濟州大學校 産業技術硏究所 論文集, 第8卷, 1號, pp.122~130, 1997. 6
Cheju National Univ. R.I.I.T. Jour., Vol.8, No.1, pp.122~130, 1997. 6

Numerical Modeling of Continuous-time Fully Coupled Neural Networks

Min-Jae, Kang*

완전결합된 Continuous-time 신경회로망의 수치해석적 모델링

강 민 제*

ABSTRACT

This paper discusses numerical performance modeling of fully coupled continuous-time networks utilizing continuous activation functions, finite input resistance of neurons, and other parasitic components within the neural system. Both time-domain performance modeling and static numerical modeling of the networks are characterized and compared.

Key words : Gradient-type networks, Dynamic analysis, Static analysis, Vector field method, Convergency

Ⅰ. Introduction

Single-layer feedback neural networks, also called gradient-type networks, have recently received widespread attention in the technical literature [1-4]. Most of the published results, however, deal with the discrete-time, discrete-output networks. The discrete-time networks represent limit case of continuous-time networks which involve continuous activation functions and change the output values continuously rather than discretely. Some of the postulates regarding both the convergence and performance of discrete-time networks remain valid for their continuous-time counterparts. There are also numerous differing aspects in both types of networks. These aspects are discussed in this paper.

Ⅱ. Description of Continuous-time Actual Neural Networks

The model of a fully coupled neural system consists of n neurons each

* 제주대학교 전자공학과
Dept. of Electronic Eng., Cheju Nat'l Univ.

mapping the input voltage u_i of the i-th neuron into the output voltage v_i through the activation function f(u) which is the common static voltage transfer characteristic(VTC) of the neuron. The common choice is the so-called sigmoidal function

$$v_i = f(u_i) = [1+\exp(-\lambda u_i)]^{-1},$$
$$i = 1,2, \ldots, n.$$

The neurons' gain value λ is assumed to be finite. Conductance w_{ij} connects the output of j-th neuron to the input of the i-th neuron. The i-th input conductance and capacitance are of non-zero values, and equal to g_i, C_i, respectively.

The network is described by following equation

$$\underline{C}\frac{d\underline{u}}{dt} = \underline{W}\underline{v}(t) + \underline{i} - \underline{G}\underline{u}(t) \qquad \text{(1a)}$$

$$\underline{v}(t) = \underline{f}(\underline{u}(t)) \qquad \text{(1b)}$$

where, using customary notation, we have

$$\underline{C} \equiv diag[C_1, C_2, \cdots, C_n]$$

$$\underline{G} \equiv diag[\sum_{j=1}^{n} w_{1j} + g_1, \sum_{j=1}^{n} w_{2j} + g_2, \cdots, \sum_{j=1}^{n} w_{nj} + g_n]$$

The postulated energy function $E(\underline{v})$ for this system has the following form [1,7]

$$E(v) = -\frac{1}{2}\underline{v}^t\underline{W}\underline{v} - \underline{i}^t\underline{v} + \sum_{i=1}^{n} G_i \int_{0.5}^{v_i} f_i^{-1}(z)\,dz \qquad (2)$$

The negative gradient vector of the energy function (2) can be computed as

$$-\nabla E(\underline{v}) = \underline{W}\underline{v} - \underline{G}\underline{u} + \underline{i} \qquad (3)$$

By comparing (3) with the right hand side of (1a) it can be written as

$$-\nabla E(\underline{v}) = \underline{C}\frac{d\underline{u}}{dt} \qquad (4)$$

III. Dynamic Analysis of Continuous-Time Networks

The vector field method is presented below as a tool for analysis of gradient-type neural networks. This method can generate trajectories capturing the transients and equilibrium points in v^n space. it enables the complete dynamic solutions for all possible initial conditions, finite gain values of neurons, while including parasitic conductances and capacitances which occur in actual neural systems.

$$\dot{u}_i = \frac{1}{C_i}(i_i + \sum_{j=1}^{n} W_{ij}v_i - G_i u_i), \quad i = 1,2, \ldots, n \qquad (5)$$

Equation(5) can now be expressed in the output space as follows

$$\frac{dv_i}{dt} = \psi_i[\underline{v}(t)], \quad i = 1,2, \ldots, n \qquad (6)$$

and used to compute the derivatives $\psi_1(\underline{v}), \psi_2(\underline{v}), \ldots, \psi_n(\underline{v})$. As a result, the vector field is obtained that indicates the complete trajectories of the system. The components $\psi_i(\underline{v})$ of the vector field for single-layer feedback networks can be explicitly computed as

$$\phi_i(\underline{v}) = \frac{\lambda_i(v_i - v_i^2)}{C_i} \left(i_i + \sum_{j=1}^{n} W_{ij}v_i - G_i f^{-1}(v_i)\right), \quad (7)$$
$$i = 1, 2, \ldots, n$$

Components ϕ_i of the computed vector determine the motion of the system output in the direction v_i. Approximated actual displacements of the output are equal to products $\phi_i \Delta t$.

The approximation for v_i^{k+1} is

$$v_i^{k+1} = v_i^k + \Delta v_i^k$$
$$i = 1, 2, \ldots, n \quad (8)$$

where Δv_i is the vector component of a displacement-step. It is equal to the product of the normalized vector component by a displacement-step

$$\Delta v_i^k = n(\phi_i(\underline{v}^k))d$$
$$I = 1, 2, \ldots, n \quad (9)$$

where d is a user-selectable displacement-step, and $n(\phi_i(\underline{v}^k))$ called normalized vector field component is defined as follows

$$n(\phi_i(\underline{v}^k)) \equiv \frac{\phi_i(\underline{v}^k)}{\left(\sum_{i=1}^{n} \phi_i^2(\underline{v}^k)\right)^{\frac{1}{2}}}$$
$$i = 1, 2, \ldots, n \quad (10)$$

Thus, the length of the sum of the vector components of a displacement-step is equal to the displacement-step. The stable displacement-step depends on a system. The output values of the Hopfield networks are in the region [0,1]. The reasonable displacement-step for the stable system can be chosen as d=0.01. Thus, the system moves by 0.01 each iteration.

Case Study

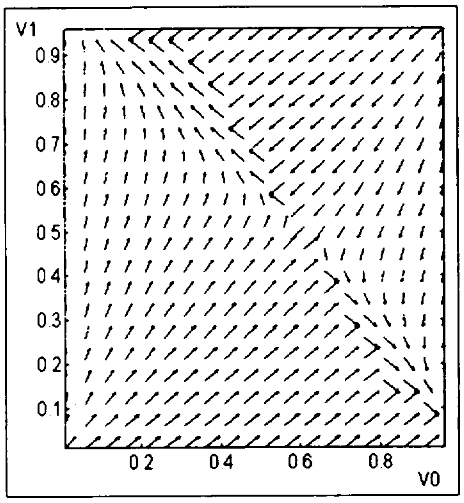
A 2-bit A/D converter representing a class of optimization, or gradient-type networks[4,5], is selected as a case study to test the method. The state space equation(1) describing the converter are

$$\begin{aligned} C_0\dot{u}_0 &= x - 0.5 - 2v_1 - (g_0 - 2)u_0 \\ C_1\dot{u}_1 &= 2x - 2 - 2v_0 - (g_1 - 2)u_1 \end{aligned} \quad (11)$$

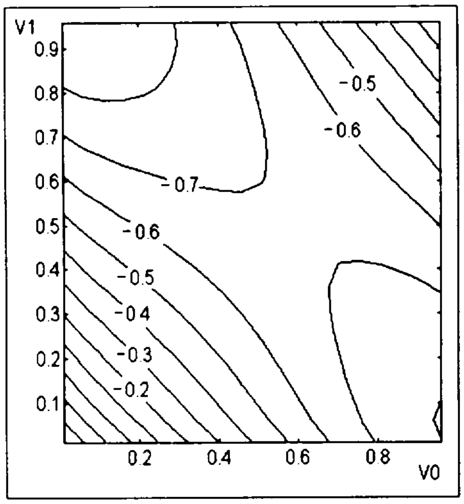
Where x is the analog voltage value to be converted to the binary reprsentation $\underline{v} = [v_0, v_1]^t$. Equation (11) can be rearranged with form (7) to follow form

$$\begin{aligned} \dot{v}_0 &= \frac{\lambda_0}{C_0}(v_0 - v_0^2)[x - 0.5 - 2v_1 \\ &\quad - (g_0 - 2)\ln(\frac{v_0}{1 - v_0})] \\ \dot{v}_1 &= \frac{\lambda_1}{C_1}(v_1 - v_1^2)[2x - 2 - 2v_0 \\ &\quad - (g_1 - 2)\ln(\frac{v_1}{1 - v_1})] \end{aligned} \quad (12)$$

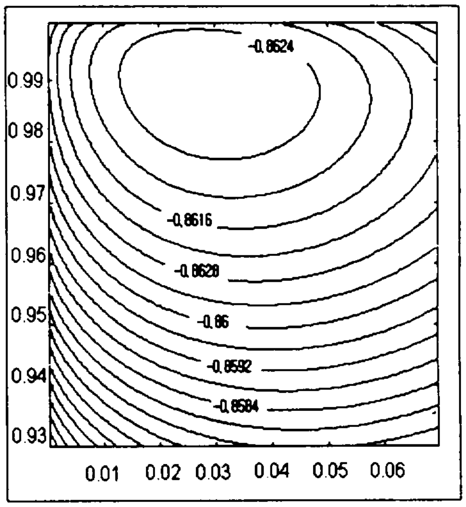
The normalized vector field of this system can now be produced for known values of x, g_i, C_i, λ_i (i=1,2). Fig. 1 provides a comparison of the vector field analysis with the actual energy map of the system for the case x=1.6, $\lambda = 2$, $C_1/C_0 = 1$, g=2.5. There is one saddle point and two minima indicated by vector in Fig. 1a. The actual energy map with the third therm of energy in Fig. 1b indicates the direction of movement of output in time. To gain more detailed insight into convergence, the energy map near one of the two minima $\underline{v}$ = [0, 1] have been expanded



a) vector field



b) energy map



c) macro map for upper left corner

Fig. 1 Vector field energy maps comparison $x=1.6,\ \lambda=2,\ C_1/C_0=1,\ g=2.5$

as shown in Fig. 1c. It can be seen that the energy minima near the corner is indeed within the unity square and very close to the corner.

In summary, the vector field approach provides a detailed insight into the transient behavior and stability conditions of the network. Although the method can be graphically illustrated only for n≤3, it can be applied to networks of any dimensional size.

Ⅳ. Static Numerical Modeling

In contrast to the dynamic analysis involving time-domain analysis in the preceding section, here we introduce the static numerical methods. They are relaxation algorithm and Newton-Raphson method. These methods are based on the assumption that the continuous-time system stabilizes in space v when the energy gradient (3) reaches zero.

4.1 Relaxation Algorithm and Convergence

The relaxation algorithm is based on the contraction mapping theorem. Here this algorithm is applied to solve equilibrium points for the continuous-time networks having electronic components. However, convergence of the algorithm is limited. this limitation is presented in section 4.1.2.

4.1.1 Relaxation Algorithm for continuous-time System

The continuous-time system moves down its energy surface. Thus, the system stops moving and becomes stable when the gradient (3) is zero. Using this particular property of the network, the solution for minimum E(v) in the output space can be written as

$$\underline{\nabla E(v)} = \underline{0} \qquad (13a)$$

$$\underline{Wv} - \underline{Gu} + \underline{i} = \underline{0} \qquad (13b)$$

By using the above equation v_i can be expressed as follows

$$v_i = f\left[\frac{1}{G_i}\sum_{j=1}^{n}(w_{ij}v_j + i_i)\right],$$
$$i = 1,2,\ldots,n \qquad (14a)$$

The iterative numerical solution for v_i can be modeled using the contraction mapping theorem as follows

$$v_i^{k+1} = f\left[\frac{1}{G_i}\left(\sum_{j=1}^{n} w_{ij}v_j^k + i_i\right)\right],$$
$$i = 1,2,\ldots,n \qquad (14b)$$

This is a static relaxation algorithm suitable for numerical calculation of the stable solutions as opposed to dynamic, numerical integration related, algorithms depicting the transients within the continuous-time network(6).

4.1.2 Sufficient Condition for Convergence of (14)

The convergence of (14) to the unique fixed point is guaranteed by the sufficient condition

$$\left|\frac{\partial f_i(\underline{v})}{\partial v_j}\right| \le \frac{K}{n},$$
$$\text{where } K < 1 \qquad (15)$$

for each j=1, 2, ..., n and each component function f_i(4). For f_i specified by each equation in (14) we obtain

$$\frac{\partial f_i(\underline{v})}{\partial v_j} = \frac{\partial f_i(u_i)}{\partial u_i}\frac{\partial u_i}{\partial v_j} \qquad (16)$$

where

$$\frac{\partial f_i(u_i)}{\partial u_i} = \frac{\lambda e^{-\lambda u_i}}{(1+e^{-\lambda u_i})^2}$$

Based on $u_i(\underline{v})$ specified in square brackets of the right-hand side of (16), the second derivative of the chain in (16) can be expressed as

$$\frac{\partial u_i}{\partial v_j} = \frac{w_{ij}}{G_i} \qquad (17)$$

Combining (16) and (17) allows the rewriting of (15) as follows

$$\left| \frac{\lambda e^{-\lambda u_i}}{(1+e^{-\lambda u_i})^2} \cdot \frac{w_{ij}}{G_i} \right| < \frac{1}{n} \quad (18)$$

The condition (18) can be further simplified by noting that it is equivalent to

$$\frac{1}{n}\left[1+\left(2-n\lambda\left|\frac{w_{ij}}{G_i}\right|\right)e^{-\lambda u_i}+e^{-2\lambda u_i}\right] \geq 0 \quad (19a)$$

The binomial in brckets of the left-hand side of the inequality (19a) should remain positive for all values of the variable $\exp(-\lambda u_i)$. This is fulfilled when the quadratic equation equating it to zero has no real roots. The above condition translates to

$$1-\frac{1}{4}\left(2-n\lambda\left|\frac{w_{ij}}{G_i}\right|\right)^2 \geq 0 \quad (19b)$$

It is obvious that (19b) is fulfilled for

$$-2 \leq 2-n\lambda\left|\frac{w_{ij}}{G_i}\right| \leq 2 \quad (20a)$$

The above condition can be briefly expressed as

$$\lambda \leq \frac{4}{n}\left|\frac{G_i}{w_{ij}}\right| \quad (21b)$$

Thus, the convergence of the relaxation algorithm (14b) to the unique fixed point of (14a) is guaranteed by the sufficient condition

$$\lambda \leq \min_{\substack{i=1,2,\dots,n \\ j=1,2,\dots,n \\ i\neq j}} \left\{\frac{4}{n}\cdot\left|\frac{G_i}{w_{ij}}\right|\right\} \quad (22)$$

It can be seen that the condition (22) for the network with fixed conductances w_{ij}, G_i imposes bounds on the highest values of neruons′ gain.

Case Study

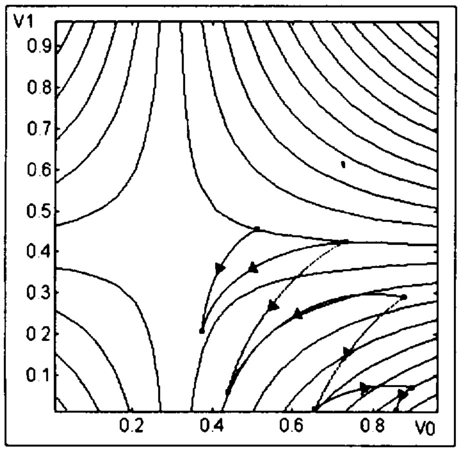
The stability of the numerical solution using the relaxation algorithm (14b) has been tested using the same circuit as before.

Fig. 2 shows the fixed point iterations for x=1.3. The numerical solution has been shown in the output space using the background of energy contours given (without the third term of (2)) as

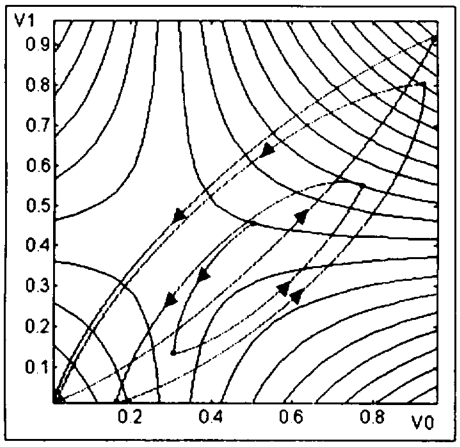
$$E(\underline{v}) = 2v_0v_1+\frac{v_0}{2}+2v_1-x(v_0+2v_1) \quad (23)$$

It should be noted that the numerical relaxation algoritm neither operates on the energy surface, nor in time, and the contours of $E(\underline{v})$ are provided solely for visualizing real transients in actual continuous-time neural network. It can be noticed that there are two minima of $E(\underline{v})$ at $\underline{v} = [0,1]^t$ or $\underline{v} = [1,0]^t$. There is also a saddle point at $\underline{v} = [.3\ .4]^t$ which sivides the bimodal energy surface.

Fig. 2a illustrates the stable relaxation algorithm applied for $\lambda=3.5$. Increase of λ to 5 or above brings instability of the relaxation modeling as shown is Fig. 2b. It should be noted that the values of maximum λ ensuring numerical convergence and computed from sufficient condition (22) are much smaller than those at which stable solutions could still be actually computed.



a) stable solution for $\lambda = 3.5$.



b) unstable solution for $\lambda = 5$.

Fig. 2 Relaxation Algorithm for network $x = 1.3$.

In summary, a fixed point relaxation algorithm has the potential of finding stable solutions of the actual continuous-time single-layer feedback network discussed. The stability of the numerical solution using this method, however, is severely restricted. Sufficient condition (22) for numerical stability of the algorithm would limit its application to networks with relatively low gain neurons. In addition, the stable solution obtained using iterations as in (14b), although mathematically correct, may not be identical to the solution of the dynamic system, even with the same starting point. Saddle points are among the solution; they are not, however, equilibrium points of the networks.

4.2 Newton-Raphson Method of Energy Function Minimization

Below, another static algorithm which is based on the Newton-Raphson approach is presented. Compared to the relaxation algorithm, this method can be used for any size network to find the stationary point, or solution of the equation $\underline{\nabla E}(\underline{v}) = \underline{0}$.

Formally, given the point $\underline{u}$ to be the current approximation of the stationary point, the next linear apporximation point can be obtained as

$$\underline{u}^{k+1} = \underline{u}^{k} - [\underline{J}(\underline{u}^{k})]^{-1} \underline{F}(\underline{u}^{k}) \qquad (24)$$

where $\underline{F}(\underline{u})$ is the energy gradient expressed in input u space.

$$\underline{F}(\underline{u}) = \begin{bmatrix} \sum_{j=1}^{n} W_{1j} f(u_j) - G_1 u_1 + i_1 \\ \cdots\cdot \\ \sum_{j=1}^{n} W_{nj} f(u_j) - G_n u_n + i_n \end{bmatrix} \qquad (25)$$

$$\underline{J}(\underline{u}^{k}) = \begin{bmatrix} \frac{\partial f_0}{\partial f_0} & \frac{\partial f_0}{\partial f_1} & \cdots & \frac{\partial f_0}{\partial f_{n-1}} \\ & & \cdots & \\ . & . & . & . \\ \frac{\partial f_{n-1}}{\partial f_{n-1}} & \frac{\partial f_{n-1}}{\partial f_1} & \cdots & \frac{\partial f_{n-1}}{\partial f_{n-1}} \end{bmatrix} \qquad (26)$$

This method always converges to a stable solution, dependent strongly, however, on initial point $\underline{u_0}$. The different initial point can cause a possibility for this method to diverge rather than converge to the true stationary point.

Case Study

The 2-bit A/D converter is selected to test this method for finding stationary points. Equation (24) for the 2-bit A/D converter is

$$\begin{bmatrix} u_0^{k+1} \\ u_1^{k+1} \end{bmatrix} = \begin{bmatrix} u_0^k \\ u_1^k \end{bmatrix} - [J(\underline{u})]^{-1} \begin{bmatrix} F(u_0^k) \\ F(u_1^k) \end{bmatrix} \tag{27}$$

where

$$J(u) = \begin{bmatrix} 2-g_0 & -2\lambda(v_1 - v_1^2) \\ -2\lambda(v_0 - v_0^2) & 2-g_1 \end{bmatrix}$$

$$F(u_0) = -2v_1 + x - 0.5 - (g_0 - 2)u_0$$

$$F(u_1) = -2v_0 + 2x - 2 - (g_1 - 2)u_1$$

The Newton-Raphson method iterations for x=1.3 and $\lambda=5$ contrast the relaxation algorithm which fails to converge for $\lambda=5$. This method finds all stationary points, two minima and a saddle point as shown in Fig. 3. However, the different initial point causes diverging rather than converging to one of the minima.

In summary, even though this method is difficult of its sensitivity to the initial condition and because it costs much to calculate the inverse of Jacobian matrix every iteration, the method can be applied to large actual networks.

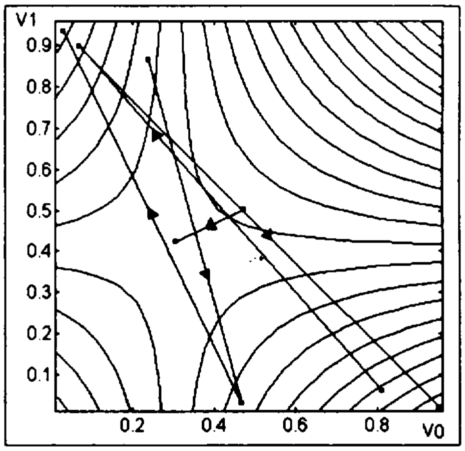


Fig. 3 Newton-Raphson Method for 2-bit A/D converter, x=1.3 and $\lambda=5$

V. Concluding Remarks

The main focus of this paper had been the dynamics and static numerical modeling of actual Hopfield-type networks. The vector field method has been presented for dynamics analysis, and it has been compared with numerical relaxation formulas and the Newton-Raphson method. It has been shown that the relaxation algorithm may not often guarantee numerical convergence while the vector field method does. So, vector field method has been considered as a reliable tool for continuous-time neural network analysis.

References

1) J. J. Hopfield, D. W. Tank, 1985, "Neural" Computation of Decisions in Optimization Problem, Biolog. Cvbern., 52, pp. 141-154.
2) J. J. Hopfield, 1984, Neurons with graded response have collective computatuional properties like those of two state neurons, Proc. Natl. Acad. Sxi., USA, Vol. 81, pp. 3088-3092.
3) J. J. Hopfield, D. W. Tank, 1986, Computing with Neural Circuits: A Model, Science, Vol. 233, pp. 625-633.
4) D. W. Tank, J. J. Hopfield, 1986, Simple "Neural" optimization Networks: An A/D Converter, Signal Decision Circuit and a Linear Programming Circuit, IEEE Trans. on Circ. and Syst., Vol. CAS33, May 1986, No. 5, pp. 533-541.
5) J. M. Zurada, 1991, Gradient-type Neural Systems for Computation and Decision-Making, in Progress in Nerral Networkd, Vol. 2, Ablex Publishing Corp., O. M. Omidvar, Editor, April.
6) E. DiZitti, et al, 1989, Analysis of Neural Algorithms for Parallel Architectures, Proc. of the 1989, pp. 2187-2190.
7) B. C. Levy, M. B. Adams, 1987, Global Optimization with Stochastic Neural Networkd, IEEE International Conference on Nerual Networks, 1987, San Diego, CA, pp. 681-689.