number of hidden layers can be problem dependent. These numbers, however, should be kept low for computational efficiency. A rule of thumb is to start with a single hidden layer with the number of nodes equal to about half the total number of variables in the input and output layers. The number of nodes and layers should be increased if convergence difficulties are encountered, but should not exceed the total number of input and output variables. A simpler network with no hidden layers may be computationally efficient, but it represents only linear mapping between input and output quantities. These are known as flat networks and can be inadequate to model nonlinear relationships. The activation function determines the response of the neuron and is the only source of introducing nonlinearities in the input-output relationships.

The details of the back-propagation scheme have been described by Rumelhart and NcCleland (ref. 7). A brief discussion of the theoretical background follows.

A typical neural net configuration consists of an input layer, an output layer, and one hidden layer (as shown in fig. 1). Each layer consists of several nodes or neurons. To gain

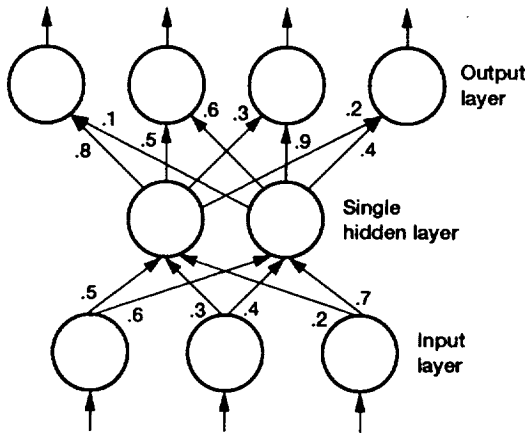

Figure 1.—A simple neural network model.

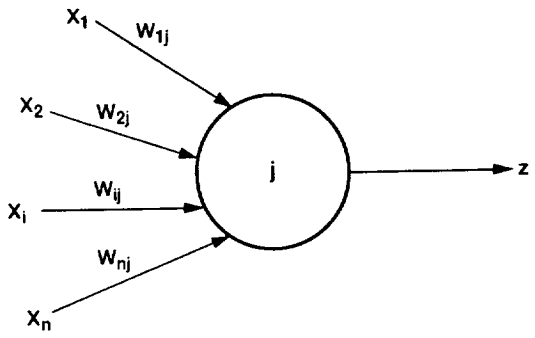

Figure 2.—A single processing element.

insight into the mechanism of information processing, it is better to focus on a single node (fig. 2), which receives a set of n inputs x_i, $i = 1,2, ..., n$. These inputs are analogous to electrochemical signals received by neurons in mammalian brains. In the simplest model, these input signals are multiplied by the connection weights w_{ij}, and the effective input to the elements is the weighted sum of the inputs as follows:

$$Z_j = \sum_{i=1}^{n} w_{ij} x_i \tag{1}$$

In the biological system, a typical neuron may only produce an output signal if the incoming signal builds up to a certain level. This biological characterstic is simulated in the artificial neural network by processing the weighted sum of the inputs through an activation function F to obtain an output signal as follows:

$$Y = F(Z) \tag{2}$$

The type of activation function that was used in the present study is a sigmoid function. The sigmoid function is given by the expression

$$F(Z) = \frac{1}{1+e^{-(Z+T)}} \tag{3}$$

This expression was adopted by the NETS computer code that was used in the present study. In equation (3), Z is the weighted input to the node, and T is a bias parameter used to modulate the element output. The principal advantage of the sigmoid function is its ability to handle both large and small input signals. The determination of the proper weight coefficients and bias parameters is embodied in the network learning process, which is essentially an error minimization problem.

In the delta error back-propagation approach, the nodes are initialized arbitrarily with random weights. The output obtained from the network is compared to the actual output (supervised learning) and the error E_i is computed as follows:

$$E_i = (T_i - Y_i) \tag{4}$$

where T_i and Y_i are the target and the actual output for node i, respectively. The error signal in equation (4) is multiplied by the derivative of the activation function for the neuron in question to obtain

$$\delta_{i,k} = \frac{\delta Y_{i,k}}{\delta Z} E_i \tag{5}$$

where the subscripts i and k denote the i^{th} neuron in the output layer k. The derivative of the output Y_i of the sigmoid function is obtained as follows:

$$\frac{\delta Y_i}{\delta Z} = Y_i(1 - Y_i) \tag{6}$$

The strength of connections between all neurons in the preceding hidden layer to the i^{th} neuron in the output layer is adjusted by an amount $\Delta w_{pi,k}$ as follows:

$$\Delta w_{pi,k} = \eta \delta_{i,k} Y_{p,j} \tag{7}$$

In equation (7), $Y_{p,j}$ denotes the output of neuron p in the hidden layer j immediately before the output layer; $\Delta w_{pi,k}$ is the change in value of the weight between neuron p in the hidden layer to neuron i in output layer k; and η denotes a learning rate coefficient (usually selected between 0.01 and 0.9). This learning rate coefficient is analogous to the step size parameter in a numerical optimization algorithm. Rumelhart and NcCleland (ref. 1) present a modification to the approach by including a momentum term as follows:

$$\Delta w^{t+1}{}_{pi,k} = \eta \delta_{i,k} Y_{p,j} + \alpha \Delta w^{t}{}_{pi,k} \tag{8}$$

Superscript t denotes the cycle of weight modification. The inclusion of the α term, which would incorporate a memory in the learning process, increases the stability of the scheme and helps in preventing convergence to a local optimum. This approach is applicable, with some variations, in the modification of weights in other hidden layers. The output of hidden layers cannot be compared to a known output to obtain an error term. Hence, the following procedure is used. The δ's for each neuron in the output layer are first computed as in equation (5) and used to determine the weights of connections from the previous layer. These δ's and w's are used to generate the δ's for the hidden layer immediately preceding the output layer as follows:

$$\delta_{p,j} = Y_{p,j} \left(\sum_i \delta_{i,k} w_{pi,k} \right) \tag{9}$$

where $\delta_{p,j}$ is the δ corresponding to the p^{th} neuron in the hidden layer and $Y_{p,j}$ is the derivative of the activation function of this neuron as computed in equation (6). Once the δ's corresponding to this hidden layer are obtained, the weight of connections of the next hidden layer can be modified by an application of equation (7) with appropriate change in the indices. This process is then repeated for all remaining hidden layers. The process must be repeated for all input training patterns until the desired level of error is attained. A modification of the approach is to present a sum of errors of all training patterns from the very beginning.

Structural Optimization

The structural design optimization problem can be described as the following:

> Find the n design variables $\bar{\chi}$ within prescribed upper and lower bounds, $(\chi_i^L \le \chi_i \le \chi_i^U,\ i = 1,2,\ldots,n)$, which make the scalar objective function, $f(\bar{\chi})$, an extremum (a minimum) subject to a set of m inequality constraints, representing the failure modes of the design problem

$$g_j(\bar{\chi}) \le 0,\ (j = 1,2,\ldots,m) \tag{10}$$

The constraints for structural design applications are typically nonlinear in the variables $\bar{\chi}$. Equality constraints could also be included, but generally do not occur in structural design problems. Behavior parameters considered are stress, displacement, and frequency constraints g_j under multiple load conditions. For each load condition, the stress constraints are specified by

$$g_j = \left|\frac{\sigma_j}{\sigma_{jo}}\right| - 1.0 \le 0 \tag{11}$$

where σ_j is the design stress for the j^{th} element and σ_{jo} is the permissible stress for the j^{th} element. For each load condition, the displacement constraints are specified by

$$g_{js+j} = \left|\frac{u_j}{u_{jo}}\right| - 1.0 \le 0 \tag{12}$$

where u_j is the j^{th} displacement component, u_{jo} is the displacement limit for the j^{th} displacement component, and js is the total number of stress constraints. Constraints on frequencies are specified by

$$g_f = \left(\frac{f_{no}}{f_n}\right)^2 - 1.0 \le 0 \cdot \tag{13}$$

where f_n represents natural frequencies of the structure and f_{no} represents the limitations on these frequencies.

In a mathematical programming technique, the optimal design point $\bar{\chi}_{opt}$ is reached starting from an initial design $\bar{\chi}_o$ in, say, K iterations. The design is updated at each iteration by the calculation of two quantities, a direction $\bar{\phi}$ vector, and a step length α. The design process can be symbolized as

$$\bar{\chi}_{opt} = \bar{\chi}_o + \sum_{k=1}^{K} \alpha_k \bar{\phi}_k, \tag{14}$$

where $\bar{\phi}_k$ is the direction vector at the k^{th} iteration, and α_k is the step length along the direction vector $\bar{\phi}_k$. At the k^{th} iteration, the direction vector $\bar{\phi}_k$ is generated from the gradients of the objective function and the active constraint subset following one of the available direction generation algorithms. Along the direction vector $\bar{\phi}_k$ a one-dimensional search is carried out to obtain the step length α_k, again utilizing one of several available procedures. The updated design is then checked against one or more stop criteria and the iterative process is repeated until it converges. The details of the nonlinear mathematical programming techniques, although well documented in the literature, are not elaborated here.

Code CometBoards

The basic structure of the optimization code CometBoards (Berke, L., Guptill, J., and Patnaik, S.N., 1993, NASA TP, to be published) is depicted in figure 3. The code has a central command unit, control via command level interface, as shown in figure 3. This unit establishes links between the three modules of the code (optimizer, analyzer, and data files) to solve the optimization problem. The solution is stored in an output device. Several options for optimizers and analyzers are available. The optimization options are

(1) Fully utilized design (FUD)
(2) Optimality criteria (OC) technique
(3) Method of feasible directions (FD)
(4) International Mathematical and Scientific Library (IMSL) sequential quadratic programming (SQP) method
(5) Sequential linear programming (SLP)
(6) Sequential quadratic programming (SQP)
(7) Sequential unconstrained minimization technique (SUMT)

The analyzer options are the displacement method, the integrated force method, and the simplified force method.

There are three input data files: ANLDAT, DISDAT, and OPTDAT. A typical command to execute the code CometBoards is

Optimize SUMT disp other stress disp freq (Output sdf a

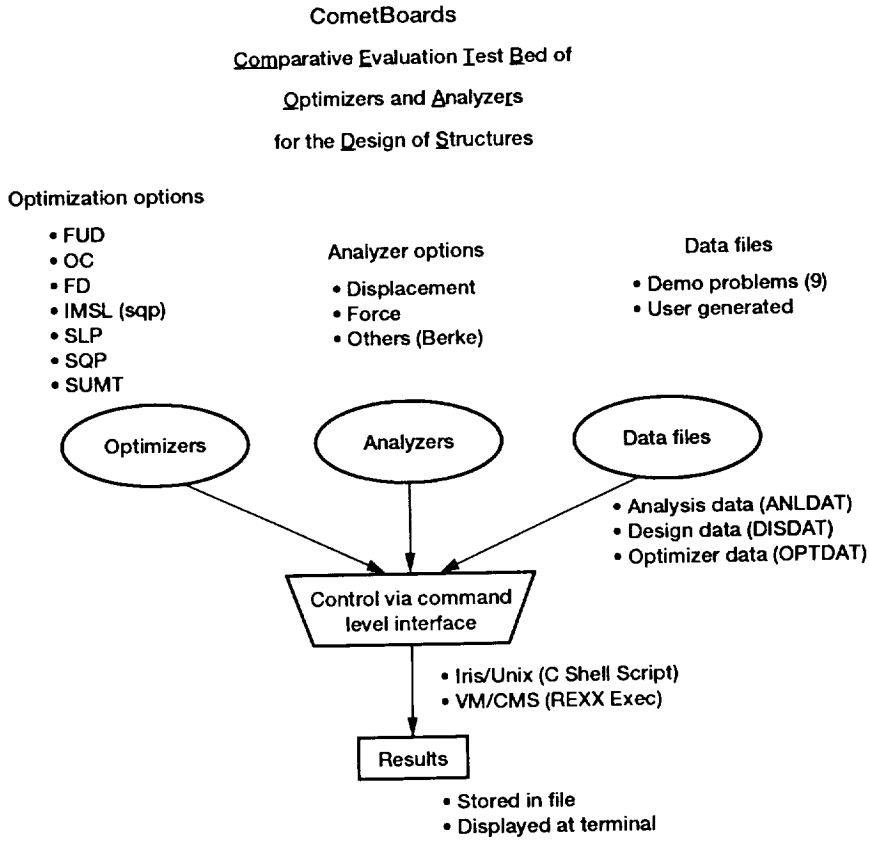

Figure 3.—Optimization code CometBoards.

followed by three data files (prompted interactively), ANLDAT file1 a, DISDAT file1 a, and SUMTDAT file1 a. The first two arguments "Optimize SUMT" represent optimization using SUMT. The third argument "disp" means displacement method will be used as the analysis tool. The fourth argument "other" is a name for the optimization problem. The fifth, sixth, and seventh arguments "stress," "disp," and "freq" indicate the types of behavior constraints considered: stress for stress constraints, disp for displacement, and freq for frequency constraints.

The file ANLDAT file1 a is the analysis input data file from which the finite element analysis information is read. The file DISDAT file1 a is the design input data file from which information required to set up the optimization problem is read. The file SUMTDAT file1 a is the optimization input data file for optimizer SUMT. Results of the optimization problem are stored in the file Output sdf a. In brief, the code CometBoards has considerable flexibility in solving a design problem by choosing one of several optimizers and one of three analysis methods. Space does not permit further discussion of CometBoards. However, the code is used to generate sets of optimum designs for the ring, the intermediate complexity wing, and the forward swept wing problems used to train the network. To ensure the reliability of the optimum designs, the same problems were solved using several different optimizers and analyzers.

Discussion of Neural Net Results

The predictive capabilities of the trained neural networks as expert designers are described for all three examples in this section. The ANN training's predictive capability for the ring problem is described in some detail; however, only cursory discussion will be included here for the other wing problems.

Example 1—The Trussed Ring

The trussed ring (fig. 4) is the first example to illustrate the predictive capabilities of ANN. The inner and outer radii of the ring are R_i and R_o, respectively. The ring is idealized by 60 truss elements made of aluminum with Young's modulus, E = 10 000 ksi and weight density ρ=0.1 lb/in.^{3}. The design of the ring for minimum weight under stress, displacement, and frequency constraints was used for the artificial neural network training. The ring was subjected to three static load conditions as given in table 1. Lumped

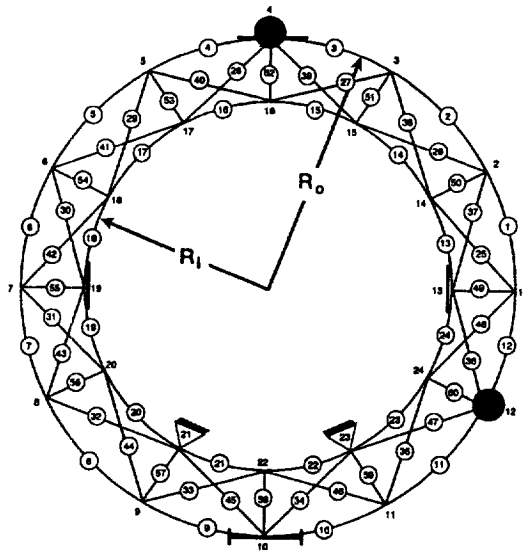

Figure 4.—Base configuration for trussed ring.

TABLE I.—LOAD SPECIFICATIONS FOR 60-BAR TRUSSED RING

Load conditions	Load components, kip		
	Node number	P_x	P_y
I	1	−10	0
	7	9	0
II	15	−8	3
	18	−8	3
III	22	−20	10
Lumped masses	4	$m_x=m_y$=200 lb	
	12	$m_x=m_y$=200 lb	

TABLE II.—CONSTRAINT SPECIFICATION FOR 60-BAR TRUSSED RING

Constraint type	Constraint description	
Stress	$\sigma_i \le \sigma_o (i=1,2,...,60)$ σ_o=10 ksi	
Displacement	Magnitude in directions x and y	
	Node number, i	U_i, in.
	4	1.75
	10	1.25
	13	2.25
	19	2.75
Frequency	$f \ge f_o$ f_o=13, 14, 15, 16, or 17 Hz	
Design bounds for areas	$A_i \ge 0.5$ in.^{2} $(i=1,2,...,60)$	

TABLE III.—DESIGN VARIABLE LINKAGE FOR 60-BAR TRUSSED RING

Design variable	Members linked
1	49 through 60
2	1,13
3	2, 14
4	3,15
5	4,16
6	5,17
7	6,18
8	7,19
9	8,20
10	9,21
11	10,22
12	11,23
13	12,24
14	25,37
15	26,38
16	27,39
17	28,40
18	29,41
19	30,42
20	31,43
21	32,44
22	33,45
23	34,46
24	35,47
25	36,48

masses were used for frequency calculations, whereas the elements were idealized as massless springs. The constraint specifications are given in table II. The optimum design of the ring was determined using a total of 195 behavior constraints, consisting of 180 stress, 24 displacement, and 1 natural frequency constraint. The 60-bar cross-sectional areas shown in table III were linked to obtain a reduced set of 25 design variables. The values for loads, masses, and bounds for displacement and frequencies were specified to ensure that, at optimum, the active set included all three types of constraints (i.e., stress, displacement, and frequency). For neural network simulation, the geometry of the ring is controlled by its inner and outer radii. These radii and the frequency limits are the three global input parameters. The 25 linked design parameters and the minimum weight make up the 26 output variables. For neural net training and predictions, 125 sets of optimum designs are generated using all combinations of 5 outer radii (R_o=80, 90, 100, 110, and 120 in.), and 5 inner radii (R_i = 0.92R_o, 0.91R_o, 0.90R_o, 0.89R_o, and 0.88R_o), along with 5 frequency limits (13, 14, 15, 16, and 17 Hz).

Figure 5 shows the composite design configuration for the trussed ring. The optimum designs were obtained using sequential unconstrained minimization techniques (SUMT). The optimum design convergence characteristics were verified by solving the problem using two other methods:

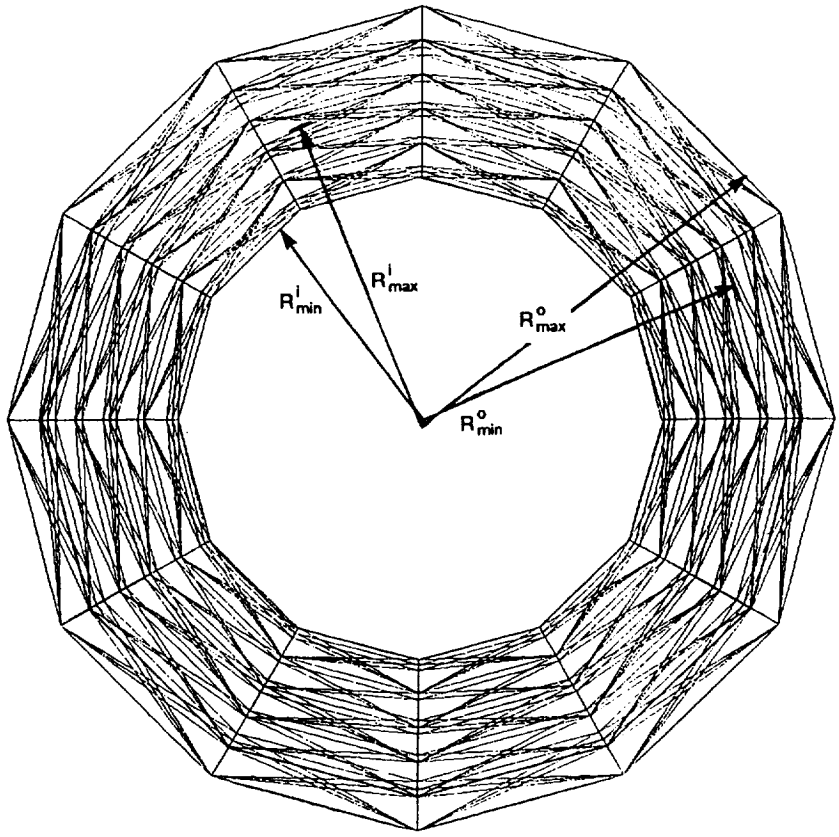

Figure 5.—Composite design configuration for trussed ring.

(1) method of feasible directions (FD) and (2) sequential quadratic programming (SQP) techniques. The optimum weights of the 125 designs varied from 1 000 lb to approximately 150 000 lb. At optimum, frequency is typically an active constraint along with other stress and displacement constraints. Simply stated, the design is complex with a highly undulated design space. The CPU time in a Convex machine to generate one optimum design of the ring using CometBoards can take between 3 to 45 min depending on the optimizer and its convergence parameters as well as on the analyzer and reanalysis schemes. Fifteen minutes of CPU time in a Convex machine can be considered typical for the solution of a ring optimization problem. However, the generation of 125 optimum designs required more than 35.125 CPU hr because of convergence difficulties encountered during optimization runs.

For the purpose of training and prediction of optimum structural design for stress, displacement, and frequency constraints through an artificial neural network, the 125 data sets were separated into a training set consisting of 120 designs and a test set of 5 designs. These input and output pairs were used to train a back-propagation neural network using the code NETS. The trained network was then used to predict the design for the test set.

The parameters specified to train the ring design by using the ANN code NETS in a SUN SPARC workstation were (1) the number of hidden layers, one with 30 nodes (Two hidden layers with 15 nodes each were also used with other parameters unchanged, but no significant training efficiency could be attributed to such variations.) and (2) the root mean square error at 1/10 of 1 percent with a learning rate of 0.9 and training iterations of 10 000 cycles. The SUN SPARC workstation required about 10 hr to complete the 10 000 training cycles. At this training level, appreciable errors in both training and test sets were observed. For superior convergence of the weights, the ANN training was augmented by another 10 000 cycles with the total training

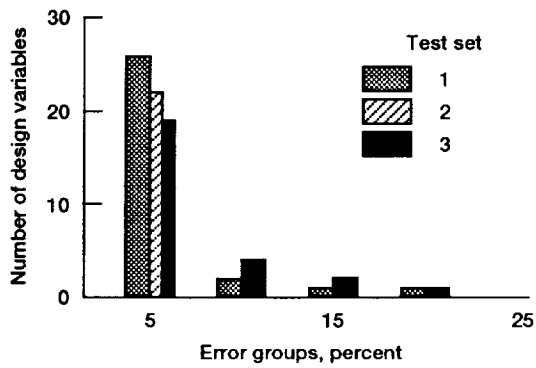

Figure 6.—Error histogram for three test data sets for trussed ring.

cycles at 20 000. At intermediate training steps, the learning rate was progressively reduced from 0.9 to 0.1. Weights, checked at intermediate training cycles, remained stationary with overall rms error at 0.029 at the end of 10 000 cycles and 0.025 at the end of 20 000 training cycles.

The NETS simulation results indicated that the training process was satisfactory for most of the variables. Predictions displayed individual error rates between 0 to 10 percent for approximately 80 percent of the variables. However, a few variables exhibited a much higher incidence of error. The test set followed the pattern of the training set with minor differences. Considering the complexity of the optimum designs (shift of load-carrying paths and weight range of 1 000 to 150 000 lb) the training and predictions cannot be considered unsatisfactory because even a veteran designer would have experienced difficulty in estimating for the situation.

To improve the performance of ANN predictions, the complexity of the optimum designs and associated undulations of the design space was reduced to a certain extent by considering only 34 data sets out of the 125 designs. The optimum weights of these 34 designs varied between 1 000 and 5 000 lb. The 34 data sets were separated into a training set of 31 data sets and 3 test data sets. As before, these input and output pairs were used to train the same back-propagation network, which was then used to predict the design for the test sets. The training parameters used were identical to the ones used previously.

As expected, substantial improvement in accuracy was noticed for this training set. Training set predictions exhibited reduced error between 0 to 5 percent for most of the variables with some exceptions. The test set followed the pattern of the training set with minor differences. The performance of the neural network to predict the ring design for three test data sets is shown in figure 6 in the form of an error histogram. Here the neural net output was compared with the target values and errors were computed for each output variable. These errors were then grouped according to their magnitudes and plotted in the form of a histogram. The procedure was repeated for each test data set. Thus, for example, in figure 6 it can be seen that all 26 output variables were predicted within 5 percent for the test data set 2. The other two data sets exhibited error in 10-, 15-, and 20-percent groups for a few output variables. Note that an error group of 15 percent, for example, suggests that all output variables in this group were predicted with an absolute error in the range of 10 to 15 percent. Most of the predictions for all three test data sets exhibited error of approximately 5 percent. A few variables exhibited error exceeding 5 percent, but there was no variable for which the incidence of error exceeded 25 percent.

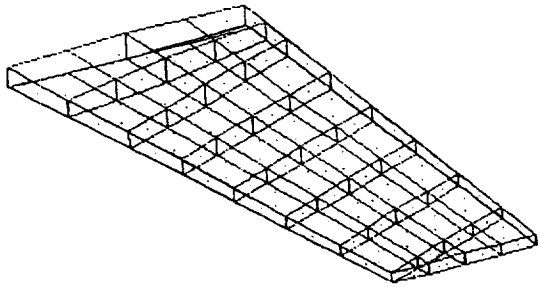

Figure 7.—Base configuration for intermediate complexity wing.

Example 2—Optimum Design of an Intermediate Complexity Wing

The intermediate complexity wing represents the second illustrative example. Figure 7 shows the geometrical configuration of the wing. The approximate length of the wing is 90 in. Its width at the base is about 48 in. and about 29 in. at the apex. Its depth varies between 2.25 in. at the base to 1.125 in. at the apex. The finite element model has 88 grid points and a total of 158 elements consisting of 39

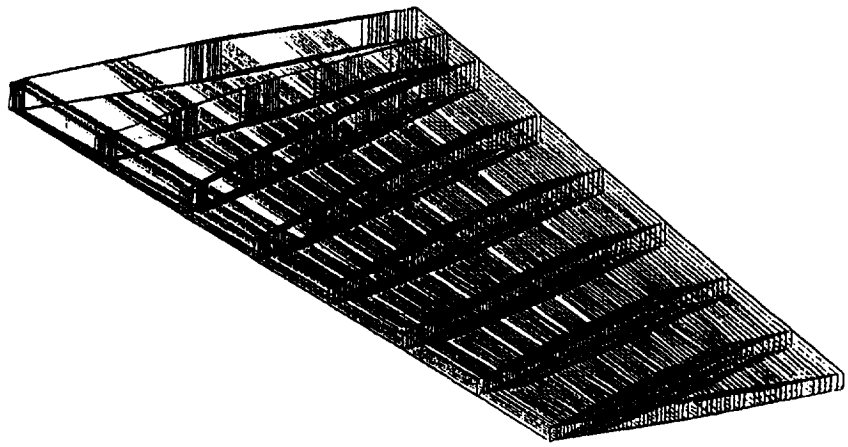

Figure 8.—Composite design configuration for intermediate complexity wing.

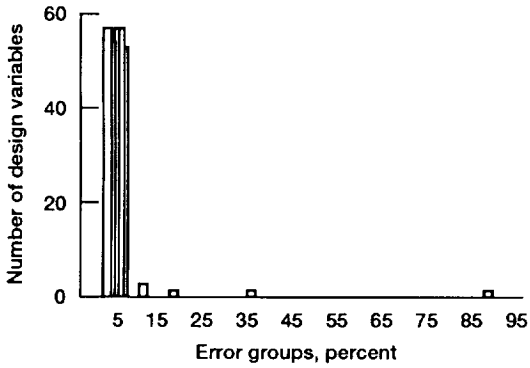

Figure 9.—Error histogram for several trained data sets for intermediate complexity wing.

bars, 2 triangular membranes, 62 quadrilateral membranes, and 55 shear panels. The structure is made of aluminum with Young's modulus E = 10 500 ksi, Poisson's ratio ν=0.3, and weight density ρ=0.1 lb/in.^{3}. The design of the wing for minimum weight, under displacement constraints specified at the apex of the wing, was used for the artificial neural network calculations. The 158 design variables consisting of bar areas and plate thicknesses were linked to obtain a reduced set of 57 design parameters for optimization. For neural network simulation, the geometry of the wing was changed to approximately 1.25 times its plan area. The basic design and the composite-perturbed configurations are shown in figures 7 and 8, respectively. Each geometrical configuration was specified by a single master parameter as the input variable. The 57 optimum design parameters and the minimum weight are considered the 58 output parameters. Fifteen sets of optimum designs are generated for 15 different geometrical configurations. The optimum designs were obtained by using sequential unconstrained minimization techniques (SUMT), which were verified further by two other optimizers: (1) method of feasible directions (FD) and (2) sequential quadratic programming (SQP) techniques. The influence of compatibility or indeterminacy, which changes the load paths in the structure for different optimum designs, was observed among the 15 data sets. The 15 data sets were separated into a training set consisting of 13 data sets and a test set of 2 data sets to predict the optimum structural design of the wing by using an artificial neural network. These input and output pairs were used to train a neural network. The ANN parameters were kept identical to those of the ring problem. Figure 9 shows the results for the wing in an error histogram form. This histogram shows that most predictions exhibited error of about 5 percent. There are a few variables that strayed into higher error brackets.

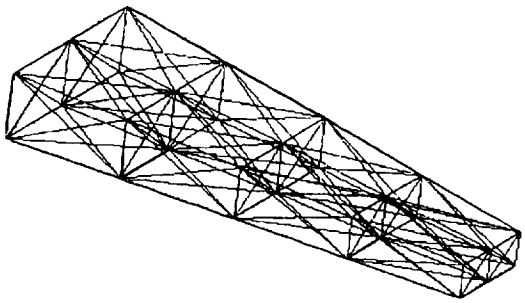

Figure 10.—Base configuration for forward swept wing.

Example 3—The Forward Swept Wing

The forward swept wing (fig. 10) represents the final illustrative example. The approximate length of the wing is 160 in. Its width at the base is about 80 in. and about 40 in. at the apex. Its depth varies between 20 in. at the base to 10 in. at the apex. The finite element model has 30 grid points and 135 truss elements. The structure is made of aluminum with Young's modulus E=10 000 ksi, Poisson's ratio ν=0.3, and weight density ρ=0.1lb/in.^{3}. The optimum design of the wing for minimum weight, under displacement constraints at its apex, was used for the artificial neural network

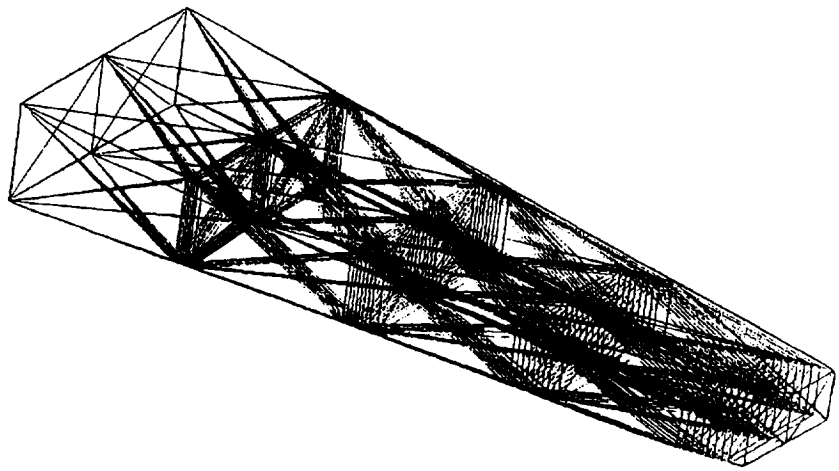

Figure 11.—Composite design configuration for forward swept wing.

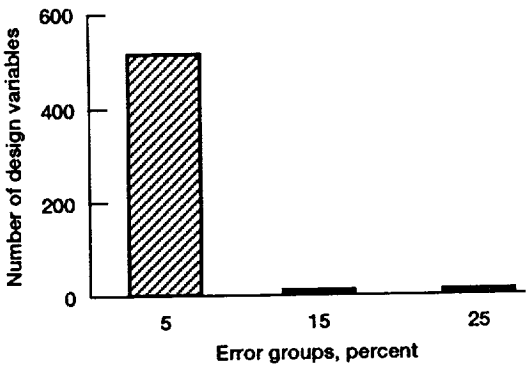

Figure 12.—Error histogram for four data sets for forward swept wing.

calculations. All 135 member areas were regarded as independent design variables. For neural network simulation, the geometry of the wing was changed to approximately 1.5 times its base line plan area. Figure 11 shows the composite configuration of the forward swept wing. Each 1-percent change in its configuration was considered one data set for the training of the artificial neural network. Fifty optimum designs were obtained for the 50 geometrical configurations. The 50 designs were separated into a training set consisting of 45 cases and a test set consisting of the remaining 5 sets. As in the previous problems, a back-propagation neural network was trained by using NETS with training parameters identical to the previous cases. Figure 12 presents the results in histogram form. Most predictions exhibited about 5-percent error. However, a few variables exhibited 15- to 25-percent error, as shown in the error historgram for four test data sets.

Conclusions

Artificial neural network predictions of optimum designs under difficult design requirements were found to be satisfactory for most of the output design parameters. A few design parameters, however, strayed into higher error brackets.

The errors can be reduced to some extent when the complexity of the design space is reduced by using a much narrower design range, or by increasing the number of training sets to provide a better supervised learning environment for the neural network. Within the context of structural design data, caution should be exercised because load paths can change as a result of the indeterminacy of the structure. Discontinuities can be created in the design space that are difficult to model with the simple ANN code. Techniques are being made available for clustering the training data that represent similar behavior and then decomposing the network for training within the clusters. Training data preprocessing and other training paradigms, such as radial base functions (RBF), are worth exploring because they show potential to increase the robustness necessary for a neural network to act as an expert designer.

More research is required to assess the viability and usefulness of a neural network as an expert designer in routine design applications. The power of a trained ANN is in its capability to generalize and in its instantaneous response regardless of the original complexity of the problem. A trained ANN can provide very good estimates for optimum designs for what-if situations under changing conditions and at trivial computing cost. Such estimates can be used, not only in the structural problem setting, but in a multidisciplinary environment because the calculation of sensitivities becomes trivial once the network is trained. There are many other intriguing possibilities, particularly if one considers the rapidly expanding ANN capabilities and improvements in associated hardwares. These factors will open applications by reducing the training effort to acceptable computational costs even for much larger problems than those illustrated here.

Lewis Research Center
National Aeronautics and Space Administration
Cleveland, Ohio, July 1, 1992

References

1. Rumelhart, D.E.; and NcCleland, J.L.: Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vols. 1&2. The MIT Press, Cambridge, MA, 1988.
2. Bahr, B.; and Nabeel, T.M: Neural Networks for Detecting Defects in Aircraft Structures. IAR Report 90–4, Institute for Aviation Research, The Wichita State University, KS, 1990.
3. Tank, D.W.; and Hopfield, J.J.: Simple "Neural" Optimization Networks: An A/D Converter, Signal Decision Circuit, and a Linear Programming Circuit. IEEE Trans. Circuits Syst., vol. CAS–33, 1986, pp. 533–541.
4. Berke, L.; and Hajela, P.: Application of Artificial Neural Nets in Structural Mechanics. NASA TM–102420, 1990.
5. Hajela, P.; Fu, B.; and Berke, L.: ART Networks in Automated Conceptual Design of Structural Systems. Presented at the Second International Conference on the Application of Artificial Intelligence Techniques to Civil and Structural Engineering, Oxford, England, Sept. 3–5, 1991.
6. Pao, Y.H.: Adaptive Pattern Recognition and Neural Networks. Addision-Wesley Publishing Co., Reading, MA, 1989.
7. Ming, G.; and Xila, L.: A Preliminary Design Expert System (SPRED–1) Based on Neural Networks. Presented at the Second International Conference on the Application of Artificial Intelligence Techniques to Civil and Structural Engineering, Oxford, England, Sept. 3–5, 1991.
8. Brown, D.A.; Murthy, P.L.N.; and Berke, L.: Computational Simulation of Composite Ply Micromechanics Using Artificial with Neural Networks. Microcomput. Civil Eng., vol. 6, 1991, pp. 87–97.
9. Swift, R.A.; and Batill, S.M.: Application of Neural Networks to Preliminary Structural Design. AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, 32nd, AIAA, New York, vol. 1, 1991, pp. 335-343.
10. Baffes, P.T.: NETS 2.0 Users Guide. LSC–23366, NASA Lyndon B. Johnson Space Center, Sept., 1989.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 1993	Technical Memorandum

4. TITLE AND SUBTITLE

Application of Artificial Neural Networks to the Design Optimization of Aerospace Structural Components

5. FUNDING NUMBERS

WU 505-63-5B

6. AUTHOR(S)

Laszlo Berke, Surya N. Patnaik, and Pappu L.N. Murthy

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio 44135-3191

8. PERFORMING ORGANIZATION REPORT NUMBER

E-6994-1

9. SPONSORING/MONITORING AGENCY NAMES(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Washington, D.C. 20546-0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

NASA TM-4389

11. SUPPLEMENTARY NOTES

Laszlo Berke and Pappu L. Murthy, NASA Lewis Research Center, Cleveland, Ohio, and Surya N. Patnaik, Ohio Aerospace Institute, 2001 Aerospace Parkway, Brook Park, Ohio 44142. Responsible person, Surya N. Patnaik (216) 433-8468.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited
Subject Category 39

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 words)***

The application of artificial neural networks to capture structural design expertise is demonstrated. The principal advantage of a trained neural network is that it requires trivial computational effort to produce an acceptable new design. For the class of problems addressed, the development of a conventional expert system would be extremely difficult. In the present effort, a structural optimization code with multiple nonlinear programming algorithms and an artificial neural network code NETS were used. A set of optimum designs for a ring and two aircraft wings for static and dynamic constraints were generated by using the optimization codes. The optimum design data were processed to obtain input and output pairs, which were used to develop a trained artificial neural network with the code NETS. Optimum designs for new design conditions were predicted by using the trained network. Neural net prediction of optimum designs was found to be satisfactory for most of the output design parameters. However, results from the present study indicate that caution must be exercised to ensure that all design variables are within selected error bounds.

14. SUBJECT TERMS

Artificial neural network; Structures; Design; Optimization; Expert designer

15. NUMBER OF PAGES

16

16. PRICE CODE

A03

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	