Identification and Control of Induction Motor Stator Currents Using Fast On-Line Random Training of a Neural Network

Bruce Burton, *Associate Member, IEEE,* Farrukh Kamran, *Member, IEEE,* Ronald G. Harley, *Fellow, IEEE,* Thomas G. Habetler, *Senior Member, IEEE,* Martin A. Brooke, *Member, IEEE,* and Ravi Poddar

***Abstract*—Artificial neural networks (ANN's), which have no off-line pretraining, can be trained continually on-line to identify an inverter-fed induction motor and control its stator currents. Due to the small time constants of the motor circuits, the time to complete one training cycle has to be extremely small. This paper proposes and evaluates a new form of the random weight change (RWC) algorithm, which is based on the method of random search for the error surface gradient. Simulation results show that the new form of the RWC, termed continually on-line trained RWC (COT-RWC), yields performance very much the same as conventional backpropagation with on-line training. Unlike backpropagation, however, the COT-RWC method can be implemented in mixed digital/analog hardware and still have a sufficiently small training cycle time. The paper also proposes a VLSI implementation which completes one training cycle in as little as 8 μs. Such a fast ANN can identify and control the motor currents within a few milliseconds and, thus, provide self-tuning of the drive while the ANN has no prior information whatsoever of the connected inverter and motor.**

***Index Terms*—Induction motor, motor current regulator, neural network, on-line training.**

I. Introduction

THE induction motor is a nonlinear system, the parameters of which vary with time and operating conditions. For high-performance applications, such as vector control and direct self control, it is necessary for the controller design to be based on observers and estimation techniques [1], which depend on a simplified model of the motor. Artificial neural networks (ANN's) provide an alternative method of observing the input/output relationships of the motor. A previously presented scheme [2] proposed that an ANN be trained *off-line*, i.e., with data obtained *a priori* to mimic existing stator current controllers. Once sufficiently well-trained, the ANN could replace the original current controller with the advantage of increased speed of execution and fault tolerance. With this approach, no further training of the network is possible, after the drive is commissioned. Therefore, the performance of such an off-line trained ANN approach depends upon the amount and quality of training data used, which in turn depends on system complexity and the range of operating conditions involved and is also sensitive to parameter variations.

Since the induction machine is a deterministic system for which the equations are well known, training of an ANN from a random initial condition is not necessarily required. Reference [3] proposes a current regulator for induction machines which maps the electrical equivalent circuit equations onto a feedforward neural network and does not require training. However, like the off-line trained ANN scheme [2], the approach in [3] is also prone to degraded performance because of parameter variations. In order to account for unknown parameter variations, an observer-based scheme like [1] may be used, but unmodeled nonlinearities, such as magnetic saturation, can only be accounted for using an adaptive nonlinear-model-based controller, or by using an ANN which is trained while the drive controller (including the ANN) is operating *on-line*. Such an ANN scheme was proposed [4] which used continual on-line training (with no off-line training) to identify and adaptively control the currents and, therefore, the torque and, thus, the speed of an induction machine. Simulated results showed that this scheme could produce high dynamic performance similar to that achieved with conventional vector control. With an on-line trained scheme, custom tailoring of the ANN architecture and weights to match the structure of the motor equations, although possible, is not necessary.

The scheme in [4] incorporates three ANN's which are on-line trained using backpropagation, with two different rates of execution; a relatively slow rate for the two ANN's which accomplishes the rotor speed identification and control functions and a much faster rate for the ANN performing the stator current regulation. The stator current loop control scheme, illustrated in Fig. 1, must run at a high enough rate to cope with the comparatively small electrical time constants of the machine. In order to achieve on-line training at a reasonable sampling frequency, e.g., 10 kHz, one epoch (one ANN weights update cycle based on the training error) needs to be completed in approximately 50 μs. Due to the lack of a suitable ANN application specific integrated circuit (ASIC), the adaptive on-line trained current controller ANN of [4] was implemented in software [5], [6] on a transputer, but

Paper IPCSD 96–53, approved by the Industrial Drives Committee of the IEEE Industry Applications Society for presentation at the 1995 Industry Applications Society Annual Meeting, Lake Buena Vista, FL, October 8–12. Manuscript released for publication October 7, 1996.

B. Burton and R. G. Harley are with the Department of Electrical Engineering, University of Natal, Durban, 4001 South Africa.

F. Kamran, T. G. Habetler, M. Brooke, and R. Poddar are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: thabetler@ee.gatech.edu).

Publisher Item Identifier S 0093-9994(97)02940-X.

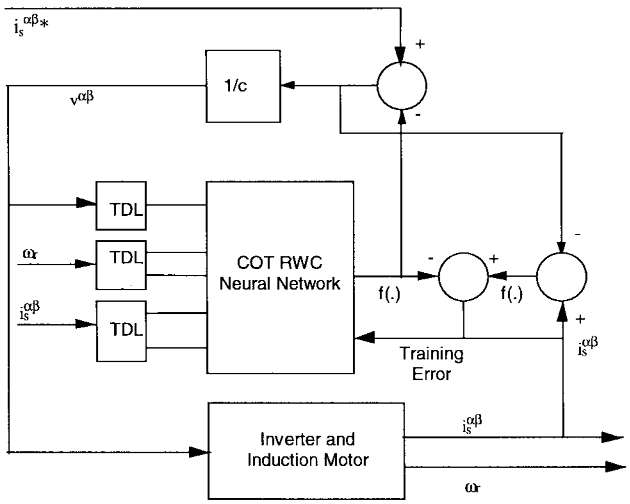


Fig. 1. Adaptive ANN stator current controller block diagram.

the sampling frequency was limited by the ANN computation overhead to approximately 500 Hz.

This paper proposes and investigates the use of a new fast on-line training algorithm for feedforward ANN's suitable for hardware implementation and which can meet the timing constraints described above. As opposed to backpropagation, this new method called *continually on-line trained random weight change* (COT-RWC) training, is designed to be robust and insensitive to the nonidealities of analog VLSI circuits.

II. ANN Current Control Method

The stator current controller of [4] is based on continual online training (COT) of an ANN to adaptively identify the nonlinear autoregressive moving average with exogenous (NARMAX) inputs model of the electrical dynamics of the induction motor. This model, derived in [4], gives the value of the next stator current sample $i_s(k+1)$ as some function $f(.)$ of the present "state" of the machine $\underline{x}(k)$ and the present voltage $\underline{v}_s(k)$ applied to the stator, i.e.,

$$\underline{i}_s(k+1) = \underline{f}[x(k)] + c\underline{v}_s(k) \tag{1}$$

where c is called the voltage constant and the state vector, $\underline{x}(k)$, is a vector made up of present and delayed values of stator current $\underline{i}_s$ and shaft speed, ω_r, and a delayed value of v_s, i.e.,

$$x(k) = [\underline{i}_s(k), \underline{i}_s(k-1), \omega_r(k), \omega_r(k-1), \underline{v}_s(k-1)]. \tag{2}$$

From (1), it can be seen that there exists some voltage $\underline{v}_s^*(k)$ which can be applied to the stator of the motor to produce some desired value of stator current $i_s^*(k+1)$, i.e.,

$$\underline{i}_s^*(k+1) = f[x(k)] + c\underline{v}_s^*(k). \tag{3}$$

Equation (3) can, thus, be rearranged to produce a nonlinear control law:

$$\underline{v}_s^*(k) = \frac{\underline{i}_s^*(k+1) - f[x(k)]}{k}. \tag{4}$$

Thus, the exact voltage required to produce the desired current can be calculated if the value of $\underline{f}[x(k)]$ and k are known. However, k and $\underline{f}(.)$ depend on the electrical parameters of the motor. Since these parameters are time-varying, both k and $\underline{f}(.)$ are implicitly time-varying [the time variation is not explicit in the values of k and $\underline{f}(.)$]. Nevertheless, Fig. 1 shows how a COT ANN can be used to adaptively identify $\underline{f}(.)$ and k and produce approximate values $g(.)$ and k', respectively, so that control (4) can be implemented as

$$\underline{v}_s^*(k) = \frac{\underline{i}_s^*(k+1) - g[\underline{x}(k)]}{k'}. \tag{4}$$

III. Commercially Available ANN Hardware

Recently, a zero instruction set computer chip, ZISC036, was introduced by IBM that uses radial basis functions (RBF's) as the training method [7]; a similar, but more powerful chip using RBF for training, has also been developed by Intel and Nestor Inc. [7]. The RBF neural networks are not as powerful as the feedforward ones, when compared in terms of extrapolation and generalization capabilities. Another possibility explored by some researchers is to combine a feedforward ANN ASIC like the Intel Electrically Trainable Analog Neural Network (ETANN) with external high-speed processors to implement the backpropagation [8]. The forward pass is carried out in the ETANN and the weight update computation is done on the external processor; although this method can achieve considerably higher speeds than a software implementation of the ANN, the achieved speed is still too slow for the continual on-line training requirement of the motor application considered in this paper.

A complete VLSI implementation of the feedforward neural net using backpropagation or one of its variants for continual on-line training has not been accomplished to date. The major obstacle in this regard is the sensitivity of these gradient descent training algorithms to the nonlinearities and offsets present in hardware analog multipliers and adders. In contrast, all-digital implementation takes up much larger chip areas than the analog ones and, therefore, a fully *parallel* digital implementation can be realized, but only for small networks and with lesser number of bits. However, backpropagation is sensitive to the bit resolution and fails to converge if the resolution is inadequate [9]. If, instead, the computation is carried out *serially* in digital circuits, learning speed must be sacrificed. Analog implementation of weight circuits and multipliers, on the other hand, has the advantage of fast operating speed and small chip areas, therefore allowing larger circuits to be realized. However, the nonidealities of these analog circuits, as mentioned above, render the use of backpropagation or its modified versions impractical.

All the above-mentioned problems prompted the search, not for faster hardware to implement continuous on-line training using backpropagation, but, instead, for an alternate training algorithm which is more robust and less complex and, therefore, would take much less time to execute on VLSI. Such an algorithm is described in the next section.

IV. The RWC Training Algorithm

A practical variation of the previously reported ANN training algorithm called RWC [9] is proposed in this paper;

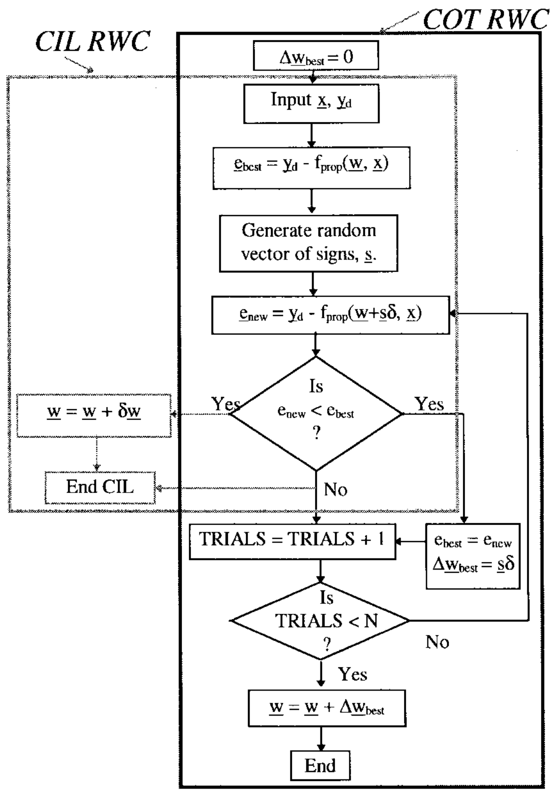


Fig. 2. flowchart depicting one training epoch of the continuous on-line trained random weight change (COT-RWC) training algorithm.

RWC updates the ANN weights based on a random search for the gradient of the error surface, instead of calculating this gradient with the complex backpropagation algorithm. For applications with hundreds of weights and weight update times in the microsecond range, the RWC algorithm can be implemented on hardware which is lower in cost and simpler than commercially available hardware, since compact analog mixed signal circuitry can be used to perform the weight updates and the forward propagation of the network. It is important to note that with RWC, convergence does not take place along steepest descent, however, convergence will still take place since *there is always a good probability that a relatively small number of random trials will find a direction in which the weights can be changed in order to reduce the error*. This paper will show that this type of convergence is sufficiently fast to ensure good adaptive control of induction motor currents.

The original RWC algorithm was designed for chip-in-the-loop (CIL) training, i.e., training on real-time data where a sample can be discarded in the event that the gradient is not found; CIL-RWC is suitable for applications requiring very fast off-line training on real-time data, such as speech and image processing. However, CIL-RWC is not suitable for continual on-line training of adaptive real-time ANN controllers, since the gradient must be found in every sampling period in order to eliminate the control error. In power electronic applications, this amounts to changing the weights in the correct direction once in every switching period to ensure stability. Thus, CIL-RWC must be significantly modified to

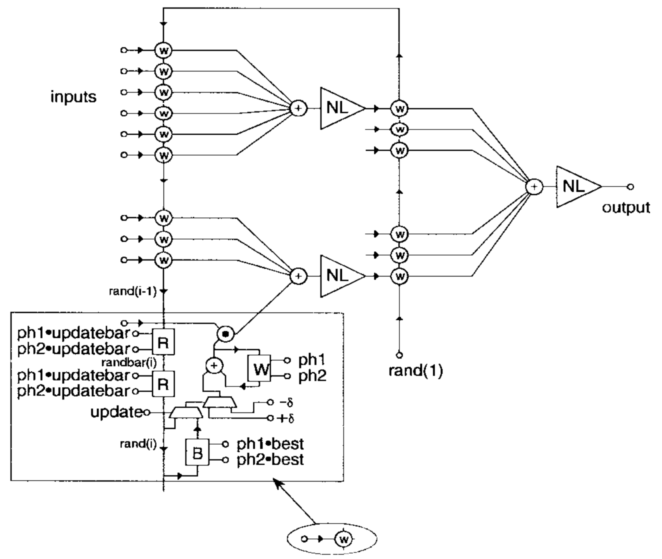


Fig. 3. The RWC learning hardware schematic.

obtain a form that is suitable for COT. Fig. 2 illustrates how CIL-RWC is modified to obtain COT-RWC.

While the practical form of the RWC algorithm works well for the induction motor application, it is generic in nature and is potentially useful for many other applications. During each training cycle, or *epoch*, each of the network weights is perturbed by a number which has a fixed magnitude δ and a random sign. The ANN output error is computed after the weight change. This error is compared to the value of the previous error before the weight change and, based on this comparison, a decision is taken whether to keep the new weights or not. Keeping the ANN input vector fixed, this process is repeated a number of times (i.e., *trials*) during each epoch, and the final weights at the end of the epoch are chosen to be the ones that result in the smallest error during that epoch. The flowchart in Fig. 2 explains one training cycle or epoch in more detail. The step size δ is a training parameter that needs to be determined heuristically for a specific problem. This is very similar to the gain coefficient β in backpropagation and is best thought of as the radius of a hypersphere in the n-dimensional error hypersurface, where n is the total number of network weights. It has been observed during experimentation that the value of δ affects the convergence speed and accuracy and needs to be small, about two orders of magnitude less than the weight magnitudes. As is clear from Fig. 2, each epoch contains N trials, therefore, the forward propagation and the random weight change has to be done N times during each epoch. This appears to be a large amount of computation, but, because of the fact that this scheme can be implemented with fully parallel nodes, and the random numbers can be generated very efficiently using shift registers, the all-hardware implementation can achieve very high speeds.

V. RWC Hardware

A proposed hardware schematic for the RWC algorithm-learning ANN is presented in Fig. 3. The hardware is controlled by a conventional microcontroller which generates

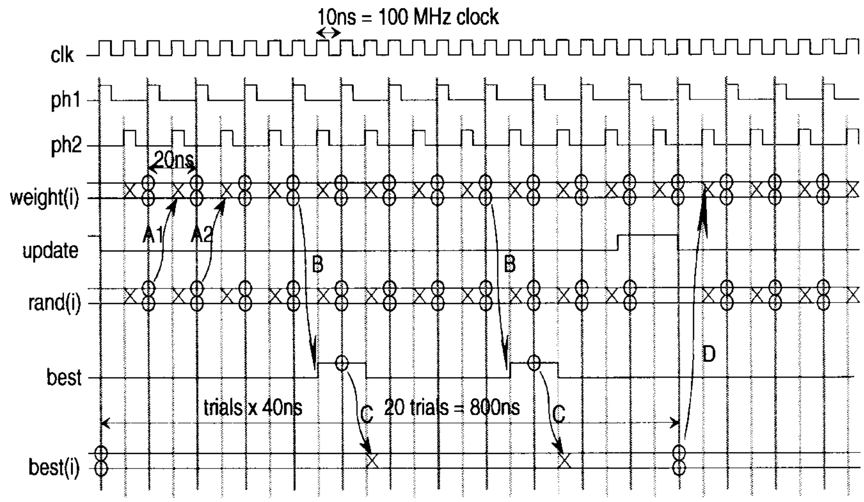


Fig. 4. Timing diagram for COT-RWC learning hardware.

the clock signals (ph1 and ph2) and several control signals [update, best, rand(i), δ]. These control signals can be generated with a low-cost single-chip microcontroller. The parallel mixed-signal analog-digital hardware carries out all the parallel weight update and forward-propagation operations. It is anticipated that with a 100-MHz digital clock, the training epoch proposed in Fig. 3 would take 800 ns (20 RWC trials). With a less aggressive 10-MHz clock, the epoch would take 8 ms.

Each weight circuit in the ANN contains three registers (R, W, and B in Fig. 3) that store the random weight changes, the current weight, and the best weight change for the current epoch, respectively. The random weight changes are shifted into the register R and used to update the weight register W. Then, the opposite weight change is shifted into R and used to return W to the original value. The timing of this process is shown in Fig. 4. The step A1 represents the trial of the current weight change, while A2 represents the return to the original value of the weights. The external signal "best" is generated externally from the hardware by calculating the output error and indicates to the hardware that the current weight change is the best so far among the completed trials in the current epoch. This signal causes the current weight change to be saved in the B (best) register. When one epoch is complete, the update signal is raised and the best weight change is permanently saved in the weight register (W). Referring to Fig. 4, in step B, the external processor computes that the current weight change is the best for the epoch. In step C, the value of the register B [best(i)] is updated. In step D, the best overall weight change is made permanent.

VI. Results

In order to assess the suitability of the RWC algorithm to identify and control the motor stator currents, the response of the system in Fig. 1 is computed for several step changes in the magnitude and the frequency of the demanded stator currents; typical results from several case studies are presented. The ANN in Fig. 1 consists of eight inputs and two outputs; the number of middle layer neurons m is varied from one case study to the next. The middle layer outputs pass through sigmoidal nonlinearities, while the final outputs are linear. The equations for the system in Fig. 1 were given in [5], [6] and are not presented here. During each epoch, the process in Fig. 3 is repeated for No_of_Trials $N = 20$, while keeping the inputs constant.

A. Case Study One: RWC Compared with Backpropagation

In order to compare the convergence properties of the RWC and the backpropagation algorithms, the simulation of the response of the system in Fig. 1 is repeated for both algorithms. Fig. 5 shows the desired current and the actual current, first using RWC [Fig. 5(a)] and then using backpropagation [Fig. 5(b)]. No pretraining of the ANN takes place, and in each of Fig. 5(a) and (b), the ANN is allowed to identify and control the current from a random initial set of weights. From $t = 0$, initial conditions prevail, but the desired current magnitude and frequency are both set to zero, and the ANN lies dormant with initial weights. On-line training, current identification, and control then all commence at $t = 100$ ms when the first of three step changes in current is demanded.

Step 1:
$$100 \text{ ms} < t < 600 \text{ ms}, \quad \omega = 30 \text{ rad/s}, \quad I = 0.7 \text{ pu},$$
Step 2:
$$600 \text{ ms} < t < 800 \text{ ms}, \quad \omega = 90 \text{ rad/s}, \quad I = 0.3 \text{ pu},$$
Step 3:
$$t > 800 \text{ ms}, \quad \omega = 15 \text{ rad/s}, \quad I = 1.0 \text{ pu}.$$

The particular sequence and nature of the step changes do not represent any particular mode of induction motor operation, but are the same as the sequence used by [4] to illustrate that convergence occurs each time, following any rapid change.

Measurement noise is artificially introduced into the simulations in order to more closely approximate the real system, but the PWM and the inverter are not modeled. The results in Fig. 5 show that both training methods allow the ANN to quickly identify the stator currents from a random initial

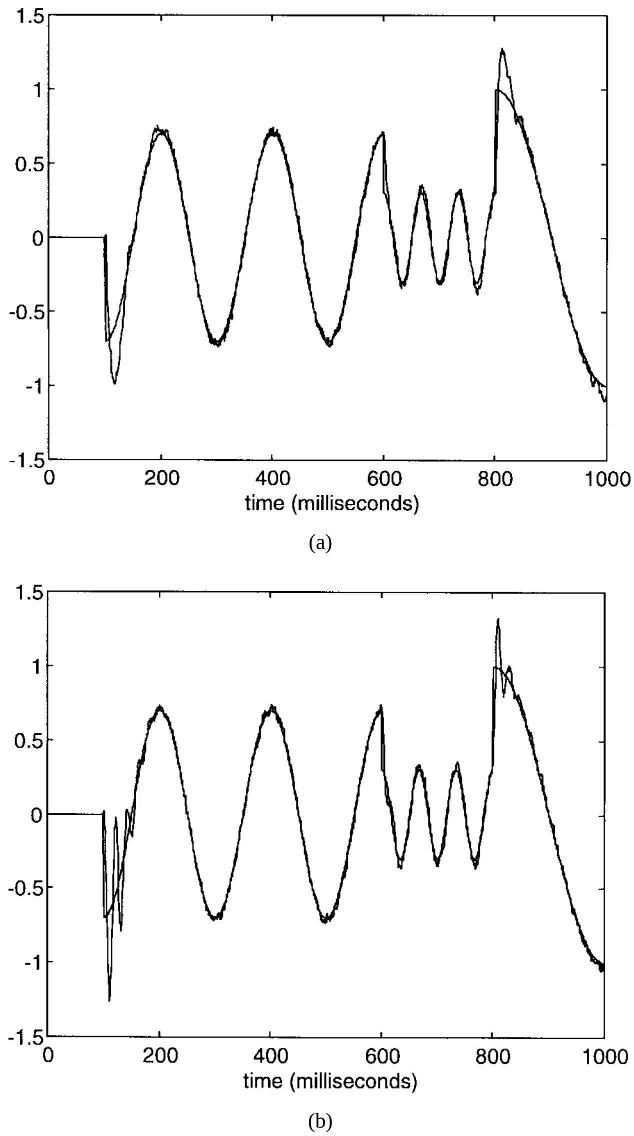


Fig. 5. Desired and actual current (d-axis). (a) Using COT-RWC training. (b) Using backpropagation.

state and, thereafter, closely track all three steps of desired response. The RWC results of Fig. 5(a) are at least as good as those of backpropagation in Fig. 5(b). The *training error* (which is a variable in Fig. 1) for each method is shown in Fig. 6 and confirms that RWC, in fact, converges at least as well as backpropagation, although neither method has been optimized for these results.

B. Case Study Two: Convergence of RWC

Both RWC and backpropagation in section VI-A started with a random initial set of weights. In order to show that the RWC convergence does not depend on the values of the initial set, the RWC simulation in section VI-A is repeated nine more times, each time starting with a different random set of weights. This yields ten sets of curves like the ones in Figs. 5(a) and 6(a), but with each one differing slightly from the other, due to the random nature of the RWC algorithm. Rather than overlaying ten such curves, only a single curve which represents the average of the ten training errors is shown

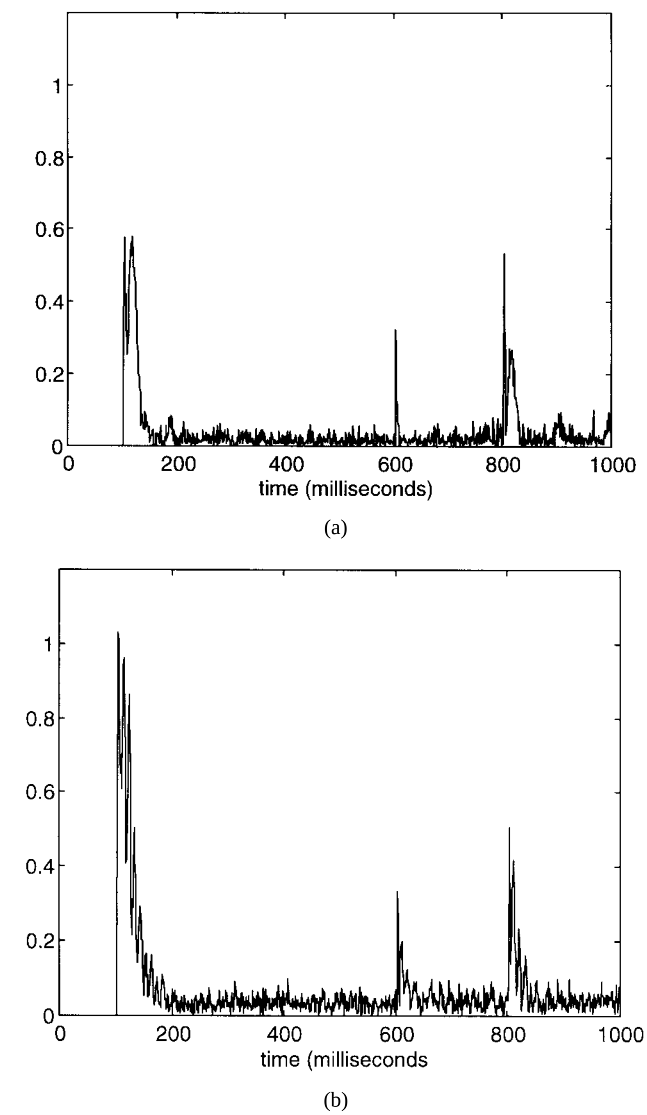


Fig. 6. The *training error* used to make weight updates. (a) COT-RWC method. (b) Backpropagation.

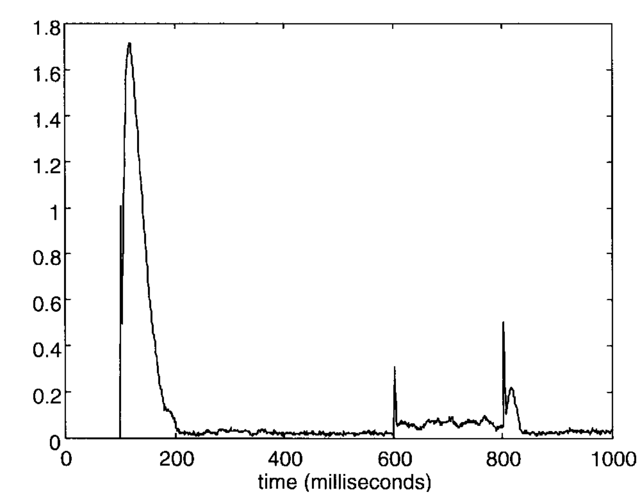


Fig. 7. Averaged *training error* over ten independent simulation runs using the RWC method from arbitrary initial conditions.

in Fig. 7, and it clearly shows an average convergence as good as the single result of Fig. 6(a). This means that the good

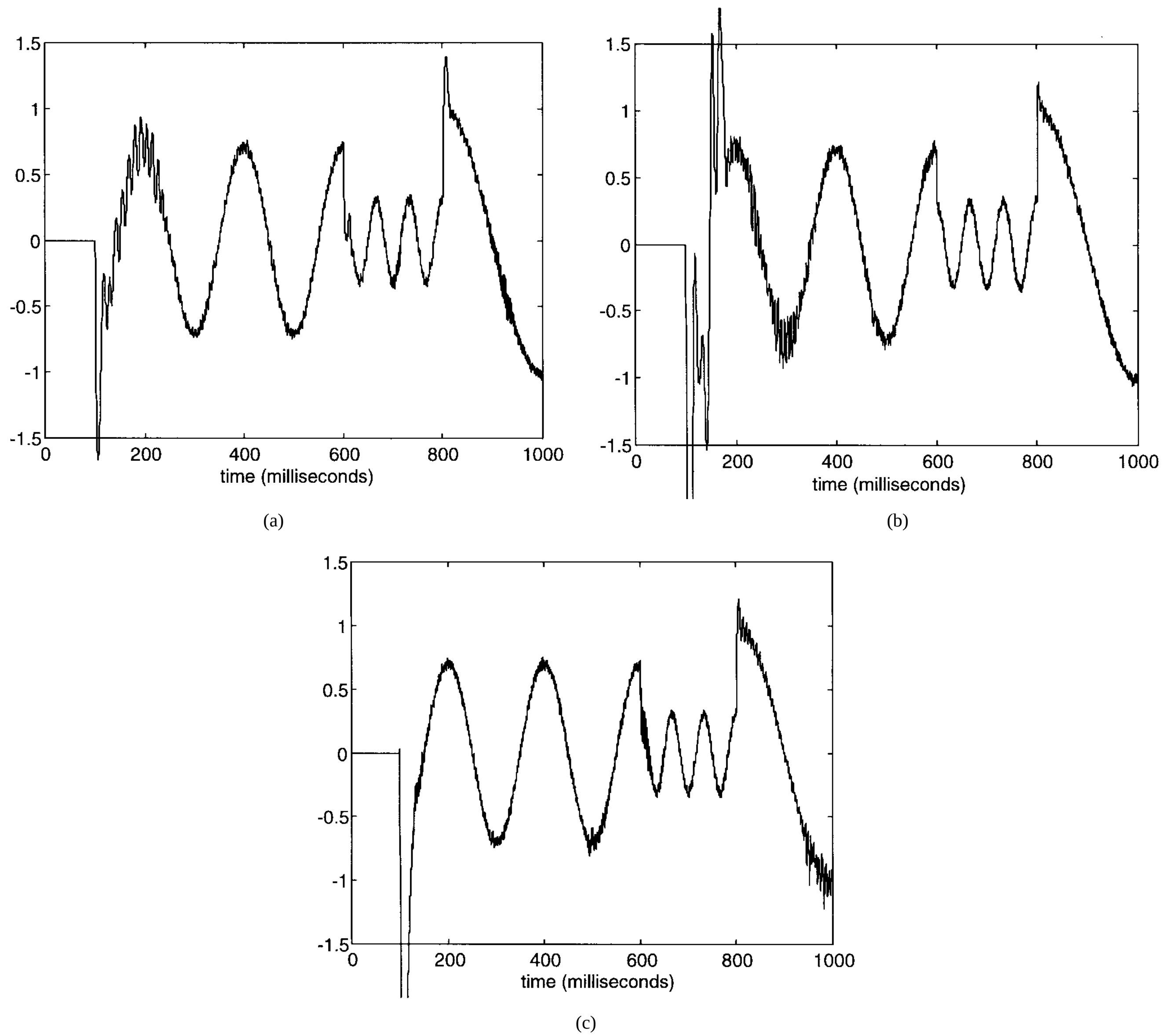


Fig. 8. Depicting the effect of number of nodes while using RWC training. Simulation results using: (a) 12 nodes, (b) 20 nodes, and (c) 30 nodes.

result in Fig. 5(a) is not the consequence of a "lucky break" in choosing the initial weights, but that RWC converges each time, irrespective of the initial values.

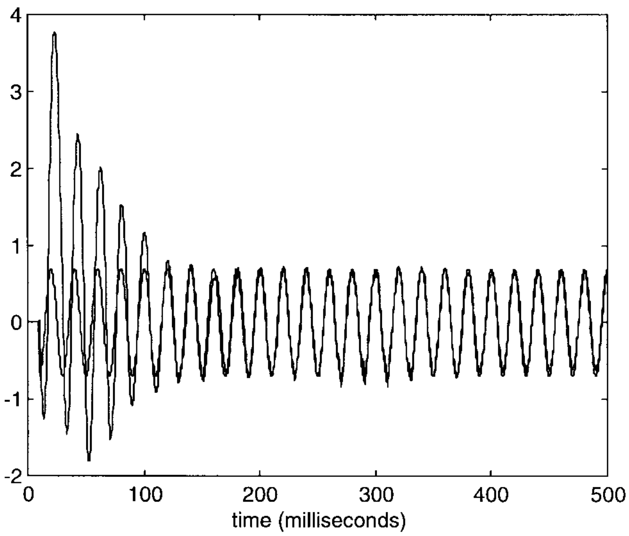
C. Case Study Three: Number of Inner Layer Nodes

The ANN in sections VI-A and VI-B used $m = 12$ inner layer nodes. In order to test whether the convergence properties of the RWC algorithm depend on the value of m, the test of Fig. 5(a) is repeated in Fig. 8 for an ANN with $m = 12$, 20, and 30 nodes, respectively. Obviously, the initial set of weights for each number of nodes differs from that of the other set, because of the different lengths of the weight vectors. However, the same value of the step size $\delta = 0.005$ is used for all the results in Fig. 8.

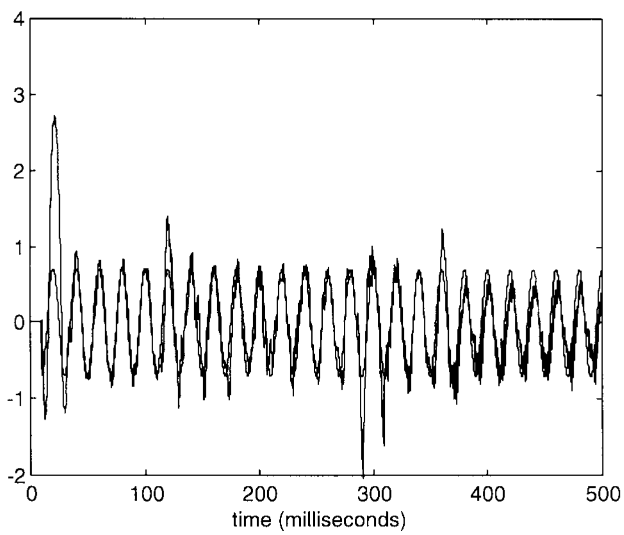
The marked difference in the initial responses (from $t = 100$ ms to about 150 ms) of the 12, 20, and 30 node systems is influenced very much by their different random starting weights. After this initial period, it appears that the 20-node ANN tracks the desired curve better than the 12-node ANN and that the 30-node ANN tracks best of all three. Once the desired current settles into a steady sinewave, the differences in tracking ability become almost insignificant. When the second and third transients occur, the 20- and 30-node systems seem to track only slightly better than the 12-node system; this is to be expected, since a 30-node system contains more high-frequency training information than a 12-node system. Nevertheless, the performance of the 12-node system appears acceptable and has the advantage of being a smaller system with a lower computational burden to implement.

D. Case Study Four: Influence of δ

In the previous case studies, the frequency of the current was stepped to 30, then to 90, and then to 15 rad/s. Induction motors often operate at frequencies higher than these and, so, for the next case study, the frequency of the 12-node system is stepped to 314 rad/s (50 Hz) in order to verify that the



(a)



(b)

Fig. 9. The desired and actual currents showing the effect of δ on learning. (a) $\delta = 0.005$. (b) $\delta = 0.02$.

same system which previously converged [Fig. 8(a)] for the slower frequencies will also converge at a higher frequency. The simulation results appear in Fig. 9 and are repeated for different values of δ, but each simulation now starts with the same random set of initial weights, so that the differences in the results are not due to different initial conditions.

The results of Fig. 9(a) [when $\delta = 0.005$ which is the same value as for Fig. 8(a)] show that the actual current needs a few cycles to converge to the commanded sinewave. Thereafter, the tracking is close, although a small steady-state error continues to exist at the end. Further simulations with slightly larger values of δ yield a smaller steady-state error, but soon result in an oscillatory response; all these results are not shown, except for a selected case when $\delta = 0.02$ in Fig. 9(b), in order to illustrate the oscillatory response when δ is too large.

The results of Fig. 9 show that the influence of the training step size δ is like the proportional gain of a traditional feedback controller. In other words, small oscillations set in before the steady-state error can be driven to zero by increasing δ. This suggests that the step size should not be a fixed value δ, but should also contain a term sensitive to the size of the error, for example.

VII. Conclusions

Backpropagation has been previously used for identification and control of the stator currents of an induction motor. However, ANN hardware implementation of backpropagation must be capable of executing one epoch in less than 50 μs, a requirement imposed by the continual on-line training and a sampling frequency of 10 kHz. An execution speed of this order, using backpropagation, is not realizable with existing VLSI technology. This paper has presented a practical form of the fast on-line RWC training algorithm for feedforward ANN's with a potential for mixed signal (analog/digital) VLSI realizability able to meet the above time constraint. The RWC algorithm is based on the method of random search, is computationally simple, and suitable for VLSI implementation; moreover, it produces results comparable to backpropagation.

Results have been presented to show that an ANN with 12 nodes in the inner layer can be trained to identify and control the motor currents almost as well as one with 30 nodes, but the 12-node ANN is preferred because of its lower computational requirement on the implementation hardware. The results also show that the 12-node ANN system performs well both at low- and high-frequency motor currents. Moreover, the value of the training step size δ influences the system behavior in the same way as the proportion gain of a traditional feedback controller.

An inverter-fed adjustable-speed induction motor could be identified, and its stator currents controlled, within a few milliseconds of the startup and, thus, provide self-commissioning, while the ANN has no prior information whatsoever of the inverter and the motor connected to it.

References

[1] D. J. Atkinson, P. P. Acarnley, and J. W. Finch, "Observers for induction motor states and parameter estimation," *IEEE Trans. Ind. Applicat.,* vol. 27, pp. 1119–1127, Nov./Dec. 1991.

[2] M. R. Buhl and R. D. Lorenz, "Design and implementation of neural networks for digital current regulation of inverter drives," in *Conf. Rec. IEEE IAS Annu. Meeting,* Detroit, MI, Oct. 1991, pp. 415–421.

[3] D. R. Seidl, D. A. Kaiser, and R. D. Lorenz, "One-step space vector PWM current regulation using a neural network," in *Conf. Rec. IEEE IAS Annu. Meeting,* Denver, CO, Oct. 1994, pp. 867–874.

[4] M. T. Wishart and R. G. Harley, "Identification and control of an induction machine using artificial neural networks," in *Conf. Rec. IEEE IAS Annu. Meeting,* Toronto, Ont., Canada, Oct. 1993, pp. 703–709.

[5] B. Burton, R. G. Harley, G. Diana, and J. R. Rodgerson, "Implementation of a neural network to adaptively identify and control VSI fed induction motor stator currents," in *Conf. Rec. IEEE IAS Annu. Meeting,* Denver, CO, Oct. 1994, pp. 1733–1740.

[6] B. Burton and R. G. Harley, "Reducing the computational demands of continually online trained artificial neural networks for system identification and control of fast processes," in *Conf. Rec. IEEE IAS Annu. Meeting,* Denver, CO, Oct. 1994, pp. 1836–1843.

[7] Å. Eide, T. Lindblad, C. S. Lindsey, M. Minerskjöld, G. Sekhniaidze, and G. Székely, "An implementation of the zero instruction set computer (ZISC036) on a PC/ISA-bus card," presented at WNN/FNN, Washington, DC, Dec. 1994.

[8] C. Park, K. Buckmann, J. Diamond, U. Santoni, S. The, M. Holler, M. Glier, C. L. Scofield, and L. Nunez, "A radial basis function

neural network with on-chip learning," in *Proc. IJCNN*, Oct. 1993, pp. 3035–3038.
[9] G. Cancelo and S. Hansen, "Analog neural network development system with fast on line training capabilities," in *Conf. Rec. IEEE IECON*, Sept. 1994, pp. 1396–1400.
[10] K. Hirotsu and M. Brooke, "An analog neural network chip with random weight change learning algorithm," in *Proc. IJCNN*, Oct. 1993, pp. 3031–3034.

Bruce Burton (S'94–A'95) was born in Johannesburg, South Africa, in 1971. He received the B.Sc. Eng. degree in electronic engineering and the M.Sc. Eng. degree in electrical engineering from the University of Natal, Durban, South Africa, in 1992 and 1995, respectively. He is currently working toward the Ph.D. degree in the Motion Control Research Group, University of Natal.

His research interests include artificial intelligence, control systems, electrical drives, and digital controller hardware.

Farrukh Kamran (S'92–M'92) received the B.Sc. degree in electrical engineering from the University of Engineering and Technology, Lahore, Pakistan, in 1988 and the M.S.E.E. and Ph.D. degrees from Georgia Institute of Technology, Atlanta, GA, in 1992 and 1995, respectively.

He was a Design Engineer with EGS (pvt) Ltd., Pakistan, from 1988 to 1989. He joined HiPo Electronics (pvt) Ltd., Pakistan, in 1989 as a Senior Design Engineer, where he worked until 1991. Since 1995, he has been with Silicon Power Company, Inc., Malvern, PA, as a Senior Research Engineer.

Mr. Kamran was awarded a scholarship for graduate studies by the Government of Pakistan.

Ronald G. Harley (M'77–SM'86–F'92) received the M.Sc. Eng. degree from the University of Pretoria, Pretoria, South Africa, in 1965 and the Ph.D. degree from Imperial College, London, U.K., in 1969.

In 1971, he was appointed Professor of Electrical Machines and Control at the University of Natal, Durban, South Africa. He was a Visiting Professor at Iowa State University of Science and Technology, Ames, in 1977, Clemson University, Clemson, SC, in 1987, and Georgia Institute of Technology, Atlanta, GA, in 1994. His areas of research include power system dynamics, electrical machines, power electronics, and control of ac variable speed drives.

Dr. Harley is a member of the IEEE Power Electronics and Industry Applications Societies. He is also a fellow of the Institution of Electrical Engineers, U.K., and the South Africa Institute of Electrical Engineers.

Thomas G. Habetler (S'83–M'83–SM'92) received the B.S.E.E. and M.S. degrees in electrical engineering from Marquette University, Milwaukee, WI, in 1981 and 1984, respectively, and the Ph.D. degree from the University of Wisconsin, Madison, in 1989.

From 1983 to 1985, he was with the Electro-Motive Division of General Motors as a Project Engineer. While there, he was involved in the design of switching power supplies and voltage regulators for locomotive applications. He is currently an Associate Professor of Electrical Engineering, Georgia Institute of Technology, Atlanta, GA. His research interests are in switching converter technology and electric machine protection and drives.

Dr. Habetler was co-recipient of the 1989 First Prize Paper Award and the 1991 Second Prize Paper Award of the Industrial Drives Committee, and the 1994 Second Prize Paper Award of the Electric Machines Committee of the IEEE Industry Applications Society (IAS). Dr. Habetler is an Associate Editor of the IEEE TRANSACTIONS ON POWER ELECTRONICS. He also serves as Secretary of the IAS Industrial Power Converter Committee and as Educational Activities Chair and Member-at-Large of the AdCom of the IEEE Power Electronics Society.

Martin A. Brooke (S'85–M'86) received the B.E. (Elect.) degree (1st Class Honors) from Auckland University, Auckland, New Zealand, in 1981 and the M.S. and Ph.D. degrees in electrical engineering from the University of Southern California, Los Angeles, in 1984 and 1988, respectively. His doctoral research focused on reconfigurable analog and digital integrated circuit design.

He is currently the Analog Device Career Development Assistant Professor of Electrical Engineering, Georgia Institute of Technology, Atlanta, and is developing electronically adjustable parallel analog circuit building blocks that achieve high levels of performance and fault tolerance. His current research interests are in high-speed high-precision signal processing. Current projects include development of adaptive neural network and analog multipliers and dividers, precision analog amplifiers, communications circuits, and sensor signal processing circuitry. To support this large analog and digital systems research, he actively pursues systems level circuit modeling research. He has developed software that reduces the complexity of large analog electronic system models to a user-specified tolerance.

Prof. Brooke is the recipient of the 1990 NSF Research Initiation Award given in the analog signal processing area.

Ravi Poddar received the B.E.E. degree with highest honors in 1991 and the M.S. degree in 1995 from the Georgia Institute of Technology, Atlanta, where he is currently working toward the Ph.D. degree.

His research interests include computer-aided design and simulation of integrated circuits, device and interconnect modeling, neural networks, integrated passive and multichip module modeling, and characterization and high-speed automated testing.