Floating Point Error Analysis of Recursive Least Squares and Least Means Squares Adaptive Filters

Sasan H. Ardalan

Center for Communications and Signal Processing
Dept. of Electrical and Computer Engineering
North Carolina State University
Raleigh, NC

April 1985

CCSP-TR-85/5

Abstract

A floating point error analysis of the Recursive Least Squares algorithm is presented. It is shown that the algorithm diverges exponentially and cubically with the number of iterations. Divergence occurs after initial convergence, however. The floating point error introduced by adding the correction to the weight vector update is seen to be the source of divergence. Other errors are non-catastrophic. The methodology of analysis is extended to the floating point error analysis of the Least Mean Squares (LMS) algorithm. It is shown that the variance of the floating point error increases inversely with loop gain. Furthermore, as the loop gain approaches zero the algorithm diverges exponentially.

TABLE OF CONTENTS

1. Introduction

The Recursive Least Squares (RLS) algorithm has found wide application in adaptive filtering problems [1,2,3]. In particular, the algorithm offers very fast convergence and tracking capability. The convergence is also independent of the signal statistics, i. e. eigenvalue spread of the input signal. This has lead to the application of the RLS algorithm to many adaptive filtering problems. However, finite wordlength effects on the digital implementation of this algorithm have not been thoroughly analyzed until recently. While much analytical work exists for the finite wordlength analysis of fixed digital filters, there is much work to be done in the analytical treatment of adaptive filters. Recently a fixed point and floating point error analysis of the Least Mean Squares algorithm was presented in [4]. In [5] a fixed point roundoff error analysis of the RLS algorithm is presented. In that paper, a closed form solution to the variance of the roundoff error is derived. It was found that the algorithm diverges as the number of iterations increases. The result parallels that of [4] where it is shown that the roundoff error increases inversely to the adaptation loop gain for the LMS algorithm. In this paper the work in [5] is extended to the floating point error analysis of the RLS algorithm. Furthermore, Using the same techniques the work is extended to the floating point error analysis of the LMS algorithm. It is shown that as the loop gains approaches zero, the algorithm diverges exponentially. This was not predicted in [4].

2. Outline and Summary

2.1 The RLS Algorithm

The RLS algorithm can be summarized as follows The desired signal d(n) is predicted from the input samples by convolving the input samples with a weight vector The prediction error is then calculated as the difference between these two signals. The prediction error is used to update the weight vector to minimize the accumulated sum of the square of the resid al error The weight vector that minimizes this criterion at each iteration is written recursively. This leads to the Kalman gain vector which when multiplied by the scalar prediction error forms the weight vector update The algorithm thus involves two primary calculations The calculation of the prediction of the desired signal d(n) through an inner product of the weight vector and the vector of delayed elements of the input samples and the calculation of the weight vector update The weight vector update involves the calculation of the weight vector correction through the multiplication of the Kalman gain vector and the scalar prediction error.

2.2 Outline of Method

For the purposes of analysis we assume that the Kalman gain is calculated with infinite precision and a floating point quantized version is available. This assumption is based on the fact that the Kalman gain can be calculated using different algorithms and that as a preliminary step we are attempting to isolate the causes of error in the floating point implementation of the algorithm.

As in the case with the floating point analysis of fixed digital filters [6,7] and the LMS algorithm in [4] the errors introduced by floating point operations are modeled as follows: For summation $fl(x+y)=(x+y)(1+e)$ and for multiplication $fl(xy)=xy(1+q)$. The relative error terms e and q are zero mean white independent random numbers. They are uncorrelated with x+y and xy. Using the above models the errors introduced by floating point implementation are incorporated into the algorithm. Where appropriate the error sequences are written as vectors and matrices.

We show as in [4] that the floating point errors in the calculation of the prediction of the desired signal (vector inner product) can be represented as an additive noise sequence to the infinite precision calculation. This sequence is a zero mean white independent random process which has a variance related to signal statistics, the weight vector covariance, and the floating point errors. The calculation of the prediction error also introduces a relative error term. More significantly are the errors introduced by the weight vector update recursion. The desired signal is represented in terms of a convolution of the input signal and an optimum weight vector. The prediction error is written in terms of this desired signal and the the predicted desired signal including the additive noise term due to finite precision and substituted into the weight update equation. The weight vector is then translated in coordinates by the optimum weight vector. Starting from the initial condition an exact relationship between the floating point translated weight vector

and the driving signal,the Kalman gain, and the floating point noise sources is derived.

Observing this equation we find that two terms are immediately separable. One term involves a weight vector error that includes the floating point noise sources. The other term describes the evolution of the translated weight vector using infinite precision but including error terms to be described. If these error terms are removed, this term will converge to the null vector. In other words the weight vector will converge to the optimum weight vector. It describes the evolution of the infinite precision weight vector neglecting the error terms. However, the floating point error terms are present and their effects are analyzed. The analysis is done in terms of the weight vector covariance matrix. This matrix is derived. Neglecting the cross-correlation between the two vectors the matrix is seen to be the sum of two matrices. One is the covariance of the weight error vector. We derive the exact equation for this matrix and using some approximations simplify the result.

If we define an error between the infinite precision calculation of the prediction of the desired signal and the floating point calculation, then the variance of this error is related to the trace of the covariance of the weight error vector. It is shown that the floating point error variance is related to the cube of the number of iterations multiplied by the exponent of the number of iterations. In other words the algorithm diverges as the number of iterations increase. If the noise sources in the translated weight vector are taken into consideration, then

another term related to exponential growth with the number of iterations is introduced This also leads to divergence. Finally the methodology is applied to the LMS algorithm It is shown that the floating point errors increase inversely to loop gain In fact as the loop gain approaches zero the LMS algorithm diverges exponentially

3. Recursive Least Squares (RLS) Algorithm _ Infinite Precision

Consider a linear system with sampled input signal x(n) and output d(n) Suppose that the samples x(n) and d(n) can be related by the system impulse response coeffients w^*_i

$$d(n) = \sum_{i=0}^{N-1} w_i^* \, x(n-i) = \underline{w}^{*T}\underline{x}(n) \qquad (3.1)$$

where the underscore denotes N-length vectors Further, we have assumed that the impulse response has insignificant terms beyond N samples. The notation x(n) signifies the vector of the last N samples of the input:

$$\underline{x}(n) = [x(n) \; x(n-1) \; .. \; x(n-N+1)]^T \qquad (3.2)$$

This paper considers the systems identification problem. that is we are interested in estimating $\underline{w}^*$ from the sequence x(n) and d(n). The Recursive Least Squares (Kalman) algorithm achieves this by updating a current coefficient estimate $\underline{w}(n-1)$ through the following recursion,

$$\underline{w}(n) = \underline{w}(n-1) + \underline{K}(n) \, e(n) \qquad (3\ 3)$$

where

$$e(n) = d(n) - d(n) \qquad (3\ 4)$$

and

$$d(n) = \underline{w}^T(n-1)\underline{x}(n) \qquad (3\ 5)$$

is the prediction of d(n) based on N previous samples of $\underline{x}(n)$. The Kalman gain can be shown [1] to be

$$\underline{K}(n) = \left[\sum_{i=0}^{n} \underline{x}(i)\underline{x}^T(i) \right]^{-1} \underline{x}(n) \tag{3.6}$$

The derivation leading to (3.3) with the Kalman gain given by (3.6) is based on minimizing the accumulated sum of the square of the residual errors up to time n with respect to the weight vector $\underline{w}(n)$ and writing $\underline{w}(n)$ recursively.

4. Floating Point Roundoff Error Models

In this paper the errors introduced by floating point operations are modeled as follows:

For multiplication

$$fl[x \cdot y] = x \cdot y[1+\varepsilon] \tag{4.1}$$

were ε is the relative error and is modeled as a zero mean random variable independent of x,y and $(x \cdot y)$. It has been found [6,8] that $\sigma_\varepsilon^2 = E(\varepsilon^2) \approx (0.18)2^{-2B}$ where B is the number of bits used to represent the mantissa.

For addition

$$fl[x+y] = (x+y)\,[1+\delta]$$

where δ is the relative error modeled as a zero mean random variable with variance σ_δ^2. The actual distribution of δ and ε is not important as long as the assumptions of zero mean independent random variables is valid.

Note that in floating point operations, addition also introduces an error in contrast to fixed point operations (assuming no overflow).

5. Floating Point Implementation of the RLS Algorithm

For this section we incorporate the errors introduced by the floating point implementation of the RLS algorithm into the algorithm.

5.1 Desired Signal Prediction

Consider the calculation of the prediciton of the desired signal through the inner product (3.5). Let $\eta(n)$ denote the error introduced by floating point operations in the calculation of the inner product. Then,

$$\hat{d}'(n) = \underline{x}^T(n)\underline{w}'(n-1) + \eta(n) \tag{5.1}$$

For the above equation $d'(n)$ is the prediction of the desired signal using the floating point RLS algorithm and $\underline{w}'(n)$ is the floating point weight vector. The primes denote the floating point variables as opposed to the infinite precision variables of (3.1-3.6).

Figure 1 illustrates the error introduced by floating point operations. From the figure,

$$\begin{aligned}\eta(n) = {} & x_0(n)w_0'(n)\varepsilon_0(n) + \ldots + x_{N-1}(n)w_{N-1}'(n)\varepsilon_{N-1}(n) + \\ & [x_0(n)w_0'(n)+x_1(n)w_1'(n)]\,\nu_1(n) + \ldots + \\ & [x_0(n)w_0'(n) + \ldots + x_{N-1}(n)w_{N-1}'(n)]\,\nu_{N-1}(n)\end{aligned} \tag{5.2}$$

Since $\varepsilon_i(n)$ and $\nu_i(n)$, $i=0,\ldots,N-1$ are zero mean white random sequences then $\eta(n)$ is a zero mean white random sequence. We have

$$\begin{aligned}\sigma_\eta^2 = E[\eta^2(n)] = {} & \sigma_\varepsilon^2 \sum_{k=0}^{N-1} \sigma_x^2\, E[w_i'^2(k)] + \sigma_\nu^2\{E[x_0(n)w_0'(n)+x_1(n)w_1'(n)]^2 + \\ & \ldots + E[x(n)w_0'(n)+\ldots+x_{N-1}(n)w_{N-1}'(n)]^2\}\end{aligned} \tag{5.3}$$

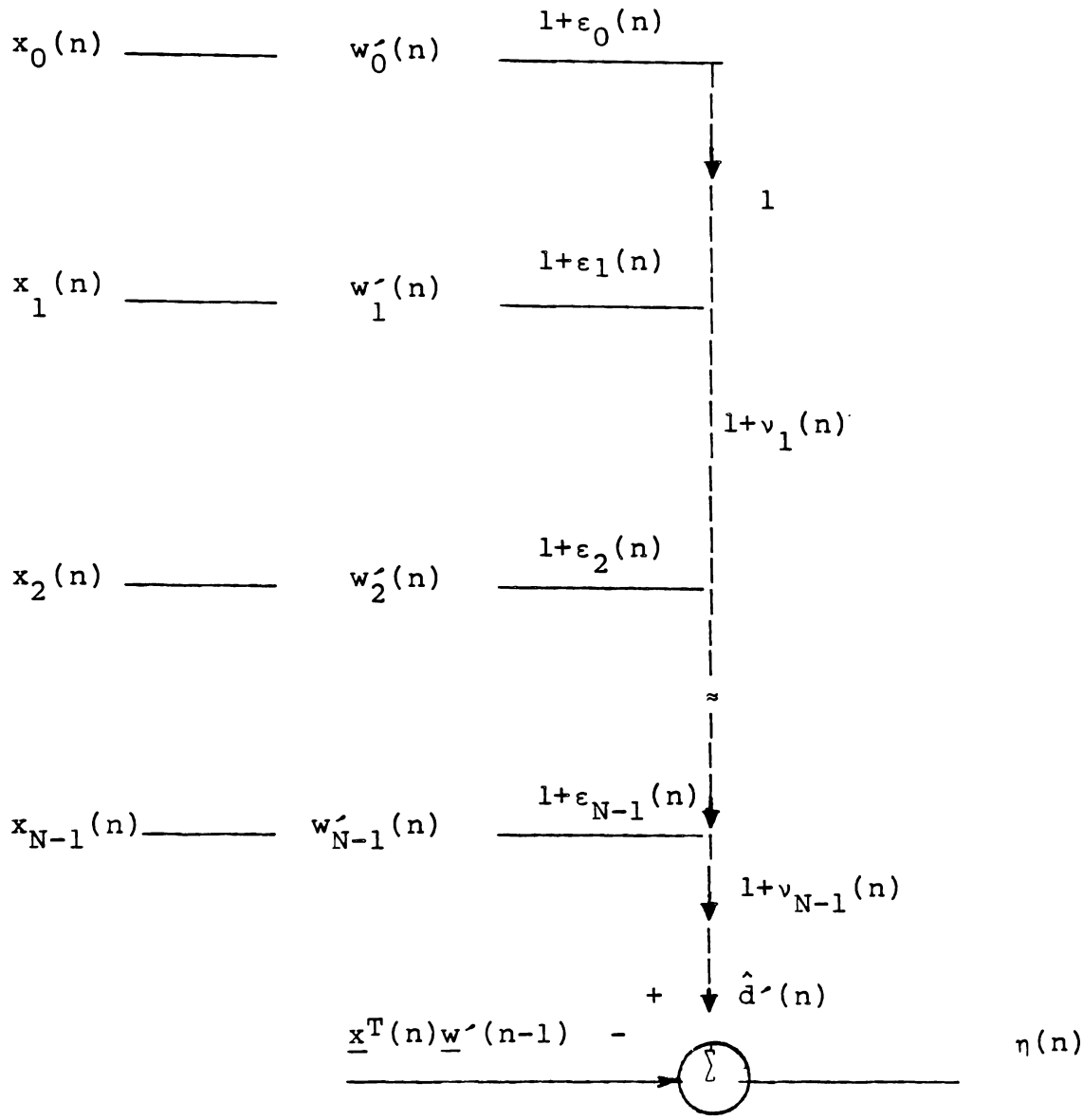

Figure 1. Floating Point Errors Due to Inner Product Calculation of $\hat{d}'(n)$.

The expression (5.3) for σ_η^2 can be written in terms of the autocorrelation matrix R_x and the optimal (Wiener) vector $\underline{w}^*$ as in [4]. But this assumes that $\underline{w}'(n)$ converges. We will show, however, that $\underline{w}'(n)$ will diverge and thus σ_η^2 will diverge also due to floating point errors introduced in the weight vector update recursion. Nevertheless, the point to consider is that prior to divergence $\sigma\eta^2$ depends on the floating point noise sources and the statistics of $x(n)$ and $\underline{w}'(n)$.

5.2 Prediction Error Calculation

Next consider the floating point error introduced in the computation of the prediction error (3.4). We have,

$$e'(n) = [d(n) - \hat{d}'(n)][1+\delta(n)] \quad , \tag{5.4}$$

where the relative error $\delta(n)$ is a zero mean white random sequence. If we substitute (3.1) for $d(n)$ and (5.1) for $\hat{d}'(n)$ then we can write

$$e'(n) = \underline{x}^T(n)\underline{w}^* - \underline{x}^T(n)\underline{w}'(n-1) - \eta(n) + \delta(n)\underline{x}^T(n)\underline{w}^* - \delta(n)\underline{x}^T(n)\underline{w}'(n-1) \quad , \tag{5.5}$$

where we have neglected second order noise terms [i.e., $\eta(n)\delta(n)$].

$$e'(n) = \underline{x}^T(n)[\underline{w}^*-\underline{w}'(n-1)] + \delta(n)\underline{x}^T(n)[\underline{w}^*-\underline{w}'(n-1)] - \eta(n) \tag{5.6}$$

5.3 Weight Vector Update Recursion

Finally, once the prediction error is computed, the weight vector is updated by the recursive equation (3.3). Assuming the floating point quantized Kalman gain vector elements are available at each step, the recursive update equation for each weight vector element including floating point noise sources is

$$w_i'(n) = \{w_i(n-1) + K_i(n)e'(n)[1+\mu_i(n)]\}[1+\alpha_i(n)] \tag{5.7}$$

where $\mu i(n)$ and $\alpha i(n)$ are floating point noise sources.

Expanding (5.7) and neglecting second order noise terms we obtain,

$$w_i(n) = w_i(n-1) + K_i(n)e(n) + w_i(n-1)\alpha_i(n) + K_i(n)e(n)[\alpha_i(n)+\mu_i(n)] \quad (5.8)$$

If we define the diagonal matrices,

$$\alpha_{NN}(n) = \text{diag}[\alpha_i(n)] \quad (5.9)$$

$$\mu_{NN}(n) = \text{diag}[\alpha_i(n)+\mu_i(n)] \quad (5.10)$$

then the weight vector update equation becomes

$$\underline{w}(n) = \underline{w}(n-1)+\underline{K}(n)e(n)+\alpha_{NN}\underline{w}(n-1)+\mu_{NN}\underline{K}(n)e(n) \quad (5.11)$$

Substituting for e (n) from (5.6) and neglecting second order noise terms we obtain,

$$\underline{w}(n) = \underline{w}(n-1) - \underline{K}(n)\underline{x}^T(n)[\underline{w}(n-1)-\underline{w}^*] - \underline{K}(n)\eta(n) - \delta(n)\underline{K}(n)\underline{x}^T(n)[\underline{w}(n-1)-\underline{w}^*] + \alpha_{NN}(n)\underline{w}(n-1) + \mu_{NN}(n)\underline{K}(n)\underline{x}^T(n)\{\underline{w}(n-1)-\underline{w}^*\} \quad (5.12)$$

It is convenient to write the recursion (5.12) in terms of the translated coordinate system

$$\underline{\theta}(n) = \underline{w}(n) - \underline{w}^* \quad (5.13)$$

Therefore, subtracting $\underline{w}^*$ from both sides of (5.12) and substituting (5.13) we obtain,

$$\underline{\theta}(n) = \underline{\theta}(n-1) - \underline{K}(n)\underline{x}^T(n)\underline{\theta}(n-1) - \delta(n)\underline{K}(n)\underline{x}^T(n)\underline{\theta}(n-1) + \mu_{NN}(n)N\underline{K}(n)\underline{x}^T(n)\underline{\theta}(n-1) + \alpha_{NN}(n)[\underline{\theta}(n-1)+\underline{w}^*] - \underline{K}(n)\eta(n) \quad (5.14)$$

Collecting terms,

$$\underline{\theta}(n) = [I - \underline{K}(n)\underline{x}^T(n) - \mu_{NN}\underline{K}(n)\underline{x}^T(n) + \alpha_{NN}(n)]\underline{\theta}(n-1) + \alpha_{NN}(n)\underline{w}^* - \eta(n)\underline{K}(n) \quad (5.15)$$

where

$$\underset{NN}{u}(n) = \underset{NN}{u}(n) + \delta(n)I \tag{5.16}$$

Define the matrix

$$\beta_{NN}(n) = \alpha_{NN}(n) - \underset{NN}{u}(n)\,\underline{K}(n)\underline{x}^T \tag{5.17}$$

and the vector

$$\underline{\xi}(n) = \alpha_{NN}(n)w^* - \eta(n)\underline{K}(n) \tag{5.18}$$

Then substituting into (5.15) we get

$$\underline{\theta}(n) = [I - \underline{K}(n)\underline{x}^T(n) + \beta_{NN}(n)]\underline{\theta}(n-1) + \underline{\xi}(n) \tag{5.19}$$

5.4 Discussion

Equation (5.19) represents the recursive update equation of the translated coordinate weight vector in terms of the noise sources due to floating point errors. For infinite precision,

$$\beta_{NN}(n) \to 0$$

$$\underline{\xi}(n) \to 0$$

and

$$\underline{\theta}(n) \to \underline{\theta}(n) = [I - \underline{K}(n)\underline{x}^T(n)]\underline{\theta}(n-1) \tag{5.20}$$

which is the exact recursion for the infinite precision update equation of the translated coordinate weight vector.

For the next section we will analyze the effects of the floating point noise sources on the steady state behaviour of the translated coordinate weight vector $\underline{\theta}(n)$.

It must be noted that in the infinite precision case, as $n \to \infty$, $\theta(n)$ converges to the null (0) vector and $\underline{w}(n)$ converges to the optimal (Wiener) weight vector $\underline{w}^*$

6. Derivations and Analysis of Floating Point Roundoff Effects on the RLS Algorithm

If we write the recursion (5.19) for $\underline{\theta}'(n)$ in terms of the initial condition $\underline{\theta}'(0)$, then we obtain

$$\underline{\theta}'(n) = \Big(\prod_{i=1}^{n} [I - \underline{K}(i)\underline{x}^T(i) + \beta_{NN}(i)]\Big)\underline{\theta}(0) + \sum_{i=1}^{n} \underline{\xi}^T(i)\Big(\prod_{j=i+1}^{n} [I - \underline{K}(j)\underline{x}^T(j) + \beta_{NN}(j)]\Big) \tag{6.1}$$

Note that we define

$$\prod_{j=n+1}^{n} [\cdot] = 1 \tag{6.2}$$

Define the weight error vector as

$$\underline{\Psi}(n) = \sum_{i=1}^{n} \underline{\xi}^T(i) \Big\{\prod_{j=i+1}^{n} [I - \underline{K}(j)\underline{x}^T(j) + \beta_{NN}(j)]\Big\} \tag{6.3}$$

Also define

$$\underline{\lambda}(n) = \Big\{\prod_{i=1}^{n} [I - \underline{K}(i)\underline{x}^T(i) + \beta_{NN}(i)]\Big\}\theta(0) \tag{6.4}$$

Then

$$\underline{\theta}'(n) = \underline{\lambda}(n) + \underline{\Psi}(n) \tag{6.5}$$

Neglecting the floating point noise sources in the matrix, $\beta_{NN}(i)$, $\underline{\lambda}(n)$ describes the convergence of the weight vector towards $\underline{w}^*$. That is $\underline{\lambda}(n)$ approaches the null vector as $n \to \infty$. In this case $\underline{\Psi}(n)$ represents a weight error vector due to floating point errors in the algorithm. In the sequel we will neglect the crosscorrelation between $\underline{\lambda}(n)$ and $\underline{\Psi}(n)$. Thus, the covariance of $\underline{\theta}'(n)$ becomes,

$$R_{\underline{\theta}} = E\{\underline{\theta}'(n)\underline{\theta}^{T\prime}(n)\} = E\{\underline{\lambda}(n)\underline{\lambda}^T(n)\} + E\{\underline{\Psi}(n)\underline{\Psi}^T(n)\} \tag{6.6}$$

We now proceed to calculate

$$R_{\Psi} = E\{\underline{\Psi}(n)\underline{\Psi}^T(n)\} = E\{\sum_{i=1}^{n}\sum_{i=j}^{n}\prod_{k=i+1}^{n-m}[I - \underline{K}(k)\underline{x}^T(k) + \beta_{NN}(k)]$$

$$\underline{\xi}(i)\underline{\xi}^T(j)\cdot\prod_{m=j+1}^{n}[I - \underline{x}(m)\underline{K}^T(m) + \beta_{NN}^T(m)] \quad (6.8)$$

A major simplification of (6.8) results when we note that the vectors $\underline{\xi}(i)$ and $\underline{\xi}(j)$ are independent zero means for $i\neq j$. Thus

$$R_{\xi}(j,k) = E\{\underline{\xi}(i)\underline{\xi}^T(j)\} = \begin{matrix} R_{\xi}(i) & i=j \\ 0_{NN} & i\neq j \end{matrix} \quad (6.9)$$

Hence terms where $i\neq j$ drop out in (6.8). Now,

$$R_{\xi}(i) = E\{\underline{\xi}(i)\underline{\xi}^T(i)\} = E\{\alpha_{NN}(i)\underline{w}*\underline{w}*^T\alpha_{NN}(i)\} + E\{\eta^2(i)\underline{K}(i)\underline{K}^T(i)\} \quad (6.10)$$

where we have substituted (5.18) for $\underline{\xi}(i)$. From [5] we show that

$$E\{\underline{K}(i)\underline{K}^T(i)\} = \frac{1}{i^2}R_x^{-1} \quad (6.11)$$

where

$$R_x = E\{\underline{x}(n)\underline{x}^T(n)\} \quad (6.12)$$

is the autocorrelation matrix of the input sequence x(n). Therefore

$$R_{\xi}(i) = \sigma_{\alpha}^2\underline{w}*\underline{w}*^T + \sigma_{\eta}^2\frac{1}{i^2}R_x^{-1} \quad (6.13)$$

Using (6.9) we can write (6.8) as follows,

$$R_{\Psi} = \sum_{i=1}^{n}E\{\underline{\xi}^T(i)\prod_{k=i+1}^{n-1}[I - \underline{x}(k)\underline{K}^T(k) + \beta_{NN}(k)]$$

$$[I - \underline{K}(k)\underline{x}^T(k) + \beta_{NN}(k)]\underline{\xi}(i)\} \quad (6.14)$$

Define,

$$\Xi(k) = E\{[I - \underline{x}(k)\underline{K}^T(k) + \beta_{NN}(k)][I - \underline{K}(k)\underline{x}^T(k) + \beta_{NN}(k)]\} \quad (6.15)$$

From [5] we show that

$$E\{\underline{x}(k)\underline{K}^T(k)\underline{K}(k)\underline{x}^T(k)\} = \frac{N+2}{k^2} \tag{6.15}$$

$$E\{\underline{x}(k)\underline{K}^T(k)\} = \frac{1}{k} \tag{6.16}$$

In deriving (6.15) and (6.16) we assume that x(n) is an ergodic, stationary white zero mean Gaussian random sequence. The results are valid in the steady state.

Now, from (5.17) the elements of $\beta_{NN}(k)$ are zero mean. We have using (5.17), (5.16), (5.1) and (6.16),

$$E\{\beta_{NN}(k)\beta_{NN}^T(k)\} = E\{\alpha_{NN}(k)\alpha_{NN}^T(k) + \mu_{NN}(k)\underline{K}(k)\underline{x}^T(k)\underline{K}(k)\underline{K}^T(k)\mu_{NN}^T(k) - \alpha_{NN}(k)\underline{x}(k)\underline{K}^T(k)\mu_{NN}^T(k) - \mu_{NN}(k)\underline{K}(k)\underline{x}^T(k)\alpha_{NN}^T(k)\} \tag{6.17}$$

$$E\{\beta_{NN}(k)\beta_{NN}^T(k)\} = \sigma_\alpha^2 I + \frac{N+2}{k^2}(\sigma_\alpha^2 + \sigma_\mu^2 + \sigma_\delta^2)I - \frac{2\sigma_\alpha^2}{k} I \tag{6.18}$$

Substituting (6.15), (6.16) and (6.18) into (6.14) we obtain

$$\Xi(k) = \left[1 - \frac{2}{k} + \frac{N+2}{k^2} + \sigma_\alpha^2 + \frac{N+2}{k^2}\sigma_\tau^2 - \frac{2\sigma_\alpha^2}{k}\right]I \tag{6.19}$$

In the above equations we have defined

$$\sigma_\tau^2 = \sigma_\alpha^2 + \sigma_\mu^2 + \sigma_\delta^2 \tag{6.20}$$

We note that $\Xi(k)$ is a diagonal matrix. Thus, we can write R_ψ as follows,

$$R_\psi = \sum_{i=1}^{n} \prod_{k=i+1}^{n-1} \Xi(k)\, E\{\underline{\xi}^T(i)\underline{\xi}(i)\} \tag{6.21}$$

Finally we obtain,

$$R_\psi = \sum_{i=1}^{n} \prod_{k=i+1}^{n} \left[1 - \frac{2}{k} + \frac{N+2}{k} - \frac{2}{k}\sigma_\alpha^2 + \frac{1}{k^2}(N+2)\sigma_\tau^2 + \sigma_\alpha^2\right] \left[\sigma_\alpha^2 \underline{w}^*\underline{w}^{*T} + \sigma_\eta^2 i^2 R_x^{-1}\right] \tag{6.22}$$

Equation (6.22) describes the exact steady state covariance matrix of the weight error vector $\psi(n)$ defined in (6.3) In order to analyze this result, we make use of the fact that as i becomes large, terms with orders $O(k^{-1})$ and $O(k^{-2})$ become insignificant compared to other terms. Hence,

$$R_{\psi} \sim \sum_{i=1}^{n} C(i) \prod_{k=M}^{n} (1+\sigma_{\alpha}^2) R_{\xi}(i) \tag{6.23}$$

The number M is related to $\frac{1}{\sigma_{\alpha}^2}$ and defines the index at which σ_{α}^2 becomes significant compared to terms of the order $O(k^{-1})$. We proceed as follows,

$$R_{\psi} = \sum_{i=1}^{n} C(i) \, (1+\sigma_{\alpha}^2)^{(n-M)} R_{\xi}(i) \tag{6.24}$$

Now, since $\sigma_{\alpha}^2 << 1$ we make use of the following approximation,

$$(1+\sigma_{\alpha}^2)^{n-M} = e^{\sigma_{\alpha}^2(n-M)} \tag{6.25}$$

This can be shown as follows,

Let $\varepsilon << 1$. Then,

$$\ell n(1+\varepsilon) \sim \varepsilon + \varepsilon^2 + \ldots$$

or,

$$x = (1 + \varepsilon) \sim e^{\varepsilon}$$

Now,

$$x = (1+\varepsilon)^n$$

or,

$$x = (1 + \varepsilon)^n \sim e^{n\varepsilon} \tag{6.26}$$

Substituting (6.25) into (6.24)

$$R_{\psi} = \sum_{i=1}^{n} C(i) e^{\sigma_{\alpha}^2(n-M)} R_{\xi}(i) \tag{6.27}$$

The term C(i) can be approximated as follows,

$$C(i) \sim \frac{i^2}{M^2} \tag{6.28}$$

The justification of (6.28) is that when i<M, the terms

$\frac{2}{k}\sigma_\alpha^2$, $\frac{1}{k^2}(N+2)\sigma_\tau^2$, and σ_α^2 can be neglected

in (6.22). Then (6.28) follows by expanding C(i) as,

$$C(i) = \prod_{k=i+1}^{M} [1 - \frac{2}{k} + \frac{1}{k^2} + \frac{N+1}{k^2}] \sim \prod_{k=i+1}^{M} (\frac{k-1}{k})^2 = \frac{i^2}{M^2} \tag{6.29}$$

Substituting into (6.27)

$$R_\psi = \frac{e^{-M\sigma_\alpha^2}}{M^2} \cdot e^{n\sigma_\alpha^2} \sum_{i=1}^{n} i^2 R_\xi(i) \tag{6.30}$$

Substituting for $R_\xi(i)$ from (6.13) we obtain,

$$R_\psi = \frac{e^{-M\sigma_\alpha^2}}{M^2} \quad e^{n\sigma_\alpha^2} [\frac{n^3}{3}\sigma_\alpha^2 \underline{w}^* \underline{w}^{*T} + n\sigma_\eta^2/\sigma_x^2 I] \tag{6.31}$$

where we have used

$$\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6} \sim \frac{n}{3} \tag{6.32}$$

An examination of (6.31) shows that the RLS algorithm diverges as the number of iterations (n) becomes large. This divergence is exponential and cubic with respect to n. It is observed that the divergence terms are due to floating point errors in the calculations of the weight vector update (3.3), i.e., $\alpha_i(n)$. Assuming no errors in the calculation of the update for $\underline{w}(n)$, $\sigma^2\alpha \to 0$ and $M \to n$. Thus (6.31) becomes

$$R_\Psi = \frac{1}{n} \sigma^2/\sigma^2 I \tag{6.33}$$

In otherwords, the error due to the floating point calculation of d(n) is averaged out by the RLS algorithm. Therefore, $\eta(n)$ indeed

behaves as an additive noise term. It will not cause a bias in $\underline{w}(n)$ as it converges to $\underline{w}^*$. However, it will contribute a noise term with variance σ_η^2 to the prediction error [5].

We must also consider the covariance of $\underline{\lambda}(n)$ which contains the information regarding the convergence of the translated weight vector. Define

$$R_\lambda = E\{\underline{\lambda}(n)\ \underline{\lambda}^T(n)\} \tag{6.34}$$

From (6.4) substituting for $\underline{\lambda}(n)$

$$R_\lambda = \underline{\theta}(0)E\Big(\prod_{i=1}^{n}[I-\underline{K}(i)\underline{x}^T(i)+\beta_{NN}(i)]\ \prod_{i=1}^{n}[I-\underline{x}(i)\underline{K}^T(i)+\beta_{NN}(i)]\Big)\underline{\theta}^T(0) \tag{6.35}$$

From (6.15)

$$R_\lambda = \underline{\theta}(0)\ \prod_{i=1}^{n}\Xi\ (i)\ \underline{\theta}^T(0) \tag{6.36}$$

where from (6.19),

$$\Xi(i) = \Big(1-\frac{2}{i}+\frac{N+2}{i^2}+\sigma_\alpha^2+\frac{N+2}{i^2}\sigma_\tau^2-\frac{2\sigma_x^2}{i}\Big)I \tag{6.37}$$

As before, consider the index M for which terms of the order $0(k^{-1})$become insignificant compared to σ_α^2. Then (6.32) becomes

$$\Xi(i) \sim [1+\sigma_\alpha^2]I \qquad i > M \tag{6.38}$$

Hence,

$$R_\lambda = \underline{\theta}(0)\ C_1\ \prod_{i=M}^{n}(1+\sigma_\alpha^2)\underline{\theta}^T(0) \tag{6.39}$$

$$R_\lambda = \underline{\theta}(0)\ C_1\ (1+\sigma_\alpha^2)^{n-M}\underline{\theta}^T(0) \tag{6.40}$$

$$R_\lambda = \underline{\theta}(0)\ C_1\underline{\theta}^T(0)\ e^{(n-M)\sigma_\alpha^2} \tag{6.41}$$

If the floating point noise sources are neglected, $\sigma_\alpha^2 \to 0$. In this case $C_1 \to 0_{NN}$ the null matrix. In otherwords, the algorithm converges to the optimal weight vector.

7. Extension to the Least Mean Squares Algorithm

In this section, the errors introduced by floating point operations in the LMS algorithm are derived using the approach for the RLS algorithm. In the LMS algorithm the weight vector is updated by first calculating the prediction of the desired signal:

$$\hat{d}(n) = \underline{x}^T(n)\underline{W}(n-1) \tag{7.1}$$

From which the error is obtained and used to update the weight vector,

$$e(n) = d(n) - \hat{d}(n) \tag{7.2}$$

$$\underline{w}(n) = \underline{w}(n-1) + \gamma\, e(n)\, \underline{x}(n) \tag{7.3}$$

In (7.3) γ is the loop gain . It controls the convergence rate of the LMS algorithm and must satisfy,

$$\gamma < \frac{2}{\lambda max}$$

where λ_{max} is the maximum eigenvalue of the autocorrelation matrix

$$R_x = E\{\underline{x}(n)\underline{x}^T(n)\} \tag{7.4}$$

Comparing (7.3) to (3.3) we observe that the development leading to (6.1) in the case of the RLS algorithm can be carried out by replacing $\underline{K}(n)$ with $\gamma\underline{x}(n)$ for the LMS algorithm. Hence, for the LMS algorithm, the floating point translated wieght vector becomes:

$$\theta'(n) = \prod_{i=1}^{n} [I - \gamma\underline{x}(i)\underline{x}^T(i) + \beta_{NN}(i)](0) + \sum_{i=1}^{n} \underline{\xi}^T(i)[\prod_{j=i+1}^{n} [I - \gamma\underline{x}(j)\underline{x}^T(j) + \beta_{NN}(j)] \tag{7.5}$$

where

$$\underline{\xi}(i) = \alpha_{NN}(i)\underline{w}^* - \eta(i)\gamma\underline{x}(i) \tag{7.6}$$

and

$$\beta_{NN}(n) = \alpha_{NN}(i) - \mu_{NN}'(n)\gamma\underline{x}(n)\underline{x}T(n) \tag{7.7}$$

Note that the floating point errors, $\alpha_i(n)$, $\mu_i(n)$, $\delta(n)$, $\eta(n)$ now apply to the LMS algorithm.

We can write

$$\underline{\theta}'(n) = \underline{\lambda}(n) + \underline{\Psi}(n) \tag{7.8}$$

We proceed to calculate

$$R_{\underline{\Psi}} = E\{\underline{\Psi}(n)\ \underline{\Psi}^T(n)\} \tag{7.9}$$

To evaluate (7.9) we need to calculate

$$R_\xi(j,K) = E\{\xi(i)\xi^T(j)\} = \begin{matrix} R_\xi(i) & i=j \\ O_{NN} & i\neq j \end{matrix} \tag{7.10}$$

Thus

$$R_\xi(i) = E\{\alpha_{NN}(i)\underline{w}^*\ \underline{w}^{*T}\ \alpha_{NN}(i)\} + E\{\eta^2(i)\gamma^2\underline{x}(i)\underline{x}^T(i)\} \tag{7.11}$$

$$R_\xi(i) = \sigma_\alpha^2\underline{w}^*\underline{w}^{*T} + \sigma_\eta^2\ \gamma^2 R_x \tag{7.12}$$

As in (6.15) for the RLS algorithm define $\Xi(k)$ for the LMS algorithm

$$\Xi(k) = E\{[I-\gamma\underline{x}(k)x^T(k)+\beta_{NN}(k)]\ [I-\gamma\underline{x}(k)\underline{x}^T(k)+\beta_{NN}(k)]\} \tag{7.13}$$

To evaluate (7.13) note that [5]

$$E\{\underline{x}(k)\underline{x}^T(k)\underline{x}(k)\underline{x}^T(k)\} = (N+2)\sigma_x{}^4 I \tag{7.14}$$

Also,

$$E\{\beta_{NN}(k)\beta_{NN}{}^T(k)\} = \sigma_\alpha{}^2 I+(N+2)\gamma^2\sigma_x{}^4(\sigma_\alpha{}^2+\sigma_\mu{}^2+\sigma_\delta{}^2)I-2\sigma_\alpha{}^2\gamma\sigma_x{}^2 I \tag{7.15}$$

Hence (7.13) becomes

$$\Xi(k) = [1-2\gamma\sigma_x{}^2+(N+2)\sigma_x{}^4\gamma^2+\sigma_\alpha{}^2-2\sigma_\alpha{}^2\gamma\sigma_x{}^2+(N+2)\gamma^2\sigma_x{}^4\sigma_\tau{}^2]I$$

$$\Xi(k) = \nu I \tag{7.16}$$

where we have defined

$$\sigma_\tau{}^2 = \sigma_\alpha{}^2 + \sigma_\mu{}^2 + \sigma_\gamma{}^2 \tag{7.17}$$

Note that $\Xi(k)$ is diagonal. Hence,

$$R_\Psi = \sum_{i=1}^{n}\ \prod_{k=i+1}^{n-1} \Xi(k)\ E\{\underline{\xi}^T(i)\underline{\xi}(i)\} \tag{7.18}$$

Or,

$$R_\Psi = \sum_{i=1}^{n} R_\xi \nu^{n-i} = R_\xi \sum_{i=1}^{n} \nu^{n-i} \tag{7.19}$$

$$R_\Psi = R_\xi \sum_{k=0}^{n-1} \nu^k = R_\xi \, (1-\nu)^{-1} \tag{7.20}$$

where we have used

$$\frac{1}{1-\varepsilon} = 1+\varepsilon+\varepsilon^2\ldots \qquad \varepsilon<1 \tag{7.21}$$

(Note $\nu<1$ since $\gamma << \frac{1}{\sigma_x^2}$ for practical purposes).

Hence,

$$\text{Trace } R_\Psi = \left[\sigma_\alpha^2 \|\underline{w}^*\|^2 + \gamma^2 \sigma_\eta^2 N \sigma_x^2\right] \frac{1}{1-\nu} \tag{7.22}$$

For γ small, $\nu \to 1-2\,\gamma\sigma_x^2$

From (7.16)

$$\boxed{\text{Trace } R_\Psi = \frac{\sigma^2 \|w^*\|^2}{2\gamma\sigma_x^2} + \frac{1}{2}\,\gamma N \sigma_\eta^2} \tag{7.23}$$

If we neglect the floating point errors in $\lambda(n)$ in (7.8) then defining,

$$\zeta(n) = e(n) - e'(n) \tag{7.24}$$

where $e(n)$ is the infinite precision prediction error and $e'(n)$ is the floating point error we can show that

$$\sigma_\zeta^2 = E\{\zeta^2(n)\} = \text{Trace}[R_x R_\Psi] + N\sigma_\eta^2 \tag{7.25}$$

We can interpret σ_ζ^2 as the mean square error between the floating point and infinite precision LMS algorithm. Substituting for Trace R_Ψ from (7.23),

$$\boxed{\sigma_\zeta^2 = \frac{1}{2}\,\sigma_\alpha^2 \|w^*\| \, \frac{1}{\gamma} + \frac{1}{2}\,\gamma N \sigma_\eta^2 \sigma_x^2 + N\sigma_\eta^2} \tag{7.24}$$

Consider now the floating point errors introduced in $\underline{\lambda}(n)$. From (6.15) we obtain for the LMS algorithm,

$$R_{\lambda}(n)= \underline{\theta}(0) \prod_{i=1}^{n} \Xi(i)\underline{\theta}^{T}(0) \tag{7.25}$$

where $\Xi(i)$ is defined by (7.13). From, (7.16)

$$R_{\lambda}(n) = \underline{\theta}(0)\underline{\theta}^{T}(0)\nu^{n} \tag{7.26}$$

where

$$\nu = 1-2\gamma\sigma_x^2+(N+2)\gamma^2\sigma_x^4+\sigma_\alpha^2-2\sigma_\alpha^2\gamma\sigma_x^2+\sigma_\tau^2(N+2)\gamma^2\sigma_x^4 \tag{7.27}$$

Examining (7.27) we note that in the absence of floating point errors, $\lim_{n\to\infty}\nu^n\to 0$ must hold for the algorithm to converge. Assuming γ to be very small (this is the case in many applications of the LMS algorithm) we can write (7.27) as

$$\nu \sim 1 - 2\gamma\sigma_x^2 + \sigma_\alpha^2 \tag{7.28}$$

For stability

$$2\gamma\sigma_x^2 << 1 \tag{7.29}$$

Hence,

$$\nu \sim e^{-2\gamma\sigma_x^2} \; e^{\sigma_x^2} \tag{7.30}$$

Substituting into (7.26)

$$R_{\lambda}(n) = \underline{\theta}(0)\underline{\theta}^{T}(0)e^{-2\gamma\sigma_x^2 n} \cdot e^{n\sigma_\alpha^2} \tag{7.31}$$

Hence, the algorithm diverges exponentially for $\gamma \to 0$.

8. Summary

An examination of the results for the RLS algorithm shows that the algorithm is not sensitive to floating point errors in the calculation of the desired signal. The error here leads to an additive noise term. The errors that lead to algorithm divergence (cubic and exponential) are those involved in the calculation of the weight vector update recursion; in particular, the floating point errors due to adding the correction term to the weight vector.

It must be pointed out that the algorithm initially converges, but as the number of iterations becomes large compared to the inverse of the variance of the floating point errors, the algorithm diverges. The initial assumption on the availability of the quantized Kalman gain vector is seen to be justified since the algorithm diverges even with infinite precision calculation of the Kalman gain. Thus independent of the algorithm for efficient computation of the Kalman gain the RLS algorithm will diverge. Furthermore, using our analysis we are able to pin point the floating point operations that lead to divergence and those that only degrade performance in a non-catastrophic manner. These results point to tradeoffs and design issues in the digital implementation of the RLS algorithm.

For the LMS algorithm, the floating point errors increase inversely to the loop gain. As the loop gain approaches zero, the errors increase exponentially with the number of iterations (if $\gamma(n) = \frac{1}{n}$ for example). This was not predicted in [4]. Increasing the loop gain causes the errors due to the desired signal

prediction to increase. In many applications the loop gain is chosen small in order to reduce the variance of the error in the weight vector estimate and the variance of the error due to additive noise (these errors are averaged out by the RLS algorithm). Decreasing the loop gain, however, increases the floating point errors.

References

(1) D.D. Falconer and L. Ljung, "Application of Fast Kalman Estimation to Adaptive Equalization," IEEE Trans. on Comm., Vol. COM-26, No.10, October 1978, pp 1436-1446.

(2) B. Friedlander, "System Identification Techniques for Adaptive Noise Cancelling," IEEE Trans. on Acous.,Speech, and Sig. Processing, Vol. ASSP-30, October 1982.

(3) E.H. Satorius and J.D. Pack, "Application of Least Squares Lattice Algorithms to Adaptive Equalization," IEEE Trans. on Comm., Vol. COM-29, No. 2, February 1981.

(4) C.Caraiscos, and B. Liu, "A Roundoff Error Analysis of the LMS Adaptive Algorithm," IEEE Trans. on Acoustics, Speech and Sig. Proc., Vol. ASSP-32, No. 1, February 1984, pp.34-41.

(5) Sasan Ardalan, S.T. Alexander, "Finite Wordlength Analysis of the Recursive Least Squares Algorithm", Proceedings Eighteenth Annual Asilomar Conference on Circuits, Systems and Computers, Monterey , CA ,November 4, 1984.

(6) Bede Liu, T. Kaneko, "Error Analysis of Digital Filters Realized with Floating-Point Arithmetic," Proceedings of the IEEE, Vol 57, No. 10, October 1969.

(7) A. V. Oppenheim and R. W. Shafer, Digital Signal Processing Englewood Cliffs, NJ: Prentice-Hall, 1975.

(8) A.B. Spirad and D.L. Snyder, Quantization Errors in Floating-Point Arithmetic," IEEE Trans Acous. Speech, and Sig. Proc., vol. ASSP-26, pp. 456-463, Oct. 1978.