A FAST ON-LINE NEURAL NETWORK TRAINING ALGORITHM FOR A RECTIFIER REGULATOR.

by

Farrukh Kamran*, Ronald G Harley**, Bruce Burton**, Thomas G Habetler* and Martin Brooke*

*Georgia Institute of Technology
School of Electrical and Computer Engr.
Atlanta, GA 30332
Tel: (404)-853-9829
Fax: (404)-853-9171
E-mail: thabetler@ee.gatech.edu

**Dept. of Electrical Engineering
University of Natal
King George V Ave
Durban, 4001 South Africa
Tel: 27-31-260-2725
Fax: 27-31-260-1300

Abstract — This paper addresses the problem of deadbeat control in fully controlled high power factor rectifiers. Improved deadbeat control can be achieved through the use of neural network-based predictors for the input current reference to the rectifier. In this application, on-line training is absolutely required. In order to achieve sufficiently fast on-line training, a new random search algorithm is presented and evaluated. Simulation results show that this type of network training yields equivalent performance to standard backpropagation training. Unlike backpropagation, however, the random weight change method, can be implemented in mixed digital/analog hardware for this application. The paper proposes a VLSI implementation which achieves a training epoch as low as 8 μsec.

INTRODUCTION

Rectifiers are an integral part of almost every power electronic system that connects to the utility supplying AC. In the industrial power range, mainly three-phase rectifiers are employed with either one-way or bi-directional power flow capabilities. In case of the three-phase bi-directional rectifiers, the boost-derived six-switch topology has been adopted by most researchers because of its advantages. The most popular method of control for these rectifiers consists of two nested control loops; a slower outer loop that computes reference signals for the AC input currents and a faster inner current regulator that forces the actual AC input currents to follow these desired trajectories as illustrated in Fig. 1. This control method guarantees stability and also achieves high control bandwidth and good steady state performance. The input current reference, I_{dq}^{ref}, is derived so as to balance the instantaneous AC input and DC output powers and to keep the input currents proportional to the corresponding input phase voltages. The input power demand is computed as a combination of the instantaneous output power and the DC bus voltage error.

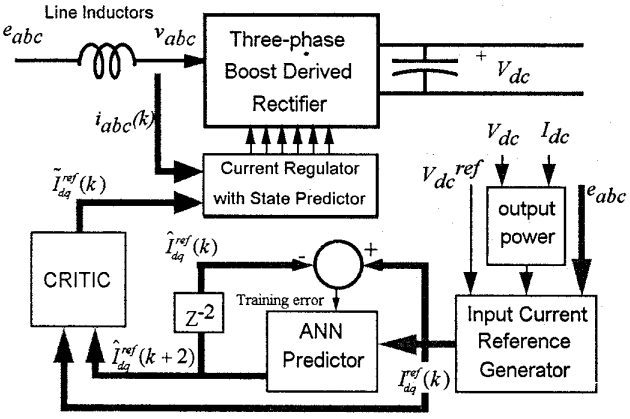

Fig. 1. Rectifier control scheme using a neural net predictor, illustrating the error to be minimized.

The current regulator typically falls into one of the following two categories; a hysteresis current regulator or one of its improved forms [1], or a predictive deadbeat control [2]. Both techniques have their advantages and disadvantages and the choice of a particular technique depends upon factors like the switching frequency used, the ease of implementation etc. A number of deadbeat control techniques have been presented previously with increasing degrees of complexity and improved performance. One such technique [2] with good controller performance was subsequently improved [3] by using prediction schemes (see Fig. 1) for the system states and the reference currents which dramatically improved the performance particularly with small DC bus energy storage. The input current reference signals, I_{dq}^{ref}, in the stationary dq frame of

0-7803-3026-9/95 $4.00 © 1995 IEEE

reference, are derived from a combination of the DC bus voltage error feedback and the output power feedforward. This I_{dq}^{ref} is fed to a current regulator that uses the system model to compute the required voltage, v_{abc}, at the rectifier side of the line inductors; v_{abc} is realized by space vector pulse width modulation in the rectifier.

Because of the computation delay, by the time v_{abc} is calculated, other system variables i.e., the input currents, i_{abc}, and voltages, e_{abc}, have changed to new values. In order to fully utilize the sampling time, T_s, for PWM purposes, this computed v_{abc} is applied at the beginning of the following sampling interval. This results in a control error since the computation of v_{abc} was based on the actual (old) sampled system variables. However, if one-sampling-interval-ahead-predicted system variables are used in the control computation, it is possible to eliminate or reduce this error. Another source of error stems from the fact that the I_{dq}^{ref} computation is based on instantaneous input power demand to supply the outward flow of energy and to compensate for any errors in the DC bus. Because of the computation delay and the fact that the deadbeat control drives the input currents to the reference values in exactly one sampling interval, the input currents actually lag I_{dq}^{ref} by $4\pi T_S$, where T_S is the sampling interval. This lag results in input power being slightly less than the desired power commanded by I_{dq}^{ref}. If the DC bus energy storage is small compared to the output power, this delay manifests itself as small oscillations on the DC bus voltage. If the sampling time T_S is large or the DC bus capacitor is too small, these oscillations can cause instability. This error in deadbeat control can be cured by using a $2T_S$ ahead predicted value of I_{dq}^{ref} in the control computation. Because the I_{dq}^{ref} calculation requires the output power and the input voltages which can not be modeled because of their statistical nature, this prediction of I_{dq}^{ref} must use statistical or AI methods. An Artificial Neural Net (ANN) based predictor was used in reference [3] along with a state predictor to greatly improve the performance of the rectifier regulator. Fig. 2 shows a typical result where the impact of the prediction schemes on the DC bus voltage ripple is obvious.

The feedforward ANN in [3] was trained *on-line* using standard backpropagation as opposed to most applications of the feedforward neural nets where training is performed *off-line* using pre-stored data. In general, each on-line training ***epoch*** consists of propagating the ANN input vector forward through the ANN to compute its output, comparing of this output with some reference to compute the training error, and finally modifying the ANN weights in such a way as to reduce the magnitude of this error. Since the ANN input vector changes from one sampling interval to the next, one training epoch has to be completed in one sampling interval. The typical desired sampling frequency is in the kHz range, the corresponding sampling time being in hundreds of microseconds; in reference [3], the sampling time was 128 μs. During this sampling time, the processor has to sample, compute the control and do housekeeping chores; the remaining time is available and can be dedicated to ANN computations; a typical duration being 50 μs [3]. Due to the lack of a suitable ANN ASIC, the ANN of [3] was originally implemented in software on a microprocessor, but for a practical sized ANN (e.g. 20 inputs, 20 middle-layer nodes and two outputs), it was not possible to complete all the computations involved in one training epoch during this 50 micro-seconds.

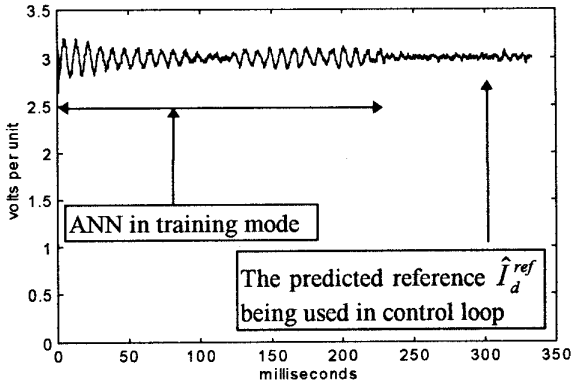

Fig. 2. Rectifier output DC voltage illustrating the improvement in control performance due to state and reference signal prediction.

This paper proposes and investigates the use of a new fast on-line training algorithm for feedforward ANNs suitable for hardware implementation and which can meet the timing constraints described above. As opposed to backpropagation, this method is robust and insensitive to the non-idealities of the analog VLSI circuits.

COMPARISON OF AVAILABLE HARDWARE

Although attempts have been made to implement the feedforward ANN and its backpropagation algorithm in software on single or multiple processor arrays [4], the speed achieved is too slow for the real time application of this rectifier regulator. Higher speeds may be possible using parallel implementations on multiprocessor architectures [5] but the cost of the system becomes prohibitive. Another alternative is the use of ANN ASICs, but to date, there is no commercial ASIC available that

integrates the backpropagation training algorithm along with the forward pass suitable for on-line training at these speeds. More recently, a zero instruction set computer chip, ZISC036 was introduced by IBM that uses radial basis functions (RBF) as the training method [6]; a similar but more powerful chip using RBF for training has also been developed by Intel and Nestor Inc. [7]. The radial basis function 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 rectifier regulator application considered in this paper.

A complete hardware implementation of the feedforward neural net using backpropagation or one of its variants for continuous 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 non-linearities and offsets present in hardware analog multipliers and adders. The all-digital implementation takes up much larger chip areas than the analog ones and therefore, a fully parallel digital implementation can be realized only for small networks and with lesser number of bits. Backpropagation is sensitive to the bit resolution and fails to converge if the resolution is inadequate. 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 non-idealities of these analog circuits, as mentioned above, render the use of backpropagation or its modified versions unsuitable from a practical standpoint.

All the above 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. This is described in the next section.

THE RANDOM WEIGHT CHANGE TRAINING ALGORITHM

A new ANN training algorithm called Random Weight Change (RWC) has been developed as a variation of a previously proposed method of ANN training based on random search for a minimum on the error surface [9]. As opposed to the deterministic methods of weight training like backpropagation, the RWC algorithm is a statistical or probabilistic method. In very simple terms, during each training cycle or ***epoch***, each of the network weights is perturbed randomly with a fixed magnitude, ±delta, and 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 called ***Trials*** and the final weights are chosen to be the ones that result in the smallest error during one epoch. The flow chart in Fig. 3 explains one training cycle or epoch in more details.

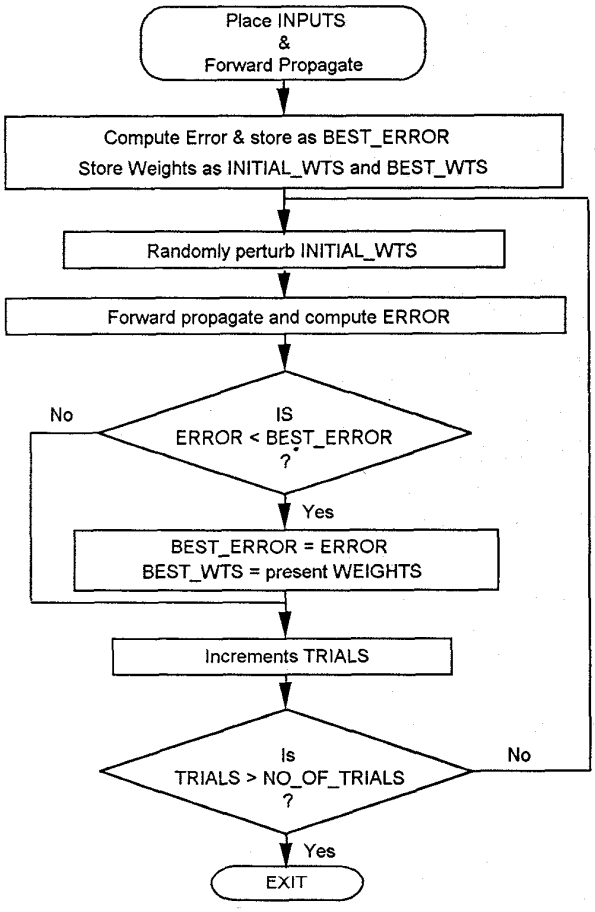

Fig. 2. Flow chart depicting one training epoch of the Random Weight Change training algorithm.

The step size, delta, is a training parameter that needs to be determined heuristically for a specific problem. This is similar to the gain coefficient, β, in backpropagation and

can be thought of as the radius of a hyper-sphere in the n-dimensional error hyper-surface, where 'n' is the total number of network weights. It was observed during experimentation that the value of delta needs to be small, about two orders of magnitude less than the weight magnitudes. Also the final convergence is sensitive to delta which must be determined heuristically for each problem. As is clear from Fig. 3, 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.

RANDOM WEIGHT CHANGE HARDWARE

The primary reason for using the RWC algorithm is that the hardware required is significantly easier to fabricate than for backpropagation. To implement backpropagation training in hardware requires high precision multiplication [9] and this limits the size and/or speed of the hardware that can be fabricated. For applications with hundreds of weights and weight update times in the micro-seconds, the RWC algorithm hardware is, therefore, superior in the cost and complexity of the hardware required. This is because compact analog mixed signal circuitry can be used to perform the weight updates and the forward-propagation of the network.

A proposed hardware schematic for the RWC algorithm learning ANN is presented in Fig. 4. The hardware is controlled by a conventional micro-controller which generates the clock signals (ph1 and ph2) and a few control signals (update, best, rand, delta). These control signals are changed, at most, every weight update cycle (i.e. epoch) and so do not present a challenge to even low cost single chip micro-controllers. 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. 4, would take 800 ns (20 RWC trials). With a less aggressive 10 MHz clock, the epoch would take 8 micro-seconds.

Each weight circuit in the ANN contains three registers (R, W, and B in Fig. 4) that store the random weight changes, the current weight, and the best weight change for the current epoch. 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. 5. The step A1 represents the trial of the current weight change, while A2 represents the return to the original value of the weights.

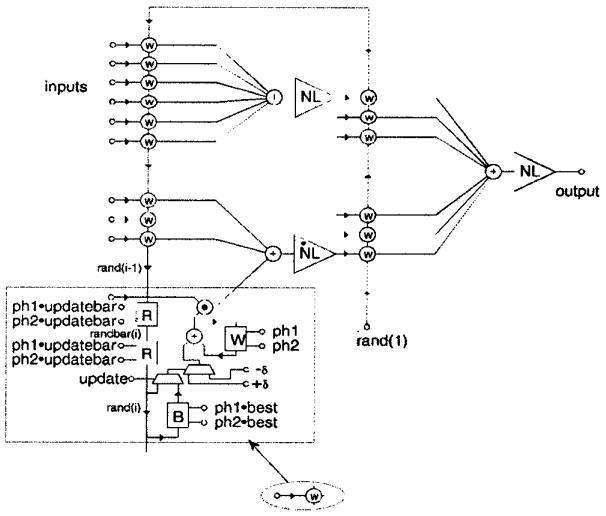

Fig. 3. The Random Weight Change learning hardware schematic.

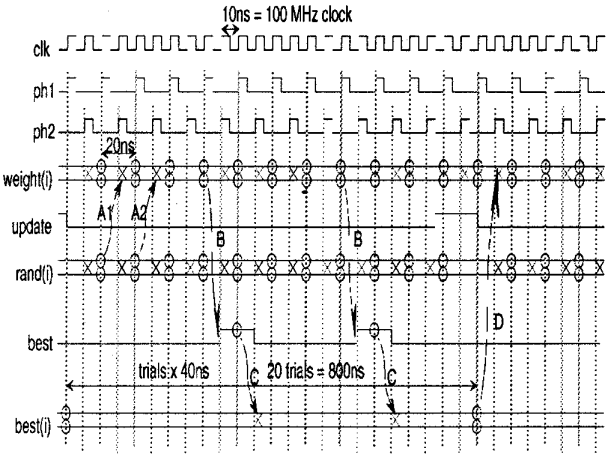

Figure 4. Timing diagram for RWC learning hardware.

The external signal "best" is computed externally from the hardware and indicates 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). 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.

RESULTS

In order to assess the suitability of the RWC algorithm for the rectifier regulator prediction problem, a computer simulation was performed. The ANN used in the simulation consists of 20 inputs, 20 middle layer neurons and two outputs. The middle layer outputs pass through sigmoidal non-linearities while the final outputs are linear. The complete system of Fig. 1 consisting of the three-phase rectifier along with the deadbeat controller and the ANN predictor was simulated. The neural network, which serves as a time-series predictor of the input current reference, uses the previous ten values of I_{dq}^{ref} as ANN inputs for one epoch. During each epoch, the process shown in Fig. 3 are repeated for N=20 times keeping constant the inputs for that epoch.

For comparison purposes, the simulation was repeated with an identical rectifier system and the same ANN, using backpropagation for training. Fig. 6 compares the output errors produced by the two (RWC and backpropagation) training methods as a function of the number of epochs. Clearly, the RWC method converges faster than backpropagation.

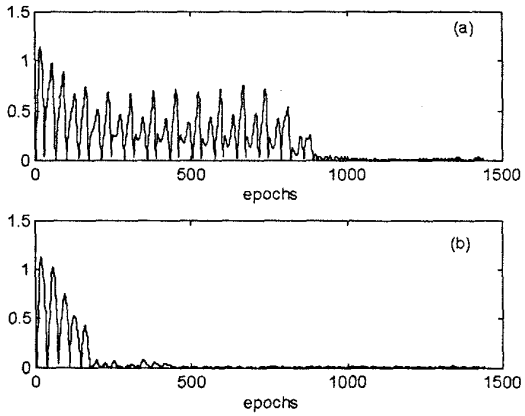

Fig. 6. Absolute value of Output Error illustrating training using (a) backpropagation and (b) using Random Weight Change algorithm.

Fig. 7 shows the computed current reference I_d^{ref} and the predicted current reference signal $\hat{I}_d^{ref}$ using the two training algorithms . Note in Fig. 7 that both the backpropagation method and the RWC method predict essentially the same current reference, and that the predicted current reference leads the computed reference, as expected. The good agreement between the results of Fig. 6 and Fig. 7 prove that the new fast on-line random training method gives results which are as good as those obtained using backpropagation. Figure 8 illustrates the effect of the RWC-based predictor on the rectifier dc output voltage in the presence of a load change, which causes the ANN to retrain.

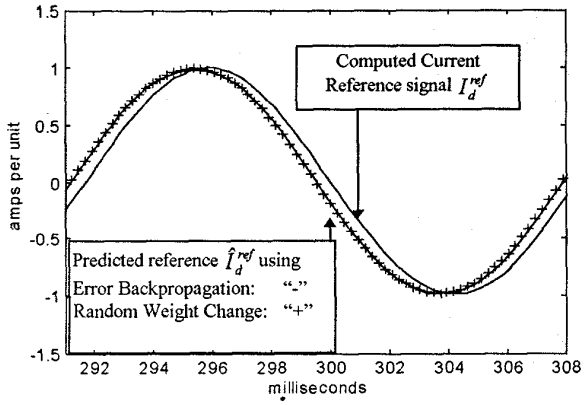

Fig. 7. Computed current reference signal along with the neural net predicted reference signal using backpropagation and random weight change.

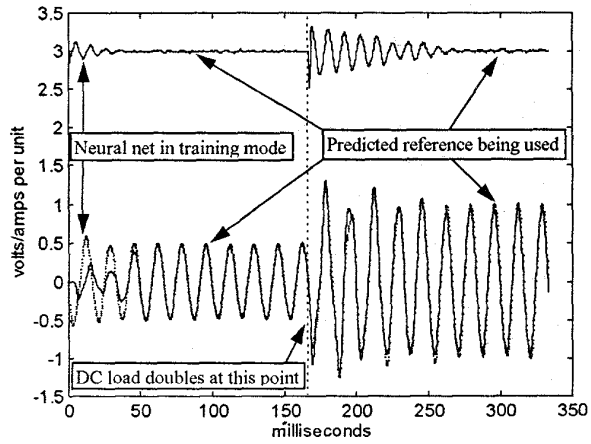

Fig. 8. The DC bus voltage (upper trace) illustrating the improvement in control performance using the predicted current reference signal (using RWC ANN) with a step change in DC load. Lower traces are the computed current reference signal I_d^{ref} '.', and the predicted current reference signal $\hat{I}_d^{ref}$ '-'.

The ANN size and the training parameter delta, are chosen by trial and error. There is a minimum required number of nodes determined by the problem complexity. A network with smaller size cannot retain the information necessary for determining the correct response. This number is determined through experiments. Similarly, the convergence speed and the final resulting error are functions of the training parameter delta and the number of trials. These parameters were also determined through experiments.

Though this paper does not include results for varying network size and the effect of delta, a similar study was performed on an induction motor control problem. Results have been presented in [10] showing 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. Moreover, the value of the training step size, delta, influences the system behavior in the same way as the proportion gain of a traditional feedback controller.

Conclusions

A new fast on-line training algorithm for feedforward artificial neural networks with a potential for hardware realizability has been proposed for use in a rectifier control scheme. The algorithm was demonstrated to produce results comparable to backpropagation when applied to a three-phase rectifier control problem. The algorithm is based on the method of random search, is computationally very simple and therefore, is suitable for VLSI implementation.

References

[1] L. Malesani, L. Rossetto, P. Tenti, P. Tomasin and A. Zuccato, "Performance Optimization of Quasi-Direct Converters," PCC Yokohama, 1993, pp 99-104.

[2] Thomas G. Habetler, "A Space Vector Based Rectifier Regulator for AC/DC/AC Converters," IEEE Trans. on Power Electronics, Vol 8, No 1, Jan. 1993, pp 30-36.

[3] Farrukh Kamran and Thomas G. Habetler, "An Improved Dead-Beat Rectifier Regulator using a Neural Net Predictor," IEEE Power Electronics Specialists Conference, 1994, Taiwan, pp 1431-1436.

[4] Bruce Burton, Ronald G Harley, Gregory Diana. James L Rodgerson, "Implementation of a Neural Network to Adaptively Identify and Control VSI Fed Induction Motor Stator Currents," Conf. Rec. of IEEE IAS Annual Meeting, Oct. 1994, Denver, pp 1733-1740.

[5] Vipin Kumar, Shashi Shekhar and Ninesh B. Amin, "A Scalable Parallel Formulation of the Back Propagation Algorithm for Hypercubes and Related Architectures," IEEE Trans. on Parallel and Distributed Systems, Vol 5, No 10, Oct. 1994, pp 1073-1090.

[6] Å Eide, Th. 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," WNN/FNN, Washington DC, Dec. 1994

[7] Chin Park, Kenneth Buckmann, Jay Diamond, Umberto Santoni, Siang-Chun The, Mark Holler, Michael Glier, Christopher L. Scofield and Linda Nunez, "A Radial Basis Function Neural Network with On-chip Learning," Proceeding of the ICJNN, Oct. 1993, Japan, pp 3035-3038.

[8] Gustavo Cancelo and Sten Hansen, "Analog Neural Network Development System with fast on line training capabilities," IEEE IECON, Sep. 1994, Italy, pp 1396-1400.

[9] Kenichi Hirotsu and Martin Brooke, "An Analog Neural Network Chip with Random Weight Change Learning Algorithm," Proceeding of the ICJNN, Oct. 1993, Japan, pp 3031-3034.

[10] Bruce Burton, Farrukh Kamran, Ronald G Harley, Thomas G Habetler, Martin Brooke and Ravi Poddar, "Identification and Control of Induction Motor Stator Currents Using Fast On-Line Random Training of a Neural Network," To be presented at the IEEE IAS Annual Meeting, 1995, to be held in Orlando.