Effects of Analog-VLSI Hardware on the Performance of the LMS Algorithm

Gonzalo Carvajal[1], Miguel Figueroa[1], and Seth Bridges[2]

[1] Department of Electrical Engineering, Universidad de Concepción, Chile
[2] Computer Science and Engineering, University of Washington, USA
gcarvaja@udec.cl, mfigueroa@die.udec.cl, seth@cs.washington.edu

Abstract. Device mismatch, charge leakage and nonlinear transfer functions limit the resolution of analog-VLSI arithmetic circuits and degrade the performance of neural networks and adaptive filters built with this technology. We present an analysis of the impact of these issues on the convergence time and residual error of a linear perceptron using the Least-Mean-Square (LMS) algorithm. We also identify design tradeoffs and derive guidelines to optimize system performance while minimizing circuit die area and power dissipation.

1 Introduction

Modern embedded and portable electronic systems use adaptive signal processing techniques to optimize their performance in the presence of noise, interference, and unknown signal statistics. Moreover, these systems are also severely constrained in size and power dissipation, making custom-VLSI neural network implementations of these techniques attractive.

Analog VLSI circuits can compute using orders of magnitude less power and die area than their digital counterparts, thus potentially enabling large-scale, portable adaptive systems. Unfortunately, device mismatch, charge leakage, and nonlinear behavior limit the resolution of analog arithmetic circuits so that the learning performance of even small-scale analog-VLSI neural networks rarely exceeds 5-6 bits. Traditional circuit-design techniques can reduce these effects, but they increase power and area and render analog solutions less attractive.

We claim that it is possible to build large-scale neural networks in analog VLSI with good learning performance at low power and area by combining on-chip circuit calibration, design techniques, and the natural adaptation of the algorithm to compensate for the limitations of analog hardware. In this paper, we present an analysis of the performance of the well-known Least-Mean-Square (LMS) algorithm under the constraints of analog VLSI arithmetic. Unlike previous work that uses mainly system simulations [2, 5, 1], we base our analysis on the mathematical properties of the algorithm, obtaining more general results that allow us to derive design guidelines and techniques to improve performance at minimal cost. Using these techniques, we have built a 64-input perceptron that adapts with 9-10 bits of accuracy, uses 0.25mm^2 of die area and dissipates 200μW in a 0.35μm CMOS process [3].

S. Kollias et al. (Eds.): ICANN 2006, Part I, LNCS 4131, pp. 963–973, 2006.

2 Convergence Properties of the LMS Algorithm

An adaptive linear combiner [8] computes the function $y_k = \mathbf{x}_k^{\mathrm{T}}\mathbf{w}_k$, where y_k is the output, and $\mathbf{x}_k = [x_{1k} \cdots x_{nk}]^{\mathrm{T}}$ and $\mathbf{w}_k = [w_{1k} \cdots w_{nk}]^{\mathrm{T}}$ are the n-dimensional input and weight vectors at time k. The weights are chosen to minimize a quadratic function of the error $\epsilon_k = d_k - y_k$, where d_k is an external reference. Both the inputs and the reference are taken from stationary zero-mean random distributions. The Mean Square Error (MSE) is defined as:

$$\xi(\mathbf{w}) = \mathrm{E}[\epsilon_k^2] = \mathrm{E}[d_k^2] - 2\mathbf{p}^{\mathrm{T}}\mathbf{w} + \mathbf{w}^{\mathrm{T}}\mathbf{R}\mathbf{w} \tag{1}$$

where $\mathbf{p} = \mathrm{E}[d_k\mathbf{x}_k]$ represents the correlation between the reference and the input, and $\mathbf{R} = \mathrm{E}[\mathbf{x}_k\mathbf{x}_k^{\mathrm{T}}]$ is the input correlation matrix. The MSE defines a quadratic surface with a single global minimum at the point where its gradient is equal to zero. The Wiener solution defines the optimal value of the weights as $\mathbf{w}^* = \mathbf{R}^{-1}\mathbf{p}$, which yields a minimal MSE of $\xi_{\min} = \mathrm{E}[d_k^2] - \mathbf{p}^{\mathrm{T}}\mathbf{w}^*$.

The LMS algorithm uses gradient descent to iteratively compute an approximation of $\mathbf{w}^*$. The algorithm uses an instantaneous estimation of the MSE gradient $\boldsymbol{\nabla}_k$ as $\hat{\boldsymbol{\nabla}}_k = 2\epsilon_k\mathbf{w}_k = \boldsymbol{\nabla}_k - \boldsymbol{\Psi}_k$, where $\boldsymbol{\Psi}_k$ is the zero-mean estimation noise. An each iteration, the LMS algorithm updates the weights as:

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \mu\hat{\boldsymbol{\nabla}}_k = \mathbf{w}_k + 2\mu\epsilon_k\mathbf{x}_k \tag{2}$$

where the learning rate μ is a parameter which controls stability and convergence time. Widrow shows [8] that $\mathrm{E}[\hat{\boldsymbol{\nabla}}_k] = \boldsymbol{\nabla}$, therefore LMS converges to the Wiener solution $\mathbf{w}^*$ in its mean value. However, the gradient estimation noise results in an oscillation around the solution which depends on the learning rate and the statistics of the input. For a small μ, the MSE at convergence is $\xi_\infty = \xi_{\min} + \mathrm{E}[\mathbf{v}_\infty\mathbf{R}\mathbf{v}_\infty^{\mathrm{T}}]$, where $\mathbf{v}_k = \mathbf{w}_k - \mathbf{w}^*$. The misadjustment is defined as:

$$M = \frac{\text{excess MSE}}{\xi_{\min}} = \frac{\xi_\infty - \xi_{\min}}{\xi_{\min}} \approx \mu\sum_{p=1}^{n}\lambda_p = \mu\ \mathrm{tr}(\mathbf{R}) \tag{3}$$

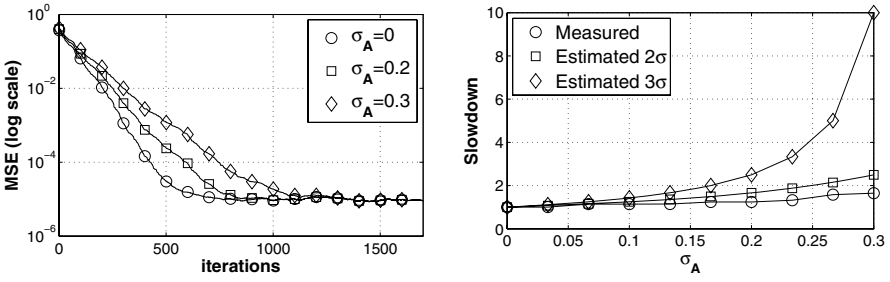
where λ_p are the eigenvalues of $\mathbf{R}$. Eqn. (3) shows that we can control the misadjustment with the learning rate. The MSE decreases as a sum of exponentials, where the time constant of each mode p is given by $\tau_p = 1/(4\mu\lambda_p)$. Therefore, decreasing the learning rate also increases the convergence time of the algorithm.

Hardware implementations of LMS requires multiplication and addition to compute the output (forward path) and weight updates (feedback path), and memory cells to store the weights. Addition is performed by summing currents on common wires and is not subject to device mismatch. The following sections focus on the effects of nonlinear circuits and mismatch on the multipliers, and of charge leakage and limited resolution on the memory cells and weight updates.

3 Effect of Analog Multipliers

We use the following general expression to model the analog multipliers [2]:

$$\mathrm{m}(i_1, i_2) = [a_1\ \mathrm{f}_1(\theta_1, i_1) + \gamma_1] \times [a_2\ \mathrm{f}_2(\theta_2, i_2) + \gamma_2] \tag{4}$$



(a) LMS learning curves (b) Algorithm slowdown

Fig. 1. Effect of gain mismatch on LMS performance. (a) Mismatch in multiplier gains do not affect the MSE at convergence, but do increase the convergence time. (b) Assuming that the minimal gain lies within two standard deviations below the mean provides a good bound for convergence time.

where i_1 and i_2 are the inputs to the multiplier, $\mathrm{f}_1(\cdot)$ and $\mathrm{f}_2(\cdot)$ are saturating, monotonic, and odd nonlinear functions, and a_p, γ_p and θ_p control the gain, offset, and linearity of the multiplier. When $f_p(\theta, x) = \frac{\tanh(\theta x)}{\tanh(\theta)}$, Eqn. (4) models the normalized transfer function of a Gilbert multiplier [6] operating in subthreshold regime. Device mismatch results in variations in the values of a_p, γ_p and θ_p for different multipliers within the same chip. The rest of this section independently analyzes the impact of each factor on the performance of the algorithm.

3.1 Gain Mismatch

Feedback Path: We first analyze the effect of gain mismatch between ideal multipliers rewriting Eqn. (4) as $\mathrm{m}_p(i_1, i_2) = a_p\ i_1 i_2$, where a_p is the gain associated with multiplier p. Mismatched gains in the feedback path modify the gradient estimation implemented by Eqn. (2) to:

$$\mathbf{w}_{k+1} = \mathbf{w}_k + 2\mu\mathbf{A}\epsilon_k\mathbf{x}_k = \mathbf{w}_k + 2\mathbf{U}'\epsilon_k\mathbf{x}_k \tag{5}$$

where $\mathbf{A} = \mathrm{diag}([a_1 \cdots a_n])$ is the diagonal matrix that represents the multiplier gains and $\mathbf{U}' = \mu\mathbf{A}$ represents a synapse-dependent learning rate. Gain mismatch does not modify ϵ_k, therefore $\xi'_{\min} = \xi_{\min}$. The new misadjustment is:

$$M' = \mu \sum_{p=1}^{n} \lambda_p a_p = \mu\ \mathrm{tr}(\mathbf{AR}) \tag{6}$$

We assume that the elements of $\mathbf{A}$ have a Gaussian distribution of unitary mean and variance $\sigma^2_{\mathbf{A}}$, and are uncorrelated with the inputs [7]. In this case, $\mathrm{tr}(\mathbf{AR}) \approx \mathrm{tr}(\mathbf{R})$ for a sufficiently large number of inputs, and thus $\xi'_\infty \approx \xi_\infty$.

Fig. 1 shows results from a simulated 16-input linear perceptron with mismatched gains in the feedback path. Fig. 1(a) shows the evolution of the MSE for different $\sigma_{\mathbf{A}}$. The graph shows that the gain variation does not affect the

MSE after convergence. However, the figure also shows that the convergence time of the algorithm increases as a function of the gain variance. Indeed, the time constant of each new mode p is given by $\tau'_p = 1/(4\mu'_p\lambda_p)$. If we assume that the MSE follows the slowest mode, the slowdown in convergence time is:

$$\frac{\tau'_{\text{conv}}}{\tau_{\text{conv}}} = \frac{\max_p[\tau'_p]}{\max_p[\tau_p]} = \frac{4\min_p[\mu\lambda_p]}{4\min_p[a_p\mu\lambda_p]} \leq \frac{4\mu\lambda_{\min}}{4\mu a_{\min}\lambda_{\min}} = \frac{1}{a_{\min}} \tag{7}$$

The value of $a_{\min}$ is unknown at design time, but we can derive a bound based on the expected distribution of the gains, which in turn can be obtained from previous experimental data or from statistical models of device mismatch [7]. In a Gaussian distribution, 95.4% and 99.7% of the gains will lie within $2\sigma_{\mathbf{A}}$ and $3\sigma_{\mathbf{A}}$ from the mean, respectively. Fig. 1(b) depicts the simulated convergence time, and the bounds estimated using $2\sigma_{\mathbf{A}}$ and $3\sigma_{\mathbf{A}}$ to estimate $a_{\min}$. In practice, it is sufficient to assume $2\sigma_{\mathbf{A}}$, because the bound established in Eqn. (7) conservatively assumes that the convergence time tightly follows the slowest mode, and that the smallest gain is in turn associated with the smallest eigenvalue of $\mathbf{R}$.

Notice that, if the designer has individual control over the learning rate of each synapse after fabrication, then setting $\mu_p = \mu/a_p$ normalizes the effective learning rate and achieves the same convergence time as the original network.

Forward Path: Gain mismatch in the forward-path multipliers modifies the error as $\epsilon'_k = d_k - \mathbf{x}_k^{\mathrm{T}}\mathbf{A}\mathbf{w}$, leading to the following expression for the MSE:

$$\xi' = \mathrm{E}[\epsilon'^2_k] = \mathrm{E}[d_k^2] - 2\mathbf{A}\mathbf{p}^{\mathrm{T}}\mathbf{w} + \mathbf{w}^{\mathrm{T}}\mathbf{A}\mathbf{R}\mathbf{A}\mathbf{w} \tag{8}$$

and the learning rule:

$$\mathbf{w}_{k+1} = \mathbf{w}_k + 2\mu(d_k - \mathbf{x}_k^T\mathbf{A}\mathbf{w}_k)\mathbf{x}_k = \mathbf{w}_k + 2\mathbf{U}'(d_k - \mathbf{x'}_k^{\mathrm{T}}\mathbf{w}_k)\mathbf{x'}_k \tag{9}$$

where $\mathbf{U}' = \mu\mathbf{A}^{-1}$ and $\mathbf{x}'_k = \mathbf{A}\mathbf{x}_k$, Eqn. (9) has the same form as the original LMS learning rule, but with nonuniform learning rates and a modified input with correlation matrix $\mathbf{R}' = \mathbf{A}\mathbf{R}\mathbf{A}^{\mathrm{T}}$. The learning rule of Eqn. (9) converges in its mean to $\mathbf{w}'^* = \mathbf{A}^{-1}\mathbf{w}^*$, and thus from Eqn. (8) $\xi'_{\min} = \xi_{\min}$.

In general, it is difficult to determine the misadjustment from the gains. If we assume that the inputs are decorrelated ($\mathbf{R}$ is diagonal), then the eigenvalues of $\mathbf{R}'$ are $\lambda'_p = a_p^2\lambda_p$, where λ_p are the eigenvalues of $\mathbf{R}$. The misadjustment is:

$$M' = \sum_{p=1}^{n}\frac{\mu}{a_p}\lambda'_p = \mu\sum_{p=1}^{n}\frac{a_p^2\lambda_p}{a_p} = \mu\sum_{p=1}^{n}a_p\lambda_p = \mu\ \mathrm{tr}(\mathbf{A}\mathbf{R}) \tag{10}$$

which is equivalent to Eqn. (6) for gain mismatch in the learning rules. Therefore, mismatched gains in the forward path do not affect the MSE, but increase the learning time as depicted in Eqn. (7). Multiplier gains also modify the Wiener solution to $\mathbf{w}'^* = \mathbf{A}^{-1}\mathbf{w}^*$, so they may also change the effect of initial conditions on convergence time, although modeling this effect is difficult without knowledge of the original solution [4].

3.2 Multiplier Offsets

We rewrite Eqn. (4) as $\mathrm{m}_p(i_1, i_2) = (i_1+\gamma_{1p})(i_2+\gamma_{2p})$, where γ_{1p} and γ_{2p} are the offsets associated with the inputs to multiplier p. The remainder of this section analyzes the effect of each offset separately.

Forward Path: Let $\boldsymbol{\gamma}_{\mathbf{w}} = [\gamma_{w1} \cdots \gamma_{wn}]$ be the vector of weight offsets in the multipliers of the forward path. The instantaneous error is $\epsilon'_k = d_k - \mathbf{x}_k^{\mathrm{T}}\mathbf{w}'_k$ and the MSE is $\xi' = \mathrm{E}[d_k^2] - 2\mathbf{p}^{\mathrm{T}}\mathbf{w}' + \mathbf{w}'^{\mathrm{T}}\mathbf{R}\mathbf{w}'^{\mathrm{T}}$, where $\mathbf{w}' = \mathbf{w} + \boldsymbol{\gamma}_{\mathbf{w}}$. A simple analysis shows that the LMS algorithm converges to the new Wiener solution $\mathbf{w}'^{*} = \mathbf{w}^{*} - \boldsymbol{\gamma}_{\mathbf{w}}$, which compensates for the weight offsets and achieves the same residual MSE as the original network. The eigenvalues of the input are not modified, thus the weight variance is the same and $M' = M$. The weight offsets modify the solution vector $\mathbf{w}^{*}$, so they also affect convergence time [4]. However, because the distribution of the weight offsets is independent of $\mathbf{w}^{*}$, it is not possible to relate the convergence time to the offset variance.

Let now $\boldsymbol{\gamma}_{\mathbf{x}} = [\gamma_{x1} \cdots \gamma_{xn}]$ be the input offsets in the multipliers of the forward path. The error is $\epsilon'_k = d_k - \mathbf{x}'^{\mathrm{T}}_k\mathbf{w}_k$ where $\mathbf{x}'_k = \mathbf{x}_k + \boldsymbol{\gamma}_{\mathbf{x}}$, and $\xi' = \xi + \boldsymbol{\gamma}_{\mathbf{x}}\boldsymbol{\gamma}_{\mathbf{x}}^{\mathrm{T}}$. Because the learning rule operates with a zero-mean $\mathbf{x}$, $\mathrm{E}[\hat{\boldsymbol{\nabla}}'_k] = \boldsymbol{\nabla}$ and the mean value of the weight converges to the original solution $\mathbf{w}^{*}$. The minimal MSE and the misadjustment quadratically increase with the offset:

$$\xi'_{\min} = \xi_{\min} + \mathbf{w}^{*\mathrm{T}}\boldsymbol{\gamma}_{\mathbf{x}}\boldsymbol{\gamma}_{\mathbf{x}}^{\mathrm{T}}\mathbf{w}^{*} \tag{11}$$

$$M' = M + \mu\sum_{p=1}^{n}\gamma_{xp}^2 \tag{12}$$

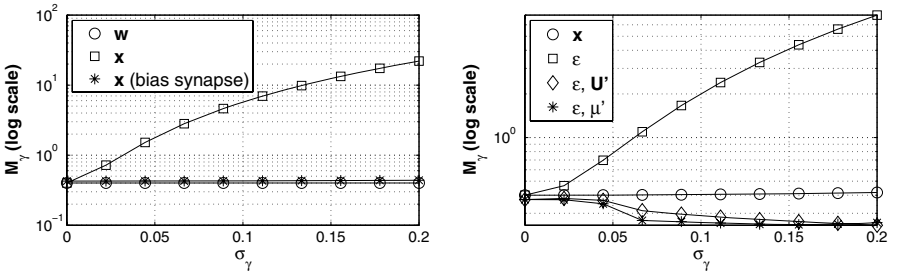
The last term in the Eqn. (11) introduces a large increase in the error which is not controllable with the learning rate. However, we can add *bias synapse* w_0 with offset γ_0 and a constant input c to cancel the accumulated offset at the output. The synapse converges to:

$$w_0 = \frac{-\boldsymbol{\gamma}_{\mathbf{x}}{}^{\mathrm{T}}\mathbf{w}}{c + \gamma_0} \tag{13}$$

which compensates for the accumulated effect of the input offsets, allowing the weights to converge to the Wiener solution and $\xi'_{\min} = \xi_{\min}$.

The bias synapse also affects the weight variance. It can be shown that if $\mathbf{x}_k$ has zero mean, then $\mathrm{tr}(\mathbf{R}') = \mathrm{tr}(\mathbf{R}) + c^2$. Therefore, from Eqn. (12) the misadjustment is $M' = M + \mu(\sum_{p=0}^{n}\gamma_{xp}^2 + c^2)$.

Fig. 2(a) shows simulation results for $M_{\boldsymbol{\gamma}}$ (which we define as the misadjustment with respect to the original $\xi_{\min}$) as a function of the standard deviation of the offsets in the forward-path multipliers. As the figure shows, offsets in the weights do not affect the MSE. Input offsets quadratically increase the MSE, but the addition of a bias synapse successfully compensates for this effect even without the reducing learning rate.



(a) M_γ vs. offset in forward path (b) M_γ vs. offset in learning rule

Fig. 2. Misadjustment versus random multiplier offsets taken from a Gaussian distribution variance σ_γ^2. (a) Forward path: Weight offsets have no effect on the MSE. Input offsets quadratically increase the MSE, but with a bias synapse the effect is almost negligible. (b) Feedback path: input offsets have little effect on the MSE, while error offsets quadratically increase its value. Using learning-rate correction fully compensates for this effect.

Feedback Path: Adding an offset vector $\boldsymbol{\gamma}_\mathbf{x}$ to $\mathbf{x}_k$ in Eqn. (2) yields a new estimated gradient $\hat{\boldsymbol{\nabla}}'_k = 2\epsilon_k(\mathbf{x}_k + \boldsymbol{\gamma}_\mathbf{x})$, which converges to the original Wiener solution $\mathbf{w}^*$. The covariance of the new gradient estimation noise is $\mathrm{cov}[\boldsymbol{\Psi}'_k] = 4\xi_{\min}(\mathbf{R}+\boldsymbol{\gamma}_\mathbf{x}\boldsymbol{\gamma}_\mathbf{x}^\mathrm{T})$ [8]. For small μ and assuming uncorrelated inputs, the gradient noise propagates directly into $\mathbf{v}_k$, leading to a new misadjustment:

$$M' = M + \mu \sum_{p=1}^{n} \gamma_{xp}^2 \tag{14}$$

Eqn. (14) shows that the MSE increases quadratically with the multiplier offsets but this effect is small and can be compensated with the learning rate.

Adding offsets to the error signal ϵ_k at each synapse computing its weight update results in a new estimated gradient $\hat{\boldsymbol{\nabla}}'_k = -2(\epsilon_k\mathbf{I} + \boldsymbol{\Gamma}_\epsilon)\mathbf{x}_k$, where $\boldsymbol{\Gamma}_\epsilon = \mathrm{diag}([\gamma_{\epsilon 1} \cdots \gamma_{\epsilon n}])$ is the diagonal matrix of error offsets. Assuming that $\mathbf{x}$ has zero mean, it is easy to show that $\mathbf{w}'^* = \mathbf{w}^*$, and therefore $\xi'_{\min} = \xi_{\min}$.

However, the new estimated gradient quadratically increases the covariance of $\mathbf{v}_k$ to $\mathrm{cov}[\mathbf{v}'_k] = \mathrm{cov}[\mathbf{v}_k] + \mu\boldsymbol{\Gamma}_\epsilon^2$, where for simplicity we assume that the inputs are uncorrelated. The misadjustment is:

$$M' = M + \frac{\mu \sum_{p=1}^{n} \lambda_p \gamma_{\epsilon p}^2}{\xi_{\min}} \tag{15}$$

Eqn. (15) shows that M' depends quadratically of $\gamma_{\epsilon p}$ and linearly of $\xi_{\min}^{-1}$, so the effect of offsets is much larger than the previous case. We can define a new learning rate that compensates for the misadjustment:

$$\mathbf{U}' = \mu\xi_{\min}(\xi_{\min}\mathbf{I} + \boldsymbol{\Gamma}_\epsilon^2)^{-1} \tag{16}$$

Note that Eqn. (16) defines a different learning rate for each synapse and requires knowledge of the offset values. If the circuit does not support individually

programmable learning rates, the following global rate assumes that most offsets lie within one standard deviation from the mean and yields good results:

$$\mu' = \frac{\mu \xi_{\min}}{(\xi_{\min} + \sigma^2_{\gamma\epsilon})} \tag{17}$$

The simulation results in Fig. 2(b) shows the effects of multiplier offsets in the feedback path. As expected, the effect of input offsets is almost negligible, even without modifying the learning rate. Error offsets have a dramatic impact with the original learning rate. Using Eqns. (16) and (17) to set the learning rate fully compensates for the effect on the MSE.

3.3 Nonlinear Multipliers

Eqn. (4) models an analog multiplier where the parameter θ_p, which varies among multipliers because of device mismatch, modulates the linearity of an odd, saturating, monotonic nonlinear function $f_p(\cdot)$. For example, the normalized transfer function of a Gilbert multiplier [6] is $f_p(\theta_p, x_p) = \tanh(\theta_p x_p)/\tanh(\theta_p)$.

Forward Path: Applying a nonlinear function to the weights in the forward path results in a new error signal $\epsilon'_k = d_k - \mathbf{x}_k^{\mathrm{T}}\ \mathbf{f}(\mathbf{w}_k)$, which yields the MSE:

$$\xi'_k = \mathrm{E}[d_k^2] - 2\mathbf{p}^{\mathrm{T}}\mathbf{f}(\mathbf{w}_k) + \mathbf{f}(\mathbf{w}_k^{\mathrm{T}})\mathbf{R}\mathbf{f}(\mathbf{w}_k) \tag{18}$$

The LMS algorithm converges to the new Wiener solution $\mathbf{w}'^* = \mathbf{f}^{-1}(\mathbf{w}^*)$, and the minimal MSE is $\xi'_{\min} = \xi_{\min}$.

Because the learning rate is small, it is possible to estimate the gradient by linearizing around the Wiener solution:

$$\hat{\boldsymbol{\nabla}}' = \left[d_k - \mathbf{x}_k^{\mathrm{T}} \left(\boldsymbol{f}(\mathbf{w}'^*) + \frac{\partial \boldsymbol{f}}{\partial \mathbf{w}}\Big|_{\mathbf{w}=\mathbf{w}'^*} (\mathbf{w}_k - \mathbf{w}'^*) \right) \right] \mathbf{x}_k \tag{19}$$

Eqn. (19) shows that the estimation noise depends on the values of $\mathbf{w}'^*$ and $[\partial \boldsymbol{f}/\partial \mathbf{w}](\mathbf{w}'^*)$. The worst case occurs when the target weights are at the point where the slope of $f_p(\cdot)$ is maximal, which corresponds to $\mathbf{w}'^* = \mathbf{0}$ for common functions such as $\tanh(\cdot)$. In that case, the estimated gradient is

$$\hat{\boldsymbol{\nabla}}' = (d_k - \mathbf{x}_k^{\mathrm{T}} \mathbf{A}_\theta \mathbf{w}_k)\mathbf{x}_k \tag{20}$$

where $\boldsymbol{A}_\theta = \mathrm{diag}(\partial \boldsymbol{f}/\partial \mathbf{w}|_{\mathbf{w}=0})$ is a diagonal matrix representing the slope of $\boldsymbol{f}(\cdot)$ at the solution. Eqn. (20) reduces the analysis of nonlinear weights to a problem of mismatched multiplier gains. For a normalized Gilbert multiplier, $[\tanh(\theta x)/\tanh(\theta)] > 1$, which increases the MSE. We can achieve $\xi'_\infty = \xi_\infty$ by normalizing the learning rate to the mean gain:

$$\mu' = \frac{\mu}{\mathrm{mean}[\mathbf{A}_\theta]} \tag{21}$$

If the nonlinearity affects the inputs in the forward path, the new error signal is $\epsilon'_k = d_k - \mathbf{f}(\mathbf{x}_k^{\mathrm{T}})\ \mathbf{w}_k$, yielding the new MSE:

$$\xi' = \mathrm{E}[\epsilon_k'^2] = \mathrm{E}[d_k^2] - 2\mathbf{p}'^{\mathrm{T}}\mathbf{w} + \mathbf{w}^{\mathrm{T}}\mathbf{R}'\mathbf{w} \tag{22}$$

where $\mathbf{p}' = \mathrm{E}[d_k\mathbf{f}(\mathbf{x}_k)]$ and $\mathbf{R}' = \mathrm{E}[\mathbf{f}(\mathbf{x}_k)\mathbf{f}(\mathbf{x}_k^{\mathrm{T}})]$. The MSE at the Wiener solution $\mathbf{w}'^* = \mathbf{R}'^{-1}\mathbf{p}'$ is:

$$\xi'_{\min} = \mathrm{E}[d_k^2] - \mathrm{E}[d_k\mathbf{f}(\mathbf{x}_k^{\mathrm{T}})]\mathbf{w}'^* \tag{23}$$

which is always greater than $\xi_{\min}$ when d_k is generated by a linear function. Furthermore, LMS converges to $\mathbf{w}_\infty = (\mathrm{E}[\mathbf{x}_k\mathbf{f}(\mathbf{x}_k^{\mathrm{T}})])^{-1}\mathrm{E}[d_k\mathbf{x}_k^{\mathrm{T}}]$, which differs from $\mathbf{w}'^*$ as a function of the nonlinearity of $\mathbf{f}$.

Feedback Path: Applying a nonlinear function $\mathbf{f}$ to the inputs in the feedback path yields the LMS rule $\mathbf{w}_{k+1} = \mathbf{w}_k + 2\mu\epsilon_k\mathbf{x}'_k$ with $\mathbf{x}'_k = \mathbf{f}(\mathbf{x}_k)$, which still converges to the Wiener solution in its mean value. Therefore, $\xi'_{\min} = \xi_{\min}$, but the nonlinearity of $\mathbf{f}$ affects the variance of $\mathbf{w}_k$ and increases the residual MSE. The misadjustment is given by the modified correlation matrix:

$$M' = \mu\ \mathrm{tr}(\mathrm{E}[\mathbf{f}(\mathbf{x}_k)\mathbf{f}(\mathbf{x}_k)^{\mathrm{T}}]) \tag{24}$$

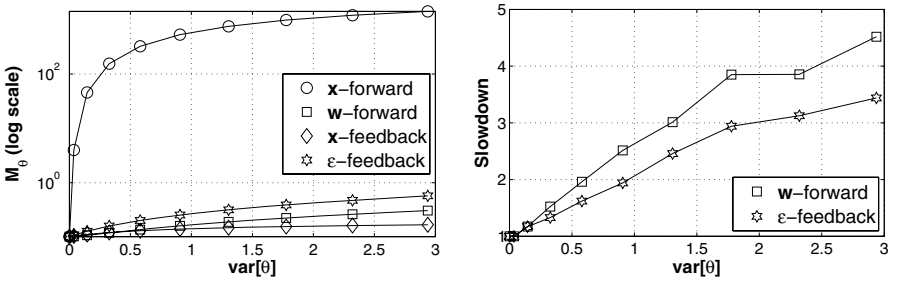
For nonlinear functions such as $[\tanh(\theta\mathbf{x})/\tanh(\theta)]$, $\mathbf{f}(\mathbf{x}) > \mathbf{x}$ and larger values of θ increase the difference between M' and M. In the limit, $\tanh(\theta\mathbf{x})$ saturates and behaves like $\mathrm{sign}(\mathbf{x})$, and we obtain an upper bound for the misadjustment as $M' \leq \mu n$. Note that this also increases the robustness of the algorithm to outliers.

Applying a nonlinear function to the error at each synapse modifies the learning rule to $\mathbf{w}_{k+1} = \mathbf{w}_k + 2\mu\mathbf{E}'_k\mathbf{x}_k$, with $\mathbf{E}'_k = \mathrm{diag}[f_1(\epsilon_k)\cdots f_n(\epsilon_k)]$. Because the error converges to a small value, we can linearize around this point and rewrite the rule as:

$$\mathbf{w}_{k+1} = \mathbf{w}_k + 2\mu(\mathbf{A}_\theta\epsilon_k)\mathbf{x}_k \tag{25}$$

where $\mathbf{A}_\theta = \mathrm{diag}[\partial f_1/\partial\epsilon|_{\epsilon=0}\cdots\partial f_n/\partial\epsilon|_{\epsilon=0}]$. The expression above is equivalent to Eqn. (5) for mismatched gains in the feedback path, therefore the misadjustment increases quadratically with the variance of θ_p and can be controlled with the learning rate using Eqn. (21). Also, for saturating functions such as $[\tanh(\theta\mathbf{x})/\tanh(\theta)]$, the nonlinearity also limits the effects of outliers in the performance of the algorithm.

Fig. 3(a) shows the effect of nonlinear multipliers on the MSE. As predicted by Eqn. (22), nonlinear inputs in the forward path increase the MSE independently of the learning rate. The effect on the forward-path weights and the feedback-path inputs and error is much lower and can be further controlled by reducing the learning rate. Fig. 3(b) shows the effect that this rate reduction has on the convergence time, which is similar to the case of mismatched multiplier gains.



(a) Misadjustment vs. nonlinearity (b) Algorithm slowdown

Fig. 3. Effects of nonlinear multipliers on the performance of LMS. (a) Nonlinear inputs in the forward path increase the MSE when the reference is generated by a linear process, while the effect on the rest of the signals is much lower. (b) It is possible to control the learning rate to trade residual for convergence speed.

4 Effect of Signal Noise and Limited Resolution

Degradation of signal resolution can arise from system noise, charge leakage in capacitor-based weight storage, or quantization effects in digital or mixed-mode implementations. We model the noise $\boldsymbol{\eta}_k$ as a Gaussian process of zero mean and variance σ_η^2. The analysis is equivalent to Section 3.2 with uncorrelated inputs.

Forward Path: In the presence of zero-mean random noise in the weights and inputs of the forward path, the LMS algorithm still converges in the mean to the original Wiener solution $\mathbf{w}^*$. From Section 3.2, we obtain:

$$\xi_{\min}^{\boldsymbol{\eta}_\mathbf{w}} = \xi_{\min} + \mathrm{E}[\boldsymbol{\eta}_k^{\mathrm{T}} \mathbf{x}_k \mathbf{x}_k^{\mathrm{T}} \boldsymbol{\eta}_k] \tag{26}$$

$$\xi_{\min}^{\boldsymbol{\eta}_\mathbf{x}} = \xi_{\min} + \mathbf{w}^{*T} \mathrm{E}[\boldsymbol{\eta}_k \boldsymbol{\eta}_k^{\mathrm{T}}] \mathbf{w}^* = \xi_{\min} + \sigma_\eta^2 \mathbf{w}^{*T} \mathbf{w}^* \tag{27}$$

$$M^{\boldsymbol{\eta}_\mathbf{w}} = M = \mu \mathrm{tr}(\mathbf{R}) \tag{28}$$

$$M^{\boldsymbol{\eta}_\mathbf{x}} = M + \mu \mathrm{tr}(\mathrm{E}[\boldsymbol{\eta}_k \boldsymbol{\eta}_k^{\mathrm{T}}]) = M + \mu n \sigma_\eta^2 \tag{29}$$

Only input-noise modifies M, but both input and weight noise modify $\xi_{\min}$.

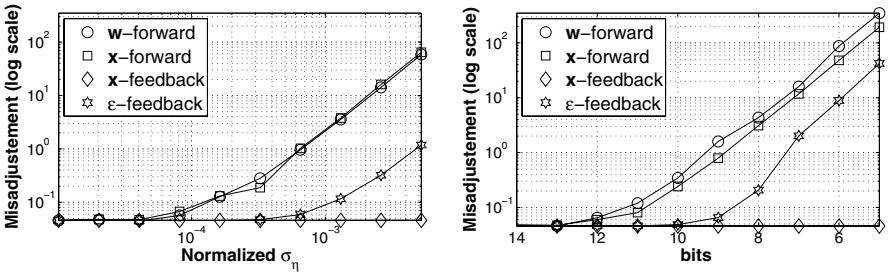
Feedback Path: With random noise in the forward-path signals the algorithm still converges to the Wiener solution and $\xi_{\min}$ is not modified because the output is not directly affected. The new misadjustments are:

$$M^{\eta_\mathbf{x}} = M + \mu \mathrm{tr}(\mathrm{E}[\boldsymbol{\eta}_k \boldsymbol{\eta}_k^{\mathrm{T}}]) = M + \mu n \sigma_\eta^2 \tag{30}$$

$$M^{\eta_\epsilon} = M + (\mu/\xi_{\min}) \sum_{p=1}^{n} \lambda_p \ \mathrm{var}[\eta_{kp}] = M + (\mu/\xi_{\min}) \sigma_\eta^2 \mathrm{tr}\ (\mathbf{R}) \tag{31}$$

The effect error noise on the misadjustment is large, but we can extend Eqn. (17) to derive a new learning rate that guarantees that $\xi_\infty^{\eta_\epsilon} \leq \xi_\infty$:

$$\mu' = \frac{\mu \xi_{\min}}{(\xi_{\min} + \sigma_\eta^2)} \tag{32}$$



(a) M vs. normalized standard deviation (b) M vs. bit width

Fig. 4. Simulated effect of signal noise and digital arithmetic. (a) Noise in the forward path has a strong effect on $\xi_{\min}$ and degrades the learning performance of LMS. The effect of noise in the feedback path is much lower, and can be further reduced with the learning rate. (b) The same analysis applies to the resolution of digital signals.

Fig. 4(a) shows the simulated misadjustment (with respect to the original $\xi_{\min}$) versus the standard deviation of the noise, normalized to the signal range. The simulated plots follow closely the results predicted by the expressions above. Fig. 4(b) shows the misadjustment versus the resolution of digital arithmetic circuits. The bit-widths were chosen match the signal-to-noise ratio used in Fig. 4(a) according to $[\text{bits}] = \log_2\left(\frac{[\text{signal range}]}{6\sigma_\eta}\right)$. The figure shows that the analysis presented in this section can also be used to predict the performance of digital arithmetic circuits implementing parts of the algorithm.

5 Conclusions

We presented an analysis of the effects of analog and mixed-signal hardware on the performance of the LMS algorithm. We derived bounds for the degradation in MSE and convergence time caused by effects such as multiplier offsets, gain mismatch, nonlinear transfer functions, noise, and charge leakage. We discussed design techniques to compensate for these effects such as local and global learning rate adjustment and bias synapses, and quantified their impact on the performance of the algorithm. We are currently extending this work to the design of dimensionality-reduction networks using Principal Components Analysis.

Acknowledgments

This work was financed in part by a FONDECYT grant No. 1040617.

References

1. H. C. Card, B. K. Dolenko, D. K. McNeill, C. R. Schneider, and R. S. Schneider. Is VLSI Neural Learning Robust Against Circuit Limitations? In *IEEE International Conference on Neural Networks*, volume 3, pages 1889–1893, Miami, FL, USA, 1994.

2. B. Dolenko and H. Card. Tolerance to Analog Hardware of On-Chip Learning in Backpropagation Networks. *IEEE Transactions on Neural Networks*, 6(5):1045–1052, 1995.
3. M. Figueroa, E. Matamala, G. Carvajal, and S. Bridges. Adaptive Signal Processing in Mixed-Signal VLSI with Anti-Hebbian Learning. In *IEEE Computer Society Annual Symposium on VLSI*, pages 133–138, Karlsruhe, Germany, 2006. IEEE.
4. A. Flores and B. Widrow. Assessment of the Efficiency of the LMS algorithm Based on Spectral Information. In *Asilomar Conf. on Signals, Systems and Computers*, volume 1, pages 120–124, Pacific Grove, CA, 2004.
5. D. K. McNeill and H. C. Card. Analog Hardware Tolerance of Soft Competitive Learning. In *IEEE International Conference on Neural Networks*, volume 4, pages 2004–2008, Miami, FL, USA, 1994.
6. C. Mead. *Analog VLSI and Neural Systems*. Addison-Wesley, Reading, MA, 1989.
7. M. J. M. Pelgrom, A. C. J. Duinmaijer, and A. P. G. Welbers. Matching Properties of MOS Transistors. *IEEE Journal of Solid-State Circuits*, 24(5):1433–1440, 1989.
8. B. Widrow and E. Walach. *Adaptive Inverse Control*. Prentice-Hall, Upper Saddle River, NJ, 1996.