EFFICIENT METHOD FOR SOLUTION OF CONTINOUS HOPFIELD NETWORKS

BOGDAN M. WILAMOWSKI AND STANLEY M. KANAROWSKI
Department of Electrical Engineering
University of Wyoming

ABSTRACT:
An expanded contraction mapping theorem to single layer feedback neural networks of a gradient-type is discussed. The derivation of a modified relaxation algorithm is presented. The algorithm converges well for continuous recurrent neural networks with low and high neurons gains. Two different implementations of the proposed algorithm are experimentally compared.

INTRODUCTION

Recurrent neural networks such as Hopfield networks are used for solving many practical problems including optimization (Zurada,1991), storing and retrieving patterns (Tank and Hopfield,1986), solving linear equations systems (Chichocki and Unbehauen,1992), linear and nonlinear programming (Kennedy and Chua,1988) and other related problems as listed in (Zurada and Kang,1991). In all cases a numerical simulation is always required even if hardware implementations are desired.

An exact transient solution requires the set of differential equations (1) to be solved. The most commonly used solution is the forward or backward Euler formula. Such an approach typically requires a large number of iteration steps with small time steps before a steady-state solution is reached. An alternate procedure, the Newton-Raphson method, can only be used when a steady-state solution is required. This method is commonly used for electronic circuit analysis. Although this technique requires only a few iteration steps, it is computationally very intensive. Another possible approach is to use the relaxation method. It is much easier to implement, but converges only for networks with small neuron gains. The gain limitations become more severe for larger systems (Zurada and Shen,1992). In many applications, high gain neurons are required and thus this method can not be used. In this discussion the modified incremental relaxation method is presented. Similar to the relaxation method no equations need to be solved. This method converges very well for all neuron gains, even for extremely large gains as required for hard threshold discrete neurons.

ALGORITHM PRINCIPLES

The basic system diagram is shown in Fig. 1. A typical activation function $v_i = f(u_i) = [\ 1 + \exp(-\lambda u_i)\ ]^{-1}$ is used in this network. The input conductances and capacitances are equal to g_i and C_i, respectively. The network transient response is

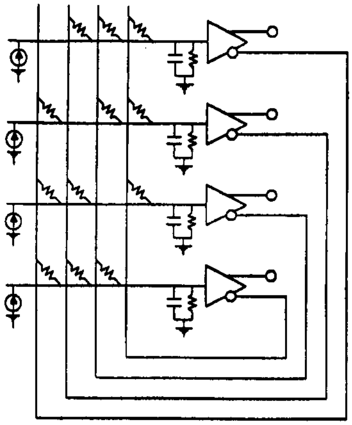

Figure 1: Single-layer recurrent neural network with four neurons.

described by the following set of differential equations (Zurada,1991)(Tank and Hopfield,1986):

$$C_i \frac{du_i}{dt} = \sum_{j=0}^{n} w_{ij} v_j - G_i u_i + i_i = cur_i \quad , \text{where} \quad G_i = \sum_{j=1}^{n} w_{ij} + g_i \tag{1}$$

Each side of equation (1) is also equal to the unbalanced sum of the currents cur_i flowing toward the node. This also corresponds to the appropriate i-th component of the energy function gradient. Furthermore the system energy reaches a minimum when all components of the gradient vector are equal to zero.

The relaxation algorithm requires the iterative computation of successive output values using the formula:

$$u_i^{k+1} = \frac{1}{G_i} \sum_{j=0}^{n} \left(w_{ij} v_j^k + i_i \right) \quad , \text{and} \quad v_i^{k+1} = f\left(u_i^{k+1}\right) \tag{2}$$

The relaxation method does not converge for networks with high neuron gains as required for many practical networks. An alternate approach for obtaining transient solutions is based on the charge conservation principle [8]. This simple algorithm follows the physical phenomena in the circuit and it is very efficient for transient analysis of VLSI digital sub-circuits. For medium size circuits (20 transistors) it is about 100 times faster than the standard SPICE algorithm (Wilamowski, Staszak, Hamilton,1986). After each iteration step the unbalanced current of every node is computed using equation (1). This current is either charging or discharging the corresponding capacitor depending upon it's sign. Assuming a small time increment, the corresponding voltage change on the input nodes can be found using:

$$\Delta u_i = \frac{\Delta t \; cur_i}{C_i} \tag{3}$$

The above algorithm will converge if the time increment Δt is chosen small enough. It can be proved that if the value of Δt is equal to the time constant of the input node then the computation process is equivalent to the relaxation method. Thus the relaxation method as described in (Zurada and Shen,1992) is a special case of the method based on the charge conservation principle. Therefore the algorithm with Δt=τ will not converge for large values of neuron gain λ, just like the relaxation method. To assure convergence, Δt in equation (3) should be reduced so that the voltage changes at each iteration step are small. A further simplification of the algorithm is possible since solely steady state solutions are required for recurrent neural networks. One can prove that the relaxation formula (2) can also be useful, if the results obtained in equation (2) are modified in the following way(Wilamowski, Staszak, Hamilton,1986):

$$u_i^{new} = u_i^{old} + \frac{1}{A_i + 1}\left(u_i^{relax} - u_i^{old} \right) \tag{4}$$

where A_i is the gain of the corresponding neuron and u_i^{relax} is obtained from the relaxation formula (2). The activation function A_i will always have a gain smaller than λ/4. Instead of using the maximum gain of $A_i = \lambda/4$, it is possible to speedup the computation process and still guarantee convergence by computing the gain of each neuron at every iteration step using:

$$A_i = \frac{\partial v_i}{\partial u_i} = \lambda_i \, v_i(1 - v_i) \tag{5}$$

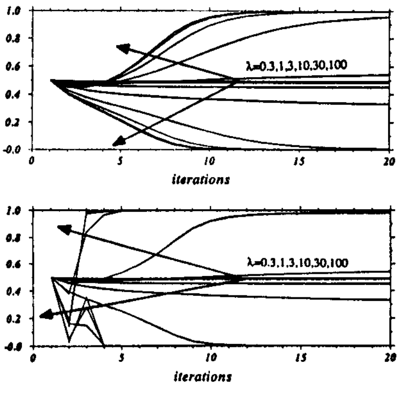

Figure 2: Output results for a 2-bit A/D converter with x = 1.3 and λ = 0.3, 1, 3, 10, 100 which converges to the binary state *01*; (a) algorithm uses A_i=λ/4; (b) algorithm uses equation (5).

EXAMPLE RESULTS

The algorithm was proven to converge by using the same example as in (Zurada and Shen, 1992) in addition to many other experiments. As an example two-bit, four- and eight-bit analog to digital converters were tested with single-layer recurrent Hopfield networks. Fig. 2(a) shows the results for the 2-bit A/D converter, with x = 1.3, λ = 0.3,1,3,10,30,100 and $A_i = \lambda/4$. This circuit converges to a binary state of *01*. The results for the same 2-bit A/D converter using equation (5) are depicted in Fig. 2(b). Thus one can see that when using the formula (5), the system converges more rapidly. Figure 3 presents results for four bit AD converter converging to the input value of x = 7.0 using recurrent neural networks with (a) λ = 10 and (b) λ = 1000. Note that network converges very rapidly for very large neuron gains.

Fig. 4 illustrates the convergence of an 8-bit A/D converter with an input value x = 182.3. This recurrent network converges to the binary state of *11000000*. Fig. 4(a) illustrates the results for a network with λ = 100, while Fig. 4(b) is for λ = 10,000.

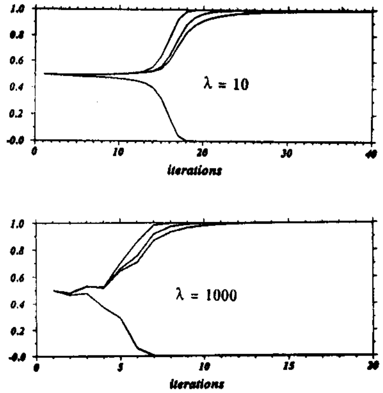

Figure 3: Output results using the algorithm with equation (5) for an 4-bit A/D converter with x = 7.0 which converges to the binary state *0111*; (a) λ = 10; (b) λ = 1000.

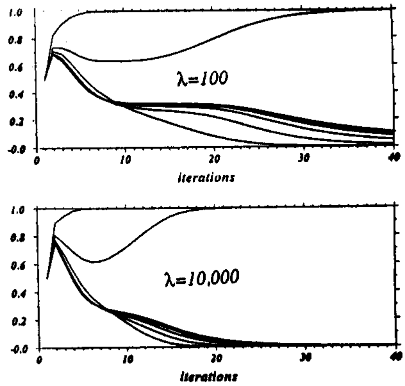

Figure 4: Output results using the algorithm with equation (5) for an 8-bit A/D converter with x = 182.3 which converges to the binary state *11000000*; (a) λ = 100; (b) λ = 10,000.

CONCLUSION

It was demonstrated that through a slight modification of the relaxation algorithm it is possible to insure convergence for any neuron gain (λ in the activation function). This was proven by the same example as in (Zurada and Shen, 1992) but also by many other examples. This enables one to simulate recurrent Hopfield networks with both "soft" and "hard" continuous activation functions. Furthermore the proposed algorithm requires far less computation effort than applying the Newton-Raphson to equation set (1), or any other method of solving differential equations. It should be noted that algorithm is suitable for large systems, since the computational effort is proportional only to the system size in comparison to the commonly used Newton-Raphson method where power relationship exist. Thus the use of the modified relaxation method is applicable to both small and large size networks regardless of neuron gain.

REFERENCES

Cichocki, A., Unbehauen, R. (1992). Neural networks for solving system of linear equations and related Problems, *IEEE Trans. on Circuits and Syst. I*, CASI-39,124-138.

Hopfield, J.J. (1984). Neurons with graded response have collective computational properties like those of two state neurons, *Proc. Nat. Acad. Sci.*, vol. 81,3088-3092.

Kennedy, M.P., Chua, L.O. (1988). Neural networks for nonlinear programming, *IEEE Trans. Circuits Sys.*,vol. CAS-35,May,554-562.

Tank, D.W., Hopfield, J.J. (1986), Simple neural optimization networks: An A/D converter, signal decision circuit and a linear programming circuit, *IEEE Trans. Circuits Syst.*, vol. CAS 33,May,533-541.

Wilamowski, B.M., Staszak, Z.J., and Hamilton, D.J. (1986). CHARCO - IC transient analysis program, *IEEE International Circuit and System Symposium*, San Jose, California,May,548-551.

Zurada, J.M. (1991). Gradient-type neural systems for computation and decision-making," in *Progress in Neural Networks, Vol. II*, O. M. Omidvar, Ed. Norwood, NJ:Ablex.

Zurada, J.M., Kang, M.J. (1991). Computational circuits using neural optimization concepts," *Int. J. Electron.*, vol. 67, no. 3,311-320.

Zurada, J.M., Shen, W. (1992). Sufficient condition for convergence of a relaxation algorithm in actual single-layer neural networks, *IEEE Trans. Neural Networks*, vol. 1, no. 1,Dec, 300-303.

ASME PRESS

INTELLIGENT ENGINEERING SYSTEMS THROUGH ARTIFICIAL NEURAL NETWORKS

VOLUME 3

Editors:
Cihan H. Dagli
Laura I. Burke
Benito R. Fernández
Joydeep Ghosh