CONF-930401--12

Full paper to be presented at the Topical Meeting on Nuclear Plant Instrumentation, Control, and Man-Machine Interface Technologies, April 18-21, 1993, Oak Ridge, Tennessee

ANL/RA/CP--77255

DE93 009985

Statistical and Optimization Methods to Expedite Neural Network Training for Transient Identification

J. Reifman,[1] J. Vitela E.,[2] and J. C. Lee[3]

[1]Reactor Analysis Division
Argonne National Laboratory
9700 S. Cass Avenue
Argonne, Illinois 60439

[2]Instituto De Ciencias Nucleares
Universidad Nacional Autonoma De Mexico
04510 Mexico D.F.

[3]Department of Nuclear Engineering
University of Michigan
Ann Arbor, Michigan 48109

*Work supported by the U. S. Department of Energy, Nuclear Energy Programs under Contract W-31-109-ENG-38.

STATISTICAL AND OPTIMIZATION METHODS TO EXPEDITE NEURAL NETWORK TRAINING FOR TRANSIENT IDENTIFICATION

Jaques Reifman
Argonne National Laboratory
Reactor Analysis Division
Argonne, Illinois 60439
jreifman@anl.gov

Javier Vitela E.
Universidad Nacional Autonoma de Mexico
Instituto de Ciencias Nucleares
04510 Mexico D.F.
jvitela@unamvm1.bitnet

John C. Lee
University of Michigan
Department of Nuclear Engineering
Ann Arbor, Michigan 48109
John.C.Lee@um.cc.umich.edu

ABSTRACT

Two complementary methods, statistical feature selection and nonlinear optimization through conjugate gradients, are used to expedite feedforward neural network training. Statistical feature selection techniques in the form of linear correlation coefficients and information-theoretic entropy are used to eliminate redundant and non-informative plant parameters to reduce the size of the network. The method of conjugate gradients is used to accelerate the network training convergence and to systematically calculate the learning and momentum constants at each iteration. The proposed techniques are compared with the backpropagation algorithm using the entire set of plant parameters in the training of neural networks to identify transients simulated with the Midland Nuclear Power Plant Unit 2 simulator. By using 25% of the plant parameters and the conjugate gradients, a 30-fold reduction in CPU time was obtained without degrading the diagnostic ability of the network.

I. INTRODUCTION

Training a feedforward artificial neural network (ANN) to model the relationships between input-output patterns is a time-consuming process. Since most training algorithms can at best guarantee convergence to one of the various local minima, a number of training runs with different sets of initial weights are generally required. Also, some training algorithms require the pre-specification of input constants. These constants usually need to be fine tuned, through trial-and-error training runs, in order to assure that convergence will be achieved and to expedite training. Furthermore, the lack of a general prescription for determining the architecture of an ANN usually requires repetitive training with different network architecture or varying numbers of hidden layers and nodes in the hidden layers.

In this paper we describe two complementary methods, statistical feature selection and nonlinear optimization techniques, for expediting the training process of multilayer feedforward ANNs for nuclear power plant transient identification. The process of feature selection is applied to eliminate non-informative and redundant features or plant parameters such that the number of nodes in the input and hidden layers of an ANN can be vastly reduced. For eliminating features that are redundant or interrelated and hence do not contribute additional information in the characterization of the transients, we use the linear correlation coefficient.[1] For selecting the most informative features that clearly discriminate each transient from the remaining transient types, we use the information-theoretic entropy.[2,3] By eliminating non-informative and redundant plant parameters we can reduce the dimensionality of the network input patterns, and hence the number of nodes in the network which expedites the training process.

Nonlinear optimization techniques which offer better convergence properties than the commonly used backpropagation (BP) algorithm[4] and which do not require the pre-specification of input constants are also used to expedite ANN learning. Here we apply the more powerful unconstrained optimization method of conjugate gradients[5,6,7] (CGs) to accelerate ANN learning. The method of CGs provides a systematic way to determine both the learning and momentum constants, which are dynamically updated at each iteration, eliminating the need for trial-and-error training runs required to fine tune these constants. For verifying the advantages of these two complementary methods in expediting neural network training, we use a data base of transients simulated with the Midland Nuclear Power Plant Unit 2 (MNP-2) simulator.[8]

II. STATISTICAL FEATURE SELECTION

When neural networks are used to model the relationships between N' measured plant parameters F_n and K transient event types T_k of nuclear plants, in general, the N' parameters correspond to the N' elements of the network input patterns and the K transient types correspond to the K elements of the network output patterns. The process of training a neural network to identify transient events consists of determining the decision boundaries that partition the N'-dimensional feature space spanned by the N' input plant parameters F_n, such that each one of the K transient types T_k is discriminated from the other transient event types. The determination of the decision boundaries that provide the best partition of the feature space, i.e., the training process, is time-consuming and increases with the dimensionality of the feature space. Hence, by retaining only a subset of N salient plant parameters or features $F_1,F_2,...,F_N$ from a set of N' features (N<<N'), such that the determination of the decision boundaries is implemented in a vastly reduced feature space we may expedite neural network training without degrading its performance.

In this work, the objective of statistical feature selection is to select N key plant parameters or features F_n, from the N' monitored parameters, to be used as the N elements of the ANN input patterns. This process of feature selection stems from the actual operation of nuclear power plants where during normal operation, hundreds of signals are monitored to determine the status of various systems and components of the plant. However, when the plant goes into an upset condition, only a few of these signals or features are important in the characterization of the transient. Here, we discuss the use of two statistical techniques, linear correlation coefficients and information-theoretic entropy, for selecting non-redundant and informative features as the elements of the ANN input patterns.

A. Linear Correlation Coefficients

Features that share common information with other features are redundant, i.e., they do not contribute additional information about their common events, and should not be selected. Including redundant features would only increase the dimensionality of the feature space, where the decision boundaries are found, with no extra information. Therefore, it is necessary to check each pair of features and delete those features that are redundant.

Feature selection on the basis of redundancy among features is attained by calculating the linear correlation coefficient[1] between each pair of features and keeping only one of two or more features when their linear correlation coefficient is beyond a specified threshold. The linear correlation coefficient r_{ij} between the *i-th* and the *j-th* features is defined as the ratio of the covariance Cov(i,j) to the product of the standard deviations $\sigma_i \sigma_j$ for the two features

$$r_{ij} = \frac{Cov(i,j)}{\sigma_i \sigma_j} . \qquad (1)$$

Two features are perfectly correlated if $r_{ij} = 1.0$ and perfectly anti-correlated if $r_{ij} = -1.0$. They are uncorrelated if $r_{ij} = 0.0$. Threshold values on the linear correlation coefficient between two features is used to characterize correlated features. Here, two features with absolute values of their correlation coefficient greater than 0.90 are considered to be correlated and only one is retained.

B. Information-Theoretic Entropy

After the redundant features have been removed from the initial list of monitored plant parameters, the remaining features can be ranked based on their information content. The information content of a feature is characterized by its capacity to discriminate among the various transient types T_k. For instance, if we are interested in discovering patterns associated with two transients, T_1=feedwater pump trip and T_2=feed flow transmitter fails high, the pressure difference across the feedwater control valves should be used as one of the features. Since the pressure difference across the control valves decreases when the pump trips and increases when the transmitter fails high, this feature can be used to discriminate the two transient event types and should be selected. On the other hand, the pressurizer pressure would initially increase for both transients, and hence would not yield discriminating information and should not be selected.

To systematically determine and rank feature variables based on their information content we use measures defined by information entropy.[2,3] The information content of each feature or plant parameter F_n, n=1,2,...,N', is obtained by projecting the D data points of a transient data base onto each feature F_n and partitioning F_n into a maximum of I feature intervals $F_{i,n}$ (i=1,...,I≤4) that maximize the mutual information exchange $\Delta S(T|F_n)$ between the transient events T_k (k=1,2,...,K) and the feature intervals $F_{i,n}$

$$\Delta S(T|F_n) = S(T) - S(T|F_n)$$
$$= \sum_{i=1}^{I} P(F_{i,n}) \sum_{k=1}^{K} P(T_k|F_{i,n}) \ln \frac{P(T_k|F_{i,n})}{P(T_k)}, \quad (2)$$

where

$S(T)$ = total entropy over the D transient events,
$$= -\sum_{k=1}^{K} P(T_k) \ln P(T_k),$$

$S(T|F_n)$ = conditional entropy,
$$= -\sum_{i=1}^{I} P(F_{i,n}) \sum_{k=1}^{K} P(T_k|F_{i,n}) \ln P(T_k|F_{i,n}),$$

$P(T_k)$ = marginal probability of occurrence of event T_k,

$P(F_{i,n})$ = marginal probability of events of any type to occur in $F_{i,n}$, and

$P(T_k|F_{i,n})$ = conditional probability of transient event T_k given that the event occurs in feature interval $F_{i,n}$.

The mutual information exchange $\Delta S(T|F_n)$ can be interpreted as a measure of interdependence between the event's feature value or location in the feature intervals and the transient event types. This can be understood if we recognize that $\Delta S(T|F_n)$ attains its minimum $\Delta S(T|F_n) = 0$ for $P(T_k|F_{i,n}) = P(T_k)$, i.e., the knowledge of the location of the events in the feature F_n gives no information about their type, and that $\Delta S(T|F_n)$ attains its maximum $\Delta S(T|F_n) = S(T)$ if $S(T|F_n) = 0$, i.e., no information will be gained by observing the event type once we have already learned its location in the feature intervals. In essence, a feature F_n is considered informative, i.e., it has large values of $\Delta S(T|F_n)$, if after projecting the D data points onto F_n, feature intervals $F_{i,n}$ are obtained that clearly discriminate events of different types, i.e., events of the same type tend to be located in the same feature interval.

III. NON-LINEAR OPTIMIZATION TECHNIQUES

The process of training a multilayer feedforward ANN with differentiable mapping functions can be cast as an unconstrained nonlinear optimization problem. A set of weights **w** that represent links between the network nodes and thresholds of the nodes is sought such that the prediction error E of mapping P input-output patterns is minimized,

$$E(\mathbf{w}) = \sum_{p=1}^{P} E_p = \sum_{p=1}^{P} \frac{1}{2} \sum_{j=1}^{J_L} (t_{pj} - x_{pj}^{(L)})^2 . \quad (3)$$

Here t_{pj} and $x_{pj}^{(L)}$ denote the *j-th* target and network output, respectively, for the *p-th* training pattern and J_L is the number of nodes in the *L-th* or output layer of the network.

A. The Backpropagation Algorithm

The most widely used method for training multilayer feedforward ANNs is the backpropagation algorithm which was popularized by Rumelhart, et al.[4] The BP algorithm is basically a gradient descent algorithm which minimizes the prediction error E of Eq. (3) in an iterative fashion. Starting from a randomly selected initial position $\mathbf{w}_1$, the gradient descent algorithm arrives at the next position $\mathbf{w}_2 = \mathbf{w}_1 - \eta \mathbf{g}_1$ by moving along the negative gradient direction $-\mathbf{g}_1$ an amount η, where η is a positive constant termed the learning constant. At the following iteration, the algorithm arrives at the next position $\mathbf{w}_3 = \mathbf{w}_2 - \eta \mathbf{g}_2$ by moving along $-\mathbf{g}_2$ an amount η. This sequence is repeated until the convergence criterion $|t_{pj} - x_{pj}^{(L)}| < \varepsilon$ is satisfied for p=1,2,...,P and j=1,2,...,J_L. Since the best one can expect from any gradient-based optimization method is that it will lead as quickly as possible to the bottom of whatever valley it starts in, it may be necessary to repeat the entire process starting from different initial positions in weight-space to avoid convergence to undesired local minima.

In the standard BP algorithm, at each iteration the weights **w** are adjusted slightly differently than in the gradient descent algorithm. In addition to the gradient term $-\eta \mathbf{g}_m$ the change in weights $\Delta\mathbf{w}_m = \mathbf{w}_{m+1} - \mathbf{w}_m$ at iteration m has a momentum term $\alpha \Delta\mathbf{w}_{m-1}$

$$\Delta\mathbf{w}_m = -\eta \mathbf{g}_m + \alpha \Delta\mathbf{w}_{m-1}, \quad (4)$$

where α is the momentum constant with arbitrarily selected values between 0 and 1. The momentum term $\alpha \Delta\mathbf{w}_{m-1}$ is used to accelerate convergence in regions of weight-space with relatively constant gradient and oscillatory gradient directions by including the effect of past weight changes on the current direction of movement.[4,9] In essence, the BP algorithm is an effective method of calculating the gradient $\mathbf{g}_m$.

Because of the similarities of the BP algorithm with the method of gradient descent, it is plagued with the same convergence-related drawbacks. First, the convergence is slow and becomes slower the closer the algorithm gets to the optimum solution. This is

caused by the fact that the rate of convergence of the gradient descent method is proportional to the norm of the gradient |g| which becomes increasingly small as the minimum is approached.[10] Second, the algorithm requires judicious selection of the learning constant η and the momentum constant α to achieve reasonable convergence and avoid oscillations about the optimum solution. The "optimum" values for η and α that balance the effects of the current and previous gradient directions, respectively, depend on the problem being solved and are generally obtained through trial-and-error training runs. Finally, the rate of convergence of the BP algorithm is very sensitive to the initial set of weights. The BP algorithm is prone to the premature "saturation" of the network nodes which may require a large number of iterations to converge depending on the initial position of the weight $\mathbf{w}_1$ in weight-space.[7,11,12] These drawbacks increase the number of iterations and CPU time necessary for convergence and require additional trial-and-error runs to determine reasonable values for η and α for the particular problem at hand, thereby making ANN training with the BP algorithm a painstaking and time-consuming process.

B. The Method of Conjugate Gradients

The unconstrained nonlinear optimization method of conjugate gradients,[5] which has better convergence properties than the method of gradient descent, is applied here to expedite neural network training. Like the BP algorithm, the method of CGs is iterative, but for quadratic objective functions of v variables it is guaranteed that the minimum will be located exactly, apart from rounding errors, in a maximum of v iterations. For functions that are not quadratic, which is the case for the prediction error $E(\mathbf{w})$ in Eq. (3), the method is based on the current local quadratic approximation to the function which causes the search process to be iterative rather than a v-step process, and a test of convergence is required. As the function $E(\mathbf{w})$ approaches the minimum, its approximation of a quadratic function becomes more realistic which may even cause the method to converge faster. This is in contrast with the BP algorithm whose rate of convergence decreases as the minimum is approached.

The method of CGs starts the iterative process of finding the minimum of $E(\mathbf{w})$ by randomly selecting an initial position $\mathbf{w}_1$ and arriving at the next position $\mathbf{w}_2 = \mathbf{w}_1 + \eta_1 \mu_1$ by moving along the conjugate direction μ_1 an amount η_1, where η_1 minimizes $E(\mathbf{w}_1 + \eta \mu_1)$ with respect to η. Like the BP algorithm, this sequence is repeated until the convergence criterion $|t_{pj} - x_{pj}^{(L)}| < \varepsilon$ is satisfied for p=1,2,...,P and j=1,2,...,J_L. However, unlike the BP algorithm, the method of CGs provides a systematic procedure to calculate the learning constant η_m which is updated at each iteration m. Here, the estimate of η_m^* which minimizes $E(\mathbf{w}_m + \eta \mu_m)$ with respect to η is obtained through a one-dimensional search that combines the golden section rule[13] with a cubic interpolation.[14] We have found that this combination provides the necessary balance between accuracy and computational speed.[7] Also, the method of CGs arrives at the next position in weight-space by moving along conjugate directions μ_m rather than the negative gradient direction $-\mathbf{g}_m$. In this work, we compute the conjugate directions μ_m as defined by Fletcher and Reeves[5]

$$\mu_m = \begin{cases} -\mathbf{g}_m + \dfrac{\mathbf{g}_m^T \cdot \mathbf{g}_m}{\mathbf{g}_{m-1}^T \cdot \mathbf{g}_{m-1}} \mu_{m-1} & ; \quad \text{for } m > 1 \\ -\mathbf{g}_1 & ; \quad \text{for } m = 1 \end{cases} \tag{5}$$

where the gradients $\mathbf{g}_m$ and $\mathbf{g}_{m-1}$ for iterations m and m-1, respectively, are calculated just like in the BP algorithm.

Similarly, the change in weights $\Delta \mathbf{w}_m = \mathbf{w}_{m+1} - \mathbf{w}_m$ at iteration m is given by

$$\Delta \mathbf{w}_m = \eta_m \mu_m . \tag{6}$$

By substituting μ_m of Eq. (5), for m > 1, into Eq. (6) and using the fact that $\mu_{m-1} = \Delta \mathbf{w}_{m-1} / \eta_{m-1}$ we obtain

$$\Delta \mathbf{w}_m = -\eta_m \mathbf{g}_m + \frac{\eta_m}{\eta_{m-1}} \frac{\mathbf{g}_m^T \cdot \mathbf{g}_m}{\mathbf{g}_{m-1}^T \cdot \mathbf{g}_{m-1}} \Delta \mathbf{w}_{m-1} . \tag{7}$$

We may now compare Eq. (7) with Eq. (4) to identify the learning constant η and the momentum constant α of the BP algorithm as the following terms of the CG algorithm

$$\eta = \eta_m , \quad \text{and} \tag{8}$$

$$\alpha = \alpha_m = \frac{\eta_m}{\eta_{m-1}} \frac{\mathbf{g}_m^T \cdot \mathbf{g}_m}{\mathbf{g}_{m-1}^T \cdot \mathbf{g}_{m-1}} .$$

Hence, the two methods use the same expression to update the weights at each iteration. The difference is that the method of CGs provides a systematic way to determine both the learning and momentum constants which are dynamically updated at each iteration. This systematic approach eliminates the need for trial-and-error training runs frequently used in the BP algorithm to obtain reasonable values of η and α such that the

algorithm has a reasonable rate of convergence and avoids oscillations about the minimum.

IV. NETWORK TRAINING FOR TRANSIENT IDENTIFICATION

With the statistical feature selection techniques described in Sec. II and the nonlinear optimization method of conjugate gradients discussed in Sec. III, we have trained a feedforward neural network to identify nuclear power plant transients. The network has also been trained by using the entire set of available plant parameter measurements, i.e., without feature selection, with the BP as the training algorithm. Comparison of these training sessions was then performed regarding the number of iterations and CPU time to converge as well as the prediction error for transient events not used in training.

A. Data Base of Transients

The MNP-2 full-scope operator training simulator[8] representing all major systems of a two-loop pressurized water reactor plant and control room instrumentation was used to construct a realistic data base of transients. Each one of the 3 transient event types T_k (k=1,2,3) illustrated in Table I occurring separately, i.e., single failures, was simulated on the MNP-2 simulator a dozen times. For each one of the 12 simulations of an event, a different combination of failure severity and initial conditions, e.g., different initial power level and fuel burnup, was used. The forty plant parameters, $F_1,F_2,...,F_{40}$, illustrated in Table II, which were arbitrarily selected from a set of signals in eight systems throughout the plant that are sent to the control console were used to characterize the 3 transient event types.

Table I. List of Transient Events.

T_1 = Feedwater pump trip
T_2 = Feed flow transmitter fails high
T_3 = Turbine control valve fails closed

The first 40 s after the start of the transients was selected as the time range for diagnostics. To represent the dynamic behavior of each of the 3 transient types over the 40-s time interval in the data base of transients, we converted the 40 plant parameters of Table II into the time rate of change $\Delta F_n/\Delta t$ at three arbitrarily selected discrete times, 10, 25, and 40 s. Thus, each simulation of a transient contributed three data points to the data base, forming

Table II. List of Plant Parameters.

1. Reactor Coolant System (RCS)
F_1 = Quench tank pressure [MPa]
F_2 = Quench tank water level [m]
F_3 = RCS average temperature [K]
F_4 = RCS hot leg average temperature [K]
F_5 = RCS cold leg average temperature [K]
F_6 = RCS leg A temperature difference [K]
F_7 = RCS leg B temperature difference [K]
F_8 = Pressurizer total pressure [MPa]
F_9 = Pressurizer water level [m]
F_{10} = F_8/F_9 [MPa/m]
2. Nuclear Instrumentation (NI) System
F_{11} = Reactor power [% full power]
3. Steam Generator (SG)
F_{12} = SG loop A downcomer temperature [K]
F_{13} = SG loop B downcomer temperature [K]
F_{14} = SG loop A water level [m]
F_{15} = SG loop B water level [m]
F_{16} = SG loop A exit pressure [MPa]
F_{17} = SG loop B exit pressure [MPa]
4. Main Steam (MS) Module
F_{18} = MS loop A header pressure [MPa]
F_{19} = MS loop B header pressure [MPa]
F_{20} = Turbine throttle pressure [MPa]
F_{21} = Turbine exhaust pressure [MPa]
F_{22} = Turbine exhaust temperature [K]
5. Feedwater (FW) System
F_{23} = Deaerator A water level [m]
F_{24} = Deaerator B water level [m]
F_{25} = Deaerator A temperature [K]
F_{26} = Deaerator B temperature [K]
F_{27} = Condenser total pressure [MPa]
F_{28} = Condenser hot well water level [m]
F_{29} = DP across FW A control valves [MPa]
F_{30} = DP across FW B control valves [MPa]
F_{31} = FW loop A inlet header pressure [MPa]
F_{32} = FW loop B inlet header pressure [MPa]
6. Makeup (MU) System
F_{33} = MU tank water level [m]
F_{34} = Letdown water flow [kg/s]
F_{35} = MU water flow [kg/s]
7. Reactor Building (RB) Spray System
F_{36} = Borated water storage tank level [m]
F_{37} = Hydrazine tank level [m]
F_{38} = RB sump level [m]
8. Reactor Containment (CH)
F_{39} = CH temperature [K]
F_{40} = CH pressure [MPa]

an entire transient data base of 108 data points, i.e., D=108, where each of the 3 transient events was represented by 36 entries. Before the time rates of change $\Delta F_n/\Delta t$ were used as input to the neural network they were normalized to a [0,1] range with respect to the maximum time rate of change $(\Delta F_n/\Delta t)_{max}$ of the corresponding variable.

B. Selection of Plant Parameters

Both statistical feature selection techniques, linear correlation coefficients and information-theoretic entropy, discussed in Sec. II were used to select the N most meaningful features from the initial set of N = 40 plant parameters illustrated in Table II. By calculating the linear correlation coefficient r_{ij} in Eq. (1) for each pair of plant parameters and using a threshold of $|r_{ij}| > 0.90$ for considering two plant parameters as correlated, we were able to eliminate F_{16} = steam generator loop A exit pressure and F_{29} = pressure difference (DP) across feedwater loop A control valves. Plant parameters F_{16} and F_{29} were found to be correlated to F_{17} and F_{30}, respectively, which are the equivalent plant parameter measurements for loop B. By calculating the mutual information exchange $\Delta S(T|F_n)$ in Eq. (2) for the remaining 38 plant parameters of Table II, we were able to select the 10 most informative features, i.e., N=10. These 10 selected plant parameters which are listed in Table III were then used as the 10 elements of the network input patterns.

Table III. List of Ten Selected Features.

F_{12}	F_{13}	F_{17}	F_{20}	F_{21}
F_{22}	F_{23}	F_{24}	F_{30}	F_{32}

C. Neural Network Training

To determine the advantages of training a network with a reduced set of input variables and the more powerful method of CGs, we performed four sets of training runs. In each one of the four sets of training runs the neural network was designed with three layers, one input, one hidden, and one output layer. For the first set of runs, the BP algorithm was used to train an input-hidden-output network topology of 40-40-3 nodes, respectively. The 40 nodes of the input layer corresponded to the 40 plant parameters in Table II and the 3 nodes of the output layer corresponded to the 3 transient event types in Table I. For the second set of runs, the method of CGs was used to train the same 40-40-3 network topology. For the third set of runs, the BP algorithm was used to train an input-hidden-output network topology of 10-10-3 nodes, respectively. The 10 nodes of the input layer corresponded to the 10 most informative and non-redundant plant parameters illustrated in Table III. For the fourth set of runs, the method of CGs was used to train the same 10-10-3 network topology. In each one of the four sets of training runs the convergence criterion $\varepsilon=0.01$, i.e., $|t_{pj} - x_{pj}^{(L)}| < 0.01$ for all j=1.2,3 and p=1,2,...,108, was used and the target values t_{pj} (j=1,2,3) for pattern p representing transient event T_k (k=1,2,3) was set to 0.9 if k=j or to 0.1 if k≠j. In both BP and CG algorithms each iteration consisted of the presentation of the 108 data patterns after which the weights w were updated. In the two sets of BP runs, the learning constant η and the momentum constant α were set to 0.1 and 0.9, respectively.

Each one of the four sets of training runs was simulated ten times using a SPARC workstation 2, where at each one of the 10 simulations a different initial set of weights w_1 was used. Table IV shows the average CPU time and number of iterations obtained for each one of the four sets, BP (40-40-3), CG (40-40-3), BP (10-10-3), and CG (10-10-3), of training runs. The training of the 40-40-3 network topology updated a total of 1,763 weights w (links between the network nodes and thresholds for the hidden and output layer nodes) at each iteration, where the 10-10-3 topology updated a drastically reduced network of 143 weights. For a given network topology, the method of CGs outperformed the BP algorithm both in CPU time and number of iterations. For a given optimization method the 10-10-3 network topology outperformed the 40-40-3 topology in CPU time only, with a split in the number of iterations. Because of the different number of weights updated at each iteration, the CPU time is a more meaningful metric than the number of iterations when comparing networks of different topologies.

Table IV. Comparison of Four Training Runs.

	CPU (s)	Iterations
BP (40-40-3)	26,545	23,562
CG (40-40-3)	7,522	1,113
BP (10-10-3)	14,290	140,365
CG (10-10-3)	817	809

Each one of the two network topologies, 40-40-3 and 10-10-3, constitutes a different optimization problem which, in general, have differently shaped error-surfaces. This fact is evidenced by the large number of iterations to converge for the BP (10-10-3)

training run as compared with the BP (40-40-3) case. From our preliminary study, it appears that the error-surface for the smaller network is not as smooth as the error-surface for the large network, requiring a much larger number of iterations to converge. This fact is not as pronounced for the CG training runs due to the systematic adjustment of the learning and momentum constants at each iteration. It takes an average of 809 iterations to obtain the minimum for the 143 weights in the 10-10-3 network topology as compared to an average of 1,113 iterations for the 1,763 weights in the 40-40-3 topology. Of the four sets of training runs, the BP (40-40-3) was the worst performer while the CG (10-10-3) was by far the best one. By applying feature selection techniques and training the network with CGs, instead of BP, we obtained a 30-fold reduction in training time.

From Table IV we can also observe that the CPU time per iteration for a given network topology is larger for the CG training runs. This is due to the fact that the method of CGs requires a one-dimensional search at each iteration m to obtain η_m which is a time-consuming process. However, this larger CPU time per iteration does not offset the advantages of the method of CGs due to the much larger reduction in the number of iterations. The relative performance of the method of CGs with respect to the BP algorithm improves with tighter values of the convergence criterion ε.[7] This is due to the fact that the rate of convergence of the BP algorithm decreases as the minimum is approached, while the rate of convergence of the method of CGs may even increase due to a better approximation of $E(\mathbf{w}_m)$ as a quadratic function when m increases and the minimum is approached.

D. Neural Network Testing

To determine the diagnosis capability of the four sets of training runs discussed in Sec. IV.C we performed tests with two simulated transient events, one representing transient T_1 = feedwater pump trip and the other representing transient T_2 = feed flow transmitter fails high, which were not used in training the four networks. At every second, for the first 40 s of each one of the two simulated transients, the 40 or 10 plant parameters, depending on whether the network had 40 or 10 input nodes, respectively, were converted into their corresponding time rates of change which were then normalized to define 40 input patterns, i.e., P=40. The 40 input patterns, for each one of the two transients, T_1 and T_2, were then presented to the four networks, BP (40-40-3), CG (40-40-3), BP (10-10-3), and CG (10-10-3), and the average prediction error $E(\mathbf{w})$ in Eq. (3) over the 40 patterns was calculated.

We obtained similar results for the four trained networks, which indicate that neither the reduction in the number of plant parameters presented to the network, nor the method of CGs reduced the network diagnostic capabilities. Each one of the four networks correctly diagnosed transient T_2 with an average error per pattern presentation ranging from 10^{-6} to 10^{-5}, which was found to be independent of the initial set of weights $\mathbf{w}_1$ used to train the networks. However, none of the four networks was able to correctly diagnose transient T_1 for all 40 input patterns representing the first 40 s of the transient. The average error $E(\mathbf{w})$ in Eq. (3) per pattern presentation ranged from 0.3 to 0.7 depending on the initial set of weights $\mathbf{w}_1$. The inability of the four networks to correctly identify transient T_1 was due to the fact that the initial plant conditions and location of the simulated transient T_1 were different enough from the 12 simulations of T_1 used to train the networks. Transient T_1 corresponded to a loop B feedwater pump trip with the reactor operating at 99% of nominal power, while the only two simulations of a loop B pump trip, out of the 12 simulated transients, were performed at 80% and 60%, respectively, of nominal power. In any case, no significant differences in diagnosing T_1 were observed by the four trained networks.

V. SUMMARY AND CONCLUSIONS

By applying the two complementary methods, statistical feature selection and nonlinear optimization through conjugate gradients, we were able to expedite neural network training for nuclear power plant transient identification. Statistical feature selection techniques in the form of linear correlation coefficients and information-theoretic entropy were used to eliminate 75% of the initial set of measured plant parameters which were found to be redundant or non-informative. In addition to expediting neural network training through the reduction of the size of the network and hence the dimensionality of the minimization problem, feature selection may also play a role in avoiding "memorization" in ANNs.[15] If the network is too large such that the number of weights is much larger than the number of input-output patterns, the network may "memorize" the relationships between specific input-output patterns without generalizing between similar input-output pairs.

The method of conjugate gradients which has better convergence properties than the gradient descent-based backpropagation algorithm was also shown to accelerate ANN training by reducing both the number of iterations and the CPU time. In

addition, the training of ANNs with the method of CGs eliminates the need to select values for the learning and momentum constants. The method of CGs provides a systematic mechanism to determine both constants which are updated at each iteration. A 30-fold reduction in CPU time, without any reduction in the network diagnosis capability, was obtained by training a smaller network with the method of conjugate gradients, as compared to training a full-blown network with the backpropagation algorithm.

ACKNOWLEDGEMENTS

The first author was supported by the U.S. Department of Energy, Nuclear Energy Program, under contract number W-31-109-ENG-38.

REFERENCES

1. A. PAPOULIS, *Probability, Random Variables, and Stochastic Processes*, McGraw-Hill Book Company (1984).

2. C. E. SHANNON and W. WEAVER, *The Mathematical Theory of Communication*, University of Illinois Press (1949).

3. H. C. ANDREWS, *Introduction to Mathematical Techniques in Pattern Recognition*, John Wiley & Sons (1972).

4. D. E. RUMELHART, G. E. HINTON, and R. J. WILLIAMS, "Learning Internal Representations by Error Propagation," *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Vol. I, p. 319, D. E. RUMELHART and J. C. McCLELLAND, Eds., MIT Press (1986).

5. R. FLETCHER and C. M. REEVES, "Function Minimization by Conjugate Gradients," *The Computer Journal*, **7**, 149 (1964).

6. J. VITELA E. and J. REIFMAN, "Improving Learning of Neural Networks for Nuclear Power Plant Transient Classification," *Trans. Am. Nucl. Soc.*, **66**, 116 (1992).

7. J. REIFMAN and J. VITELA E., "Accelerating Learning of Neural Networks with Conjugate Gradients for Nuclear Power Plant Applications," paper submitted to *Nuclear Technology*, November (1992).

8. *Midland Plant Simulator Units I & II Manual*, Consumers Power Company, Midland, Michigan (1984).

9. R. L. WATROUS, "Learning Algorithms for Connectionist Networks: Applied Gradient Methods of Non-linear Optimization," *Proc. IJCNN First Int. Conf. Neural Networks*, San Diego, California, June 21-24, 1987, Vol. II, p. 619, M. CAUDILL And C. BUTLER, Eds., SOS Print, Piscataway, N.J. (1987).

10. D. A. WISMER and R. CHATTERGY, *Introduction to Nonlinear Optimization: A Problem Solving Approach*, Elsevier North-Holland (1978).

11. A. REZGUI and N. TEPEDELENLIOGLU, "The Effect of the Slope of the Activation Function on the Back Propagation Algorithm," *Proc. IJCNN Int. Conf. Neural Networks*, Washington, DC, January 15-19, 1990, Vol. I, p. 707, M. CAUDILL, Ed., IEEE Tab Neural Network Conference (1990).

12. J. R. CHEN and P. MARS, "Stepsize Variation Methods for Accelerating the Back-Propagation Algorithm," *Proc. IJCNN Int. Conf. Neural Networks*, Washington, DC, January 15-19, 1990, Vol. I, p. 707, M. CAUDILL, Ed., IEEE Tab Neural Network Conference (1990).

13. E. H. HAUG and J. S. AURORA, *Applied Optimal Design*, John Wiley & Sons (1979).

14. D. G. LUENBERGER, *Linear and Nonlinear Programming*, Addison-Wesley (1984).

15. A. WIELAND and R. LEIGHTON, "Geometric Analysis of Neural Network Capabilities, " *Proc. IEEE First Int. Conf. Neural Networks*, San Diego, California, June 21-24, 1987, Vol. III, p. 385, M. CAUDILL and C. BUTLER, Eds., SOS Print, Piscataway, N.J. (1987).

END

DATE
FILMED
6/17/93