REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*
May 1990

2. REPORT TYPE
Conference paper

3. DATES COVERED *(From - To)*

4. TITLE AND SUBTITLE
See report.

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
See report.

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

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

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
See report.

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT
Distribution Statement A - Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
Presented at the IEEE 1990 National Aerospace and Electronics Conference (NAECON 1990) held in Dayton, Ohio, on 21-25 May 1990.

14. ABSTRACT
See report.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER *(include area code)*
Unclassified	Unclassified	Unclassified	UU		

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

USING 3-D SURFACE MAPS TO ILLUSTRATE NEURAL NETWORK PERFORMANCE

Peter G. Raeth
Wright Research and Development Center
Avionics Laboratory; WRDC/AAWP-1
Electronic Support Measures Research Group
Wright-Patterson AFB, OH 45433-6543

Abstract

This paper suggests a possible means for evaluating the performance of neural networks from a global perspective in parameter-space. Traditional evaluations tend to focus on performance in weight-space or on overall output error during one training session. However, a global perspective of performance in parameter-space may be of primary importance during the initial stages of problem solution. During these stages, the researcher is typically trying to determine a network configuration and suitable values for its training equation parameters. Instead of a hit-or-miss approach, this paper describes an organized experimental method that identifies network configuration and parameter value choices which are not sensitive to minor variations for a standard training metric. The technique is illustrated for the network used by Hopfield and Tank to solve a traveling salesman problem and with traditional Backpropagation as described by Lippmann.

Introduction

The application of neural networks to new and complex problems would be greatly aided by a global view, in parameter-space, of neural network performance. It is the author's experience that researchers tend to offer combinations of training equation parameter values and network configurations without explanation or apparent systematic choice. For instance, in traditional backpropagation, the values chosen for the Gain and Momentum parameters are typically not explained. When a new problem is tried, the original values may or may not permit the network to learn the new mappings even if they are of the same class as the original. The researcher then has to try many variations of parameter values and network configurations or do detailed studies in error or weight-space in order to get the network to learn the new mappings. It would be better if there were some systematic method to show ranges of parameter values and network configurations that would work well for a given class of mappings. This desire has led to this paper and a longer term research effort aimed at neural network performance evaluation.

Before proceeding, it is necessary to define two terms as used in this paper.

Performance: The number of training cycles the network needs to carry out its intended task. For input/output vector mapping networks: the number of random exposures to the training vector pairs the network needs in order to learn the input/output mappings represented by the vectors. For energy minimization networks used to solve optimization problems (such as Hopfield and Tank's): the number of node updates needed before the network settles into its minimum energy state within a given tolerance.

Global/Local Perspective: Instead of looking at performance during one training session (local perspective), the global perspective looks at performance over many training sessions.

Convergence Maps

Convergence maps are N-dimensional plots which show the ability of a neural network to converge on (learn) a given training metric. The traveling salesman optimization problem is a classic metric for testing energy minimization networks. This is the metric discussed in this paper.

Two dimensional convergence maps have been used in the past to illustrate the performance of neural networks. Among the recent papers are Cherkassky and Vassilas, Perugini and Engeler, and Levine. Once such global measures are taken in parameter space, plots of the error surface or of weight space can be developed during a specific training run. These latter plots are useful for observing the behavior of a network at the local level. Based on global observations, specific changes may be indicated in the values of training equation parameters or in network configuration. Modifications to the training method and/or training equation can be made based on local observations. Either global and/or local performance measures can be taken again to judge the results of the changes. In this way, an organized experimental approach to the selection of training equation parameter values, network architecture, or the training method/equation for the problem class represented by the training metric could develop.

Maps of Hopfield and Tank's Traveling Salesman Neural Network

Hopfield and Tank's neural network for solving the traveling salesman problem presents an opportunity for trying out the ideas behind convergence maps. The thought here is to take a global look at the network's performance relative to its goal of arriving at valid tours. In their summary, Wilson and Pawley state, "Our simulations indicate that Hopfield and Tank were very fortunate in the limited number of TSP simulations they attempted. .... their basic method is unreliable ...." If we could take a global look at the performance of Hopfield and Tank's network, we could see for ourselves whether or not the training of the network is reliable. (In this sense "reliable" means "convergence on valid tours is not sensitive to variations in parameter values".) The following paragraphs show convergence maps which give us a start at getting this global look.

The training equation of the Hopfield and Tank network as used in this paper is shown below. The form presented here is due to Little's analysis of Hopfield and Tank's original mathematics. To initialize the network, the Uxi's are set to small random values. These are the values present in the network when t=DeltaT. Time is allowed to advance in steps of DeltaT until there are no further changes in the Uxi's (within a given tolerance). At this point, minimum energy has been achieved and a valid route should have resulted. As you will see, a valid tour does not always result. The frequency with which this happens led to Wilson and Pawley's remark.

$$U_{xi}(t+\Delta t) = U_{xi}(t) - \Delta t \Big\{ A\sum_{j\neq i} V_{xj} + B\sum_{y\neq x} V_{yi} + C\Big[\sum_{y}\sum_{i} V_{yi} - N\Big] + D\sum_{y} d_{xy}\big[V_{y,i+1}+V_{y,i-1}\big] + \big[U_{xi}(t)/\tau\big]\Big\}$$

The parameters in the training equation described above were set as follows except in the cases where they were varied to produce a given plot:

A	= 500
B	= 500
C	= 200
D	= 500
dxy	= Distance between cities x and y
DeltaT	= 0.0001 (change in time t)
N	= 15
t	= time
Tau	= So big that (Uxi(t) / Tau) could be assumed = 0
U0	= 0.02
Uinit	= -0.5 * U0 * Ln(#CITIES - 1)
Vxi	= 0.5 * (1.0 + TANH[Uxi(t)/U0])
x,y	= City number
i,j	= Tour position

Region of random selection for Uxi initialization = U(-0.1Uinit,+0.1Uinit)
Training cutoff = At valid tour, limit of 12000 node updates

It is possible to plot the number of node updates the network needed to converge on a valid tour. The plot could be based on 2500 training sessions where 50 value variations of one equation parameter are made for each of 50 variations of another parameter. The axes for such a plot are shown in Figure 1. Figures 2, 3, and 4 show plots for variations of the D & N, A & B, and Tau & DeltaT parameters respectively. By observing these plots it is possible to take an organized look at the Hopfield and Tank network from a global perspective. A similar method could be used to study the performance of other network types (see the author's other two papers on this subject). Some details on Figures 2, 3, and 4 are given below.

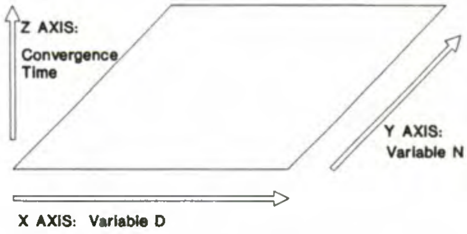


Figure 1: Example 3-Dimensional Convergence Map Axes

Figure 2. Choosing the above values for the network's training equation parameters and Hopfield and Tank's city locations as determined by Wilson and Pawley, the convergence map given in Figure 2 results if D = 0 - 600 and N = 0 - 30. As is readily evident, the map is generally a high table with a narrowing trench and one fairly wide pit where valid tours are reliably achieved within 12000 iterations of the equation for Uxi(t). This range of D and N was chosen because Hopfield and Tank suggest the above set of values and then say that some variation about those values may be necessary. The surface shows that between N=10.41 and N=19.59 over the entire range of D, there is plenty of opportunity for converging on a valid tour.

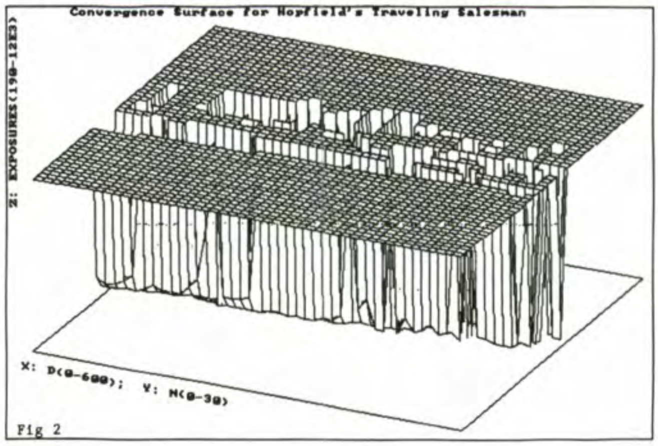


Figure 3. Fixing D at 12.24 and N at 11.63, the map in Figure 3 results under variations of A = 0 - 600 and B = 0 - 600. Observe the many opportunities for reliable convergence available when the values of A and B range over 220.41 - 600.0.

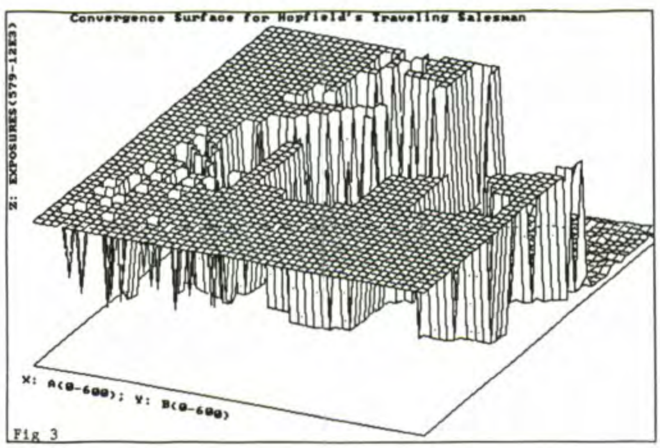
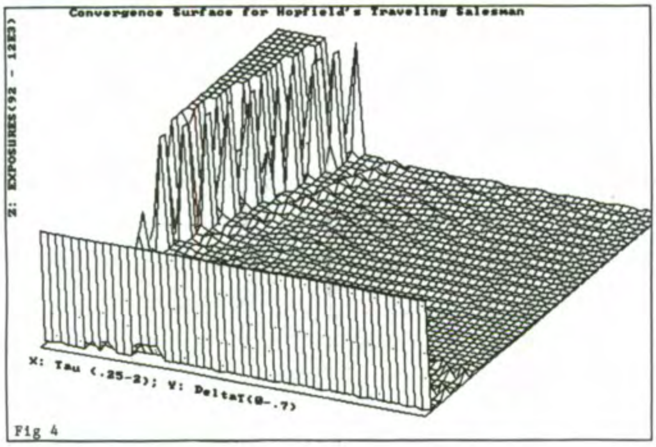


Figure 4. Setting A = 367.35, B = 428.47, D = 12.24, and N = 11.63; variations of DeltaT and Tau (0.0 - 0.7 and 0.25 - 2.0) produce the map in Figure 4. Notice that the surface is essentially low and flat except where DeltaT = 0.0 and for very low values of DeltaT coupled with very high values of Tau.

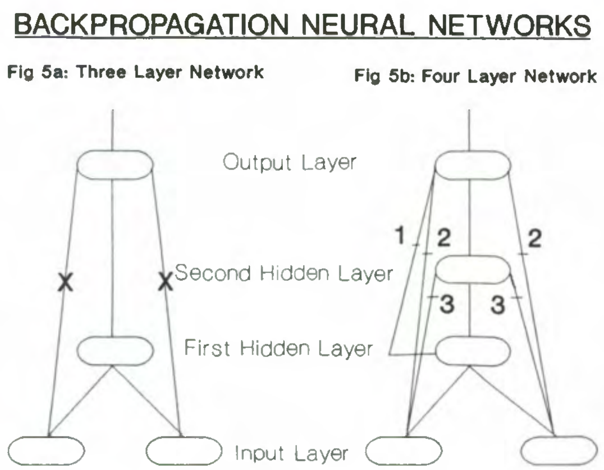


Conclusions on Hopfield & Tank's TSP Network

From the evidence provided by the above convergence maps, Hopfield and Tank's TSP network reliably finds valid tours but only in a vary narrow parametric region. It appears that the critical parameters for finding this region are D and N. As a matter of interest, this author's experience is that the list of random numbers chosen for use by the Hopfield and Tank TSP network has a critical impact on the network's performance.

Backpropagation and the 2D XOR Problem

For this set of experiments we developed Lippmann's traditional backpropagation neural network model with a modification by Klimasauskas. This development was also assisted by Gustafson's notes. The modification by Klimasauskas involved an exception to Lippmann's and Gustafson's specification in that this project's model uses a positive bias term instead of a negative bias term.



Three and four layer models were generated and tested. Their architectures ares shown in Figure 5. Using these models, runs were made to generate 3-dimensional convergence maps based on the 2-dimensional XOR problem. This involved training the network to map the XOR inputs to the desired outputs. Variations were made of parameter values (Gain, momentum, and distribution) and network configuration (interlayer connections and number of nodes in the hidden layers). Convergence maps were drawn to show the ability of the network to learn the XOR input/output mappings. Generally, there are some rather dramatic variations in the network's ability to converge on (learn) the desired mapping, depending on how one picks parameter values and network configurations. However, the results show clearly how, for this class of mapping, to pick values and configurations that are not sensitive to minor variation in parameter values or network configuration. This will benefit the project when we try more complex, but related, mappings.

In these experiments, Gain was incremented from 0 to 4 in 50 trials and either Distribution, Number of Hidden Nodes, or Momentum was incremented for each increment of Gain. The result of each trial then became a point on the 3-dimensional surface. Other facts about the runs are: Initialization: U(-.1,.1) except in cases where Distribution was one of the varied parameters; Training: random examples from the two-dimensional XOR table; Computer: DEC MicroVax III under Ultrix using Berkeley Pascal; Weight Updates: Asynchronous within layers, Synchronous between layers; Momentum: 0 except in cases where Momentum was varied; Number of Nodes in Hidden Layers: 1 except in cases where Number of Nodes was varied; Acceptable Error: 0.1 for each input/output pair. The random seed was the same for all runs and the generator was reseeded with the same seed after the initial weights were selected. Training was cut off after 100,000 exposures if convergence did not take place by then.

Four-layer backpropagation: The four-layer backpropagation connection architecture is shown in Figure 5b.

The connection architectures used in this experiment were 1) fully connected, input connected to both the output and the hidden layers and 2) input not connected to output. (Line 2 disconnected.) The experiment results given below compare the two architectures' ability to develop the 2-dimensional XOR mappings given variations of Gain, Momentum, Initialization Distribution Size, and Number of Hidden Nodes.

Variations of Momentum: Momentum was varied from 0 to 1 inclusive in 50 steps for each of 50 steps of Gain. Architecture 1): As Momentum increases, it has less and less a desirable affect. As Gain increases, Momentum lends less and less assistance to convergence. The fastest convergence occurred at Gain = 3.61 and Momentum = 0.57. However, that is deep in a pit so it is better to choose something like Gain = 1.5 and Momentum = 0.02, values that are in the middle of the low flat plain. The plot for this experiment is shown in Figure 6. Architecture 2): This experiment showed very little opportunity for reliable convergence, only with high values of Gain and low values of Momentum. It would be interesting to extend this plot to Gain = 10, something we may do in the next phase. Convergence was fastest for Gain = 3.51 and Momentum = 0.45. Very near the back-right wall. Figure 7 shows the plot.

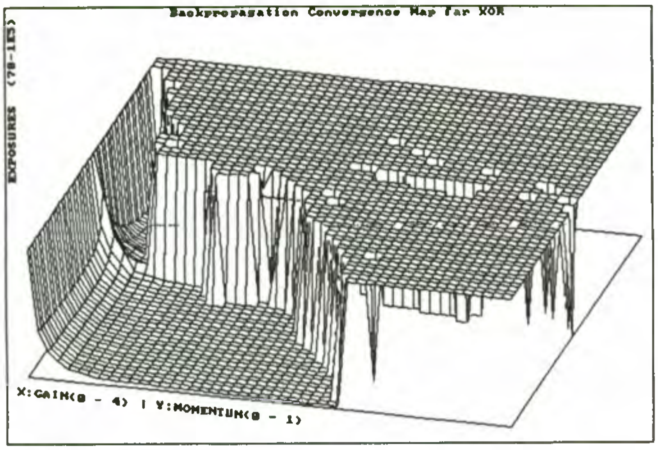


Fig 6. Fully Connected Network Using Gain and Momentum

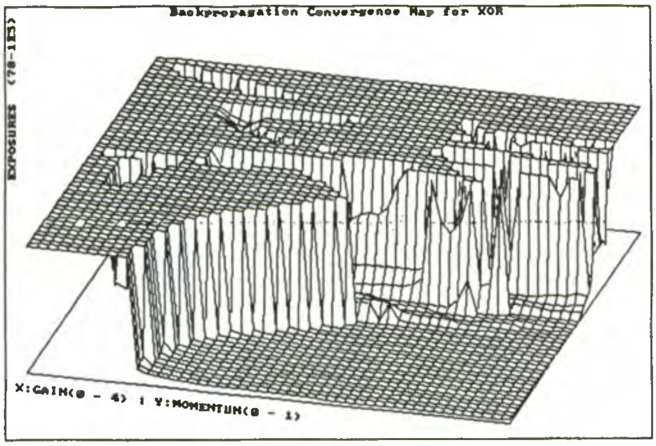


Fig 7. Input not Connected to Output Using Gain and Momentum

Variations of weight initialization distribution: To initialize the network weights, values were chosen randomly from a uniform distribution whose size varied in 50 steps for each of 50 increments of Gain. The variation was U(-.1,+.1) to U(-2, +2) inclusive. Architecture 1): Distribution variations had very little affect on this architecture's ability to converge. Notice some contrary regions, however. Convergence is difficult for very low values of Gain and very high values of Distribution. Very high values of Gain for most values of Distribution also negatively impact convergence except in the rare case where there is a combination of very high Gain and very large Distribution. Convergence was fastest for Gain = 3.92 and Distribution = 2, in the flat plain which appears in the back-right of the plot. See Figure 8 for this plot. Architecture 2) For this architecture, convergence almost never occurred except for small Distributions. As Gain increased, Distribution generally helped but the affect was minimal. The fastest convergence occurred at Gain = 0.82 and Distribution = 2, a deep pit. Figure 9 illustrates these results.

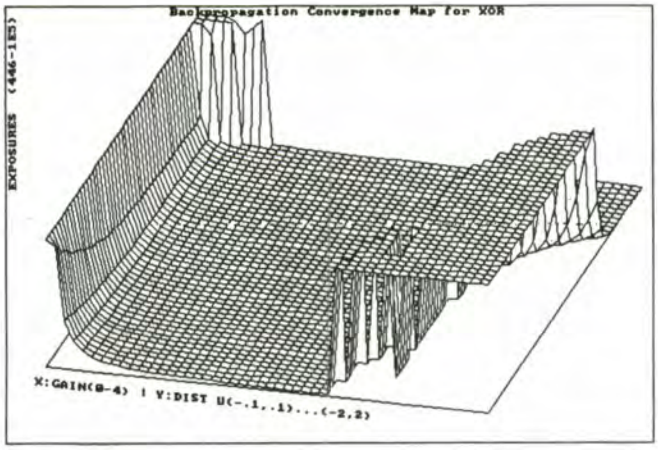


Fig 8. Fully Connected Network Using Gain and Initialization Distribution

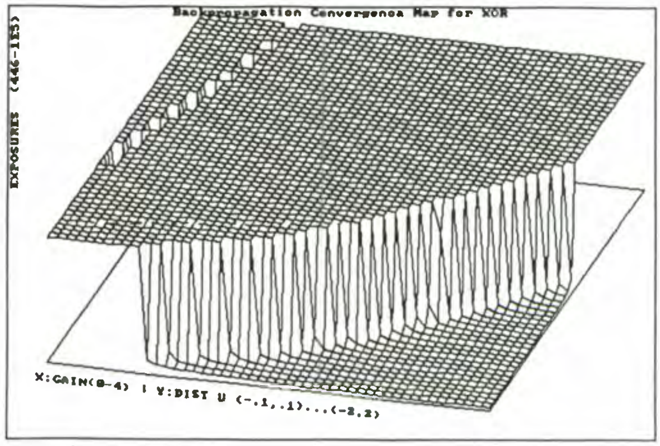


Fig 9. Input not Connected to Output Using Gain and Initialization Distribution

Variations of number of hidden nodes: The number of hidden nodes in both layers was varied from 1 - 10 in 10 steps for each of 50 increments of Gain so that each hidden layer always had the same number of nodes. Only 10 nodes were run since the amount of computer time needed would have been too great to go higher. Architecture 1): Variations of number of hidden nodes led to tremendous unreliability in convergence. The fastest time was recorded at Gain = 2.09 and Number of Nodes = 8, in the middle of an unreliable region. See Figure 10 for this plot. Architecture 2) This was a most surprising result in light of the other plots from this architecture. In this case, the second architecture resulted in better performance. The plot is generally like a slide that angles downward, left to right. Only at very low values of Gain is this not so and even then at Nodes = 1. Convergence was fastest at Gain = 3.76 and Nodes = 10. Figure 11 has this plot.

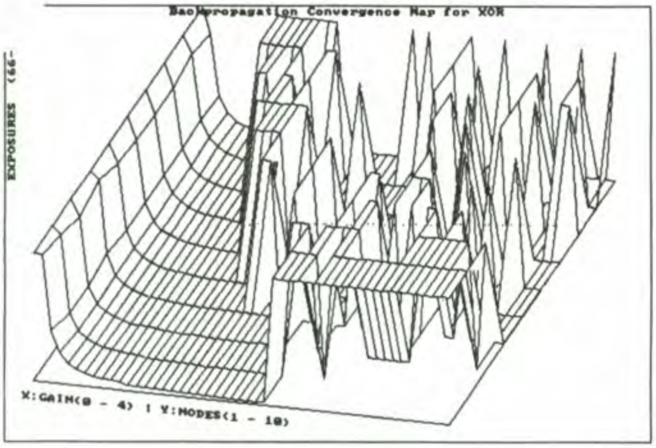


Fig 10. Fully Connected Network Using Gain and Number of Hidden Nodes

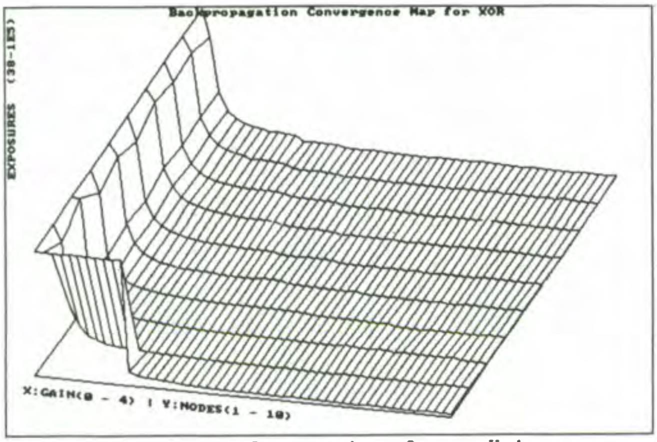


Fig 11. Input not Connected to Output Using Gain and Number of Hidden Nodes

Three-layer backpropagation: The three-layer backpropagation connection architecture is shown in Figure 5a.

Three experiments with the three-layer backpropagation model were tried:

1) data lines marked "X" connected, Gain varied from 0 - 4 in 50 steps, and Random Distribution for Initialization varied from U(-.1,.1) to U(-2,2) in 50 steps for each step of Gain.
2) data lines marked "X" connected, Gain varied from 0 - 4 in 50 steps, and Number of Hidden Layer Nodes varied from 1 - 10 in 10 steps for each step of Gain.
3) data lines marked "X" connected, Gain varied from 0 - 4 in 50 steps, and Momentum varied from 0 - 1 in 50 steps for each step of Gain.

Convergence was not achieved with data lines marked "X" disconnected.

Experiment 1: The point of convergence maps is the ability to see where the regions of reliable convergence are. This experiment achieved its fastest convergence time at Gain = 3.10 and Distribution = U(-.41,.41). These parameters are near the outer edge of the low flat region near the right wall. The conclusion is that these values are too near the wall for convergence to be insensitve to small changes in their value. One would be better advised to use something like Gain = 2 and Distribution = (-.5,.5) when trying a new problem in this problem class. These values put one near the middle of the low flat plain. Figure 12 illustrates the results of this experiment.

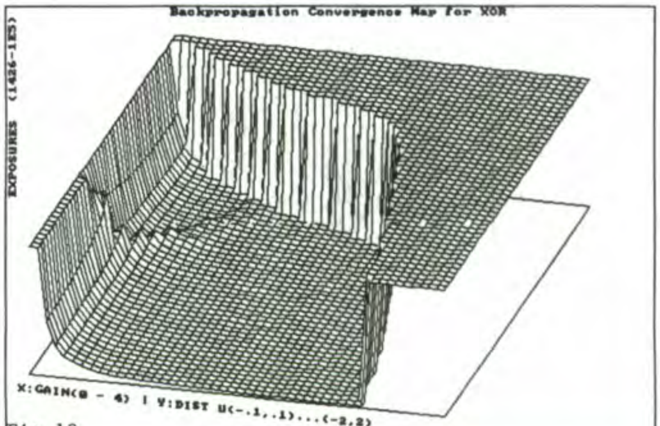


Fig 12

Experiment 2: The fastest convergence occured at Gain = 3.51 and Number of Hidden-Layer Nodes = 10. However, note that after Gain = 2.4, variations in Number of Hidden-Layer Nodes causes considerable unreliable network behavior. Thus, there would be no predicting what would happen if a new problem were tried. It would be better to try something like Gain = 1.6 and Number of Hidden-Layer Nodes = 2 in order to get into the low flat region. Figure 13 illustrates the results of this experiment.

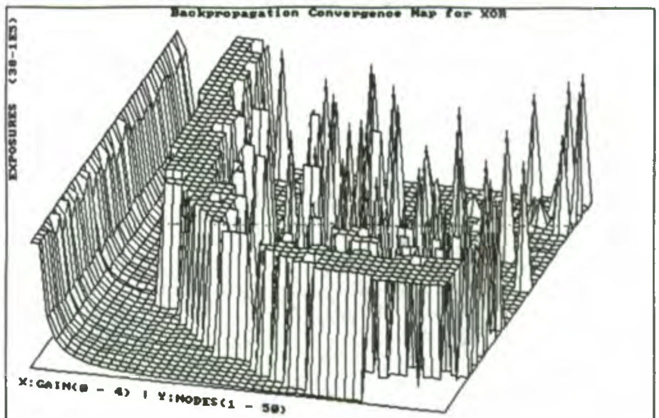


Fig 13

Experiment 3: Convergence was fastest at Gain = 2.53 and Momentum = .63. These values are due to the pit which occurs in the middle of the high flat region. AGain, One should not choose these values when trying some new problem. Rather, choose Gain = 1 (or so) and Momentum not greater that .5. Figure 14 illustrates the results of this experiment.

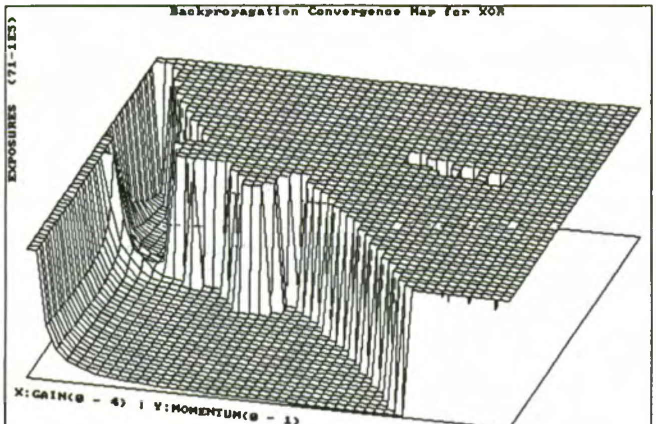


Fig 14

Conclusions on Backpropagation

It is obvious from the convergence maps shown in this report that the node connection architecture has a dramatic affect on the network's ability to learn a set of vector mappings. Less dramatic, but just as important, are the values of the network's training equation parameters. Preliminary tests have suggested that these parameters need to be set not to the values which give fastest learning for a given metric but to those values which give reliable learning. Reliable learning ability is shown on the maps as low flat plains. In most of the maps shown in this report, the fastest learning occurred in a pit found on a high flat plain.

Continuing Research

A concern which came about as a result of this study has to do with the basic theory of backpropagation. The theory says that backpropagation neural networks are guaranteed to learn an arbitrary set of vector mappings. The theory does not say how long it will take to learn a given set of vector mappings. This is no problem when the data domain is fixed. However, in most military applications, the data domain is not fixed. This leads to questions on what happens if the number of vector mappings to be learned increases and what happens if the number of components in the vectors increases. Certainly, one would expect the number of exposures required for learning to increase. But, what is the rate of increase? Is it linear, geometric, or exponential? It is not sufficient to say, "Well, just put it on your Cray and let it run". In military applications, we have to be able to guarantee reprogrammability within a given amount of time. We will address these issues in the next phase of this research as part of our work in event-train restoration.

A second concern is the fact that most of the neural models which this author is familiar with are written in either C, Pascal, or Basic. For neural networks to transition to military systems, the models will have to be in Ada. At the present time, the most likely technology for early implementation of neural networks in military hardware involves transputers. A modeling capability is needed which will generate Ada code and the Occam harnass for transputer systems.

An effort now underway is the development of 4-dimensional convergence maps. A 4-dimensional plot is produced by fixing w and then generating a three dimensional plot by varying x and y to get z [z = f(x,y)]. Then, w is changed by some fixed delta after which the three dimensional plot is again produced. This continues until the desired number of 3D plots is produced. The result is a series of 3D plots based on variations of w where each plot frame uses fixed variations of x and y to get z. If enough of these plots are produced, they can be "played back" in rapid sequence to see how the 3D plot evolves based on the variations in w and how, for a fixed w, z changes based on x and y. So, for instance, it should be possible to obtain plots of convergence time on the z axis with w, x, and y being momentum, gain, and number of hidden layer nodes.

Acknowledgments

The author wishes to thank Dr Gordon Little for his analysis of the Hopfield and Tank network and Dr Steve Gustafson for his discussions on Backpropagation. Both professors are with the University of Dayton Research Institute. They both reviewed this paper and offered many helpful comments. Their Jan 89 graduate course in neural networks saw the beginnings of this research project. Thanks are also due to Debbie Ables for her administrative support.

Bibliography

Cherkassky, Vladimir and Nikolaos Vassilas; "Performance of Back Propagation Networks for Associative Database Retrieval"; Proc: International Joint Conference on Neural Networks; pI-77; 1989

Gustafson, Steve; Class notes on Backpropagation; University of Dayton Research Institute; Jan 89

Hopfield, J.J. and D.W. Tank; "Neural Computation of Decisions in Optimization Problems"; Biological Cybernetics, 52, 141-152, 1985

Klimasauskas, Casimir C. "Neural Networks: A Short Course", PC AI Magazine, Nov/Dec 88

Levine, R.Y.; "Neural Network Performance on the Stochastic Exclusive-Or Problem"; DTIC/NTIS # AD-A197-789; Jul 88

Lippmann, Richard P.; "An Introduction to Computing with Neural Nets"; IEEE ASSP Magazine, Apr 87

Little, Gordon; Class notes on Hopfield and Tank TSP; University of Dayton Research Institute; Apr 89

Perugini, N.K. and W.E. Engeler; "Neural Network Learning Time: Effects of Network and Training Set Size"; Proc: International Joint Conference on Neural Networks; pII-395; 1989

Raeth, Peter G.; "3-D Surface Maps for Neural Network Performance Evaluation"; Proc: ACM Dayton SIGART Aerospace Applications of Artificial Intelligence Conference; Oct 89

Raeth, Peter G.; "Event-Train Restoration Via Backpropagation Neural Networks (Jan-Jun 89)"; National Technical Information Service & Defense Technical Information Center, Cameron Station, Virginia (USA); Order # AD-A216-308; Dec 89

Wilson, G.V. and G.S. Pawley; "On the Stability of the Travelling Salesman Problem Algorithm of Hopfield and Tank"; Biological Cybernetics, 58, p63, 88