distribution," *IEEE Trans. Signal Processing*, vol. 43, pp. 1262–1268, May 1995.
[21] ____, "Autoterm representation by the reduced interference distributions: A procedure for kernel design," *IEEE Trans. Signal Processing*, vol. 44, pp. 1557–1564, June 1996.
[22] S. Stankovic, L. Stankovic, and Z. Uskokovic, "On the local frequency, group shift and cross-terms in some multidimensional time-frequency distribution: A method for multidimensional time-frequency analysis," *IEEE Trans. Signal Processing*, vol. 45, pp. 1719–1725, July 1997.
[23] M. Sun, C. C. Li, L. N. Sekhar, and R. J. Sclabassi, "Elimination of cross-components of discrete pseudo Wigner distribution via image processing," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 1989, pp. 2230–2233.
[24] H. Suzuki and F. Kobayashi, "A method of two-dimensional spectral analysis using the Wigner distribution," *Electron. Commun. Jpn.*, vol. 75, no. 1, pp. 1006–1013, 1992.
[25] J. Ville, "Theorie et applications de la notion de signal analitique," *Cables et Transmission*, vol. 2A, pp. 61–74, 1948.
[26] E. Wigner, "On the quantum correction for thermodynamic equilibrium," *Phys. Rev.*, vol. 40, pp. 749–759, 1932.
[27] W. J. Williams, "Reduced interference distributions: Biological applications and interpretations," *Proc. IEEE*, vol. 84, pp. 1264–1280, 1996.
[28] Y. M. Zhu and R. Goutte, "Analysis and comparison of space/spatial-frequency and multiscale methods for texture segmentation," *Opt. Eng.*, vol. 34, no. 1, pp. 269–282, 1995.
[29] Y. M. Zhu, R. Goutte, and M. Amiel, "On the use of a two-dimensional Wigner-Ville distribution for texture segmentation," *Signal Process.*, vol. 30, pp. 205–220, 1993.
[30] Y. M. Zhu, R. Goutte, and F. Peyrin, "The use of a two-dimensional Hilbert transform for Wigner analysis of 2-dimensional real signals," *Signal Process.*, vol. 19, pp. 205–220, 1990.
[31] Y. M. Zhu, F. Peyrin, and R. Goutte, "Equivalence between the two-dimensional real and analytic signal Wigner distribution," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, pp. 1631–1634, Oct. 1989.

Finite-Precision Analysis of the Pipelined Strength-Reduced Adaptive Filter

Manish Goel and Naresh R. Shanbhag

***Abstract*— In this correspondence, we compare the finite-precision requirements of the traditional cross-coupled (CC) and a low-power strength-reduced (SR) architectures. It is shown that the filter block (F block) coefficients in the SR architecture require 0.3 bits more than the corresponding block in the CC architecture. Similarly, the weight-update (WUD) block in the SR architecture is shown to require 0.5 bits fewer than the corresponding block in the CC architecture. This finite-precision architecture is then used as a near-end crosstalk (NEXT) canceller for 155.52 Mb/s ATM-LAN over unshielded twisted pair (UTP) category-3 cable. Simulation results are presented in support of the analysis.**

I. Introduction

Strength reduction is an algebraic transformation that has been proposed [4] to trade off multipliers with adders in a complex multiplication, thereby achieving power reduction. In [7], we proposed the application of *strength reduction* transformation at the algorithmic level to adaptive systems involving complex signals and filters. It was shown in [7] that the strength-reduced (SR) filter enables power savings of 21–25% over the traditional cross-coupled (CC) filter with no loss in performance. However, the application of strength reduction increases the critical path, and, hence, an inherently pipelined SR (PIPSR) architecture was also presented. Furthermore, by trading the throughput gained through pipelining with power supply scaling [4], it was demonstrated that additional power savings of 40–69% are feasible. In this correspondence, we compare the finite-precision requirements of the SR and the PIPSR architectures developed in [7] with that of the CC architecture. It is shown that the precision requirements of the SR and PIPSR architectures are similar to those of the CC architecture. This makes the SR and the PIPSR architectures attractive alternatives to the traditional CC architecture for high bit-rate communications and digital signal processing applications.

In this correspondence, a linear model is employed for coefficient quantization noise. The filter (F) block precision $B_{\mathbf{F}}$ is chosen such that the signal-to-quantization-noise-ratio (*SQNR*) is greater than the desired signal-to-noise ratio SNR_o. The coefficient precision for weight-update (WUD) block $B_{\mathbf{WUD}}$ is determined by applying the *stopping criterion* [3], [5], which puts a lower limit upon the correction term being added to the weight update. This criterion is given by

$$\mu^2 E[|e(n)|^2]\sigma_x^2 \geq 2^{-2B_{\mathbf{WUD}}} \tag{1.1}$$

where

μ	step-size;
$E[\|e(n)\|^2]$	mean-squared error;
σ_x^2	power of the received signal $x(n)$;
$B_{\mathbf{WUD}}$	precision (including sign-bit) of the coefficients in the WUD block.

A more accurate *nonlinear* analysis presented in [1] and [2] can be employed to provide a tighter bound on $B_{\mathbf{WUD}}$. However, the purpose of this paper is to compare the precision requirements for CC and SR architectures, and hence, we employ the analysis in [3]. This analysis provides useful design guidelines for applications such as those in digital subscriber loops where the final step sizes are reasonably large.

We demonstrate an application of the finite-precision SR architecture as a near-end crosstalk (NEXT) canceller for 155.52 Mb/s [6] ATM-LAN over 100 m of unshielded twisted pair category-3 (UTP-3) cable employing 64-CAP (carrierless amplitude/phase) modulation scheme. We present the simulation results for this application in order to determine the precision requirements of various signals and to support the analytical results presented in the correspondence.

The organization of the paper is as follows. In Section II, we present PIPSR adaptive filter architecture. In Section III, we determine the finite-precision requirements of CC, SR, and PIPSR architectures. Finally, in Section IV, the finite-precision architectures are employed as a near-end crosstalk (NEXT) canceller for 155.52 Mb/s ATM-LAN.

II. A Pipelined Strength-Reduced (PIPSR) Adaptive Filter

In this section, we review the strength reduction transformation and development of the PIPSR architecture [7] from the CC architecture. The product of two complex numbers $(a + \jmath b)$ and $(c + \jmath d)$ is given by

$$(a + \jmath b)(c + \jmath d) = (ac - bd) + \jmath(ad + bc).$$

Manuscript received November 12, 1996; revised December 11, 1997. This work was supported by the NSF CAREER Award MIP-9623737. The associate editor coordinating the review of this paper and approving it for publication was Dr. Konstantin Konstantinides.

The authors are with the Coordinated Science Laboratory and Electrical and Computer Engineering Department, University of Illinois at Urbana-Champaign, Urbana, IL 61801 USA (e-mail: mgoel@uivlsi.csl.uiuc.edu; shanbhag@uivlsi.csl.uiuc.edu).

Publisher Item Identifier S 1053-587X(98)03944-0.

A direct-mapped architectural implementation would require a total of four real multiplications and two real additions to compute the complex product. Application of strength reduction involves reformulating the above multiplication as

$$\begin{aligned}(a-b)d+a(c-d)&=ac-bd\\(a-b)d+b(c+d)&=ad+bc\end{aligned}\tag{2.1}$$

where we see that strength reduction reduces the number of multipliers by one at the expense of three additional adders. Typically, multiplications are more expensive than additions, and hence, we achieve an overall savings in hardware. We now present the SR and the PIPSR architectures.

A. Strength-Reduced (SR) Architecture

The SR architecture [7] is obtained by applying strength reduction transformation at the algorithmic level instead of at the multiply-add level. Assume an N-tap adaptive filter implementing a complex LMS algorithm. Assume that the filter input is a complex signal $\mathbf{X}(n)$ given by $\mathbf{X}(n)=\mathbf{X}_r(n)+\jmath\mathbf{X}_i(n)$, where $\mathbf{X}_r(n)$ and $\mathbf{X}_i(n)$ are the real and the imaginary parts of the input signal vector $\mathbf{X}(n)$. Furthermore, if the filter $\mathbf{W}(n)$ is also complex ($\mathbf{W}(n)=\mathbf{c}(n)+\jmath\mathbf{d}(n)$), then the complex LMS algorithm is given by

$$\begin{aligned}e(n)&=d(n)-\mathbf{W}^H(n-1)\mathbf{X}(n)\\\mathbf{W}(n)&=\mathbf{W}(n-1)+\mu e^*(n)\mathbf{X}(n)\end{aligned}\tag{2.2}$$

where

μ	step size;
$d(n)$	desired signal;
$e(n)$	error;
$\mathbf{W}(n)$	coefficient vector.

In addition, $e^*(n)$ represents the complex conjugate of the signal $e(n)$, and $\mathbf{W}^H(n)$ represents the hermitian (complex conjugate transpose) of $\mathbf{W}(n)$.

From (2.2), we see that there are two complex inner products involved. Traditionally, the complex LMS algorithm is implemented via the CC architecture, which is described by

$$y_r(n)=\mathbf{c}^T(n-1)\mathbf{X}_r(n)+\mathbf{d}^T(n-1)\mathbf{X}_i(n)\tag{2.3a}$$
$$y_i(n)=\mathbf{c}^T(n-1)\mathbf{X}_i(n)-\mathbf{d}^T(n-1)\mathbf{X}_r(n)\tag{2.3b}$$
$$\mathbf{c}(n)=\mathbf{c}(n-1)+\mu[e_r(n)\mathbf{X}_r(n)+e_i(n)\mathbf{X}_i(n)]\tag{2.3c}$$
$$\mathbf{d}(n)=\mathbf{d}(n-1)+\mu[e_r(n)\mathbf{X}_i(n)-e_i(n)\mathbf{X}_r(n)]\tag{2.3d}$$

where $e(n)=e_r(n)+\jmath e_i(n)$, and the **F**-block output is given by $y(n)=y_r(n)+\jmath y_i(n)$. Equations (2.3a)–(2.3b) and (2.3c)–(2.3d) define the computations in the **F**-block and the WUD-block, respectively. A direct-mapped implementation of (2.3) would require $8N$ multipliers and $8N$ adders for power-of-two step sizes.

We see that (2.2) has two complex inner products and hence can benefit from the application of strength reduction. Doing so results in the following equations, which describe the **F**-block computations of the SR architecture [7]. We have

$$\begin{aligned}y_1(n)&=\mathbf{c}_1^T(n-1)\mathbf{X}_r(n)\\y_2(n)&=\mathbf{d}_1^T(n-1)\mathbf{X}_i(n)\\y_3(n)&=-\mathbf{d}^T(n-1)\mathbf{X}_1(n)\end{aligned}\tag{2.4a}$$

and

$$\begin{aligned}y_r(n)&=y_1(n)+y_3(n)\\y_i(n)&=y_2(n)+y_3(n)\end{aligned}\tag{2.4b}$$

where $\mathbf{X}_1(n)=\mathbf{X}_r(n)-\mathbf{X}_i(n)$, $\mathbf{c}_1(n)=\mathbf{c}(n)+\mathbf{d}(n)$, and $\mathbf{d}_1(n)=\mathbf{c}(n)-\mathbf{d}(n)$. Similarly, the WUD computation is described by

$$\mathbf{c}_1(n)=\mathbf{c}_1(n-1)+\mu[\mathbf{e}\mathbf{X}_1(n)+\mathbf{e}\mathbf{X}_3(n)]\tag{2.5a}$$
$$\mathbf{d}_1(n)=\mathbf{d}_1(n-1)+\mu[\mathbf{e}\mathbf{X}_2(n)+\mathbf{e}\mathbf{X}_3(n)]\tag{2.5b}$$

where $\mathbf{e}\mathbf{X}_1(n)=2e_r(n)\mathbf{X}_i(n)$, $\mathbf{e}\mathbf{X}_2(n)=2e_i(n)\mathbf{X}_r(n)$, $\mathbf{e}\mathbf{X}_3(n)=e_1(n)\mathbf{X}_1(n)$, $e_1(n)=e_r(n)-e_i(n)$, $\mathbf{X}_1(n)=\mathbf{X}_r(n)-\mathbf{X}_i(n)$. It is easy to show that the SR architecture (see Fig. 1) requires only $6N$ multipliers and $8N+3$ adders for power-of-two step sizes. This is the reason why the SR architecture results in 21–25% power savings [7] over the CC architecture.

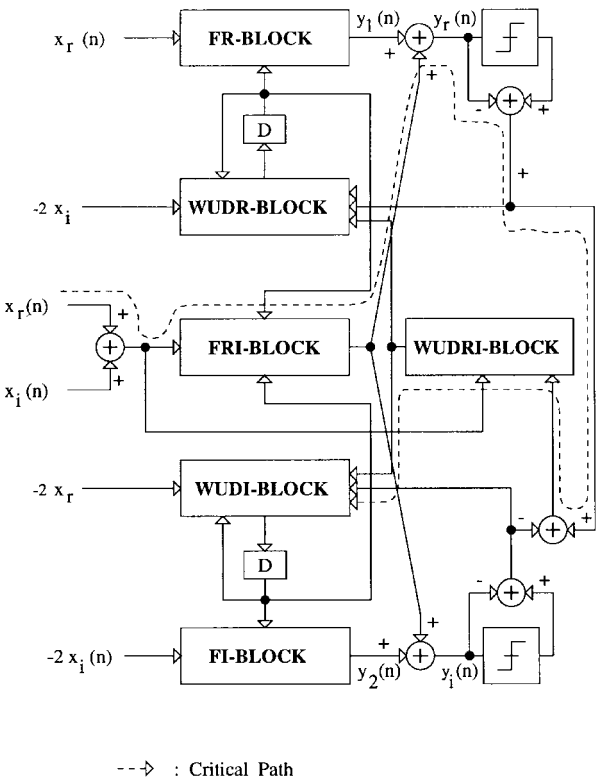

Fig. 1. Block diagram of the SR architecture.

B. Pipelined Strength-Reduced (PIPSR) Architecture

The dotted line in Fig. 1 indicates the critical path of the SR architecture. As explained in [7], both the SR as well as CC architectures are bounded by a maximum possible clock rate due the computations in this critical path. This throughput limitation is eliminated via the application of the relaxed look-ahead transformation [8] to the SR architecture [see (2.4) and (2.5)]. Application of relaxed look-ahead to the SR architecture in (2.4) and (2.5) results in the following equations that describe the **F**-block computations in the PIPSR architecture.

$$\begin{aligned}y_1(n)&=\mathbf{c}_1^T(n-D_2)\mathbf{X}_r(n)\\y_2(n)&=\mathbf{d}_1^T(n-D_2)\mathbf{X}_i(n)\\y_3(n)&=-\mathbf{d}^T(n-D_2)\mathbf{X}_1(n)\end{aligned}\tag{2.6a}$$

$$\begin{aligned}y_r(n)&=y_1(n)+y_3(n)\\y_i(n)&=y_2(n)+y_3(n)\end{aligned}\tag{2.6b}$$

where D_2 is the number of delays introduced before feeding the filter coefficients into the **F**-block. Similarly, the computation of the WUD

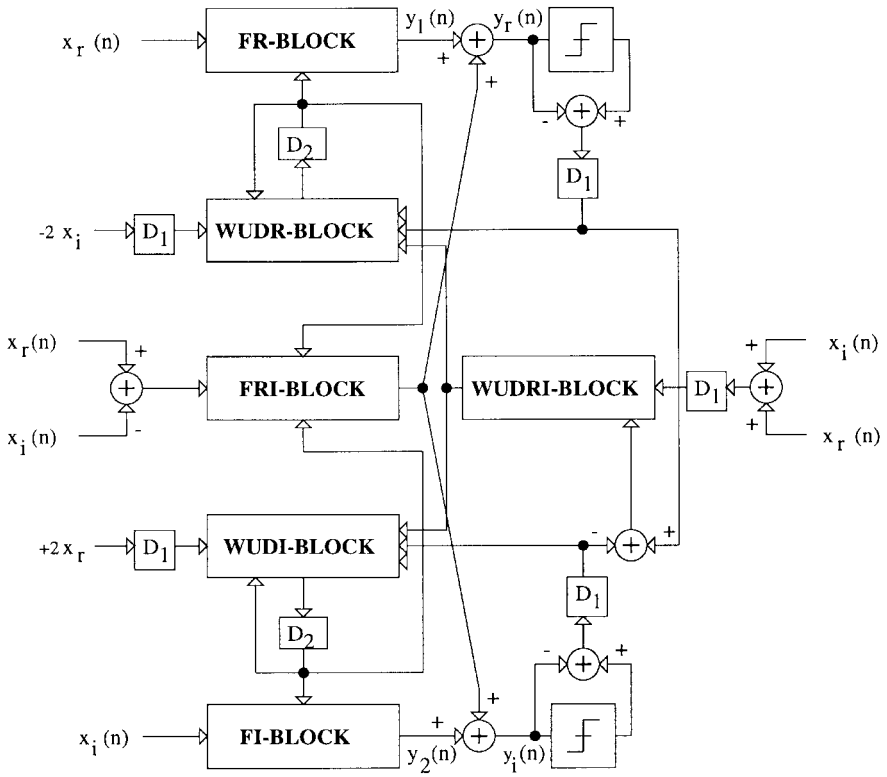

Fig. 2. Block diagram of the PIPSR architecture.

block of the PIPSR architecture are given by

$$\mathbf{c}_1(n) = \mathbf{c}_1(n-D_2) + \mu \sum_{i=0}^{LA-1} [\mathbf{e}\mathbf{X}_1(n-D_1-i) + \mathbf{e}\mathbf{X}_3(n-D_1-i)] \tag{2.7a}$$

$$\mathbf{d}_1(n) = \mathbf{d}_1(n-D_2) + \mu \sum_{i=0}^{LA-1} [\mathbf{e}\mathbf{X}_2(n-D_1-i) + \mathbf{e}\mathbf{X}_3(n-D_1-i)] \tag{2.7b}$$

where $\mathbf{e}\mathbf{X}_1(n)$, $\mathbf{e}\mathbf{X}_2(n)$, and $\mathbf{e}\mathbf{X}_3(n)$ are defined in the previous subsection, $D_1 \geq 0$ are the delays introduced into the error feedback loop, and $0 < \mathrm{LA} \leq D_2$ indicates the number of terms considered in the sum-relaxation. A block level implementation of the PIPSR architecture is shown in Fig. 2, where D_1 and D_2 delays will be employed to pipeline the various operators such as adders and multipliers at a fine-grain level. The high-throughput of the PIPSR architecture can be traded off with supply voltage reduction resulting in additional power savings [7] of 40–69%. Therefore, the PIPSR architecture results in 60–90% power savings as compared to the serial CC architecture.

III. Finite-Precision Requirements

In this section, we will present a comparison of the precision requirements of the CC and SR architectures. We employ linear models [3] for the quantization noise. Further, the **F**-block coefficient precision, $B_{\mathbf{F}}$, is determined by treating **F**-block as a constant coefficient FIR filter and choosing $J_Q \ll J_{\mathrm{fl}}$, where J_Q is the mean squared quantization error, and J_{fl} is the output mean squared error (MSE) for floating-point algorithm. The condition $J_Q \ll J_{\mathrm{fl}}$ guarantees that in case of an equalizer, the bit error rate (BER) of the fixed and floating-point receivers are close to each other.

The stopping criterion [3] is used to determine the WUD-block coefficient precision, $B_{\mathbf{WUD}}$. The stopping criterion is based on the fact that the filter will stop adapting if the correction term ($\mu e(n)x(n)$ in real LMS adaptive filter) drops below *LSB*/2, where *LSB* is least magnitude representable by the chosen precision. The precision assigned should be sufficient for the adaptive filter to converge to the specified MSE, J_o.

A. F-Block Precision

Define $B_{\mathbf{x},\mathbf{y}}$ to be the coefficient precision (including sign-bit) in $\mathbf{x}$ block of $\mathbf{y}$ architecture. Let N be the number of taps in adaptive filter. In addition, let J_{fl} be the infinite-precision MSE (IEEE 754 floating-point format offers resolution up to 10^{-37} and can be safely treated as infinite precision). If σ_d^2 is the power of symbol constellation (or the desired signal), the output SNR is given by $\sigma_d^2/J_{\mathrm{fl}}$.

Now, we determine the quantization error due to finite-precision implementation of the **F**-block. The additional error due to the finite-precision **F**-block implementation is given by $E[\Delta y_r^2(n) + \Delta y_i^2(n)]$, where $\Delta y_r(n)$ and $\Delta y_i(n)$ are the quantization errors in $y_r(n)$ and $y_i(n)$. For CC architecture, it can be seen from (2.3a)–(2.3b) that these errors are given by

$$\Delta y_r(n) = \Delta \mathbf{c}^T(n) X_r(n) + \Delta \mathbf{d}^T(n) X_i(n) \tag{3.1a}$$

$$\Delta y_i(n) = \Delta \mathbf{c}^T(n) X_i(n) - \Delta \mathbf{d}^T(n) X_r(n) \tag{3.1b}$$

where $\Delta \mathbf{c}(n)$ and $\Delta \mathbf{d}(n)$ are the errors due to quantization of coefficients $\mathbf{c}(n)$ and $\mathbf{d}(n)$, respectively. Now, assume that all the quantization errors $\Delta c_i(n)$ and $\Delta d_j(n)$ are mutually independent. In addition, assume a uniform noise model for the quantization error and noise variance of $\sigma^2_{\mathbf{F},\mathbf{CC}} = 2^{-2B_{\mathbf{F},\mathbf{CC}}}/12$. Then, the quantization error J_Q is given by

$$\begin{aligned} J_Q &= E[(\Delta y_r(n))^2 + (\Delta y_i(n))^2] \\ &= E[\Delta \mathbf{c}^T(n) \mathbf{R} \Delta \mathbf{c}(n) + \Delta \mathbf{d}^T(n) \mathbf{R} \Delta \mathbf{d}(n)] \\ &= 2\sigma^2_{\mathbf{F},\mathbf{CC}} \mathrm{tr}(\mathbf{R}) = 2N\sigma^2_{\mathbf{F},\mathbf{CC}} \sigma_x^2 \end{aligned} \tag{3.2}$$

where $\mathbf{R} = E[\mathbf{X}(n)\mathbf{X}^H(n)]$ is the input correlation matrix. Now, we can make the performance of the finite-precision **F**-block arbitrarily close to that of the infinite-precision **F**-block by choosing a factor $\alpha \ll 1$ such that

$$J_Q = \alpha J_{\mathrm{fl},\mathbf{CC}} \tag{3.3}$$

where $J_{\mathrm{fl},\mathbf{CC}}$ is the floating-point MSE for the CC implementation. From (3.2), (3.3), and the definition of $\sigma^2_{\mathbf{F},\mathbf{CC}}$, it can be seen that the **F**-block precision is given by

$$\begin{aligned} B_{\mathbf{F},\mathbf{CC}} &> \frac{1}{2}\log_2\left(\frac{N\sigma_x^2}{6\alpha J_{\mathrm{fl},\mathbf{CC}}}\right) \\ &= \frac{1}{2}\log_2\left(\frac{N\sigma_x^2}{6\alpha\sigma_d^2}\right) + \frac{\mathrm{SNR}_{\mathrm{fl},\mathbf{CC}}(\mathrm{dB})}{6} \end{aligned} \tag{3.4}$$

where $\mathrm{SNR}_{\mathrm{fl},\mathbf{CC}} = \sigma_d^2/J_{\mathrm{fl},\mathbf{CC}}$ is the floating-point SNR for the CC implementation.

From (3.4), we see that lower the value of α, higher is the precision requirement. By choosing $\alpha \ll 1$, we can make the finite-precision performance very close to the performance of the infinite-precision algorithm.

The **F**-block precision for the SR architecture can be similarly determined from (2.4). The quantization error J_Q due to finite-precision implementation of the **F**-block in the SR architecture is given by

$$\begin{aligned} J_Q &= E[(\Delta y_r(n))^2 + (\Delta y_i(n))^2] \\ &= E\big[\Delta\mathbf{c}_1^T(n)\mathbf{X}_r(n)\mathbf{X}_r^T(n)\Delta\mathbf{c}_1(n) \\ &\quad + \Delta\mathbf{d}_1^T(n)\mathbf{X}_i(n)\mathbf{X}_i^T(n)\Delta\mathbf{d}_1(n)\big] + 2E[\Delta\mathbf{d}^T(n)(\mathbf{X}_r(n) \\ &\quad - \mathbf{X}_i(n))(\mathbf{X}_r(n) - \mathbf{X}_i(n))^T\Delta\mathbf{d}(n)] \\ &= 3N\sigma^2_{\mathbf{F},\mathbf{SR}}\sigma_x^2. \end{aligned} \tag{3.5}$$

Therefore, for a given α, the **F**-block precision $B_{\mathbf{F},\mathbf{SR}}$ is obtained from (3.3), (3.5), and the fact that $\sigma^2_{\mathbf{F},\mathbf{SR}} = 2^{-2B_{\mathbf{F},\mathbf{SR}}}/12$, as

$$B_{\mathbf{F},\mathbf{SR}} > \frac{1}{2}\log_2\left(\frac{N\sigma_x^2}{4\alpha\sigma_d^2}\right) + \frac{\mathrm{SNR}_{\mathrm{fl},\mathbf{SR}}(\mathrm{dB})}{6} \tag{3.6}$$

where $\mathrm{SNR}_{\mathrm{fl},\mathbf{SR}} = \sigma_d^2/J_{\mathrm{fl},\mathbf{SR}}$ is the floating-point SNR for the SR implementation. For the infinite-precision implementation, both the CC and the SR architectures give the same performance. Therefore, $\mathrm{SNR}_{\mathrm{fl},\mathbf{CC}} = \mathrm{SNR}_{\mathrm{fl},\mathbf{SR}}$. It can be seen from (3.4) and (3.6) that for the same value of α, the coefficient precision of **F**-block for CC and SR architectures is related by

$$B_{\mathbf{F},\mathbf{SR}} = B_{\mathbf{F},\mathbf{CC}} + 0.3. \tag{3.7}$$

This shows that the **F**-block in the SR architecture requires at the most one bit more than in the CC architecture. The quantization error due to finite-precision implementation of **F**-block in PIPSR architecture [see (2.6)] is same as that of the SR architecture because both architectures involve same computations in the **F**-block. Therefore, for given α, **F**-block precision in PIPSR architecture is also given by

$$B_{\mathbf{F},\mathbf{PIPSR}} > \frac{1}{2}\log_2\left(\frac{N\sigma_x^2}{4\alpha\sigma_d^2}\right) + \frac{\mathrm{SNR}_{\mathrm{fl},\mathbf{PIPSR}}(\mathrm{dB})}{6}. \tag{3.8}$$

B. WUD-Block Precision

The finite-precision WUD-block can be analyzed by using linear model for coefficient quantization noise. Then, $B_{\mathbf{WUD}}$ is chosen based on the *stopping criterion* [3]. For CC architecture, the correction terms are given by (2.3c)–(2.3d). Therefore, the adaptive filter will stop converging if the following two conditions are simultaneously satisfied.

$$\mu|e_r(n)x_r(n) + e_i(n)x_i(n)| < 2^{-B_{\mathbf{WUD},\mathbf{CC}}} \tag{3.9a}$$

$$\mu|e_r(n)x_i(n) - e_i(n)x_r(n)| < 2^{-B_{\mathbf{WUD},\mathbf{CC}}}. \tag{3.9b}$$

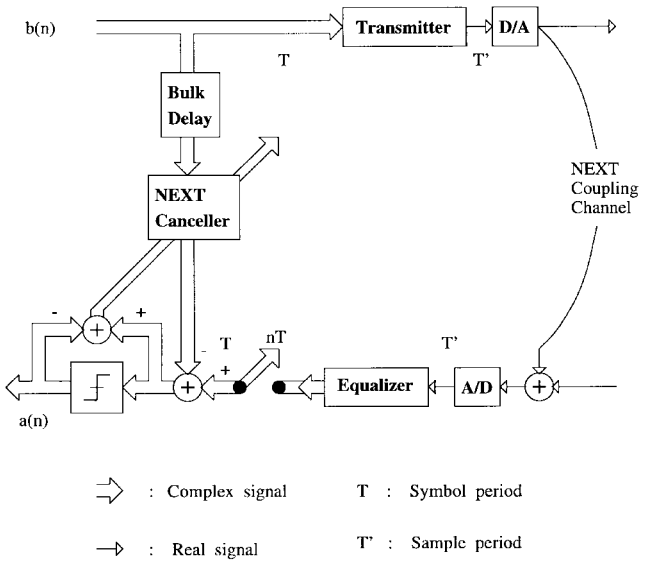

Fig. 3. 155.52 Mb/s ATM-LAN transceiver.

Squaring (3.9a) and (3.9b) and adding and using stochastic estimates for the resulting terms, we get

$$\begin{aligned} &\frac{1}{2}\mu^2 E\left[e_r^2(n) + e_i^2(n)\right]E\left[x_r^2(n) + x_i^2(n)\right] \\ &\quad < 2^{-2B_{\mathbf{WUD},\mathbf{CC}}}. \end{aligned} \tag{3.10}$$

Therefore, if $J_o = E[e_r^2(n) + e_i^2(n)]$ is the desired MSE level, the required WUD-block precision is obtained as

$$\begin{aligned} B_{\mathbf{WUD},\mathbf{CC}} &\geq \frac{1}{2}\log_2\left(\frac{2}{\mu^2 J_o\sigma_x^2}\right) \\ &= \frac{1}{2}\log_2\left(\frac{2}{\mu^2\sigma_x^2\sigma_d^2}\right) + \frac{\mathrm{SNR}_o(\mathrm{dB})}{6} \end{aligned} \tag{3.11}$$

where $\mathrm{SNR}_o = \sigma_d^2/J_o$. Typically, SNR_o is chosen to equal $\mathrm{SNR}_{\mathrm{fl},\mathbf{CC}}$.

A similar expression can be found from (2.5) for the coefficient precision of the WUD-block in the SR architecture. The stopping criterion in this case is given by

$$\mu|ex_1(n) + ex_3(n)| < 2^{-B_{\mathbf{WUD},\mathbf{SR}}} \tag{3.12a}$$

$$\mu|ex_2(n) + ex_3(n)| < 2^{-B_{\mathbf{WUD},\mathbf{SR}}} \tag{3.12b}$$

where $ex_1(n)$, $ex_2(n)$, and $ex_3(n)$ are the elements of the vectors $\mathbf{eX}_1(n)$, $\mathbf{eX}_2(n)$, and $\mathbf{eX}_3(n)$ [see (2.5)], respectively. Squaring (3.12a) and (3.12b), adding and using stochastic estimates, we get

$$\begin{aligned} &\frac{1}{2}\mu^2 E\left[e_r^2(n) + e_i^2(n)\right]E[(x_r(n) + x_i(n))^2 \\ &\quad + (x_r(n) - x_i(n))^2] < 2^{-2B_{\mathbf{WUD},\mathbf{SR}}}. \end{aligned} \tag{3.13}$$

On simplification, the coefficient precision of the WUD block in SR architecture is obtained as

$$B_{\mathbf{WUD},\mathbf{SR}} \geq \frac{1}{2}\log_2\left(\frac{1}{\mu^2\sigma_x^2\sigma_d^2}\right) + \frac{\mathrm{SNR}_o(\mathrm{dB})}{6}. \tag{3.14}$$

Comparing (3.11) and (3.14), we see that the precision requirements for WUD-block in the SR architecture are 0.5 bits less than that of the CC architecture. This is indeed an attractive result given that the SR architecture also enables power savings of 21–25% [7].

The precision requirements for WUD block of PIPSR architecture [see (2.7)] can be determined by replacing μ in (3.14) by $\mu\mathrm{LA}$. In most of the designs, we choose $\mathrm{LA} = 1$ or 2 to minimize the hardware overhead. Therefore, we conclude that the finite-precision requirements of the PIPSR architecture are similar to that of the SR architecture.

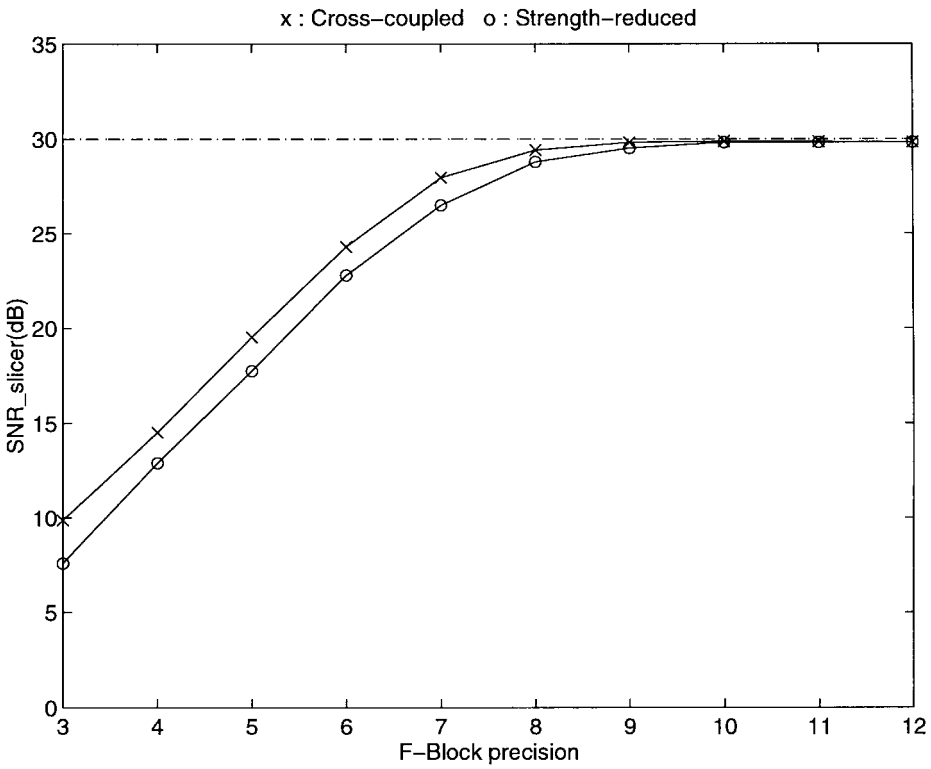

Fig. 4. Performance versus F-block precision.

IV. Application to 155.52 Mb/s ATM-LAN

In this section, we employ the finite-precision PIPSR architecture as a NEXT canceller for the 155.52 Mb/s ATM-LAN [6] over UTP-3 cable and present the simulation results. The basic transceiver block diagram is presented in Fig. 3. Two pairs are used for the dual-simplex transmission, where each direction of transmission uses a different pair though in the same cable. Two main impairments are propagation loss and NEXT. We assume the worst-case EIA/TIA model for the simulations presented in this paper, where the channel and NEXT models are obtained from [6].

A. 155.52 Mb/s ATM-LAN Transceiver

For the details on the transmitter and the equalizer block diagrams in Fig. 3, refer to [6]. The transmitter consists of a 64-CAP (6 bits/symbol) encoder that results in a bandwidth of $155.52/6 = 25.92$ Mbaud. The FCC Class B requirements specify the maximum signal frequency $f_{\max}$ of 30 MHz. This implies that the shaping filters have an excess bandwidth with $\alpha = \frac{f_{\max}-1/T}{1/T} = 0.15$ or 15%. The sampling frequency of the D/A and A/D is three times the baud-rate of 25.92 Mbaud or 77.76 MHz. At the receiver (see Fig. 3), the received signal is distorted further due to the superimposition of the NEXT signal. This composite signal is processed by the fractionally spaced linear equalizer (FSLE), which is a pair of adaptive filters. In addition, the local transmitted symbols are passed through a complex adaptive NEXT canceller, which tries to cancel the effect of the NEXT in the received signal. Note that the NEXT canceller operates at the baud rate, which is three times lower than the sampling frequency of the FSLE.

In this section, we will employ the fixed-point architectures presented in this paper as NEXT cancellers. The simulation procedure we adopt is to let the FSLE converge in presence of the distortion introduced by 100 m UTP-3 cable in the absence of NEXT. Next, the coefficients of the FSLE are frozen, and the local transmitter and the NEXT canceller are activated. This introduces the NEXT in the received signal, which the NEXT canceller attempts to cancel with an appropriately chosen bulk delay. In this section, we determine the precision requirements of CC, SR, and PIPSR NEXT canceller architectures. We will assume that the PIPSR NEXT canceller has been obtained by pipelining the serial SR architecture to the pipelining level of 105 by using $D_1 = 109$, $D_2 = 5$, and LA $= 2$ (see [7] for more details regarding this choice of D_1, D_2, and LA). For this application, $N = 32$, $\sigma_x^2 = 42$, $\sigma_d^2 = 42$, and $\text{SNR}_o = 29.75$ dB (corresponding to probability of error of 10^{-10}) are chosen. We get SNR_{fl} of 29.85 dB from the floating-point simulations. Further, we employ $\alpha = 0.025$ for finding the F-block precisions.

B. Simulation Results

F-block precisions can be determined by employing (3.4) for CC architecture and (3.6) for SR. On substituting above given parameters, we obtain $B_{\mathbf{F},\mathbf{CC}} = 8.87$ and $B_{\mathbf{F},\mathbf{SR}} = 9.17$. These values are supported via simulation results plotted in Fig. 4, which shows the variation of $\text{SNR}_{\text{slicer}}$ with the F-block precision in CC, and SR architectures. Desired SNR is attained at about 9 bits of precision for CC architecture and 10 bits for SR architecture. Fig. 4 also confirms that the coefficient precision required in F-block for the SR architecture is at the most 1 bit more as compared with the CC architecture. Recall that this conclusion was also obtained from (3.7).

Similarly, the coefficient precision in the WUD block can be determined by employing (3.11) for CC and (3.14) for SR. For proper convergence, μ was chosen to be 0.0007 for CC and SR implementations. The $B_{\mathbf{WUD}}$ precisions are determined from (3.11) and (3.14) to be $B_{\mathbf{WUD},\mathbf{CC}} = 9.45$ and $B_{\mathbf{WUD},\mathbf{SR}} = 8.95$. This is confirmed by simulation results in Fig. 5, where the desired performance is achieved with 9 bits of precision for both CC and SR architectures.

We now consider the coefficient precisions in the PIPSR architecture. The F-block precision $B_{\mathbf{F},\mathbf{PIPSR}}$ is obtained by substituting

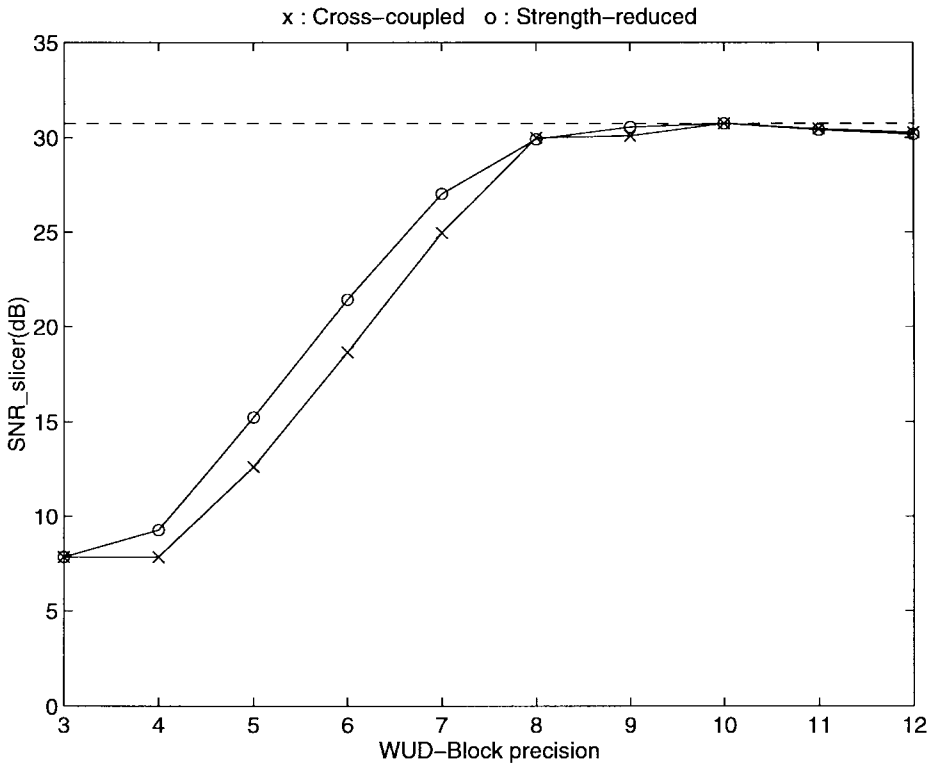

Fig. 5. Performance versus WUD-block precision.

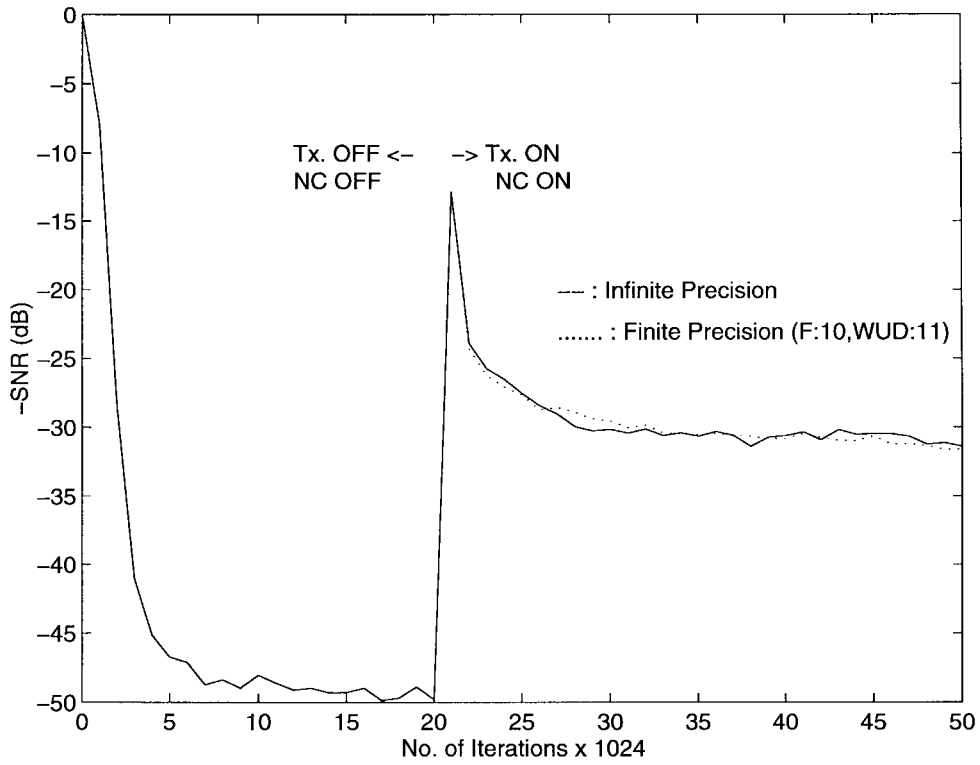

Fig. 6. Convergence plot for the PIPSR architectures.

the above specified parameters in (3.6). The WUD-block coefficient precision requirements are obtained from (3.14) by replacing μ by μLA. For the desired J_o value chosen above, we get $B_{\mathrm{F,PIPSR}} > 9.17$ and $B_{\mathrm{WUD,PIPSR}} > 10.1$ bits. The finite-precision algorithm is simulated for $B_{\mathrm{F,PIPSR}} = 10$ and $B_{\mathrm{WUD,PIPSR}} = 11$. In Fig. 6, we show the convergence plots for the fixed-point PIPSR architecture and the floating-point PIPSR architecture. The steady-state SNR of the the fixed-point algorithm match very closely to that of the floating-point algorithm.

Therefore, we conclude that PIPSR architecture is a viable low-power solution for 155.52 Mb/s ATM-LAN and other digital subscriber loop applications.

REFERENCES

[1] J. C. M. Bermudez and N. J. Bershad, "A nonlinear analytical model for the quantized LMS algorithm—The arbitrary step size case," *IEEE Trans. Signal Processing*, vol. 44, pp. 1175–1183, May 1996.

[2] N. J. Bershad and J. C. M. Bermudez, "A nonlinear analytical model for the quantized LMS algorithm—The powers-of-two case," *IEEE Trans. Signal Processing*, vol. 44, pp. 2895–2900, Nov. 1996.

[3] C. Caraiscos and B. Liu, "A roundoff error analysis of the LMS adaptive algorithm," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-32, pp. 34–41, Feb. 1984.

[4] A. Chandrakasan *et al.*, "Minimizing power using transformations," *IEEE Trans. Comput.-Aided Design*, vol. 14, pp. 12–31, Jan. 1995.

[5] R. D. Gitlin, J. F. Hayes, and S. B. Weinstein, *Data Communications Principles*, New York: Plenum, 1992.

[6] G. H. Im and J. J. Werner, "Bandwidth-efficient digital transmission up to 155 Mb/s over unshielded twisted-pair wiring," *IEEE J. Select. Areas Commun.*, vol. 13, pp. 1643–1655, Dec. 1995.

[7] N. R. Shanbhag and M. Goel, "Low-power adaptive filter architectures and their application to 51.84 Mb/s ATM-LAN," *IEEE Trans. Signal Processing*, vol. 45, pp. 1276–1290, May 1997.

[8] N. R. Shanbhag and K. K. Parhi, "Relaxed look-ahead pipelined LMS adaptive filters and their application to ADPCM coder," *IEEE Trans. Circuits Syst.*, vol. 40, pp. 753–766, Dec. 1993.

On Hetero-Associative Neural Networks and Adaptive Interference Cancellation

Chanchal Chatterjee and Vwani P. Roychowdhury

***Abstract*— We discuss two novel adaptive algorithms for *generalized* eigendecomposition that are derived from a two-layer linear feedforward hetero-associative neural network. In addition, we provide a rigorous convergence analysis of the adaptive algorithms by using *stochastic approximation theory*. Finally, we use these algorithms for on-line multiuser access interference cancellation in code-division-multiple-access-based cellular communications. Numerical simulations are reported to demonstrate their rapid convergence.**

***Index Terms*—Adaptive generalized eigen-decomposition, on-line interference cancellation.**

I. INTRODUCTION

We study two novel adaptive algorithms for generalized eigendecomposition that are derived from a two-layer linear hetero-associative neural network. We discuss applications of these algorithms in an adaptive beamforming example to solve the near–far problem in code-division-multiple-access (CDMA) based cellular communications. Note that the well-studied topic of principal component analysis [1] provides adaptive algorithms for eigendecomposition of a correlation matrix A, which is the limit matrix of a single sequence of random matrices. We, on the other hand, provide adaptive algorithms for generalized eigendecomposition of a matrix pair (A, B), which are the limit matrices of two sequences of random matrices.

A. Adaptive Beamforming for CDMA Based Cellular: A Case Study

As an example of an application that requires adaptive generalized eigendecomposition, we study the problem of on-line cochannel interference cancellation to solve the near–far problem in CDMA-based cellular communications. A number of nonadaptive methods have been proposed [3], [5]–[7] to solve this problem. A common scheme uses multiple (say, m) antennas to receive the signal at the base. The output of each antenna is put through a matched filter corresponding to the code of the desired user [7]–[9] (see Fig. 1). Although there are many methods to extract the desired signal at the base, we next consider a particular method that has been studied by several researchers [8], [9]. In the IS-95 standard, the bit period of the signal is on the order of 100 μs in duration. Within each bit period, there is roughly a 10 μs or so interval during which the desired filtered signal occurs. During this period of time, the signal plus interference correlation matrix A is estimated. In the remaining 90 μs or so, we estimate the interference correlation matrix B.

Given the correlation matrices A and B of signal plus interference and interference, respectively, we compute the weight vector $\boldsymbol{w}$ of a transversal filter such that we maximize the signal-to-interference plus noise ratio (SINR) expressed as $\max_{\boldsymbol{w}}\{\text{SINR} = (\boldsymbol{w}^H A \boldsymbol{w}/\boldsymbol{w}^H B \boldsymbol{w})\}$.[1] The solution to this problem is the generalized eigenvector of the matrix pencil (A, B) corresponding to the largest generalized eigenvalue. Although this computation appears to be relatively uncomplicated, for typical urban multipath time delay spreads, the correlation matrices A and B are of rather large dimension, even when the number of receiving antennas are relatively small. For example, if we sample two times per microsecond for a 10 μs time delay spread of the desired signal, and if we have eight receiving antennas, the resulting space-time correlation matrices A and B are of dimension 160×160. Since generalized eigendecomposition requires $O(n^3)$ computation, this computation is quite intensive.

In an attempt to simplify this problem, an alternative method [8] constructs a lower dimensional $m \times m$ matrix pencil (A, B) for an m-antenna problem. We next compute the first $p(< m)$ generalized eigenvectors of the matrix pencil (A, B) corresponding to the p largest generalized eigenvalues. The p weight vectors transform the initial m-dimensional antenna space to a p-dimensional beam space. The first principal generalized eigenvector is now computed in the reduced dimensional beam space with lower dimensional spatial correlation matrices (A, B).

In all of the above-mentioned schemes, interference cancellation can be achieved by first computing the matrix pencil (A, B) after collecting all of the samples and then the application of a numerical procedure [2], i.e., by working in a *batch* fashion. If the principal generalized eigenvectors are computed in a batch mode, the time delay needed to make a decision would not only include bit times needed to average the spatial correlation matrices but the subsequent time required to compute the generalized eigenvectors as well. In addition, the batch mode operation will not, in general, exploit the fact that there is a gradual time variation of the weight vector $\boldsymbol{w}$ in a urban mobile environment and that we need to recompute $\boldsymbol{w}$ after every few (say 4) bits. In order to reduce this computation and obtain effective interference cancellation, an adaptive (i.e., on-

Manuscript received July 24, 1997; revised October 13, 1997. This work was supported in part by NSF Grants ECS-9308814 and ECS-9523423. The associate editor coordinating the review of this paper and approving it for publication was Prof. Yu-Hen Hu.

C. Chatterjee is with GDE Systems Inc., San Diego, CA 92127 USA.

V. P. Roychowdhury is with the Electrical Engineering Department, University of California, Los Angeles, CA 90095 USA.

Publisher Item Identifier S 1053-587X(98)03937-3.

[1] Superscript H denotes Hermitian transpose matrix operation.