Fault-Tolerant Training of Neural Networks in the Presence of MOS Transistor Mismatches

Arif Selçuk Öğrenci, *Member, IEEE*, Günhan Dündar, and Sina Balkır

***Abstract*—Analog techniques are desirable for hardware implementation of neural networks due to their numerous advantages such as small size, low power, and high speed. However, these advantages are often offset by the difficulty in the training of analog neural network circuitry. In particular, training of the circuitry by software based on hardware models is impaired by statistical variations in the integrated circuit production process, resulting in performance degradation. In this paper, a new paradigm of noise injection during training for the reduction of this degradation is presented. The variations at the outputs of analog neural network circuitry are modeled based on the transistor-level mismatches occurring between identically designed transistors. Those variations are used as additive noise during training to increase the fault tolerance of the trained neural network. The results of this paradigm are confirmed via numerical experiments and physical measurements and are shown to be superior to the case of adding random noise during training.**

***Index Terms*—Backpropagation, neural network hardware, neural network training, transistor mismatch.**

I. Introduction

THE IMPLEMENTATION of analog neural networks (ANNs) in VLSI remains to be an active research area since they can be utilized in system-on-chip applications where high-performance classification, pattern recognition, function approximation, or control tasks have to be realized in real time. Major advantages of using analog circuitry for neural networks are due to higher operation speed and less silicon area consumption as compared to their digital counterparts. The power consumption of analog implementations is also preferable over digital [1]. On the other hand, analog implementations suffer from several shortcomings and nonidealities such as nonlinearities in the synapses, nonideal neuron behavior, limited precision in storing the weights, limited dynamic range for adding the synapse outputs, etc., whose effects can be minimized by various training and modeling strategies. Chip-in-the-loop training [2]–[6] and on-chip training [7]–[15] are two such strategies that have appeared extensively in the literature. However, the former requires a host computer on site for training of the hardware whereas the latter requires a larger chip area due to the extra circuitry for weight storage and training algorithm.

Training of the analog neural network hardware on software is another alternative. In this method, nonideal behavior of analog neural network circuitry is modeled based on simulation or actual measurement results, which usually cannot be expressed analytically. The training software then employs these models in the backpropagation learning [2], [8], [9], [16]. This type of modeling and training is usually required for obtaining a suitable initial weight set to be used in chip-in-the-loop training. However, the inaccuracies present in these models may lead to small deviations from the actual physical behavior, which in turn may cause the analog neural network training to fail. Although the training may be performed satisfactorily on the software, outputs of the actual circuitry may deviate heavily from those of the ideal training set [17]. In an effort to alleviate this problem, SPICE models of the circuits that closely approximate the actual behavior were used in the training to provide the best starting point for chip-in-the-loop training in [18].

Another essential issue regarding proper training besides the accuracy of the models involves statistical variations in the IC manufacturing process. These variations cause the outputs of identical blocks to exhibit interdie/intradie random distributions. Such variations impose serious constraints on the training algorithm of the analog neural network. The requirement that each individual chip-set has to be trained separately in order to avoid the effects of variations among identical blocks degrades the applicability of chip-in-the-loop training. It has been shown in [2], [8], and [9] that nonidealities do not form an obstacle for the learning ability of neural networks if they are known and invariant during the training stage. Variations of a random nature, on the other hand, cannot be tolerated.

Injecting noise into the inputs, weights, or outputs during backpropagation training has also been utilized for generating a fault-tolerant neural network [19], [20], which, by definition, should also be tolerant to process variations. However, the robustness of training for the analog neural networks is not guaranteed by improved generalization since the problem for the analog hardware extends more into the randomness of the variations between identical blocks. It has been shown in [21] that injecting analog noise to weights during the training dramatically enhances the generalization of the neural network. The effect of synaptic noise is to distribute the dependence of the output evenly across the weight set [22]. The simulations with ideal neural network elements reveal that injection of multiplicative analog noise during training would yield a more robust network

Manuscript received July 1999; revised February 2001. This work was supported by TÜBİTAK under Grant EEE-AG 183. This paper was recommended by Associate Editor A. Andreou.

A. S. Öğrenci is with the Department of Electronics Engineering, Kadir Has University, İstanbul, Turkey (e-mail: ogrenci@khas.edu.tr).

G. Dündar is with the Department of Electrical and Electronics Engineering, Boğaziçi University, Bebek 80815 İstanbul, Turkey (e-mail: dundar@boun.edu.tr).

S. Balkır is with the Department of Electrical Engineering, University of Nebraska-Lincoln, Lincoln, NE 68588-0511 USA (e-mail: sbalkir@unl.edu).

Publisher Item Identifier S 1057-7130(01)04197-0.

against faults created by removal of synapses or by perturbation of final weight values [22]. Clearly, such an improvement would also be beneficial where variations at outputs of building blocks are encountered. In [16], it is reported that an ANN hardware has been modeled in software for faults, and the enhanced fault tolerance of the network by means of weight noise is verified via simulations. However, the necessity to measure the relative slopes of each multiplier in the neurons and the limited dynamic range as a hardware restriction prove to be prohibitive.

It is necessary to overcome the problems due to hardware nonidealities and variations so that once a robust training of the system for a specific task is achieved, any chip of the same family can be directly utilized. To that end, these variations have to be incorporated into the training of analog neural networks.

In this paper, a novel approach to fault-tolerant training of ANN in the presence of manufacturing process variations is presented. The focus is on multilayer perceptron neural networks implemented in MOS technology and trained by backpropagation algorithm. However, the assertions and results can be extended to other topologies, learning strategies, and technologies as well. The following are combined into a robust training methodology.

1) Closed-form analytical expressions of statistical variations from the nominal output are derived for ANN building blocks.
2) These theoretical variations are then compared to the results of measurements performed on sample integrated circuits and to Monte Carlo simulations.
3) In order to incorporate the variations at the output, the training algorithm (backpropagation) is modified. The building blocks are modeled according to their average outputs, whereas the variations are considered to be noise with a normal distribution.

The outline of this paper is as follows. In Section II, transistor-level mismatch models are used to derive expressions for variations at the block level. To demonstrate the validity of these expressions, they are compared with the results obtained from simulations and measurements. The training is carried out using "noisy" backpropagation where the outputs of the blocks are calculated in a probabilistic manner, taking the noise into account as outlined in Section III. In Section IV, sample simulations and measurements on several examples are conducted to verify that this method of training allows a higher degree of fault tolerance in the sense that noisy forward pass outputs exhibit better performance over outputs obtained from the network trained without including the variations. Section V concludes this paper.

II. MOS Transistor Mismatch—Modeling and Verification

A. Modeling of MOS Transistor Mismatch

Mismatch between parameters of two identically designed MOS transistors is the result of several random processes that occur during the fabrication phase. The essential parameters of interest are the zero-bias threshold voltage (V_{T0}), current factor (β), and substrate factor coefficient (γ), which affect the current through the transistor. Any variations in these parameters of two matched transistors would cause a difference in the currents of equally biased transistors. Variations in any parameter may have systematic and random causes. Gradients in oxide thickness and wafer doping cause systematic variations in the parameters along a wafer and among different batches. On the other hand, random, local variations in physical properties of the wafer cause mismatch between closely placed transistors. Nonuniform distribution of dopants in the substrate and fixed oxide charges are responsible for local zero-bias threshold voltage mismatches, whereas variations in substrate doping are the only cause for γ mismatch. The mismatch in current factor β is due to edge roughness and local mobility variations [23]–[25]. The mismatch in a parameter is modeled by a normal distribution with zero mean, and the variance of the distribution for mismatches can be expressed as [23], [24], [26]–[28]

$$\sigma^2(\Delta V_{T0}) = \frac{A_{VT0}^2}{WL} + S_{VT0}^2 D_x^2 \tag{1}$$

$$\frac{\sigma^2(\Delta\beta)}{\beta^2} = \frac{A_\beta^2}{WL} + S_\beta^2 D_x^2 \tag{2}$$

$$\sigma^2(\Delta\gamma) = \frac{A_\gamma^2}{WL} + S_\gamma^2 D_x^2 \tag{3}$$

where A_{VT0}, A_β, A_γ, S_{VT0}, S_β, and S_γ are process-related constants, W and L are the length and width of the transistors, and D_x is the spacing between the matched transistors.

As can be seen from (1)–(3), the variance of the mismatch can be minimized by placing matched transistors close to each other and by choosing large area transistors. However, the latter increases the total die area which should be avoided. A detailed analysis of the circuitry is necessary in order to determine transistor pairs that have higher impact on mismatch behavior so that they are designed accordingly. This requires that mismatches and their cumulative effects on the circuitry have to be modeled. A statistical MOS model has been developed in [29] and [30], which allows the designer to determine circuit output variance due to mismatches in device parameters. However, the methodology requires that a set of test structures have to be built and measured for each specific technology in order to gain knowledge on the model parameters in question. Then, the modeling has to be incorporated into a simulation environment, which is not a straightforward task. Other studies on modeling the mismatch based on the circuit structure also exist in the literature [31]–[33]. In this paper, the focus is primarily on mismatches between pairs of transistors and the analytical derivation of their effects on outputs.

The differential pair and current mirror are among the most frequently used structures in analog integrated circuits. This is also the case for the ANN considered in this paper, where the building blocks are the synapse (Gilbert type, 4-quadrant, current output multiplier as shown in Fig. 1), an op-amp (used for adding synapse outputs and converting them to voltage), and the sigmoid block. A prototype chip for testing purposes was designed and manufactured in 2.4-μm CMOS technology. According to the experimental results of [24] and the data collected from the technology design kits, the process-dependent

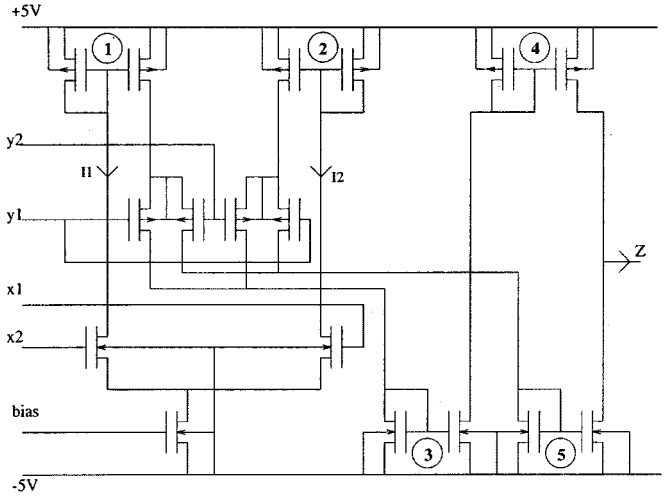

Fig. 1. Circuit diagram of the synapse.

constants are estimated as shown in Table I. Using the data, calculation of the variance of the mismatch in matched transistor pairs of differential stages and current mirrors is possible.

B. Modeling ANN Circuit Mismatches

The aim of mismatch analysis is to predict the variances at the outputs of the building blocks using the mismatches in the individual pairs. The analysis is carried out for two separate cases: variations only in the threshold voltage (due to mismatches in zero-bias threshold voltage V_{T0} and substrate factor γ) or in the current factor β of the matched pairs. Mismatches in the threshold voltage and current factor are shown to be independent [23]. Moreover, the correlation coefficients for transistor pairs with different W/L ratios have been computed using the empirical formulas of [28] between the three parameters, indicating that the variations in different parameters may be considered to be independent.

In the following, the variance of output current for the synapse circuit of Fig. 1 is derived. In the case of V_{T0} mismatches, the total mismatch in the output current Z can be attributed to mismatches in all of the current mirrors and differential pairs. For mismatches caused by the current mirrors, the analytical expression for output current is

$$Z = 7((1 - M_3)(1 - M_4)U - (1 - M_5)T) \tag{4}$$

where

$$U = \tfrac{1}{2} I_1(1 - M_1) + \tfrac{1}{2} I_2(1 - M_2) - \Delta y \frac{\sqrt{\beta_y}}{2}\left(\sqrt{I_1(1 - M_1)} - \sqrt{I_2(1 - M_2)}\right) \tag{5}$$

$$T = \tfrac{1}{2} I_1(1 - M_1) + \tfrac{1}{2} I_2(1 - M_2) + \Delta y \frac{\sqrt{\beta_y}}{2}\left(\sqrt{I_1(1 - M_1)} - \sqrt{I_2(1 - M_2)}\right). \tag{6}$$

I_1 and I_2 are currents flowing through the transistors of the input (x) differential pair, $M_i = (2\Delta V_{T,i})/(V_{GS,i} - V_{T0,i})$ are the factors of mismatch in the current mirror i due to the mismatch $\Delta V_{T,i}$, β_y is the current factor of the transistors in the weight (y) differential pairs, and Δy is the weight $(y_2 - y_1)$. There are five such current mirrors, and the output current Z can be expressed as a function

$$Z = f(\Delta V_{T,1}, \Delta V_{T,2}, \Delta V_{T,3}, \Delta V_{T,4}, \Delta V_{T,5}) \tag{7}$$

for the mismatch analysis. Here, $\Delta V_{T,i}$ are considered to be random variables with zero mean and variance, as given by (1). Then, the variance in Z due to the zero-bias threshold mismatches in current mirrors can be expressed as [34]

$$\sigma(\text{mirror}, V_{T0})_Z^2 = \sum_{i=1}^{5}\left(\frac{\partial f}{\partial \Delta V_{T,i}}\right)^2 \sigma^2_{\Delta V_{T,i}} \tag{8}$$

where the partial derivatives are computed with the zero mean value of the random variables $\Delta V_{T,i}$. Similarly, if the variation due to the threshold voltage mismatches in the differential pairs alone is considered, the output current Z becomes

$$Z = 7\sqrt{\frac{\beta_x\beta_y}{2}}\left(\Delta x + \Delta V_{T0,x} + (\gamma_x + \Delta\gamma_x) \cdot \left(\sqrt{\phi + V_{SB,x}} - \sqrt{\phi}\right)\right)(\Delta y + \Delta V_{T0,y}) \tag{9}$$

where

$\Delta x = (x_2 - x_1)$	input to the multiplier;
$\Delta\gamma_x$	mismatch of substrate factor for the input differential pair;
$\Delta V_{T0,x}$ and $\Delta V_{T0,y}$	mismatches at the differential pairs of input and weight, respectively.

Then, the variance at Z is cast as

$$\sigma(\text{diff-pair}, V_{T0})_Z^2 = \frac{49\beta_x\beta_y}{2}\left(\Delta y^2\sigma^2_{\Delta V_{T0,x}} + \Delta y^2\left(\sqrt{\phi + V_{SB,x}} - \sqrt{\phi}\right)^2 \cdot \sigma^2_{\Delta\gamma_x} + \left(\Delta x + \gamma_x\left(\sqrt{\phi + V_{SB,x}} - \sqrt{\phi}\right)\right)^2\sigma^2_{\Delta V_{T0,y}}\right). \tag{10}$$

Similarly, variations at output current due to the mismatches in the current factor β can be computed for the existence of mismatch at current mirrors and differential pairs. If mismatches at current mirrors only are considered, Z becomes

$$Z = 7((1 + Q_3)(1 + Q_4)U - (1 + Q_5)T) \tag{11}$$

where $Q_i = \Delta\beta_i/\beta_i$ are the factors of mismatch in the current mirror i due to the mismatch $\Delta\beta_i$, and all the terms $(1 - M_i)$ will be replaced by $(1 + Q_i)$ in (5)–(6). Hence, Z becomes a function of mismatches in β as

$$Z = g(\Delta\beta_1, \Delta\beta_2, \Delta\beta_3, \Delta\beta_4, \Delta\beta_5) \tag{12}$$

and the variance in Z due to β mismatches in current mirrors is

$$\sigma(\text{mirror}, \beta)_Z^2 = \sum_{i=1}^{5}\left(\frac{\partial g}{\partial \Delta\beta_i}\right)^2 \sigma^2_{\Delta\beta_i}. \tag{13}$$

Also, the variance due to β mismatches in differential pairs can be derived as

$$\sigma(\text{diff-pair}, \beta)_Z^2 = \frac{49\Delta x^2\Delta y^2}{8}\left(\frac{\beta_y}{\beta_x}\sigma^2_{\Delta\beta,x} + \frac{\beta_x}{\beta_y}\sigma^2_{\Delta\beta,y}\right) \tag{14}$$

TABLE I
PARAMETERS FOR MISMATCH ANALYSIS

	A_{VT0} $(mV\mu m)$	S_{VT0} $(\mu V/\mu m)$	A_β $(\%\mu m)$	S_β $(10^{-6}/\mu m)$	A_γ $(V^{0.5}\mu m)$	S_γ $(10^{-6}V^{0.5}/\mu m)$
NMOS	25	4	2.5	2	0.016	4
PMOS	30	4	3	2	0.012	4

TABLE II
MISMATCH IN THE CURRENT MIRRORS AND DIFFERENTIAL PAIRS OF THE SYNAPSE CIRCUIT

	M1	M2	M3	M4	M5	diff,x	diff,y
$\sigma^2_{\Delta V_{T0}}(mV^2)$	78	78	72	11	26	27	15
$\sigma^2_{\Delta\beta}(\mu A/V)^2$	0.03	0.03	0.19	0.004	0.07	0.36	0.005
$\sigma^2_{\Delta\gamma}$	0	0	0	0	0	0.63e-3	0

where β_x and β_y are current factors of the input and weight differential pairs, respectively. Finally, the total variation at the output current can be expressed by

$$\sigma_Z^2 = \sigma(\text{mirror}, V_{T0})_Z^2 + \sigma(\text{diff-pair}, V_{T0})_Z^2 + \sigma(\text{mirror}, \beta)_Z^2 + \sigma(\text{diff-pair}, \beta)_Z^2 \quad (15)$$

under the assumption that the individual mismatches due to the parameters V_{T0} and β are independent. In the same manner, analytical expressions for the variance at the outputs of the op-amp and sigmoid blocks can also be derived.

C. Verification of the Models with the Test Chip

The structure of each neuron in the test chip is such that current outputs of five synapses are connected to an op-amp summing node for current-to-voltage conversion. Voltage output of the op-amp is then applied to the sigmoid block. The chips have been tested using an automated data acquisition environment. At each step of the measurement for a neuron, four pairs of synapse inputs and weights are set to 0 V, while the fifth synapse input and weight values are swept over a range of -5 V to $+5$ V and -2 V to $+2$ V, respectively. This procedure has been repeated for each of the 200 synapses to obtain characteristics of each synapse as seen at the neuron output (sigmoid input). Meanwhile, outputs of the sigmoid block are also measured for characterization.

The variance of mismatch for the parameters V_{T0}, β, and γ are computed using (1)–(3), and they are given in Table II for current mirrors and differential pairs of the synapse circuit. One hundred Monte Carlo dc sweep runs are performed for a single synapse incorporating the mismatches in V_{T0}, β, and γ, where each mismatch parameter has zero mean and the variance as given in Table II. The average of the Monte Carlo simulations is given in Fig. 2. The variance of the sample of size 100 is computed for each input-weight pair, and the values are plotted in Fig. 3 for nonpositive weight values. The variance of the synapse current Z exhibits a symmetrical behavior as given by $\sigma_Z^2(x, w) = \sigma_Z^2(-x, -w)$. The theoretical mismatch in the synapse output current is calculated using (4)–(15). The results are again plotted for nonpositive weight values in Fig. 4, since the same symmetry also exists for the theoretical computations. It is evident from Figs. 3 and 4 that the mismatch in synapse cur-

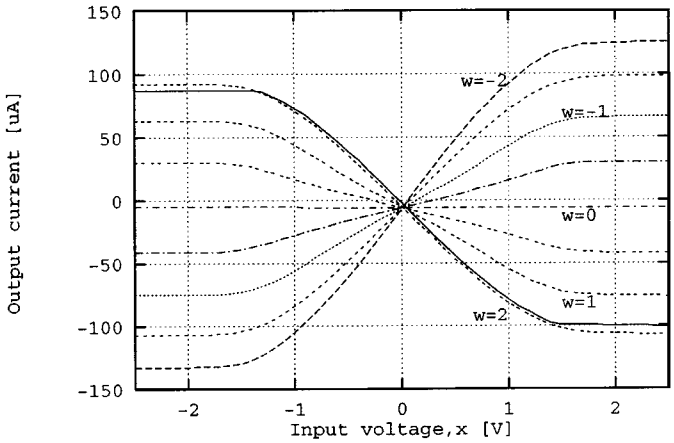

Fig. 2. Average of 100 Monte Carlo runs for the synapse with mismatched model parameters.

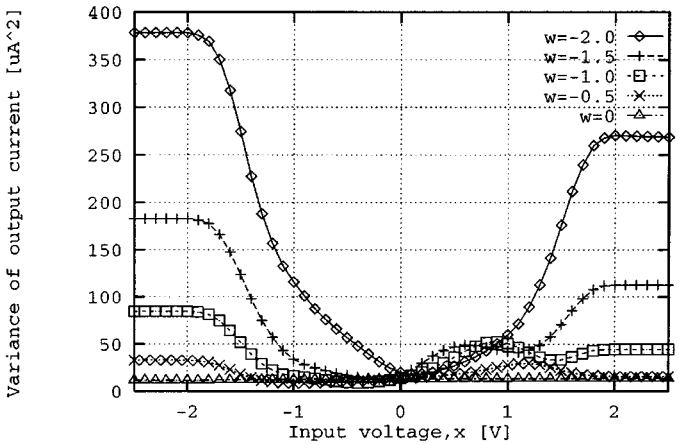

Fig. 3. Variance in synapse output obtained from 100 Monte Carlo runs with mismatched model parameters.

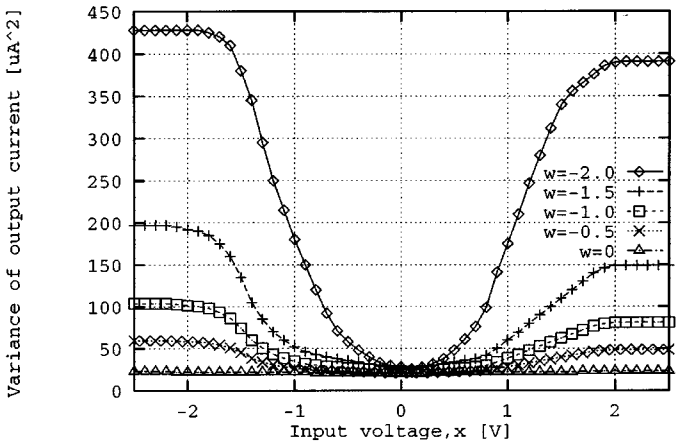

Fig. 4. Variance in synapse output obtained using the formulas.

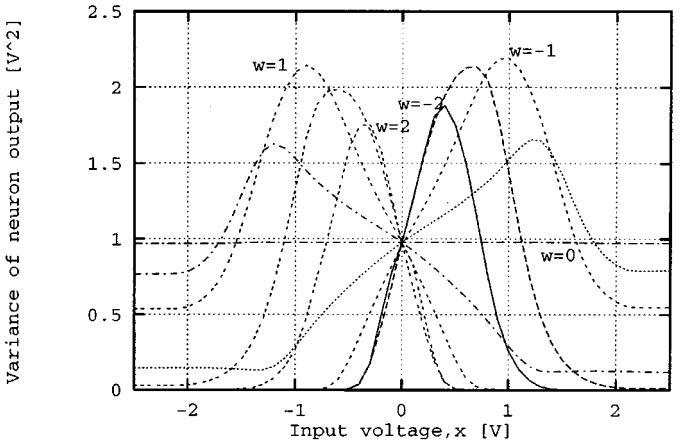

Fig. 5. Variance in neuron output obtained from 100 Monte Carlo runs with mismatched model parameters.

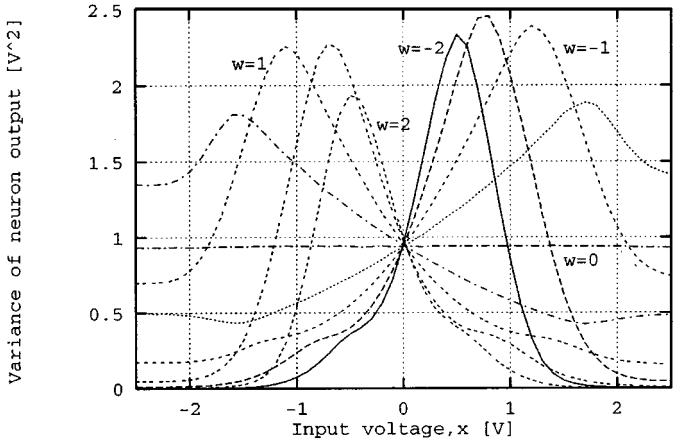

Fig. 6. Variance in neuron output obtained from actual measurements on the chips.

rent as computed by theoretical analysis and as simulated are in close agreement.

A neuron circuit made up of five synapses is also simulated for 100 Monte Carlo runs where each current mirror and differential pair in all of the five synapses and the op-amp are perturbed by the appropriate amount of mismatch. The variance obtained is given in Fig. 5, whereas Fig. 6 displays the measurement results. The close agreement between the two families of curves implies that the variations in the block outputs are due to device mismatches. Moreover, this also suggests that the process-dependent parameters have been estimated close to the actual values.

A similar analysis has also been carried out for the sigmoid block. Three types of results are given in Fig. 7 for the variance in sigmoid output: based on theoretical formulas, 100 runs of Monte Carlo simulations with mismatched model parameters, and measurements on actual chips. These results indicate that the presented approach forms a realistic way of predicting block-level variations based on individual transistor-level mismatches. This type of theoretical analysis not only provides the designer with a training methodology for ANN but also aids the designer in identifying the critical transistor pairs based on the sensitivity of the output to the individual pair mismatches.

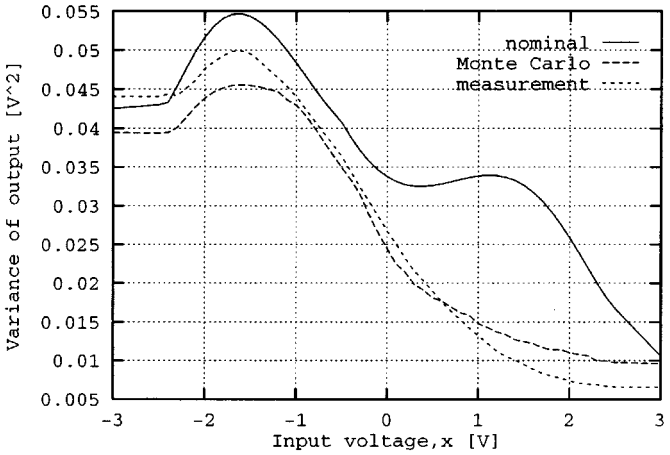

Fig. 7. Variance in sigmoid output.

III. Incorporation of Mismatch into Backpropagation Training

In this paper, the ANN hardware blocks are modeled using analytical expressions for their input–output relations in order to be used in the backpropagation algorithm, which has also been suggested previously [2], [9]. The synapse function can be modeled as a polynomial function with less than 5% deviation as follows:

$$\mu(x, w) = c_0 + (c_1 x + c_2 x^2 + c_3 x^3 + c_4 x^4) \cdot (c_5 w + c_6 w^2 + c_7 w^3 + c_8 w^4) \tag{16}$$

where $\mu(x, w)$ is the output current for the input pair x, w and the coefficients c_i are determined using nonlinear regression, which may take nominal or Monte Carlo simulations as input. The op-amp (I–V conversion) block can be represented by

$$O(\mu) = \begin{cases} VDD, & \text{if } -R\mu > VDD \\ -R\mu, & \text{if } VDD > -R\mu > VSS \\ VSS, & \text{if } -R\mu < VSS \end{cases} \tag{17}$$

where

VDD and VSS	positive and negative supply voltages;
R	implemented resistance (the value is 100 kΩ);
μ	total current supplied by the synapses.

For the sigmoid block, an exponential function with coefficients d_i is employed in regression

$$\phi(x) = d_0 + \frac{d_1}{1 + \exp(d_2 x + d_3)}. \tag{18}$$

During backpropagation training, the variations are modeled according to the mismatch analysis of the circuitry, and they are considered as zero mean additive noise with known variance depending on the input values of the blocks. For the variances in synapse and sigmoid blocks, regression is again employed based on the data obtained from theoretical computations, Monte Carlo simulations, and measurements separately to fit the following polynomials:

$$\sigma_\mu^2(x, w) = c_0 + (c_1 x + c_2 x^2 + c_3 x^3 + c_4 x^4 + c_9 x^6) \cdot (c_5 w + c_6 w^2 + c_7 w^3 + c_8 w^4) \tag{19}$$

$$\sigma_\phi^2(x) = d_0 + d_1 x + d_2 x^2 + d_3 x^3 + d_4 x^4. \tag{20}$$

The output of the op-amp is governed by the expression

$$O(\mu) = -(R + \Delta R)\mu + \Delta V_{\text{offset}} \tag{21}$$

where the implemented resistor has a certain variation ΔR (approximately 5–10% for standard processes) and the output is affected by the input offset voltage ΔV_{offset} of the op-amp only. Thus, the resulting expression for the variance at op-amp output becomes

$$\sigma_O^2(\mu) = \mu^2 \sigma_{\Delta R}^2 + \sigma_{\Delta V_{\text{offset}}}^2. \tag{22}$$

Once the "noisy" behavior of synapse and sigmoid circuits has been modeled based on analytical calculations (or Monte Carlo simulations), the backpropagation algorithm can be modified in order to incorporate those variations at the outputs. For this purpose, a scalar input–output multilayer perceptron structure is considered. The output of the multilayer perceptron (y) is the op-amp (neuron) output $O(\mu)$ (a weighted sum of the outputs of a number (h) of hidden units, net_o), filtered through the nonlinear sigmoid function $\phi(x)$ as follows:

$$net_o = \left(\sum_{i=1}^{h} \mu(T_i, H_i(x)) + \mu_n(T_i, H_i(x)) + \mu(T_0, 1) + \mu_n(T_0, 1) \right) \tag{23}$$

$$y(x) = \phi(O(net_o) + O_n(net_o)) + \phi_n(O(net_o) + O_n(net_o)) \tag{24}$$

where

x input;
T_i weights associated with the outputs of hidden units;
T_0 bias weight.

Note that the statistical variations at the outputs of the blocks are represented by additive noise terms $\mu_n(x, w)$, $O_n(\mu)$, and $\phi_n(x)$ for the synapse, op-amp, and sigmoid, respectively. The output of each hidden unit is another similar expression. Given a training set, the weights are updated using gradient descent in the backpropagation algorithm. The learning method can be implemented using any synapse and nonlinearity if $\mu(x, w)$, $\partial\mu/\partial x$, $\partial\mu/\partial w$, O, O', ϕ, and ϕ' can be computed. Hence, backpropagation learning can be realized using the analytical expressions of (16), (18), and (21). The noise terms are determined as follows. In each epoch of the training (forward pass), a random number is generated from a normal distribution of zero mean and variance as calculated by (19), (20), and (22) for each synapse, op-amp, and sigmoid block. Those values are considered to be the statistical variations at the outputs.

Training of neural networks for analog hardware implementation utilizing the blocks discussed so far requires special modifications. The input–output values need to be scaled such that they fall within the operational range of the analog circuitry. A further modification in the update rules is applied to favor small weight magnitudes so that the synapses operate in their linear region and the variations in the outputs due to mismatches are less severe. This is done by the *weight decay* technique, which has shown to be effective in the improvement of generalization in the presence of noise [35], [36]. In weight decay, weights with large magnitudes are penalized. At each iteration, there is an effect of pulling a weight toward zero. This not only assures that the blocks operate in their "close to ideal" region, but also decreases the variations. Moreover, it has also been reported in the literature that employing smaller weight magnitudes enhances the fault tolerance of the neural network by distributing the computation evenly to the neurons and synapses [22].

IV. Numerical Experiments

The noisy backpropagation approach employing transistor-based mismatches has been tested on several examples. Five different types of learning have been applied to each problem for comparison purposes. In all of the training experiments, an on-line weight update scheme is employed. The following are the training types employing different types of models for the neural network blocks.

1) *Nominal model,* based on simulations with nominal transistor model parameters (no noise terms added). The behavior of synapses and neurons were modeled based on circuit simulations using these models, and the training was done using these synapse and neuron models.
2) *Monte Carlo without noise,* based on average of simulations with induced mismatches in the model parameters (no noise terms added). The transistor models were perturbed with technology variations, and an average behavior over Monte Carlo simulations was obtained for the synapse and neuron blocks. This behavior was then modeled and observed to be more realistic compared to the behavior obtained with nominal models. Training was performed using these models.
3) *Monte Carlo with noise,* based on average of simulations with induced mismatches in the model parameters and noise terms added with variance obtained from Monte Carlo simulations. In this type of training, not only the models which are the average of many Monte Carlo simulations are utilized but also the variation in the models for the synapse and neuron are calculated. These variations are called "noise" and added during training for robustness.
4) *Measurements and noise,* based on average of simulations with induced mismatches in the model parameters for the synapse and op-amp and measurements for the sigmoid, and noise terms added with variance obtained from theoretical formulas for the synapse and op-amp and measurements for the sigmoid. This approach contains data from various sources. For modeling the neuron behavior, measurements on the chips were used. However, synapses did not contain measurement facilities on the test chips. Hence, simulations had to be employed for the synapses. To model the noise to be added during training, the formulas derived in the previous section were used for the synapse and op-amp block. To demonstrate the flexibility of this method, the variance obtained from the measurements for the sigmoid was used for the noise in the sigmoid block.
5) *Weight noise,* based on uniformly distributed random noise expressed as a percentage added to weights, and gradually reduced to zero as the network converges [16],

TABLE III
SUCCESS RATES IN % FOR XOR (3-BIT PARITY) PROBLEM FOR DIFFERENT TRAINING/FORWARD PASS TYPES

	TRAINING using models obtained from...				
Forward pass using models from...	Nominal simulation	Monte Carlo		Measurements and noise	Weight noise
		without noise	with noise		
Nominal	100(100)	100(100)	100(100)	100(100)	100(100)
Monte Carlo	100(100)	100(100)	100(93)	100(100)	100(100)
Monte Carlo + noise	84(73)	60(59)	100(99)	100(100)	78(74)
Measurements +noise	65(58)	72(66)	94(90)	100(100)	82.5(85)

[22]. The hardware model is based on Monte Carlo simulations for the synapse and on measurements for the sigmoid for the average behavior. This method is included for comparison purposes.

For testing the performance of the above training methodologies, four different simulation sets were performed in addition to chip measurements. The first forward pass is performed using nominal transistor parameters and is just included to test the validity of the training for the first approach. The second forward pass methodology is called Monte Carlo and is done using models based on the average of many Monte Carlo runs. This forward pass corresponds to the case where there are process variations, but all synapses and neurons are actually identical to each other on the chip. The third forward pass methodology is called Monte Carlo + noise and corresponds to using average models and noise during forward pass and is closer to the real world. The fourth and the last forward pass methodology approximates the real chip most closely and is based on using measured behavior for the neuron.

Although design improvements can be carried out regarding transistor geometries of critical components, variations at the outputs of blocks are still inevitable. In order to allow hardware training of analog neural networks without on-chip circuitry or chip-in-the-loop training, proper modeling of those variations is necessary. The examples below indicate that modeling them as additive noise based on transistor-level mismatches and performing the training on software to include the random effects of hardware enhances the performance of the ANN remarkably.

A. *XOR and 3-Bit Parity Problem*

Training has been carried out 30 times for the XOR and 3-bit parity check problems for each type of training mentioned above. The noisy forward passes using models from *Monte Carlo with noise* and *measurements and noise* have also been run for 20 times (i.e., 600 different training-forward pass pairs have been simulated). This allows the derivation of statistically significant results for the problems. For the *weight noise* training, three noise levels have been used: 10%, 20%, and 40%. Then, ten forward runs are performed for each noise level. Regarding the network sizes, a $2:3:1$ network is used for the XOR problem, whereas a $3:6:1$ network is employed for the 3-bit parity problem. The success rates for 600 (30 for nominal and Monte Carlo without noise) forward runs are given in Table III. The numbers in parentheses are the results for the 3-bit parity problem.

In order to test the effectiveness of the training, sample measurements have been carried out on the prototype chips as well.

TABLE IV
SUCCESS RATES OF MEASUREMENTS ON THE CHIPS IN PERCENTAGE FOR XOR PROBLEM

Training Method	Success	Failure	Reject
Nominal	49	29	22
Monte Carlo without noise	61	26	13
Monte Carlo with noise	85	6	9
Measurements and noise	85	10	5
Weight noise	81	14	5

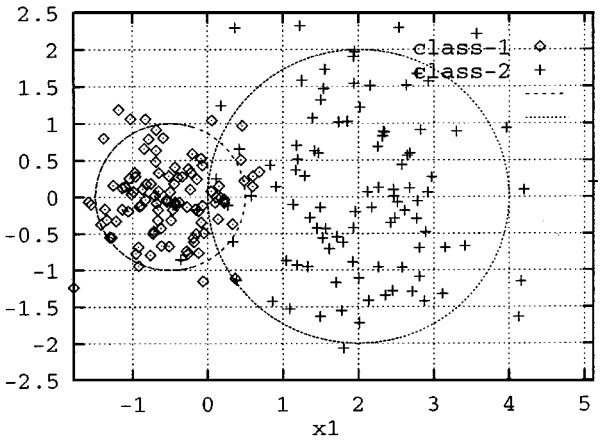

Fig. 8. Data for the classification problem.

For each one of the training methods, five different weight sets are used to test the XOR operation. For each weight set, 30 combinations of chips have been used in the test; that is, the chip for the hidden neurons and the chip for the output neuron have been selected from ten chips in 30 different ways so that the effects of statistical variations among different chips can be observed. In this way, for each training method, a total of 150 XOR networks have been constructed and tested. Although the weights and inputs that are not used have been connected to ground to imply zero input and zero weight, they still contribute to the output due to variations.

The results of the measurements on the chips are given in Table IV. "Reject" denotes the case where the outputs are not "settled" (i.e., the output can not be identified as low or high).

B. *Two-Dimensional Classification Problem*

As a continuous input classification problem, two sets of data points are generated from a normal distribution with the following properties. The one labeled by *class-1* has mean of -0.5 and zero for $x1$ and $x2$ coordinates respectively, where the standard deviation for both coordinates is 0.3. *Class-2*, on the other

TABLE V
SUCCESS RATES IN % FOR CLASSIFICATION PROBLEM FOR DIFFERENT TRAINING/FORWARD PASS TYPES

	TRAINING using models obtained from...				
Forward pass using models from...	Nominal simulation	Monte Carlo		Measurements and noise	Weight noise
		without noise	with noise		
Nominal	94(88)	96(86)	92(88)	90(86)	92(84)
Monte Carlo	46(44)	94(88)	96(92)	94(88)	96(88)
Monte Carlo + noise	32(38)	76(72)	88(92)	90(86)	82(79)
Measurements +noise	29(26)	68(64)	90(88)	95(92)	78(74)

The entries are rates for the training set and the test set in parantheses.

TABLE VI
DISTRIBUTION OF WEIGHTS FOR DIFFERENT TRAINING TYPES IN THE CLASSIFICATION PROBLEM

	TRAINING using models obtained from...				
	Nominal simulation	Monte Carlo		Measurements and noise	Weight noise
		without noise	with noise		
Mean	-0.035	-0.033	-0.015	-0.05	0.001
Standard Deviation	0.14	0.24	0.73	0.54	0.64

hand, has mean of two and zero for $x1$ and $x2$, respectively, where the standard deviation for both coordinates is one. Of the 100 data points generated for each class, half are used as the training set and the other half are used for test. Fig. 8 displays the data points. A $2:14:2$ structure is used for training. All types of training are carried out until the root-mean-square training error dropped below 1%. The training has been carried out 15 times, for each type of training mentioned above. The noisy forward passes using models from *Monte Carlo with noise* and *measurements and noise* have also been performed ten times. The results are summarized in Table V.

C. Discussion

As observed from the results, incorporation of variations into the training enhances the capability of the network strongly. The comparison is performed with respect to the noisy forward pass using the models obtained from measurements, which resemble the actual electrical characteristics of the analog circuitry. For the XOR (3-bit parity) problem, training without noise results in a high level of error. The degradation in the classification problem is more severe. Correct classification rates drop to 29% and 68% for *nominal* and *Monte Carlo* models. Inclusion of weight noise improves the fault tolerance of the network as expected. However, the performance of training with *weight noise* is worse in comparison to training with *Monte Carlo with noise* and *measurements and noise*. This implies that random injection of noise is not capable of compensating the effects of hardware variations fully.

An investigation of the weight distributions also suggests some hints for the enhancement of the fault tolerance. The weights for the training types with noise exhibit a larger standard deviation in comparison to weights obtained without noise terms (see Table VI for the classification problem). This is in agreement with the results of [22] that the "information" is distributed evenly to the weights in training with noise injection.

V. CONCLUSION

In this study, building blocks of an analog neural network—namely, the synapse, op-amp, and sigmoid circuitry—are analyzed for their mismatch characterization. Mismatches in the threshold voltages (V_T) and current factors (β) are considered to be the causes of variations on matched MOS transistor pairs, which result in deviations at outputs of identically designed blocks. Closed-form expressions of statistical variations from the nominal output are derived for these circuits. These theoretical variations are compared to actual measurements obtained from chips. It is evident from the comparison that those variations can be attributed to mismatches. In order to incorporate the variations at the outputs, the backpropagation algorithm is modified. The building blocks are modeled according to their average outputs, and the variations are considered to be noise with certain normal distributions.

Next, the training is carried out using "noisy" backpropagation where the outputs of blocks are calculated in a probabilistic manner taking the noise into account. Sample simulations and measurements are conducted to verify that this method of training allows a higher degree of fault tolerance in the sense that noisy forward pass outputs exhibit better performance over outputs after training without including the variations. The comparison of the modified backpropagation algorithm for different types of modeling also indicates that incorporating variations based on mismatch model obtained through Monte Carlo simulations and actual measurements also coincide. This verifies that noise (variations) can be estimated during the design stage of the circuitry using the data on transistor geometries so that the usage of "noisy" backpropagation can be helpful for achieving a robust training for ANN.

The following conclusions can be drawn from measurement of the chips. The measurement results do not coincide exactly with the simulation results in Table III. This is mainly due to discrepancies between the transistor-level simulation models for ANN and the fabricated circuits. However, they are correlated;

that is, injection of noise during the training considerably improves the performance of the analog neural network. Modeling the variations based on either Monte Carlo simulations or actual measurements on the chip do not differ with respect to the robustness of the training. However, both techniques perform better than random noise injection. It is observed that modeling the statistical variations using theoretical analysis, Monte Carlo simulations, or measurement results yields similar performances, hence allowing the designer to have the flexibility of choosing among these alternatives or their combinations. Even though the success rate has been found to be 100% in the simulation using measurement-based variations, the actual success rate on the test chip has been 85% only. This may be due to the fact that the precision of weights used during the measurements is not as high as the precision of computed weights. Moreover, electrical noise on the setup may have affected the outputs slightly.

The measurements indicate that further work has to be carried out in order to guarantee satisfactory operation in the presence of hardware nonidealities and variations. This can be achieved through improvement of the analog circuitry to decrease mismatch-induced variations and through utilization of a simulation-based training, as offered in [18], thus enabling the training of analog neural networks on software and possibly eliminating the need for chip-in-the-loop or on-chip training.

ACKNOWLEDGMENT

The authors wish to express their gratitude to Dr. E. Alpaydın for his suggestions and many fruitful discussions and to M. R. Becer for his assistance in hardware characterization and measurements.

REFERENCES

[1] B. J. Sheu and J. Choi, *Neural Information Processing and VLSI*. Norwell, MA: Kluwer Academic, 1995.

[2] J. B. Lont and W. Guggenbühl, "Analog CMOS implementation of a multilayer perceptron with nonlinear synapses," *IEEE Trans. Neural Networks*, vol. 3, pp. 457–465, May 1992.

[3] S. M. Gowda, B. J. Sheu, J. Choi, C.-G. Hwang, and J. S. Cable, "Design and characterization of analog VLSI neural network modules," *IEEE J. Solid-State Circuits*, vol. 28, pp. 301–313, Mar. 1993.

[4] J. Donald and L. Akers, "An adaptive neural processing node," *IEEE Trans. Neural Networks*, vol. 4, pp. 413–426, May 1993.

[5] J. Choi, S. H. Bang, and B. J. Sheu, "A programmable analog VLSI neural network processor for communication receivers," *IEEE Trans. Neural Networks*, vol. 4, pp. 484–495, May 1993.

[6] A. Hamilton, S. Churcher, P. J. Edwards, G. B. Jackson, A. F. Murray, and H. M. Reekie, "Pulse stream VLSI circuits and systems: The EPSILON neural network chipset," *Int. J. Neural Syst.*, vol. 4, no. 4, pp. 395–405, Dec. 1993.

[7] F. A. Salam and Y. Wang, "A real-time experiment using a 50-neuron CMOS analog silicon chip with on-chip learning," *IEEE Trans. Neural Networks*, vol. 2, pp. 461–464, July 1991.

[8] R. C. Frye, E. A. Rietman, and C. C. Wong, "Back-propagation learning and nonidealities in analog neural network hardware," *IEEE Trans. Neural Networks*, vol. 2, pp. 110–117, Jan. 1991.

[9] B. K. Dolenko and H. C. Card, "Tolerance to analog hardware of on-chip learning in backpropagation networks," *IEEE Trans. Neural Networks*, vol. 6, pp. 1045–1052, Sept. 1995.

[10] G. Cauwenberghs, "An analog VLSI recurrent neural network learning a continuous-time trajectory," *IEEE Trans. Neural Networks*, vol. 7, pp. 346–361, Mar. 1996.

[11] J. Hertz, A. Krogh, B. Lautrup, and T. Lehmann, "Non-linear back-propagation: Doing back-propagation without derivatives of the activation function," *IEEE Trans. Neural Networks*, vol. 8, pp. 1321–1327, Nov. 1997.

[12] A. J. Montalvo, R. S. Gyurcsik, and J. J. Paulos, "An analog VLSI neural network with on-chip perturbation," *IEEE J. Solid-State Circuits*, vol. 32, pp. 535–543, Apr. 1997.

[13] T. Morie, "Analog VLSI implementation of self-learning neural networks," in *Learning on Silicon: Adaptive VLSI Neural Systems*, G. Cauwenberghs and M. A. Bayoumi, Eds. Norwell, MA: Kluwer Academic, 1999, ch. 10, pp. 213–242.

[14] G. M. Bo, D. D. Caviglia, H. Chiblé, and M. Valle, "Analog VLSI on-chip learning neural network with learning rate adaptation," in *Learning on Silicon: Adaptive VLSI Neural Systems*, G. Cauwenberghs and M. A. Bayoumi, Eds. Norwell, MA: Kluwer Academic, 1999, ch. 14, pp. 305–330.

[15] F. Diotalevi, M. Valle, G. M. Bo, E. Biglieri, and D. D. Caviglia, "An analog on-chip learning circuit architecture of the weight perturbation algorithm," in *Proc. ISCAS2000*, vol. 1, Geneva, Switzerland, May 2000, pp. 419–422.

[16] P. J. Edwards and A. F. Murray, "Fault tolerance via weight noise in analog VLSI implementations of mlp's—A case study with EPSILON," *IEEE Trans. Circuits Syst. II*, vol. 45, pp. 1255–1262, Sept. 1998.

[17] A. Şimşek, M. Civelek, and G. Dündar, "Study of the effects of nonidealities in multilayer neural networks with circuit level simulation," in *Proc. MELECON96*, vol. 1, Bari, Italy, May 1996, pp. 613–616.

[18] İ. Bayraktaroğlu, A. S. Öğrenci, G. Dündar, S. Balkır, and E. Alpaydın, "ANNSyS: An analog neural network synthesis system," *Neural Networks*, vol. 12, no. 2, pp. 325–338, Feb. 1999.

[19] K. Matsuoka, "Noise injection into inputs in back-propagation learning," *IEEE Trans. Syst., Man, Cybern.*, vol. 22, pp. 436–440, May/June 1992.

[20] L. Holmström and P. Koistinen, "Using additive noise in back-propagation training," *IEEE Trans. Neural Networks*, vol. 3, pp. 24–38, Jan. 1992.

[21] A. F. Murray, "Analogue noise-enhanced learning in neural network circuits," *Electron. Lett.*, vol. 27, no. 17, pp. 1546–1548, Aug. 1991.

[22] A. F. Murray and P. J. Edwards, "Enhanced MLP performance and fault tolerance resulting from synaptic weight noise during training," *IEEE Trans. Neural Networks*, vol. 5, pp. 792–802, Sept. 1994.

[23] K. R. Lakshmikumar, R. A. Hadaway, and M. A. Copeland, "Characterization and modeling of mismatch in MOS transistors for precision analog design," *IEEE J. Solid-State Circuits*, vol. 21, pp. 1057–1066, Dec. 1986.

[24] M. J. M. Pelgrom, A. C. J. Duinmaijer, and A. P. G. Welbers, "Matching properties of MOS transistors," *IEEE J. Solid-State Circuits*, vol. 24, pp. 1433–1440, Oct. 1989.

[25] M.-F. Lan and R. Geiger, "Matching performance of current mirrors with arbitrary parameter gradients through the active devices," in *Proc. ISCAS98*, vol. 1, Monterey, CA, 1998, pp. 555–558.

[26] F. Forti and M. E. Wright, "Measurement of MOS current mismatch in the weak inversion region," *IEEE J. Solid-State Circuits*, vol. 29, pp. 138–142, Feb. 1994.

[27] M. Steyaert, J. Bastos, R. Roovers, P. Kinget, W. Sansen, B. Graindourze, A. Pergoot, and E. Janssens, "Threshold voltage mismatch in short-channel MOS transistors," *Electron. Lett.*, vol. 30, no. 18, pp. 1546–1547, Sept. 1994.

[28] T. Serrano-Gotarredona and B. Linares-Barranco, "Cheap and easy systematic CMOS transistor mismatch characterization," in *Proc. ISCAS98*, vol. 2, Monterey, CA, June 1998, pp. 466–469.

[29] C. Michael and M. Ismail, "Statistical modeling of device mismatch for analog mos integrated circuits," *IEEE J. Solid-State Circuits*, vol. 27, pp. 154–165, Feb. 1992.

[30] C. Michael, C. Abel, and M. Ismail, "Statistical modeling and simulation," in *Analog VLSI: Signal and Information Processing*, M. Ismail and T. Fiez, Eds. New York: McGraw-Hill, 1994, ch. 14, pp. 615–655.

[31] Z. Wang, "Design methodology of CMOS algorithmic current A/D converters in view of transistor mismatch," *IEEE Trans. Circuits Syst.*, vol. 38, pp. 660–667, June 1991.

[32] A. Yufera and A. Rueda, "Studying the effects of mismatching and clock-feedthrough in switched-current filters using behavioral simulation," *IEEE Trans. Circuits Syst. II*, vol. 44, pp. 1058–1067, Dec. 1997.

[33] T. Serrano-Gotarredona and B. Linares-Barranco, "An ART1 microchip and its uses in multi-ART1 systems," *IEEE Trans. Neural Networks*, vol. 8, pp. 1184–1194, Sept. 1997.

[34] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*, 3rd ed. ed. New York: McGraw-Hill, 1991.

[35] A. Krogh, "Generalization in a linear perceptron in the presence of noise," *J. Phys. A*, vol. 25, no. 5, pp. 1135–1147, 1992.
[36] A. Krogh and J. A. Hertz, "A simple weight decay can improve generalization," in *Advances in Neural Information Processing*, J. E. Moody, S. J. Hanson, and R. P. Lippmann, Eds. San Mateo, CA: Morgan Kauffman, 1995, vol. 4, pp. 950–957.

Arif Selçuk Öğrenci (S'91–M'00) received the B.S. degree in electrical and electronics engineering and in mathematics the M.S. and Ph.D. degrees in electrical and electronics engineering from Boğaziçi University, İstanbul, Turkey, in 1992, 1995, and 1999, respectively.

From 1992 to 1999, he was a Research Assistant. Currently, he is an Assistant Professor in the Electronics Engineering Department, Kadir Has University, İstanbul. His research interests include analog design automation, neural computation, and analog neural networks.

Günhan Dündar was born in İstanbul, Turkey, in 1969. He received the B.S. and M.S. degrees from Boğaziçi University, İstanbul, in 1989 and 1991, respectively, and the Ph.D. degree from Rensselaer Polytechnic Institute, Troy, NY, in 1993, all in electrical engineering.

Since 1994, he has been with Boğaziçi University, where he is currently an Associate Professor. Between August 1994 and December 1995, he was with the Turkish Navy at the Naval Academy. His research interests include CAD for VLSI, neural networks, and VLSI design.

Sina Balkır received the B.S. degree from Boğaziçi University, İstanbul, Turkey, in 1987 and the M.S. and Ph.D. degrees from Northwestern University, Evanston, IL, in 1989 and 1992, respectively, all in electrical engineering.

Between August 1992 and August 1998, he was with the Department of Electrical and Electronics Engineering, Boğaziçi University, as an Assistant and Associate Professor. Currently, he is with the Department of Electrical Engineering, University of Nebraska-Lincoln. His research interests include CAD of VLSI systems, analog VLSI design automation, and chaotic circuits for spread-spectrum communications.