IEEE Transactions on Power Systems, Vol. 13, No. 2, May 1998

Adaptive Hopfield Neural Networks for Economic Load Dispatch

Kwang Y. Lee and Arthit Sode-Yome
Senior Member, IEEE Student Member, IEEE
Department of Electrical Engineering
The Pennsylvania State University
University Park, PA 16802

June Ho Park
Member IEEE
Department of Electrical Engineering
Pusan National University
Pusan 609-735, Korea

***Abstract*– A large number of iterations and oscillation are those of the major concern in solving the economic load dispatch problem using the Hopfield neural network. This paper develops two different methods, which are the slope adjustment and bias adjustment methods, in order to speed up the convergence of the Hopfield neural network system. Algorithms of economic load dispatch for piecewise quadratic cost functions using the Hopfield neural network have been developed for the two approaches. The results are compared with those of a numerical approach and the traditional Hopfield neural network approach. To guarantee and for faster convergence, adaptive learning rates are also developed by using energy functions and applied to the slope and bias adjustment methods. The results of the traditional, fixed learning rate, and adaptive learning rate methods are compared in economic load dispatch problems.**

Key words– Economic load dispatch, Hopfield neural networks, adaptive Hopfield neural networks.

I. INTRODUCTION

In power system, the operation cost at each time needs to be minimized via economic load dispatch (ELD). Traditionally, the cost function of each generator has been approximately represented by a single quadratic cost function. Practically, operating conditions of many generating units require that the generation cost function be segmented as piecewise quadratic functions. Therefore, it is more realistic to represent the generation cost function as a piecewise quadratic cost function, and Lin and Viviani [1] presented the hierarchical economic dispatch for piecewise quadratic cost functions using a Lagrangian function.

PE-217-PWRS-0-01-1997 A paper recommended and approved by the IEEE Power System Engineering Committee of the IEEE Power Engineering Society for publication in the IEEE Transactions on Power Systems. Manuscript submitted July 15, 1996; made available for printing January 8, 1997.

Since Hopfield introduced in 1982 [2] and 1984 [3], the Hopfield neural networks have been used in many different applications. The important property of the Hopfield neural network is the decrease in energy by finite amount whenever there is any change in inputs [11]. Thus, the Hopfield neural network can be used for optimization. Tank and Hopfield [4] described how several optimization problem can be rapidly solved by highly interconnected networks of a simple analog processor, which is an implementation of the Hopfield neural network. Park and others [5] presented the economic load dispatch for piecewise quadratic cost functions using the Hopfield neural network. The results of this method were compared very well with those of the numerical method in an hierarchical approach [1]. King and others [6] applied the Hopfield neural network in the economic and environmental dispatching of electric power systems. These applications, however, involved a large number of iterations and often shown oscillations during transients. This suggests a need for improvement in convergence through an adaptive approach, such as the adaptive learning rate method developed by Ku and Lee [7] for a diagonal recurrent neural network.

II. ECONOMIC LOAD DISPATCH

The ELD problem is to find the optimal combination of power generation that minimizes the total cost while satisfying the total demand. The cost function of ELD problem is defined as follows:

$$C = \sum_i C_i(P_i), \tag{1}$$

$$C_i(P_i) = \begin{cases} a_{i1} + b_{i1} P_i + c_{i1} P_i^2, & \text{fuel } 1, \quad \underline{P}_i \le P_i \le P_{i1} \\ a_{i2} + b_{i2} P_i + c_{i2} P_i^2, & \text{fuel } 2, \quad P_{i1} \le P_i \le P_{i2} \\ \vdots & \\ a_{ik} + b_{ik} P_i + c_{ik} P_i^2, & \text{fuel } k, \quad P_{k-1} \le P_i \le \overline{P}_i \end{cases}, \tag{2}$$

where

$C_i(P_i)$: cost of the i^{th} generator
P_i : the power output of generator i
a_{ik}, b_{ik}, c_{ik}: cost coefficients of the i^{th} generator for fuel type k.

In minimizing the total cost, the constraints of power balance and power limits should be satisfied:

a) Power balance

The total generating power has to be equal to the sum of load demand and transmission-line loss:

$$D + L - \sum_i P_i = 0\,, \tag{3}$$

where D is total load, and L is transmission loss.

The transmission loss can be represented by the B-coefficient method as

$$L = \sum_i \sum_j P_i B_{ij} P_j\,, \tag{4}$$

where B_{ij} is transmission loss coefficient.

b) Maximum and minimum limits of power

The generation power of each generator has some limits and it can be expressed as

$$\underline{P_i} \le P_i \le \overline{P_i}\,, \tag{5}$$

where

$\underline{P}$: the minimum generation power

$\overline{P}$: the maximum generation power.

III. HOPFIELD NETWORKS AND MAPPING OF ELD.

A. The Hopfield Neural Networks

The continuous neuron model is a generalized Hopfield network in which the computational energy decreases continuously in time [3,10]. For a very high-gain parameter (λ) of the neurons, continuous networks perform in a way similar to the discrete model. Since the weight parameter vector is symmetric, the energy function of Hopfield neural network is defined as

$$E = -\frac{1}{2}\sum_i \sum_j T_{ij} V_i V_j - \sum_i I_i V_i + \sum_i \theta_i V_i\,, \tag{6}$$

where V_i is output value of neuron i, I_i is external input to neuron i, and θ_i is threshold bias.

The dynamics of the neurons is defined by

$$\frac{dU_i}{dt} = \sum_j T_{ij} V_j + I_i\,, \tag{7}$$

where U_i is the total input to neuron i and the sigmoidal function can be defined as

$$V_i = g_i(\lambda U_i) = g_i\left(\frac{U_i}{U_0}\right) = \frac{1}{1 + \exp\left(-\frac{U_i + \theta_i}{U_0}\right)}. \tag{8}$$

Stability is known to be guaranteed since the energy function is bounded and its increment is found to be nonpositive as

$$\frac{dE}{dt} = -\sum_i g_i'(U_i)\left(\frac{dU_i}{dt}\right)^2. \tag{9}$$

Since $g_i(U_i)$ is a monotone increasing function, each term in this sum is nonnegative. Therefore (9) is less than zero. The time evolution of the system is a motion in state-space that seeks out minima in E and comes to a stop at such points.

B. Mapping of ELD Into the Hopfield Neural Networks

In order to solve the ELD problem, the following energy function is defined by augmenting the objective function (1) with the constraint (2):

$$E = \frac{1}{2}A\Big(D + L - \sum_i P_i\Big)^2 + \frac{1}{2}B\sum_i (a_i + b_i P_i + c_i P_i^2)\,. \tag{10}$$

where a_{ik}, b_{ik}, c_{ik} are the cost coefficients as discrete functions of P_i defined in (1).

By comparing (10) with (6) whose threshold is assumed to be zero, the weight parameters and external input of neuron i in the network [5] are given by

$$\begin{aligned} T_{ii} &= -A - Bc_i\,,\\ T_{ij} &= -A\,,\\ I_i &= A(D+L) - \frac{Bb_i}{2}\,, \end{aligned} \tag{11}$$

where the diagonal weights are nonzero. This converts (7) into the following synchronous updating rule:

$$U_i(k) - U_i(k-1) = \sum_j T_{ij} V_j(k) + I_i\,. \tag{12}$$

Unlike the asynchronous model, the synchronous model has fixed points as well as limit cycles as attractors. However, it does not get trapped to local minima as easily as the asynchronous model. There are additional advantages of the synchronous model in computation and hardware savings [9]. The sigmoidal function (8) can be modified [5] to meet the power limit constraint as follows:

$$V_i(k+1) = (\overline{P_i} - \underline{P_i})\frac{1}{1 + \exp\left[-\frac{U_i(k) + \theta_i}{U_0}\right]} + \underline{P_i}\,. \tag{13}$$

IV. ADAPTIVE HOPFIELD NETWORKS

The traditional approach in solving the economic load dispatch (ELD) problem using the Hopfield neural network requires a large number of iterations and often oscillates during the transient [5] and [8]. In order to speed up

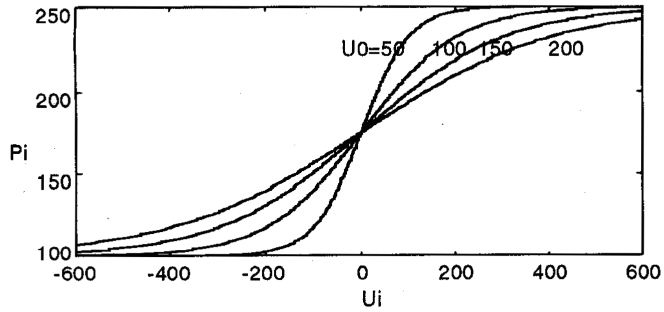

Fig. 1. Sigmoidal threshold function with different values of the gain parameter.

convergence, two adaptive adjustment methods are developed in this paper: slope adjustment and bias adjustment methods.

A. Slope Adjustment Method

In transient state, the neuron input oscillates around the threshold value, zero in Figure 1. Some neuron inputs oscillate away from the threshold value. If the gain parameter is set too high, the oscillation will occur at the saturation region. If the slope in this region is too low, the neurons can not go to the stable state and will cause instability.

Since energy is to be minimized and its convergence depends on the gain parameter U_0, the gradient-descent method can be applied to adjust the gain parameter as

$$U_0(k+1) = U_0(k) - \eta_s \frac{\partial E}{\partial U_0}, \tag{14}$$

where η_s is a learning rate.

From (10) and (13), the gradient of energy with respect to the gain parameter can be computed as

$$\frac{\partial E}{\partial U_0} = \sum_{i=1}^{n} \frac{\partial E}{\partial P_i} \frac{\partial P_i}{\partial U_0}. \tag{15}$$

The update rule of (14) needs a suitable choice of the learning rate η_s. For a small value of η_s, convergence is guaranteed but speed is too slow, on the other hand if the learning rate is too big, the algorithm becomes unstable. For faster and to guarantee convergence, a method to compute adaptive learning rates is developed following the procedure in Ku and Lee [7]. It can be shown [7,8] that convergence is guaranteed if the learning rate η_s is chosen as

$$0 < \eta_s < \frac{2}{g_{s,\max}^2}, \tag{16}$$

where $g_{s,\max} := \max\|g_s(k)\|$, $g_s(k) = \partial E(k) / \partial U_0$.

Moreover, the optimal convergence is corresponding to

$$\eta_s^* = \frac{1}{g_{s,\max}^2}. \tag{17}$$

This show an interesting result that any other learning rate larger than η_s^* does not guarantee a faster convergence.

B. Bias Adjustment Method

There is a limitation in the slope adjustment method, in that, slopes are small near the saturation region of the sigmoidal function, Fig 1. If every input can use the same maximum possible slope, convergence will be much faster. This can be achieved by changing the bias to shift the input near the center of the sigmoidal function. The bias can be changed following the similar gradient-descent method used in the slope adjustment method:

$$\theta_i(k+1) = \theta_i(k) - \eta_b \frac{\partial E}{\partial \theta_i}, \tag{18}$$

where η_b is a learning rate.

The bias can be applied to every neuron as in (8), therefore, from (10) and (13), a derivative of energy with respect to a bias can be individually computed as

$$\frac{\partial E}{\partial \theta_i} = \frac{\partial E}{\partial P_i} \frac{\partial P_i}{\partial \theta_i}. \tag{19}$$

The adaptive learning rate is also developed following the similar procedure [8]. It can be shown that convergence is guaranteed if η_b is chosen as

$$0 < \eta_b < -\frac{2}{g_b(k)}, \tag{20}$$

where

$$g_b(k) = \sum\sum T_{ij} \frac{\partial V_i}{\partial \theta} \frac{\partial V_j}{\partial \theta}. \tag{21}$$

Moreover, the optimal convergence is corresponding to

$$\eta_b^* = -\frac{1}{g_b(k)}. \tag{22}$$

Again, any other learning rate larger than η_b^* does not guarantee a faster convergence.

C. Momentum

The speed of convergence can be accelerated by adding momentum in the update processes. The momentum can be applied when updating the input in (12), the gain parameter in (14) and the bias in (18):

$$U_i(k) - U_i(k-1) = \sum_j T_{ij} V_j(k) + \alpha_u \, \Delta U_i(k-1), \tag{23}$$

$$U_0(k) = U_0(k-1) - \eta_s \frac{\partial E}{\partial U_0} + \alpha_s \Delta U_0(k-1), \quad (24)$$

$$\theta(k) = \theta(k-1) - \eta_b \frac{\partial E}{\partial \theta} + \alpha_b \Delta\theta(k-1), \quad (25)$$

where α's are momentum factors.

V. SIMULATION RESULTS

The results of our Hopfield neural network are compared with those of the numerical method [1] and an earlier Hopfield neural network [5]. Then the results of slope adjustment and bias adjustment methods with fixed learning rate are compared with those with adaptive learning rates. Finally, momentum is applied to all update processes and results are compared. Graphs for 2500 MW load are compared in order to demonstrate the convergence properties.

A. Conventional Hopfield Network

The conventional Hopfield neural network for ELD problem using piecewise cost functions has been developed based on [5]. The same data has been used as in Table 1. Each generator has three types of fuel and there are four values of load demand, that is, 2400, 2500, 2600 and 2700 MW. The results are compared with the numerical method and the earlier Hopfield network reported in [5] as shown in the Table 2.

Table. 1: Cost coefficients for piecewise quadratic cost functions.

U	GENERATION Min	F1	P1	F2	P2	F3	Max	F	COST COEFFICIENTS a	b	c
1	100		196		250		250	1	.2697e2	-.3975e0	.2176e-2
		1		2		2		2	.2113e2	-.3059e0	.1861e-2
								2	.2113e2	-.3059e0	.1861e-2
2	50		114		157		230	1	.1184e3	-.1269e1	.4194e-2
		2		3		1		2	.1865e1	-.3988e-1	.1138e-2
								3	.1365e2	-.1980e0	.1620e-2
3	200		332		388		500	1	.3979e2	-.3116e0	.1457e-2
		1		3		2		2	-.5914e2	.4864e0	.1176e-4
								3	-.2876e1	.3389e-1	.8035e-3
4	99		138		200		265	1	.1983e1	-.3114e-1	.1049e-2
		1		2		3		2	.5285e2	-.6348e0	.2758e-2
								3	.2668e3	-.2338e1	.5935e-2
5	190		338		407		490	1	.1392e2	-.8733e-1	.1066e-2
		1		2		3		2	.9976e2	-.5206e0	.1597e-2
								3	-.5399e2	.4462e0	.1498e-3
6	85		138		200		265	1	.5285e2	-.6348e0	.2758e-2
		2		1		3		2	.1983e1	-.3114e-1	.1049e-2
								3	.2668e3	-.2338e1	.5935e-2
7	200		331		391		500	1	.1893e2	-.1325e0	.1107e-2
		1		2		3		2	.4377e2	-.2267e0	.1165e-2
								3	-.4335e2	.3559e0	.2454e-3
8	99		138		200		265	1	.1983e1	-.3114e-1	.1049e-2
		1		2		3		2	.5285e2	-.6348e0	.2758e-2
								3	.2668e3	-.2338e1	.5935e-2
9	130		213		370		440	1	.8853e2	-.5675e0	.1554e-2
		3		1		3		2	.1530e2	-.4514e-1	.7033e-2
								3	.1423e2	-.1817e-1	.6121e-3
10	200		362		407		490	1	.1397e2	-.9938e-1	.1102e-2
		1		3		2		2	-.6113e2	.5084e0	.4164e-4
								3	.4671e2	-.2024e0	.1137e-2

Table 2: Results using numerical method (A), Hopfield neural network I (B), and Hopfield neural network II (C).

U	2400 MW								2500 MW					
	A		B		C				A		B		C	
					U0=100		U0=108						U0=100	
	F	GEN.	F	GEN.	F	GEN	F	GEN.	F	GEN.	F	GEN.	F	GEN.
1	1	193.2	1	192.7	1	189.1	1	192.5	2	206.6	2	206.1	2	206.0
2	1	204.1	1	203.8	1	202.0	1	203.9	1	206.5	1	206.3	1	206.3
3	1	259.1	1	259.1	1	254.0	1	258.9	1	265.9	1	265.7	1	365.7
4	3	234.3	2	195.1	3	233.0	3	234.2	3	236.0	3	235.7	3	235.9
5	1	249.0	1	248.7	1	241.7	1	248.9	1	258.2	1	258.2	1	257.9
6	1	195.5	3	234.2	1	233.0	1	195.1	3	236.0	3	235.9	3	235.9
7	1	260.1	1	260.3	1	254.1	1	260.7	1	269.0	1	269.1	1	269.6
8	3	234.3	3	234.2	3	232.9	3	234.1	3	236.0	3	235.9	3	235.9
9	1	325.3	1	324.7	1	320.0	1	324.8	1	331.6	1	331.2	1	331.4
10	1	246.3	1	246.8	1	240.3	1	247.0	1	255.2	1	255.7	1	255.4
	P	2401.2	P	2399.8	P	2400.0	P	2400.0	P	2501.1	P	2499.8	P	2400.0
	C	488.5	C	487.87	C	481.70	C	488.0	C	526.70	C	526.13	C	526.23
	itr	NA	itr	NA	itr	139,130	itr	143,382	itr	NA	itr	NA	itr	181,824

U	2600 MW						2700 MW					
	A		B		C		A		B		C	
					U0=100						U0=100	
	F	GEN.	F	GEN.	F	GEN.	F	GEN.	F	GEN.	F	GEN.
1	2	216.4	2	215.3	2	215.8	2	218.4	2	224.5	2	225.7
2	1	210.9	1	210.6	1	210.7	1	211.8	1	215.0	1	215.2
3	1	278.5	3	278.9	3	279.1	1	281.0	3	291.8	3	291.8
4	3	239.1	3	238.9	3	239.1	3	239.7	3	242.2	3	242.3
5	1	275.4	1	275.7	1	276.3	1	279.0	1	293.3	1	293.7
6	3	239.1	3	239.1	3	239.1	3	239.7	3	242.2	3	242.3
7	1	285.6	1	286.2	1	286.0	1	289.0	1	303.1	1	302.8
8	3	239.1	3	239.1	3	239.1	3	239.7	3	242.2	3	242.3
9	3	343.3	1	343.5	1	342.8	3	429.2	1	335.7	1	355.1
10	1	271.9	1	272.6	1	271.9	1	275.2	1	289.5	1	288.8
	PT	2599.3	PT	2599.8	PT	2600.0	PT	2702.2	PT	2699.7	PT	2700.0
	C	574.03	C	574.26	C	574.37	C	625.18	C	626.12	C	626.24
	itr	NA	itr	NA	itr	151,428	itr	NA	itr	NA	itr	173,742

For the total load of 2400 MW, the final values (C) compared with those of numerical method (A) are very close when the gain parameter U_0 is 108. However, this is not the case for the earlier Hopfield network (B) where the output of unit 4 and unit 6 are interchanged. Also note that since the parameters of unit 4 and 8 are identical, the output power should also be the same, which is not the case for the earlier network, Hopfield network I. When the gain parameter is set to be 100, the output power for 2400 MW load is different from those of the numerical method and Hopfield network I. However, the cost is lower and the constraints are met. Thus, the gain parameter has to be set to a suitable value and the new Network, Network II, gives reasonable and better results. In other load demand, the gain parameter is set to be 100, and the output power is very close to those of other methods. A slightly higher cost for the numerical method is due to the numerical resolution in the earlier simulation reported in [5].

B. Slope Adjustment with Fixed and Adaptive Learning Rates

For slope adjustment method, Table 3, the number of iterations is reduced to about one half of that of the conventional Hopfield network, Table 2. Oscillation is also drastically reduced from about 40,000 to less than 100 iterations, Fig 2. For the slope adjustment method, the final results of the adaptive learning rate are close to those of the fixed learning rate. A fixed learning rate gives non-smooth response when the learning rate is high, and gives an incorrect answer for 2400 MW load. However for the adaptive learning rate, the response is much smoother, and at the same time the cost is also reduced, Fig 3. Compared with the conventional Hopfield neural network, the degree of freedom of the system increases from 1, which is U_0, to 2, which are the initial conditions of U_0 and η.

C. Bias Adjustment Method with Fixed and Adaptive Learning Rates

For the bias adjustment method, Table 4, the number of iteration is also reduced to about one half of that of the conventional Hopfield network. Oscillation is also drastically reduced from about 40,000 to less than 100 iterations as seen in Fig 2. For the adaptive learning rate, the number of iterations is reduced and the final results of the adaptive learning rate are better than those of the fixed learning rate, Fig 3. The stable point after the transient period can be controlled by selecting initial values of U_0 and η; then the number of iterations will be lower.

Table 3: Results for the slope adjustment method with fixed learning rate, $\eta = 1.0$ (A) and adaptive learning rate (B).

U	2400 MW		2500 MW		2600 MW		2700 MW	
	A	B	A	B	A	B	A	B
1	196.8	189.9	205.6	205.1	215.7	214.5	223.2	224.6
2	202.7	202.9	206.7	206.5	211.1	211.4	216.1	215.7
3	251.2	252.1	265.3	266.4	278.9	278.8	292.5	291.9
4	232.5	232.9	236.0	235.8	239.2	239.3	242.6	242.6
5	240.4	241.7	257.9	256.8	276.1	276.1	294.1	293.6
6	232.5	232.9	236.0	235.9	239.2	239.1	242.4	242.5
7	252.2	253.4	269.5	269.3	286.0	286.7	303.5	303.0
8	232.5	232.9	236.0	235.8	239.2	239.3	242.7	242.6
9	320.2	321.0	331.8	334.0	343.4	343.6	355.8	355.7
10	238.9	240.4	255.5	254.4	271.2	271.2	287.3	287.8
Total P	2400.0	2400.0	2500.0	2500.0	2600.0	2600.0	2700.0	2700.0
Cost	481.83	481.71	526.23	526.23	574.36	574.37	626.27	626.24
Iters	99,992	84,791	80,156	86,061	72,993	79,495	99,948	99,811
U0	95.0	110.0	120.0	100.0	130.0	120.0	160.0	120.0
n	1.5	1.0E-04	1.0	1.0E-04	1.0	1.0E-04	1.0	1.0E-04

Table 4: Results for the bias adjustment method with fixed learning rate, $\eta = 1.0$ (A) and adaptive learning rate (B).

U	2400 MW		2500 MW		2600 MW		2700 MW	
	A	B	A	B	A	B	A	B
1	197.6	189.4	208.3	206.7	212.4	217.9	221.4	228.8
2	201.6	201.8	206.2	205.8	209.6	210.5	213.8	214.1
3	252.3	253.5	265.2	265.6	280.0	278.8	293.3	292.0
4	232.7	232.9	235.9	235.8	238.8	239.0	242.1	242.2
5	239.9	242.1	257.1	258.2	277.9	275.8	295.4	293.6
6	232.7	232.9	235.9	235.8	238.6	239.0	242.0	242.1
7	251.5	253.8	268.3	269.4	288.1	285.5	305.3	302.6
8	232.7	232.9	235.8	235.8	238.8	239.0	242.1	242.1
9	318.8	319.3	330.9	330.1	341.9	342.1	345.2	352.3
10	240.3	241.6	256.4	256.9	274.0	272.3	290.4	290.1
Total P	2400.0	2400.0	2500.0	2500.0	2600.0	2600.0	2700.0	2700.0
Cost	481.83	481.72	526.24	526.23	574.43	574.37	626.32	626.27
Iters	99,960	99,904	99,987	88,776	99,981	99,337	99,972	73,250
U0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
theta	0.0	50.0	0.0	50.0	0.0	50.0	0.0	100.0
n	1.0	1.0	1.0	5.0	1.0	5.0	1.0	5.0

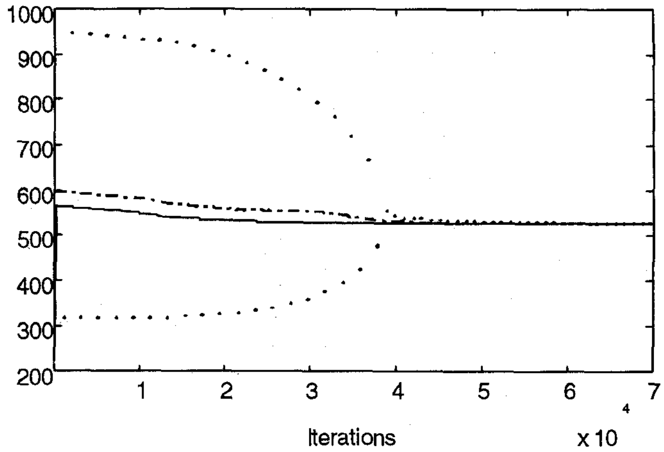

Fig. 2. Cost of the using Hopfield network (dotted line), slope adjustment method with fixed learning rate (dashed line), and bias adjustment method with fixed learning rate (solid line).

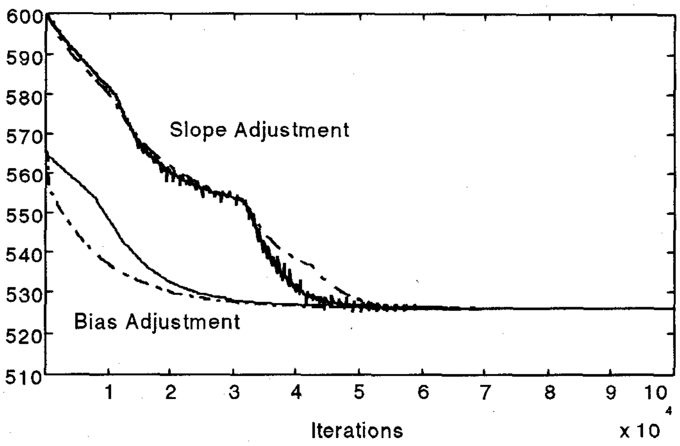

Fig. 3: Cost for fixed learning rate (solid line) and adaptive learning rate (dashed-dotted line).

D. Momentum

Momentum is applied to the input of each neuron to speed up the convergence of the system. When the momentum is applied to the system, the number of iterations is drastically reduced. In Table 5, the momentum factor of 0.9 is applied and the number of iterations is reduced to about 10 percent of those of the conventional Hopfield neural network, while lower momentum factors give slower responses. When the momentum with the same momentum factor is applied to the slope adjustment method with adaptive learning rates, the number of iterations is reduced further to be about 60 percent of that of the Hopfield neural network with the input momentum. The number of iterations, as seen in Table 6, can be reduced to about one third which is about 3 percent of the Hopfield network without momentum when the momentum factor for the gain parameter is 0.97.

Table 5. Result with momentum, α=0.9, applied to input for conventional Hopfield network (A), and adaptive slope adjustment method (B).

U	2400 MW		2500 MW		2600 MW		2700 MW	
	A	B	A	B	A	B	A	B
1	207.7	191.2	205.6	204.5	215.3	214.5	225.4	223.7
2	206.9	201.5	206.2	206.8	210.7	211.1	215.2	215.4
3	267.0	253.9	265.6	264.8	279.1	278.7	292.0	291.8
4	199.5	232.7	235.8	236.2	239.1	239.4	242.3	242.7
5	259.6	242.0	258.4	258.5	276.4	276.1	293.8	293.7
6	199.5	232.7	235.7	235.2	239.1	239.3	242.3	242.7
7	270.9	253.9	269.7	269.8	286.1	286.4	302.8	303.3
8	199.5	232.7	235.8	236.2	239.1	239.4	242.4	242.7
9	332.8	319.2	331.1	331.7	343.0	343.4	355.2	355.7
10	256.6	240.1	256.0	256.5	272.1	271.7	288.7	288.2
Total P	2400.0	2399.9	2500.0	2500.0	2600.0	2600.0	2700.0	2699.9
Cost	501.81	481.7	526.23	526.23	574.37	574.37	626.24	626.24
Iters	20,115	21,187	13,666	8,954	12,466	8,741	14,623	9,716
U0	100.0	95.0	100.0	100.0	100.0	100.0	100.0	100.0
n	NA	1.0E-05	NA	1.0E-04	NA	1.0E-04	NA	1.0E-04

Table 6: Result for numerical method (A), adaptive slope adjustment method with momentum 0.9 applied to input and 0.97 (0.9 for 2400 MW case) applied to gain parameter (B) and adaptive bias adjustment method with momentum 0.9 applied to input (C).

U	2400 MW			2500 MW		
	Numercal Method	Slope	Bias	Numerical Method	Slope	Bias
1	193.2	191.5	189.0	206.6	205.7	206.7
2	204.1	203.0	201.7	206.5	207.5	205.8
3	259.1	254.2	253.5	265.9	264.7	265.6
4	234.3	232.5	232.8	236.0	235.7	235.8
5	249.0	240.9	242.2	258.2	257.5	258.2
6	195.5	232.4	232.8	236.0	235.8	235.8
7	260.1	253.2	253.9	269.0	269.9	269.4
8	234.3	232.5	232.8	236.0	235.7	235.8
9	325.3	321.8	319.1	331.6	332.5	330.1
10	246.3	237.9	242.0	255.2	255.1	256.9
Total P	2401.2	2399.9	2400.0	2501.1	2500.0	2500.0
Cost	488.5	481.7	481.72	526.70	526.23	526.23
Iters	NA	15,148	8,707	NA	4,474	8,931
U0	NA	95.0	100.0	NA	100.0	100.0
theta	NA	NA	100.0	NA	NA	50.0
n	NA	1.00E-05	1.0	NA	1.00E-04	0.5

U	2600 MW			2700 MW		
	Numercal Method	Slope	Bias	Numerical Method	Slope	Bias
1	216.4	215.1	218.1	218.4	218.6	228.2
2	210.9	211.7	210.4	211.8	211.6	214.8
3	278.5	279.1	278.8	281.0	281.2	291.7
4	239.1	239.1	239.1	239.7	239.6	242.3
5	275.4	276.1	275.8	279.0	278.9	293.3
6	239.1	239.0	239.1	239.7	239.6	242.2
7	285.6	286.6	285.4	289.0	288.4	302.3
8	239.1	239.1	239.1	239.7	239.6	242.3
9	343.3	343.6	341.9	429.2	427.9	354.2
10	271.9	270.6	272.4	275.2	274.3	288.9
Total P	2599.3	2600.0	2600.0	2702.2	2699.9	2700.0
Cost	574.03	574.37	574.37	625.18	623.78	626.24
Iters	NA	5,224	9,303	NA	61,309	9,857
U0	NA	100.0	100.0	NA	100.0	100.0
theta	NA	NA	50.0	NA	NA	100.0
n	NA	1.00E-04	5.0	NA	1.00E-04	5.0

For the bias adjustment method, the effect of momentum is similar to the case of slope adjustment method. However, this method gives better responses during transient and at the beginning of the process. Due to the slow convergence near the stable state, the system converges to the stable state

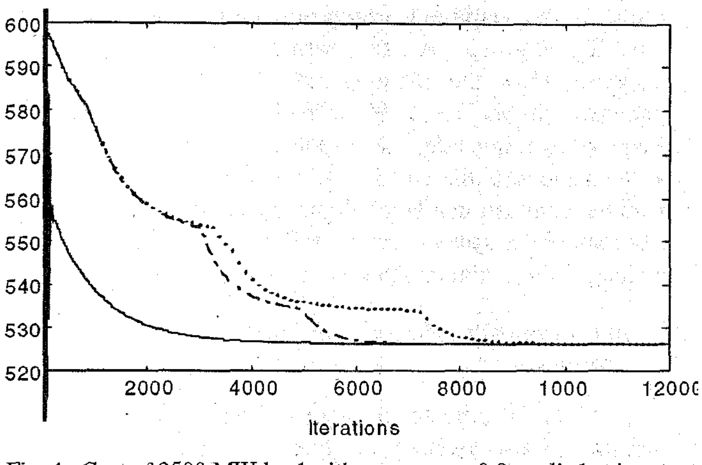

Fig. 4: Cost of 2500 MW load with momentum 0.9 applied at input for conventional Hopfield network (dotted line), the slope adjustment method (dashed-dotted line), and the bias adjustment method (solid line).

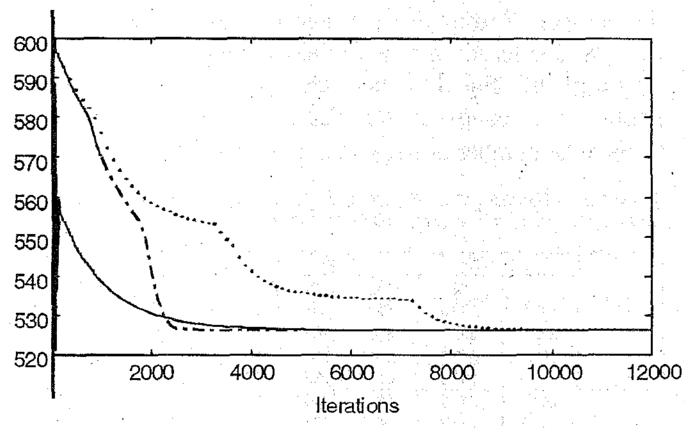

Fig. 5: Cost of 2500 MW load with momentum 0.9 applied at input for Hopfield network (dotted line), the slope adjustment method with momentum 0.97 applied at gain parameter (dashed-dotted line), and the bias adjustment method (solid line).

at the same iterations as that of the slope adjustment method.

Sensitivity analysis was also performed by forcing one or more units to hit an upper or lower limits. Since the power limit constraints have already been incorporated in the sigmoidal function (13), units can not exceed but approach the limits. For example, for 3000 MW load, Unit 3 has 498.87 MW which is close to its upper limit, 500 MW. For 1900 MW load, Units 5, 7 and 10 have 190.84, 201.17 and 200.171 MW, respectively; which are all very close to but within the respective lower limits. The simulation time of the numerical method with VAX 11/780 was a little bit more than 1 sec. The conventional Hopfield network without any adjustment took about 28 sec. in the Compaq 90 MHz Pentium PC with Window 95 (Table 2). On the other hand, the slope adjustment and the bias adjsutment methods with momentum (Fig. 5) took about 2 and 4 sec., respectively. Considering the use of a personal computer rather than a main frame, there is a great potential for the proposed methods; especially, when implemented in hardware.

VI. CONCLUSIONS

This paper presents a unified adaptive learning approach in the Hopfield neural network using the slope adjustment and bias adjustment methods for application to economic load dispatch. The methods reduced the transient period drastically. The adaptive learning rate in both methods gives better response compared to the fixed learning rate. The bias adjustment method gives good response especially in the beginning of the process. Both methods reduced the number of iterations to one half of that of the traditional Hopfield neural network. When the momentum is introduced to all methods in either input or gain, the number of iterations and the computation time are reduced in the order of magnitudes. This promises a great potential of the proposed method for real-time economic load dispatch.

VII. ACKNOWLEDGMENTS

This work is supported in part by the Electricity Generating Authority of Thailand, the Allegheny Power Company, and the NSF under grants "Research and Curriculum Development for Power Plant Intelligent Distributed Control" (EID-9212132) and "U.S.-Korea Cooperative Research on Intelligent Distributed Control of Power Plants and Power Systems" (INT-9223030).

VIII. REFERENCES

[1] C. E. Lin, and G. L. Viviani, "Hierachical economic dispatch for piecewise quadratic cost functions", *IEEE Transactions on Power Apparatus and Systems.*, Vol. PAS-10, No. 6., June 1984.

[2] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities", *Proceedings of National Academy of Science, USA.*, Vol. 79, pp. 2554-2558, April 1982.

[3] J. J. Hopfield, "Neurons with graded response have collective computational properties like those of two-state neurons", *Proceedings of National Academy of Science, USA.*, Vol. 81, pp. 3088-3092, May 1984.

[4] W. D. Tank and J. J. Hopfield, "Simple neural optimization networks: an A/D converter, signal decision circuit, and a linear programming circuit", *IEEE Transactions on Systems, Mans, and Cybernetics*, Vol. CAS-33, No. 5, pp. 553-541, January/February 1992.

[5] J. H. Park, Y. S. Kim, I. K. Eom, and K. Y. Lee, "Economic load dispatch for piecewise quadratic cost function using hopfield neural networks," *IEEE Transactions on Power system Apparatus and Systems*, Vol. 8, No. 3, pp. 1030-1038, August 1993.

[6] T. D. King, M. E. El-Hawary, and F. El-Hawary, "Optimal environmental dispatching of electric power system via an improved hopfield neural network model", *IEEE Transactions on Power Systems*, Vol. 1, No. 3, pp. 1559-1565, August 1995.

[7] C. C. Ku and K. Y. Lee, "Diagonal recurrent neural networks for dynamic systems control," *IEEE, Transactions on Neural Networks*", Vol. 6, No. 1, pp. 144-156, January 1995.

[8] A. Sode-Yome, *Development of Adaptive Hopfield Neural Networks with Applications to Economic Load Dispatch,* M.S. Thesis, The Pennsylvania State University, University Park, PA, August 1996.

[9] S. Y. Kung, *Digital Neural Networks,* Prentice Hall, Englewood Cliffs, NJ, 1993.

[10] J. J. Hopfield and W. D. Tank, "Neural computation of decisions in optimization problems", *Biological Cybernetics*, Vol. 52, pp 141-152, 1985.

[11] J. A. Freeman and D. M. Skapura, *Neural Networks Algorithms, Applications, and Programming Techniques*, Addison-Wesley Pub. Co., Inc., July 1992.

Kwang Y. Lee was born in Pusan, Korea, March 6, 1942. He received the B.S. degree in Electrical Engineering from Seoul National University, Seoul, Korea, in 1964, the M.S. degree in Electrical Engineering from North Dakota State University, Fargo, ND, in 1967, and the Ph.D. degree in System Science from Michigan State University, East Lansing, in 1971. He is currently a professor of Electrical Engineering at the Pennsylvania State University, University Park, PA. His current research interests include control theory, intelligence systems and their applications to power systems.

Dr. Lee is currently the Director of Power Systems Control Laboratory at Penn State. He is a senior member of IEEE.

Arthit Sode-Yome was born in Bangkok, Thailand, August 30, 1970. He received the B.Eng. degree in Electrical Engineering from Prince of Songkla University, Songkla, Thailand, in 1993. He worked with the Electricity Generating Authority of Thailand from 1993-1994. He received the M.S. degree in Electrical Engineering from the Pennsylvania State University, University Park, in 1996. His research interests are in neural networks, power and control system, and intelligent control.

June Ho Park was born in Masan, Korea, September 17, 1955. He received the B.S., M.S. and Ph.D. degrees in Electrical Engineering from Seoul National University, Seoul, Korea, in 1978, 1980, and 1987, respectively. From 1980 to 1981, he was a researcher at Korea Electrotechnology Research Institute. He has been on the faculties of Chung-Nam National University from 1981-1984 and Pusan National University since 1984. Dr. Park has been a member of IEEE.

Discussion

L L Lai & A G Sichanie (Energy Systems Group, City University, London, EC1V 0HB, UK) :

The authors have presented an interesting paper on application of Hopfield neural networks in economic load dispatch.

The discussers have the following comments.

In 1982, John J Hopfield, working at the California Institute of Technology and at AT&T Bell Laboratories, conceptualised a model conforming to the asynchronous nature of biological neurons. It was a more abstract, fully interconnected, random and asynchronous network. In general, the Hopfield network is an auto associative fully connected network of a single layer of nodes. The network takes two-valued inputs, namely, binary or bipolar.

Hopfield described his model in terms of an energy function. The neural network must be able to escape local minima and settle at the global minimum, that is, produce true results. Has the introduction of the slope and bias adjustment methods will also made sure that a global minimum could be achieved?

Manuscript received February 28, 1997.

K. Y. Lee, A. Sode-Yome, and J. H. Park (Department of Electrical Engineering, The Pennsylvania State University, University Park, PA 16802, U.S.A.) :

The authors appreciate the interest of the discussers and their comments.

Hopfield networks can have many local minimum points and they depend on the numerical values of parameters that define the energy function. Once an energy function is selected for economic load dispatch problem, the usual Hopfield network may converge to a local minimum. As pointed out in the paper, unlike the asynchronous network, the synchronous network adopted in the paper has fixed points as well as limit cycles as attractors. However, it does not get trapped to local minima as easily as the asynchronous model, and has additional advantages in computation and hardware savings.

In order to escape from a local mimimum, one might apply the concept of *simulated annealing,* i.e., if the lowering of the temperature is done sufficiently slow, the solid can reach thermal equilibrium at each temperature [1,2]. In fact, a close look at the sigmoid function (Fig. 1) reveals that the gain parameter is equivalent to the temperature or control parameter in the simulated annealing algorithm. By adjusting this control parameter judiciously one might be able to achieve the global minimum, which is the spirit of the adaptive Hopfield network proposed. This behavior is observed in the simulation study for low load demand (2400 MW). Table 2 shows two solutions for Case C in the 2400 MW column for two different values of gains, 100 and 108. The gain of 100 gives the global solution while the gain of 108 gives a local solution similar to Cases A and B. This same global minimum is found in the adjustment methods, Tables 3 through 6.

[1.] E. Aarts and J. Korst, *Simulated Annealing and Boltzmann Machines*, John Wiley & Sons, New York, NY, 1989.

[2.] J. Hertz, A. Krogh, and R. G. Palmer, *Introduction to the Theory of Neural Computation*, Addison-Wesley, Menlo Park, CA, 1991.