Improved Fault Coverage for Adaptive Fault Tolerant Filters

J. Jiang, C. D. Schmitz, B. A. Schnaufer and W. K. Jenkins
Department of Electrical and Computer Engineering and
The Coordinated Science Laboratory
University of Illinois
Urbana, Illinois 61801 USA

Abstract - Adaptive fault tolerance (AFT) has been developed recently for achieving reliable performance of FIR adaptive filters in the presence of certain types of hardware failures. Previous studies limited the use of AFT to one-dimensional FIR filter structures with simple "stuck-at" fault models that account for errors introduced into the adaptive process by faulty coefficients that cease to adjust properly. This paper considers a broader class of hardware errors which, in addition to "stuck-at" errors in the multiplier inputs, can be modeled by errors in the outputs of both multipliers and adders. By including a simple circuit that removes erroneous induced constants in the output of the adaptive filter, adaptive fault tolerance can be extended to a broader class of hardware failures.

1. Introduction

Adaptive Fault Tolerance (AFT) is a fault tolerant design approach applicable to adaptive systems that makes use of the inherent adaptive process as an automatic fault tolerance mechanism [1-2]. Under normal operating conditions adaptive systems, such as adaptive echo cancelers, adaptive equalizers, and adaptive controllers, adjust their own system parameters to reduce a specified error criterion. Hardware failures in such systems would presumably hamper their ability to minimize the error criterion to the greatest possible extent. However, such systems will continue to adapt their parameters to reduce this error to the greatest possible extent despite the occurrence of hardware failures. Since this is the case, it may be possible to design a system that makes use of the ongoing adaptive process to automatically compensate for certain types of hardware failures. This approach has been successfully applied to finite impulse response (FIR) adaptive filter structures. The primary advantage of AFT is that it can provide fault tolerance in adaptive systems for certain classes of hardware failures with relatively low hardware overhead compared to those for traditional fault tolerance techniques.

Previous studies limited the use of AFT to one-dimensional FIR filter structures with simple "stuck-at" fault models that account for errors introduced by faulty coefficients that cease to adjust properly. This paper considers a broader class of hardware errors which are modeled by errors in the outputs of both multipliers and adders. It is shown that by including a simple circuit that removes erroneous constant values in the output of the adaptive filter, adaptive fault tolerance can be extended to a broader class of hardware failures.

2. AFT for FIR Adaptive Filters

In most of the published literature on AFT, the concepts were developed for finite impulse response (FIR) filters using vector space concepts. The impulse response of a length-N FIR filter can be considered to be an element of the vector space $\mathbf{R}^N$ (or $\mathbf{C}^N$, if the coefficients are complex). Vectors of $\mathbf{R}^N$ (or $\mathbf{C}^N$) can be represented as a linear combination of N basis vectors which span the space. One such basis consists of the set of standard unit vectors $\underline{\mathbf{e}}_i$, where the i^{th} element of $\underline{\mathbf{e}}_i$ is one, and the rest are zero. This particular decomposition corresponds to the structure of a direct-form FIR adaptive filter, where the weights of the standard unit vectors correspond to the coefficients of the adaptive filter which change with time to match some desired response, i.e., the Wiener solution. If the time-varying impulse response of the adaptive filter is given by $\mathbf{A}(n) = [a_1(n), a_2(n), \ldots, a_N(n)]^T$, then the vector decomposition just described can be written as

$$\mathbf{A}(n) = a_1(n)\,\underline{\mathbf{e}}_1 + a_2(n)\,\underline{\mathbf{e}}_2 + \ldots + a_N(n)\,\underline{\mathbf{e}}_N . \qquad (1)$$

Note that the right-hand side of Eq (1) is an adaptive linear combination of N column vectors, each of dimension N x 1.

If a particular tap weight of the adaptive filter fails and remains fixed at an incorrect value, the other coefficients will not be able to entirely compensate for the faulty tap, i.e., the Wiener solution will not be achieved because each tap is used to span a different dimension of $\mathbf{R}^N$, specified by the N basis vectors. This prevents an error in a particular tap from being corrected by readjusting the values of the other taps. However, the incorporation of one or more appropriate, additional vectors would allow the coefficients which remain functional to readjust and achieve the Wiener solution. In this way, coefficient failures in an FIR adaptive filter can be compensated for by adding extra coefficients which, when taken with the other fault-free coefficients, allow all of $\mathbf{R}^N$ to be spanned.

For example, consider a three-tap, direct form, FIR adaptive filter which has the following impulse response:

$$\mathbf{A}(n) = a_1(n)\begin{bmatrix}1\\0\\0\end{bmatrix} + a_2(n)\begin{bmatrix}0\\1\\0\end{bmatrix} + a_3(n)\begin{bmatrix}0\\0\\1\end{bmatrix} \qquad (2)$$

If one of the adaptive coefficients incurs a fault, the other taps cannot be re-adapted to compensate for the failure. However, adding a fourth adaptive tap, whose input is the sum of the signals driving the original taps, results in a length-3 adaptive filter with impulse response:

$$\tilde{\mathbf{A}}(n) = a_1(n)\begin{bmatrix}1\\0\\0\end{bmatrix} + a_2(n)\begin{bmatrix}0\\1\\0\end{bmatrix} + a_3(n)\begin{bmatrix}0\\0\\1\end{bmatrix} + a_4(n)\begin{bmatrix}1\\1\\1\end{bmatrix} \quad (3)$$

The presence of this additional adaptively weighted column makes it possible for the remaining three adaptively-weighted columns to match any Wiener solution when any one of the four coefficients incurs a stuck-at fault. The reason for this is that the 3 x 3 matrix formed by choosing any three of the four columns in (3) is nonsingular. It should be noted that this property will only hold for certain choices of the additional vector as described later.

Most fault tolerant filter structures recently studied are based on this notion of adding extra coefficients and using the adaptive algorithm to automatically compensate for faults in the adaptive coefficients. A filter structure with R redundant coefficients, where R is greater than or equal to one, has been developed. This structure is able to achieve the fault-free minimum mean-squared error (MSE) despite the occurrence of R coefficient failures.

A significant aspect of the way in which fault tolerance is added to the adaptive filter is that the redundant components form an integral part of the structure. All the coefficients work together to match the Wiener solution, and when some of the coefficients fail, the remaining functional coefficients continue to operate normally until the Wiener solution is again achieved. This is significant because it is generally impossible to distinguish between the redundant coefficients and the primary coefficients. The locations of the coefficient faults, therefore, do not impact whether the adaptive filter has the ability to converge again to the Wiener solution, assuming less than R failures have occurred.

The fault-free learning rate is an important characteristic of a FTAF since it indicates the cost in terms of a reduced convergence rate for including fault tolerance in the adaptive filter design. In order to provide improved post-fault convergence properties, a transform domain fault tolerant adaptive filter (TDFTAF) based on the DFT has been formulated in [3]. The structure, shown in Fig. 1, is completely described by

$$\mathbf{X_e}(n) = [\mathbf{X}(n)\ 0\ 0\ \ldots\ 0]^T \quad (4)$$

$$\mathbf{V}(n) = \mathbf{F_M}\mathbf{X_e}(n) \quad (5)$$

$$y(n) = \mathbf{C}(n)^T\mathbf{V}(n), \quad (6)$$

where $\mathbf{X_e}(n)$ is a length M vector and $\mathbf{C}(n)$ is the vector of M adaptive coefficients. If the power normalized LMS algorithm is used to update the coefficients of the TDFTAF, then the relevant equations are

$$e(n) = d(n) - y(n) \quad (7)$$

$$\mathbf{C}(n+1) = \mathbf{C}(n) + e(n)\tilde{\mu}\mathbf{V}(n), \quad (8)$$

where $\tilde{\mu}$ is a diagonal matrix of time-varying step size parameters.

Mathematical analyses have shown that the fault-free convergence rate of the DFT-based TDFTAF having a white noise input be very nearly the same as for a standard direct form filter [3]. This result guarantees that there is very little penalty in convergence rate caused by providing fault tolerance to the adaptive filter.

3. Faults in Multiplier Outputs

Although the AFT method described above is able to achieve fault tolerance with respect to hardware faults that result in stuck-at coefficients, more general methods are needed to cover faults that may occur in the outputs of multipliers, the outputs of adders, and the registers that hold the intermediate values that are needed to form the filter output. In this section, we will describe how the principles of AFT can be extended to cover a broader class of faults that include stuck-at errors in multiplier and adder outputs.

Refer to Eq (6), which characterizes the convolution that produces the output y(n). In the present discussion adder faults are not considered, and we will assume that the registers containing samples of the input signal x(n), x(n-1), . . . are also fault free. The registers holding the $c_i(n)$'s can cause faults in the system, but these registers are protected by the AFT mechanism described above. However, the registers used to store the multiplication products of Eq (6), i.e. the $c_i(n)v_i(n)$'s, can also fail. Equivalently, stuck-at faults that occur in the outputs of the multipliers would result in the same condition.

Note that this class of stuck-at faults results in a very different condition than stuck-at faults in the coefficients themselves, which are the inputs to the multipliers. When the coefficient is stuck but the multiplier itself is functioning properly, the multiplier input $v_i(n)$ continues to vary properly, with the effect that the output of the multipliers remains correlated with the input signal. However, when the multiplier output takes on a stuck-at condition, a constant is induced into the filter output. This constant provides no information to the adaptive filter because it is not correlated with the input signal. It will be shown later in this section that removal of such an induced constant will allow the filter to re-adapt properly, based on the original principles of AFT for the stuck-at coefficient model.

Lastly, note that stuck-at faults in outputs of more than one multiplier still result in the creation of an induced constant. This condition of multiple errors is equivalent to multiple stuck-at faults in the registers holding the multiplication products.

Equation (5) performs the transformation of the extended input vector $\mathbf{X}_e(n)$ by multiplying by the transformation matrix $\mathbf{F}_M$. Assuming that $\mathbf{T}$, $\mathbf{X}$, and the adders are not corrupted, failures can occur in the registers holding the product of these values. In any efficient hardware processing scheme, registers are reused for successive calculations. For example, the same register will be used to hold $F_{11}x_e(n)$, then the quantity $F_{11}x_e(n) + F_{12}x_e(n-1)$, and finally $v_0(n)$. Note that a temporary register is also not needed during the addition because the output of the multiplication can be fed directly into the adder, thereby requiring a single register to produce $v_0(n)$, rather than M registers. All together, M registers are needed to obtain the complete $\mathbf{V}(n)$ vector at a given value of n. Finally, assuming that the appropriate multiplier assignments are used, any combination of faults in the M^2 multiplications will be equivalent to faults in the M registers holding the components of $\mathbf{V}(n)$.

Equation (8) performs the update for the filter coefficient vector $\mathbf{C}(n)$. With the technique of register reuse described above, the values of each $c_i(n+1)$, i = 0, . . . , M-1 are written into the same registers that stored $c_i(n)$. Registers holding the values of e(n), x(n), and $\tilde{\mu}$ are assumed to be globally protected; thus these values can be assumed to be reliable. Also, separate registers are not needed to hold the terms $e(n)\,\tilde{\mu}\,\mathbf{V}(n)$ because these terms can be fed directly into the adder after the multiplication in complete. Again, note that stuck-at faults in multiplier outputs can be traced into the convolution product and characterized as induced constants in the output of the filter.

To summarize, there are many other types of failures besides faults in the coefficient registers which must be handled properly if the principles of AFT are to be used effectively in realistic fault tolerant designs. However, with an appropriate multiplier assignment scheme all non-coefficient register stuck-at faults can be modeled by including induced constants in the convolution terms. The only question remaining is whether the filter can re-adapt to the correct Wiener solution after such constant terms are induced into the filter output via hardware faults.

Regardless of its source, when a constant is induced into the filter output due to stuck-at hardware failures, the filter will not be able to cancel it because the constant is uncorrelated with the input to the filter. The constant remains uncanceled in the filter output, with the consequence that the learning curve will converge to a constant level that is usually much higher than the natural noise floor of the system. The block diagram of Fig. 3 shows the induced constant modeled as an independent signal that is additively injected into the filter output. Figure 4 illustrates the learning characteristic of the filter when a constant of value 10 is suddenly injected at iteration 200. Note that the MSE jumps immediately to the value of this constant and remains in the vicinity of this value indefinitely, due to the fact that the filter is unable to cancel it.

A simple circuit can be designed to estimate the value of this constant, and to remove it by subtracting the estimated value from the filter output. This procedure is illustrated in Fig. 3, where the correction factor IFIX is added to cancel the induced constant, thereby allowing the filter to continue operating properly. Note that since

$$e(n) = d(n) - y(n) - \text{CONST}, \qquad (9)$$

the expected value of e(n) provides an estimate of -CONST. Therefore the average of e(n) can be calculated according to the algorithm shown in Table 1, and periodically added into the output as shown in Fig 3. This procedure is demonstrated with the experiment shown in Fig. 5, where constant errors were induced into the filter output to simulate stuck-at multiplier output errors at iterations 200, 600, and 1000. Each time a new error occurs the averaging circuit produces a new correction factor to eliminate the induced constant, allowing the adaptive filter to return over and over again to the correct Wiener solution.

Although it is beyond the scope of this paper to discuss fault in adder outputs, it can be noted that stuck-at errors in many of the adders will produce induced errors in the filter output that are similar to those produced by multiplier output faults. However for adders, a single fault will involve two or more product terms, and hence more than one multiplier may be removed from effective operation by the occurrence of a single adder failure. A thorough analysis of adder faults will be the subject of a future publication.

Acknowledgment

This work is supported by the Joint Services Electronics Program (JSEP) under contract number N00014-90-J-1270. The opinions expressed here do not necessarily represent those of the sponsoring agencies.

References

[1] B. A. Schnaufer, "Practical techniques for rapid and reliable real-time adaptive filtering," Ph.D. dissertation, Department of Electrical and Computer Engineering, University of Illinois, Urbana-Champaign, IL, 1994.

[2] W. K. Jenkins, A. W. Hull, J. A. Strait, B. A. Schnaufer, and X. Li. *Advanced Concepts in Adaptive Filtering*, Kluwer Press, 1996.

[3] B. A. Schnaufer and W. K. Jenkins, "An FFT-based fault tolerant adaptive filter," *Proc. 1994 IEEE Symp. on Circuits Syst.* (London, England, May 1994) 2261-2264 (1994).

[4] J. Jiang, C. D. Schmitz, B. A. Schnaufer and W. K. Jenkins, "The use of adaptive fault tolerance in general classes of linear systems," *Proceedings of the*

1996 Midwest Symposium on Circuits and Systems, Ames, IA, August 1996, to appear.

[5] J. Jiang, "Fault-Tolerant Linear systems Using Adaptive Filters," MS Thesis, Department of Electrical and Computer Engineering, University of Illinois, Urbana-Champaign, IL, 1995.

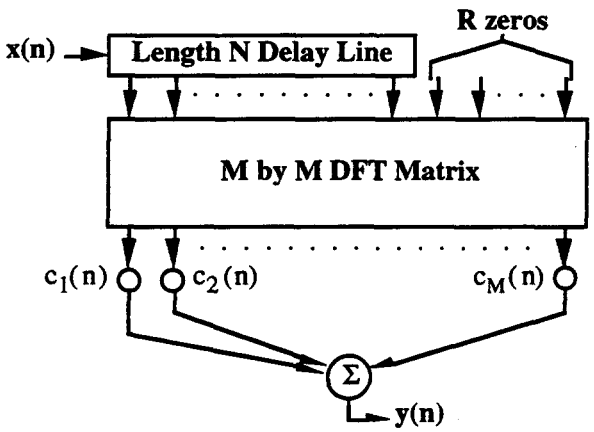

Fig. 1 Transform domain AFT filter structure.

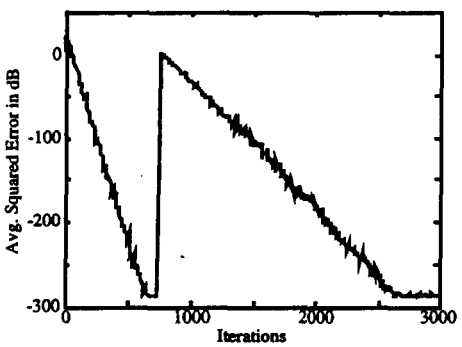

Fig. 2 Convergence plot for the FFT-based TDFTAF being driven with white noise having N = 10 and R = 2. (Stuck-at fault in Tap 5 at Iteration 750)

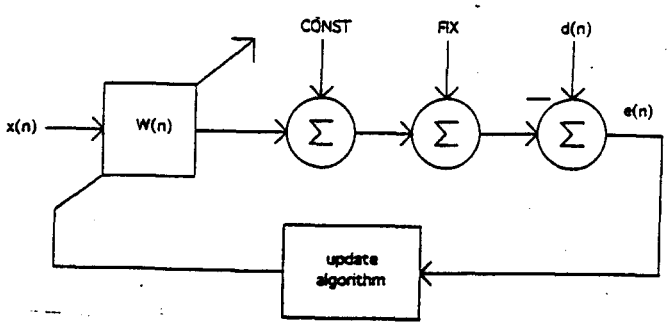

Fig 3. Block diagram of an adaptive filter with an induced constant error and an additive correction factor FIX.

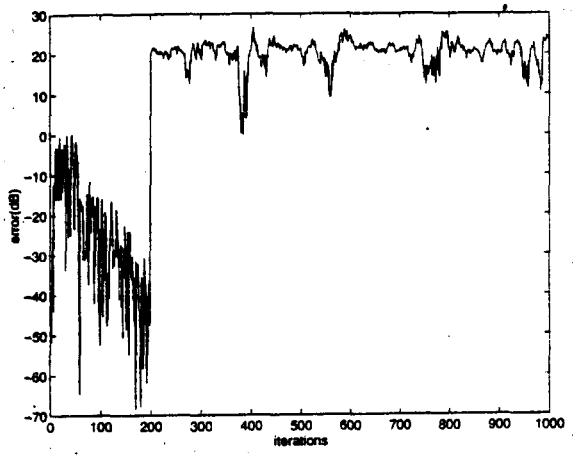

Fig 4. Learning curve for an adaptive filter with an induced constant error of value 10.

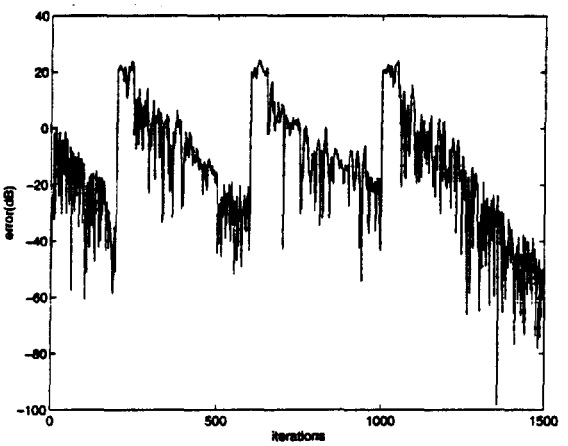

Fig 5. Learning curve with an induced constant error at iterations 200, 600, and 100 with a continuously updated and applied correction factor FIR.

Table 1. Algorithm to Obtain FIX

```
sum = 0
For every NUM iterations
        for i = 1 till NUM
                sum = sum + e(n-i)
        end
        FIX = FIX + sum / NUM
        sum = 0
end.
```