Adaptive noise reduction using numerically stable fast recursive least squares algorithm

Farid Ykhlef*†, Madjid Arezki, Abderezzak Guessoum and Daoud Berkani

LATSI laboratory, Department of Electronic, Faculty of Engineering Sciences, University of Blida, Algeria

SUMMARY

This paper is concerned with adaptive noise reduction based on the fast recursive least squares (FRLS) algorithm. It is well known that the fast recursive least squares (FRLS) algorithm suffers from numerical instability when operating under the effects of finite precision arithmetic. Several numerical solutions of stabilization were proposed in the case of stationary signals. In this work a new version of a numerically stable FRLS algorithm (NS-FRLS) is proposed. The stability characteristics of this new stabilized algorithm are analysed. The analysis is based on a linear model for the errors in the states of the adaptive filter. Experimental results confirm the merits of adaptive filtering with the NS-FRLS algorithm over optimum filtering using the solution provided by Wiener–Hopf equations. Copyright © 2006 John Wiley & Sons, Ltd.

Received 30 May 2005; Revised 14 March 2006; Accepted 18 April 2006

KEY WORDS: adaptive noise reduction; fast recursive least squares; numerical stability; normalized least mean squares; recursive least squares; propagation of errors

1. INTRODUCTION

Noise reduction is the subject of research in many different fields. Depending on the environment, the application, the source signals, the noise, and so on, the solutions look very different. For the extraction of signal buried in noise, adaptive noise reduction provides a good solution.

Noise is inevitable in applications of speech processing. Another application strongly influenced by noise is related to the hands-free phones where the background noise reduces the signal to noise ratio (SNR) and the speech intelligibility [1]. One of the main problems with
hands-free operation in a car is related to the high background noise level. The additive car noise could be from the engine, road, wind, bumps, noise when passing a car running in the opposite direction, etc. An effective method should be applied on the noise reduction in order to obtain good communication or phone voice recognition devices [1–3]. Many methods have been employed on noise reduction [4], such as spectral subtraction and the use of microphone arrays [5]. Spectral subtraction has well-known disadvantages such as limited performance at low SNR values and artificial sounding residual noise. Microphone arrays could have a good result, but due to the number of microphones used in the arrays, this is not practical.

Adaptive signal processing evolved from methods developed to enable adaptive control of time-varying systems [6]. It has gained popularity due to the advances in digital technology that have increased computing capacity and broadened the scope of digital signal processing. The key difference between classical signal processing techniques and adaptive signal processing methods is that in the latter we deal with time varying digital systems. When adaptive filters are used to process non-stationary signals, whose statistical properties vary in time, the required amount of prior information is often less than that required for processing via fixed digital filters. Adaptive filters are particularly suitable in applications dealing with non-stationary speech processes. In many applications of noise reduction the changes in signal characteristics could be quite fast. This requires the use of adaptive algorithms, which converge rapidly. The purpose of this contribution is to study an algorithm for adaptive filtering in noise reduction applications.

The two of the most frequently applied algorithms for adaptive noise reduction [7] are the normalized least mean squares (NLMS) [8] and the recursive least squares (RLS) [9] algorithms. Considering these two algorithms, it is obvious that the NLMS algorithm has the advantage of low computational complexity. On the contrary, the high computational complexity is the weakest point of the RLS algorithm but it provides a fast adaptation rate. Thus, it is clear that the choice of the adaptive algorithm to be applied is always a trade-off between computational complexity and fast convergence. Also, it is well known that fast recursive least squares (FRLS) algorithms can produce a good trade-off between convergence speed and computational complexity. However, the FRLS algorithms suffer from problems with numerical instability [10]. In this paper a new version of a numerically stable fast recursive least squares algorithm is introduced for reducing the noise of a speech signal that has been corrupted with noise. We provide also a stability analysis of the FRLS algorithm. This analysis is based on the linearized model of the perturbed states and follows the method used in [11]. The rest of the paper is organized as follows. Section 2 briefly describes each of the implemented adaptive algorithms for real-time programming. It is shown to perform much better than the normalized least mean squares (NLMS) algorithm and while it performs worse than the standard recursive least squares (RLS) algorithm it is shown to be computationally simpler. Section 3 contains the definition of the FRLS algorithm problem and we introduce a new numerically stable version (NS-FRLS) of this algorithm, in the stationary case, based on a first-order model of the propagation of numerical errors. From the analysis of the error propagation, we extract a stability condition for the NS-FRLS algorithm. Section 4 presents a number of experimental results that verify the numerical stability and the good performance of the NS-FRLS algorithm in a noise reduction application. Finally, Section 5 concludes the paper. Notations used in this paper are fairly standard. Boldface symbols are used for vectors and matrices. We also have the
following notations:

- $N$: vector length
- $n$: discrete time index
- $(\cdot)^T$: transpose
- $E\{\cdot\}$: expectation
- $\|\cdot\|$: Euclidean norm

2. ADAPTIVE NOISE REDUCTION

Adaptive noise reduction is usually based on modelling the noise path impulse response with an adaptive finite impulse response (FIR) filter. Figure 1 shows the standard adaptive filtering scheme for adaptive noise reduction using a digital filter with finite impulse response (FIR). The primary input consists of speech $s(n)$ and noise $v(n)$ while the reference input consists of noise $x(n)$ alone. The two noises $x(n)$ and $v(n)$ are correlated and $W_N(n)$ is the impulse response of the noise path. The system tries to reduce the impact of the noise in the primary input exploring the correlation between the two noise signals. This is equivalent to the minimization of the mean-square error $E\{e^2(n)\}$ where

$$e(n) = y(n) - u(n) = s(n) + v(n) - u(n)$$  \hspace{1cm} (1)

Having in mind that by assumption, $s(n)$ is correlated neither with $v(n)$ nor with $x(n)$ we have

$$E\{e^2(n)\} = E\{s^2(n)\} + E\{(v(n) - u(n))^2\}$$  \hspace{1cm} (2)

In other words the minimization of $E\{e^2(n)\}$ is equivalent to the minimization of the difference between $v(n)$ and $u(n)$. Obviously $E\{e^2(n)\}$ will be minimal when $v(n) \approx u(n)$, i.e. when the impulse response of the adaptive filter closely mimics the impulse response of the noise path. The minimization of $E\{e^2(n)\}$ can be achieved by updating the filter taps $W_N(n)$. To reduce noise, cancellers are designed with adaptive transversal FIR digital filters, and based on variants of the normalized least mean squares (NLMS), recursive least squares (RLS) and fast recursive least squares (FRLS) algorithms. The ability of updating tap-weights in adaptive filters is suitable for reducing non-stationary noise.

2.1. Normalized least mean squares

The normalized least mean squares (NLMS) [1] algorithm is perhaps the most widely used algorithm for adaptive filtering applications, primarily due to its simplicity and low
implementation cost. The algorithm is defined by the equations

\[ e(n) = y(n) - W_N^T(n-1)X_N(n) \]  

(3)

\[ W_N(n) = W_N(n-1) - \frac{\delta}{||X_N(n)||^2} e(n)X_N(n) \]  

(4)

where \( X_N(n) = [x(n)x(n-1)\ldots x(n-N+1)]^T \) denotes a vector which summarizes the past of the signal \( x(n) \) over a length of \( N \) points. The stability and convergence properties of NLMS [1] are determined by the step-size parameter \( \delta(0 < \delta < 2) \). If \( \delta \) is too large then the algorithm will not be convergent in a mean square sense. If, on the other hand, \( \delta \) is too small, then the convergence of the algorithm will be very slow. The NLMS algorithm is very computationally efficient and is of computational complexity \( \mathcal{O}(N) \).

### 2.2. Recursive least squares

The classical recursive least squares algorithm (RLS) [12] consists of the following equations:

\[ K_N(n) = - \frac{R_N^{-1}(n-1)X_N(n)}{\lambda + X_N(n)R_N^{-1}(n-1)X_N(n)} \]  

(5)

\[ R_N^{-1}(n) = \lambda^{-1}[R_N^{-1}(n-1) + K_N(n)X_N^T(n)R_N^{-1}(n-1)] \]  

(6)

\[ e(n) = y(n) - W_N^T(n-1)X_N(n) \]  

(7)

\[ W_N(n) = W_N(n-1) - e(n)K_N(n) \]  

(8)

where \( R_N^{-1}(n) \) is the inverse of the time-averaged correlation matrix and \( \lambda \) is a forgetting factor. For the initialization of \( R_N^{-1}(n) \), \( R_N^{-1}(0) = cI_N \), \( c \) being positive constant. \( I_N \) is the identity matrix of \( N \times N \) dimension. The forgetting factor \( \lambda \) in the RLS algorithm is set to a value between \( 0 < \lambda \leq 1 \) [12]. The RLS algorithm is easily found to be of computational and storage complexity \( \mathcal{O}(N^2) \).

### 2.3. Fast recursive least squares

A computational simpler version of fast RLS that seems to be widely used is the fast transversal filter (FTF) algorithm [13, 14]. It is of complexity \( \mathcal{O}(7N) \) which is much lower than that of RLS but, unfortunately, the FTF algorithm suffers from stability problems. In [15] it has been shown that all fast versions of RLS suffer from stability problems when the forgetting factor \( \lambda \) is less than one (which is a requirement for the algorithm to be able to adapt to parameter changes). Therefore, control of the propagation of numerical errors has to be included for the algorithm to work well. The study of this algorithm with the stabilization method and the new version numerically stable proposed in this paper will be investigated in the following paragraph.

Copyright © 2006 John Wiley & Sons, Ltd.  
Int. J. Adapt. Control Signal Process. (in press)  
DOI: 10.1002/acs
3. FRLS ALGORITHM

This algorithm solves the RLS problem by exploiting the time-shift invariance property of the input data [11]. In order to describe the algorithm the following variables are defined:

\( \tilde{e}(n) \): a priori forward prediction-error,
\( \tilde{r}(n) \): a priori backward prediction-error,
\( \pi(n) \): sum of forward prediction-error squares,
\( \beta(n) \): sum of backward prediction-error squares,
\( \gamma_N(n) \): likelihood variable of order \( N \),
\( \gamma_{N+1}(n) \): likelihood variable of order \( N + 1 \),
\( \tilde{K}_N(n) \): normalized Kalman gain vector (or Kalman dual),
\( A_N(n) \): forward prediction filter,
\( B_N(n) \): backward prediction filter,
\( E_0 \): initialization constant,
\( \tilde{K}_{N+1}(n) \): normalized Kalman gain vector of \( (N + 1) \times 1 \) dimension,
\( \tilde{K}_{N+1}^k(n) \): the last element of the \( \tilde{K}_{N+1}(n) \),
\( \theta_A^N \): indicate null vectors of \( N \times 1 \) dimension.

As mentioned in the introduction, for application where the fast convergence rate is vital, NLMS algorithm is not applicable. The more complex RLS algorithm maintains a good rate of adaptation but the prize to be paid is an increased complexity. The FRLS shows superior convergence speed over the NLMS, and it has a low complexity compared to RLS algorithm [3]. This reduction of complexity can lead to numerical instability.

3.1. Fast recursive least squares algorithm

Our main interest is in the application of FRLS algorithms to adaptive FIR filters for the identification of impulse responses. The equations are more complex than for the RLS algorithm and are shown in Table II. For the following, we consider the so-called fast transversal filter (FTF) version of the FRLS algorithm [10].

Due to the versatility as well as its ease of implementation, the fast transversal filters algorithm is attractive for many adaptive filtering applications. However, it is not widely used because of its undesirable tendency to diverge when operating in finite precision arithmetic [14]. The FTF algorithm is a result of taking advantage of redundancies arising from the solution of four transversal filtering problems, each related through their use of the same input data; a filter for each of the one-step forward and backward prediction problems, a filter defining the gain vector of the recursive least squares (RLS) algorithm, and a filter to provide the weight vector corresponding to the desired problem being solved. Combined in a special way, these four transversal filters provide the exact solution to the RLS problem at all times and define the FTF algorithm [14]. FTF is often separated into two parts. The prediction part generates the Kalman gain vector \( \tilde{K}_N(n) \) and the likelihood variable \( \gamma_N(n) \) which are sent to the filtering part. The filtering part uses the output of the prediction part together with the excitation signal, \( x(n) \), and the signal, \( y(n) \), to generate the a priori error output, \( \tilde{e}(n) \). Normally, the prediction part also maintains forward \( A_N(n) \) and backward \( B_N(n) \) prediction vectors and their corresponding prediction error energies as internal variables (\( \pi(n) \) and \( \beta(n) \)). The initialization procedure of the
FTF algorithm shown in Table I, is necessary to update the algorithm. The constant $E_0$ (strictly positive) is the only one which must be correctly selected. During the first iterations, the values taken by the internal variables of the algorithm are closely linked to the choice of the constant $E_0$. In practice, it is necessary to ensure a good start for the algorithm. We can choose for example the constant $E_0$ which verifies the following inequality [10]:

$$E_0 \geq \frac{N}{100} \sigma_x^2$$

(9)

where $\sigma_x^2$ is the energy of the signal $x(n)$. 

---

**Table I.** FRLS algorithm initialization.

| $A_N(n-1) = B_N(n-1) = \tilde{K}_N(n-1) = W_N(n-1) = 0$; $\gamma_N(n-1) = 1$; $\alpha(n-1) = E_0 \lambda^N$; $\beta(n-1) = E_0$. |

---

**Table II.** The FTF algorithm.

Variables available at index $n$:

$A_N(n-1) ; B_N(n-1) ; \tilde{K}_N(n-1) ; W_N(n-1) ; \gamma_N(n-1) ; \alpha(n-1) ; \beta(n-1)$

New information: $x(n)$ and $x(n-N)$

**Prediction part:**

$\tilde{e}(n) = x(n) - A_N(n-1)x_N(n-1)$

$\alpha(n) = \lambda \alpha(n-1) + \gamma_N(n-1)(\tilde{e}(n))^2$

$\gamma_{N+1}(n) = (\lambda \alpha(n-1) + \gamma_N(n-1)) / \alpha(n)$

$\tilde{K}_{N+1}(n) = \left[ \begin{array}{cc} 0 & \alpha(n) \gamma_{N}(n-1) \tilde{K}_N(n-1) \\ \tilde{K}_N(n-1) & 1 / \alpha(n) \end{array} \right]^{-1}$

$A_N(n) = A_N(n-1) - \tilde{e}(n) \gamma_{N}(n-1) \tilde{K}_N(n-1)$

$\tilde{r}(n) = -\lambda \beta(n-1) \tilde{K}_{N+1}^{N+1}(n)$

$\gamma_{N}(n) = (\gamma_{N+1}(n)) / (1 + \gamma_{N+1}(n) \tilde{r}(n) \tilde{K}_{N+1}^{N+1}(n))$

$\tilde{K}_N(n) = \tilde{K}_{N+1}(n) + \tilde{K}_{N+1}^{N+1}(n) \left[ \begin{array}{c} B_N(n-1) \\ -1 \end{array} \right]$

$B_N(n) = B_N(n-1) - \tilde{r}(n) \gamma_{N}(n) \tilde{K}_N(n)$

$\beta(n) = \lambda \beta(n-1) + \gamma_{N}(n) (\tilde{r}(n))^2$

**Filtering part:**

$\varepsilon(n) = y(n) - W_N^T(n-1)x_N(n)$

$W_N(n) = W_N(n-1) - \varepsilon(n) \gamma_{N}(n) \tilde{K}_N(n)$
Although the underlying source for this instability has not fully been understood, the symptoms leading toward it have been well-studied [10, 11]: In finite precision arithmetic an updating expression for one of the parameters in the FTF algorithm, the likelihood variable $\gamma_N(n)$, was found to take on meaningless values just prior to the catastrophic divergence of the algorithm. It is well-known that in exact arithmetic $0 < \gamma_N(n) < 1$, and a precursory condition to divergence results from this condition being violated [14].

3.2. Stability analysis of FTF algorithm

In this part, we study the propagation of errors in all quantities that are recursively computed. The analysis proposed in [11], was conducted for the FTF algorithm. Define the vector containing the states of the FTF algorithm as [16]

$$Z(n) = [A_T^N(n) \ a(n) \ K_T^N(n) \ g_N(n) \ B_T^N(n) \ \beta(n)]$$

(10)

For our stability analysis, the errors of the states at discrete time can be represented as the differences between the true quantities of the states assuming infinite precision calculation and the perturbed quantities in finite precision arithmetic, given by

$$\Delta Z(n) = Z(n) - \hat{Z}(n)$$

(11)

where $\hat{Z}(n)$ contains the perturbed values of the states at time $n$

$$\hat{Z}(n) = [\hat{A}_T^N(n) \ \hat{a}(n) \ \hat{K}_T^N(n) \ \hat{g}_N(n) \ \hat{B}_T^N(n) \ \hat{\beta}(n)]$$

(12)

The true quantities are the exact definitions of the variables, and the perturbed quantities consist of the perturbed variables and the additional noise terms due to finite precision arithmetic. Then, by the nature of the updates of these quantities, the numerical errors within these states at time $n$ are functions of the errors at time $n - 1$.

We assume that the propagation of these errors can be characterized by the linear model

$$\Delta Z(n) = F(n)\Delta Z(n - 1)$$

(13)

with $\Delta Z(n)$ being a vector containing the errors in the states of the system given by

$$\Delta Z(n) = \begin{bmatrix} \Delta a(n) \\ \Delta c(n) \\ \Delta b(n) \end{bmatrix}$$

(14)

where

$$\Delta a(n) = \begin{bmatrix} \Delta A(n) \\ \Delta z(n) \end{bmatrix}; \quad \Delta c(n) = \begin{bmatrix} \Delta K_N(n) \\ \Delta g_N(n) \end{bmatrix}; \quad \Delta b(n) = \begin{bmatrix} \Delta B_N(n) \\ \Delta \beta(n) \end{bmatrix}$$

respectively indicating the errors cumulated until the index time $n$ in the recursive forward, Kalman and backward variables. $\Delta V(n)$ stands for the first-order approximation of the error on the theoretical variable $V(n)$. The matrix $F(n)$ is an approximate transition matrix for error propagation. To obtain the matrix $F(n)$ we use (11) to substitute for the elements of $Z(n)$ and $\hat{Z}(n)$.

To simplify the results, we assume that the magnitudes of the noise terms are small enough such as the products of these noise terms can be neglected. The resulting system can be
written as

\[
F(n) = \begin{bmatrix}
F_{11}(n) & F_{12}(n) & F_{13}(n) \\
F_{21}(n) & F_{22}(n) & F_{23}(n) \\
F_{31}(n) & F_{32}(n) & F_{33}(n)
\end{bmatrix}
\]

with dimension \((3N + 3) \times (3N + 3)\), where the sub-matrices are as \(F_0(n)\) shown in Appendix A, respectively.

Relation (13) can also be represented as

\[
\begin{bmatrix}
\Delta a(n) \\
\Delta c(n) \\
\Delta b(n)
\end{bmatrix} = \begin{bmatrix}
F_{11}(n) & F_{12}(n) & F_{13}(n) \\
F_{21}(n) & F_{22}(n) & F_{23}(n) \\
F_{31}(n) & F_{32}(n) & F_{33}(n)
\end{bmatrix} \begin{bmatrix}
\Delta a(n - 1) \\
\Delta c(n - 1) \\
\Delta b(n - 1)
\end{bmatrix}
\]

Let \(F_{ss}\) be the expected value of the matrix \(F(n)\) in steady-state. Then, if all of the eigen-values of the matrix \(F_{ss}\) are less than one in magnitude, the algorithm is numerically stable locally about its optimum solution. In general, the stability analysis of the \(F(n)\) matrix is very difficult, due to the complexity of some of its components and to its dependency on the input signal [11, 17]. However, in practice, \(\lambda\) is chosen close to 1; so the matrix \(F(n)\) is close to block upper triangular, because \(F_{23}(n)\) is a diagonal matrix with terms scaled by \((1 - \lambda)\) (see Appendix A)

\[
F_{23}(n) = \tilde{K}_{N+1}^{N+1}(n) \begin{bmatrix}
I_N & 0_N^T \\
0_N & \lambda(\gamma_N(n))\tilde{K}_{N+1}^{N+1}(n)
\end{bmatrix}
\]

and

\[
\tilde{K}_{N+1}^{N+1}(n) = -\lambda^{-1}q_{N+1}^T R_{N+1}^{-1}(\lambda) X_{N+1}(n)
\]

in asymptotic mode [17, 18]:

\[
\tilde{K}_{N+1}^{N+1}(n) \approx (1 - \lambda^{-1})q_{N+1}^T R_{N+1}^{-1}(\lambda) X_{N+1}(n)
\]

\[
\tilde{K}_{N+1}^{N+1}(n) \approx -(1 - \lambda)\lambda^{-1}q_{N+1}^T R_{N+1}^{-1}(\lambda) X_{N+1}(n)
\]

where

\[
\tilde{K}_{N+1}(n) = -\lambda^{-1}R_{N+1}^{-1}(\lambda) X_{N+1}(n)
\]

\[
R_{N+1}^{-1}(\lambda) = E\{X_{N+1}(n) X_{N+1}^T(n)\}
\]

Therefore, the sub-system of the backward variables \(\Delta b(n)\) can be approximated by the following first-order linear time-varying equation [19]:

\[
\Delta b(n) = F_{31}(n) \Delta a(n - 1) + F_{32}(n) \Delta c(n - 1) + F_{33}(n) \Delta b(n - 1)
\]

This system is unstable because the sub-matrix \(F_{31}(n)\) has all its eigen-values greater than 1, for \(\lambda < 1\) (see Appendix A). The authors in [18] note that the backward recursive variables are also numerically unstable even if we assume that \(\tilde{K}_{N}(n)\) and \(\gamma_{N}(n)\) are computed exactly (i.e. without...
numerical errors); then, system (13) reduces to

\[
\Delta Z'(n) = \begin{bmatrix} \Delta a(n) \\ \Delta b(n) \end{bmatrix} = \begin{bmatrix} F_{11}(n) & 0_{N+1} \\ F_{31}'(n) & F_{33}'(n) \end{bmatrix} \Delta Z'(n - 1)
\]  

(19)

with

\[
F_{33}'(n) = \begin{bmatrix} I_N & \tilde{\lambda} \gamma_N(n) \tilde{K}_{N+1}^{N+1}(n) \tilde{K}_N(n) \\ 0_N & \tilde{\lambda}(2\tilde{\lambda}^{-1}(n) - 1) \end{bmatrix}
\]

where: \( \theta(n) = \tilde{\lambda} \beta(n - 1) / \beta(n) \in (0, 1) \) and \( \tilde{\lambda}(2\tilde{\lambda}^{-1}(n) - 1) \approx 2 - \tilde{\lambda} \) (for large \( n \) and \( \tilde{\lambda} \) close to 1).

The stability of \( \Delta a(n) \) in Equation (19) is easily verified [17]; however, \( F_{33}'(n) \) has an eigenvalue greater than 1; this bad numerical property of the backward variables comes from the computation of the \( a \ priori \) backward prediction errors [18]:

\[
\bar{r}(n) = -\tilde{\lambda} \beta(n - 1) \tilde{K}_{N+1}^{N+1}(n)
\]

(20)

when exact computations are not assumed for \( \tilde{K}_N(n) \) and \( \gamma_N(n) \) the numerical instability is increased by the connection between \( (\tilde{K}_N(n), \gamma_N(n)) \) and \( (B_N(n - 1), \beta(n - 1)) \), which implies that the \( F_{33}'(n) \) has all eigenvalues greater than 1.

### 3.3. New numerically stable FRLS algorithm version

Several numerical solutions of stabilization, with stationary signals, are proposed in literature [11, 17–23]. On the other hand, the FRLS algorithm is notoriously unstable, but it is possible to maintain stability by using a few more equations. Here, we use the method followed in [18], to propose a new numerically stable version of the FRLS algorithm. This method is based on a first-order model of the propagation of the numerical errors [11]. The general principle is to modify the numerical properties of the algorithm without modifying the theoretical behaviour of the algorithm [18]. In this case, by using some known relationships between the different backward \( a \ priori \) prediction errors, we define a ‘control variable’ \( \xi(n) \), theoretically null, given by

\[
\xi(n) = \bar{r}^c(n) - [(1 - \mu_s)\bar{r}^{f0}(n) + \mu_s\bar{r}^{f1}(n)]
\]

(21)

with

\[
\bar{r}^c(n) = x(n - N) - B_N'(n - 1)X_N(n)
\]

\[
\bar{r}^{f0}(n) = -\tilde{\lambda} \beta(n - 1) \tilde{K}_{N+1}^{N+1}(n)
\]

(22)

\[
\bar{r}^{f1}(n) = -\tilde{\lambda}^{-N} \gamma_{N+1}(n) a(n) \tilde{K}_{N+1}^{N+1}(n)
\]

where relations (22) represent the backward \( a \ priori \) prediction errors, theoretically equal, calculated differently. The scale parameter \( 0 \leq \mu_s \leq 1 \) controls the propagation of the numerical errors in the algorithm. The choice of \( \mu_s \) is found by simulations.

In addition, the choice \( \mu_s = 0 \) corresponds to the stabilized algorithm proposed in [18].

In practice, the variable \( \xi(n) \) is never null on account of the machine precision. To stabilize the algorithm, we use the variable \( \xi(n) \) to calculate a backward \( a \ priori \)
Table III. The NS-FRLS algorithm.

<table>
<thead>
<tr>
<th>Variables available at index n:</th>
</tr>
</thead>
<tbody>
<tr>
<td>(A_N(n-1); B_N(n-1); \hat{K}_N(n-1); W_N(n-1); \gamma_N(n-1); \alpha(n-1); \beta(n-1))</td>
</tr>
</tbody>
</table>

New information: \(x(n)\) and \(x(n-N)\)

Prediction part:
\[
\hat{e}(n) = x(n) - A_N^T(n-1)X_N(n-1)
\]
\[
\gamma_N(n) = \hat{\zeta}(n-1)\hat{K}_N(n-1)
\]
\[
\hat{K}_{N+1}(n) = \begin{bmatrix}
0 \\
\hat{K}_N(n-1)
\end{bmatrix} + (\hat{e}(n))/((\hat{\zeta}(n-1))
\]

Filtering part:
\[
\varepsilon(n) = x(n) - W_N^T(n-1)X_N(n)
\]
\[
W_N(n) = W_N(n-1) - \varepsilon(n)\hat{K}_N(n)
\]

Thereafter, the notation NS-FRLS indicates the numerically stabilized FTF algorithm. For adaptive filtering, the new version of FRLS (NS-FRLS) is summarized in Table III. This NS-FRLS uses also the initialization procedure defined in Table I. This new algorithm has a \(\mathcal{O}(8N)\) computational complexity. However, the analysis of numerical errors can then be made with this stabilized algorithm (NS-FRLS). Writing the propagation of numerical error equations according to the linear model (13), we can obtain a new approximate transition matrix for errors propagation \(F^{NS}(n)\) for the NS-FRLS algorithm (where the exponent \(NS\) corresponds to the new numerically stable version). This matrix is parameterized by \(\mu_s\) (see Appendix B).
Assuming that $F_{33}^{NS}(n)$ is close to the zero matrix; then, the propagation of numerical errors equations for $\Delta b(n)$ can be approximated by a first-order linear time-varying Equation [18] (Equation (18)).

The new matrix $F_{33}^{NS}(n)$ is close to a block upper triangular matrix, because the first $N$ component of its $(N + 1)$th column

$$
(\mu_s - 1)\lambda \theta^{-1}(n)y_N(n)R_{N+1}^{-1}(n)R_N(n)
$$

are scaled in asymptotic mode by $(1 - \lambda)^2$

$$
(1 - \lambda)^2(\mu_s - 1)R_N^{-1}(X)X_N(n)q_N^T R_{N+1}^{-1}(X)X_{N+1}(n)
$$

Then, the diagonal blocks of $F_{33}^{NS}(n)$ are the gains of non-perturbed, first-order:

$$
\Delta B_{N}(n) = \theta^{-1}(n)I_N + 2\theta^{-1}(n)\gamma_N(n)R_N(n)X_N(n)\Delta B_{N}(n - 1) + P_B(n)
$$

$$
\Delta \beta(n) = \lambda(1 + (1 - \mu_s)(1 - \theta^2(n)))\Delta \beta(n - 1) + P_\beta(n)
$$

where the perturbed terms $P_B(n)$ and $P_\beta(n)$ can be found by using Equation (18).

To obtain the stability condition, we constrain all of the eigen-values of $F_{33}$ to be less than one in magnitude. For this step, the choice of the parameter $\mu_s$ was set equal to 0.25 and it has been extensively tested in simulations without any tendency to divergence. Noting also that $\lambda$ is typically chosen to be positive. For this fixed choice of $\mu_s$, the recursive expressions (26) and (27) can be written as

$$
\Delta B_N(n) = \theta^{-1}(n)\{I_N - 2R_N^{-1}(n)X_N(n)X_N^T(n)\} \Delta B_N(n - 1) + P_B(n)
$$

$$
\Delta \beta(n) = (\lambda/4)(7 - 3\theta^{-2}(n))\Delta \beta(n - 1) + P_\beta(n)
$$

where $G_B(n)$ and $G_\beta(n)$ represent the matrix and scalar gains, respectively, and $\gamma_N(n)R_N(n)X_N(n) = -R_N^{-1}(n)X_N(n)$. It is easy to verify that the asymptotic scalar gain, $G_\beta(\infty) = (1/4)(7\lambda - 3/\lambda)$ is bounded between $\pm 1$ for $1/2 < \lambda < 1$.

The matrix gain $G_B(n)$ depends on the statistical properties of the reference input signal $x(n)$ is white Gaussian and with some hypotheses used in the statistical approach for the analysis of the round-off errors, we calculate the power of system (29) in backward prediction vector

$$
E\{\Delta B_N(n)\Delta B_N^T(n)\} = (1/\lambda^2)[1 - 4(1 - \lambda) + 4(N + 2)(1 - \lambda)^2]
$$

The constraint: $(1/\lambda^2)[1 - 4(1 - \lambda) + 4(N + 2)(1 - \lambda)^2]<1$ leads to the following bounds:

$$
1 > \lambda > (4N + 5)/(4N + 7) = 1 - 1/(2N + 3, 5).
$$

From this statistical approach for the analysis of the round-off errors, we can determine the range of the forgetting factor $\lambda$ that guarantees the numerical stability of the algorithm [17]

$$
1 > \lambda > 1/(2N + 3, 5)
$$
To simplify we can write

\[ \lambda > 1 - 1/(pN) \]  \hspace{1cm} (32)

where the parameter \( p \) is a real number higher than 2 to ensure the numerical stability of the algorithm. The matrix \( F_{33}^{NS}(n) \) in asymptotic mode is

\[
F_{33}^{NS}(n) = \begin{bmatrix}
(2 - 1/\lambda)I_N & * \\
* & (1/4)(7\lambda - 3/\lambda)
\end{bmatrix}
\]

Elements denoted by * are not explicitly given as they do not affect the results of the analysis.

4. EXPERIMENTAL RESULTS

As an application in this work, the studied algorithms are applied to the noise reduction problem, which can be modelled as in Figure 2. The noise path corresponds to the unknown path. To cancel the noise, we have to identify this unknown path. Two experiments are reported in this part, which are presented under two sections: test of numerical stability for the stabilized FRLS algorithm and application to adaptive noise reduction. In the NS-FRLS algorithm case, the choice of the forgetting factor must verify the stability condition (Equation (32)).

4.1. Test of numerical stability

It is well-known that in case of the stable FRLS algorithm, the likelihood variable \( \gamma_N(n) \) must be included between 0 and 1. Then, for selected input parameters of the NS-FRLS algorithm, we observe the evolution of this variable.

Initially, it should be noted that this first experiment is conducted without input speech \( s(n) \), because our research is directed only to the stability of the NS-FRLS algorithm. However, to test the numerical behaviour of the numerically stabilized and the unstabilized FRLS algorithm, the alone noise reference input \( x(n) \) was generated using white zero-mean Gaussian noise of unit variance. The impact of the stability of the algorithm on the filtering part is also observed in this part. The unknown path was synthesized by using a finite impulse response filter of an order equals to 32 \( (N = 32) \), to obtain the primary noise signal \( v(n) \). The forgetting factor is set to \( \lambda = 1 - 1/3N \) \( (p = 3) \) and the constant \( E_0 = 1 \). The parameter \( \mu_s \) was experimentally set equal to 0.25.

![Figure 2. Modelled scheme.](image-url)
The system mismatch used for these experiments is computed as a function of time and defined as

$$\text{mis}(n) = \frac{||W_{opt} - W_{N}(n)||^2}{||W_{opt}||^2}$$  \hspace{1cm} (33)$$

where $W_{opt}$ denotes the optimal path.

Figure 3 shows the evolution of the likelihood variable and the system mismatch for both FRLS algorithms (NS-FRLS and FTF). While FTF eventually diverge, the NS-FRLS algorithm did not present any divergence for different data sets.

To demonstrate the validity of the analytical results developed on the propagation of errors in precedent sections and presented in Appendices, the average eigen-values of the transition matrix for errors propagation in steady-state was computed for both algorithms (stabilized and not-stabilized) with the same input parameters. This measurement is defined for $3(N + 1)$ eigen-values as

$$\text{ave}(n) = \frac{1}{3(N + 1)} \sum_{i=1}^{3(N+1)} |\text{eigen-values}(F_{ss})_i|$$  \hspace{1cm} (34)$$

where $F_{ss} = F(n)$ or $F^{NS}(n)$ in steady-state.
Figure 4 shows the ave(n) evolution for FTF and NS-FRLS algorithms, and only 3000 iterations are shown. We can observe that in the FTF case, ave(n) exceeds the value 1, which causes a divergence of the algorithm.

In the NS-FRLS case, ave(n) does not present a value exceeding 1. We verified also that the non-respect of the stability condition causes a divergence of the stabilized algorithm (see Figure 5). The NS-FRLS algorithm is numerically stable when we respect the stability condition (simulations were run for more than $10^7$ samples).

The simulation results shown are obtained by set averaging over 100 independent trials of the experiment. The steady-state system mismatch is limited in simulation to around $-330$ dB because of the quantization errors introduced in the calculation.

4.2. Application to adaptive noise reduction

Various experiments have been performed to illustrate the performance of the NS-FRLS algorithm for noise reduction. The RLS and FRLS algorithms solve the same solution of least squares criterion. For this reason and for the computational complexity of the RLS
Figure 5. NS-FRLS algorithm: (a) stability condition non-respected ($\lambda = 1 - 1/2N$); and (b) stability condition respected ($\lambda = 1 - 1/3N$).

Figure 6. $N = 32$: (a) NLMS algorithm: $\delta = 0.1$; and (b) NS-FRLS algorithm: $\lambda = 0.98958$ ($p = 3$).
algorithm, the results illustrate only the solution of the NS-FRLS algorithm. For a comparison study, the NLMS and NS-FRLS algorithms are implemented for noise reduction application. In this second case, the input speech \( s(n) \) is added to the primary noise signal data obtained in the previous section (Section 4.1) at an initial Signal to Noise Ratio of \(-20\) dB. The used speech signals are sampled at 16 kHz (16bits/mono wav format). Figure 6 shows the pure and the corrupted speech signal. The results after noise reduction are also shown in this figure. The NS-FRLS corresponding to the choice of the stabilizing parameters discussed above have been tested on very long sequences. From the results, it becomes evident that the NS-FRLS is a better algorithm for noise reduction compared with NLMS. These results show also the superiority in the convergence speed of the NS-FRLS algorithm compared to the NLMS algorithm.

A second experiment for adaptive noise reduction is conducted with car noise. The transfer function (noise path or impulse response) used in the simulation was an acoustic transfer function experimentally measured in a duct. Figure 7 shows the measured impulse response in the car truncated at 256 points (equivalent to 0.0160 s) with the frequency response and the speech signal mixed with the vehicle interior noise at an initial SNR of \(-20\) dB.

![Figure 7. The transfer function and the speech signal (pure and noised).](image-url)
With car noise, the quality of noise reduction is more evident after 0.04 s for the NS-FRLS. For the NLMS algorithm, it comes after 0.34 s. The above-mentioned comparisons imply that the NLMS algorithm is simpler in the evaluation process, but it attains lower quality in the noise reduction. On the other hand, the NS-FRLS algorithm achieves higher quality in the disturbing signal reduction. In addition, informal listening tests have been conducted in order to provide a judgement on the quality of the cleaned signal.

5. CONCLUSION

A new numerically stable version of the FRLS algorithm has been presented employing modifications of the numerical properties of some critical recursive variables which have shown numerical instability. We have analysed the error propagation properties of this new version. Our analysis was based on the linearized model for the error propagation. The methodology introduced in this paper solves the stability problem caused by the numerical error propagation in FRLS algorithms. Tests of numerical stability presented in simulation confirm this.

The results indicate also that stability of NS-FRLS algorithms can be achieved for all \( N \) and \( \lambda \in [(1 - 1/2N), 1] \). This is a useful range for \( \lambda \) for tracking purposes in all practical applications. The application of the new NS-FRLS to noise reduction has been studied. In this aspect the obtained results show that the NS-FRLS algorithm is very satisfactory.

The main advantages of the NS-FRLS algorithm are:

- Ability to suppress non-stationary noise;
- Ability to operate with low SNR;
- No speech distortions;
- Fast adaptation;
- Low computational complexity and possible robustness in fixed-point implementations.

Figure 8. \( N = 256 \): (a) NLMS algorithm: \( \delta = 0.25 \); and (b) NS-FRLS algorithm: \( \lambda = 0.99869 \ (p = 3) \).
APPENDIX A

The sub-matrices of the $F(n)$ calculated for numerically unstable FRLS (FTF) are [17]:

\[
F_{11}(n) = \begin{bmatrix}
I_N + \gamma_N(n-1)\tilde{K}_N(n-1)X_N^T(n-1) & 0_N \\
-2\bar{e}(n)\gamma_N(n-1)X_N^T(n-1) & \lambda
\end{bmatrix}
\] (A1)

\[
F_{12}(n) = \begin{bmatrix}
-\bar{e}(n)\gamma_N(n-1)I_N & -\bar{e}(n)\tilde{K}_N(n-1) \\
0_N^T & -(&\bar{e}(n))^2
\end{bmatrix}
\] (A2)

\[
F_{13}(n) = 0_{N+1}
\] (A3)

\[
F_{21}(n) = \begin{bmatrix}
C(n) \\
U(n)
\end{bmatrix}
\] (A4)

\[
F_{22}(n) = \begin{bmatrix}
M'(n) & 0_N^T \\
-2\bar{e}(n)(\gamma_N(n))^2q_N^T & (\rho^2(n)/\theta(n))^2
\end{bmatrix}
\] (A5)

\[
F_{23}(n) = \begin{bmatrix}
\tilde{K}_{N+1}^N(n)I_N & 0_N^T \\
0_N^T & \lambda(\gamma_N(n)\tilde{K}_{N+1}^N(n))^2
\end{bmatrix}
\] (A6)

\[
F_{31}(n) = M'(n)F_{21}(n) + H(n)q_{N+1}^T L(n)
\] (A7)

\[
F_{32}(n) = M'(n)F_{22}(n) + H(n)[q_N^T 0]
\] (A8)

\[
F_{33}(n) = \begin{bmatrix}
\theta^{-1}(n)I_N & \lambda \theta^{-1}(n)\gamma_N(n)\tilde{K}_{N+1}^N(n)\tilde{K}_N(n) \\
0_N^T & \lambda \theta^{-2}(n)
\end{bmatrix}
\] (A9)

where

- $0_N$ : indicates a null matrix of $N \times N$ dimension,
- $I_N$ : indicates a identity matrix of $N \times N$ dimension,
- $0_N^T$ and $0_N^T$ : indicate null vectors of $N \times 1$ dimension and $1 \times N$ dimension, respectively,
- $q_{N+1}$ : allows to extract the $(N + 1)$th component from a vector of order $N + 1$. 

\[
L(n) = \begin{bmatrix}
X^T_N(n-1) & \bar{\epsilon}(n)/z(n-1) \\
-A_N(n-1)X^T_N(n-1) + \bar{\epsilon}(n)I_N & -A_N(n-1)\bar{\epsilon}(n)/z(n-1)
\end{bmatrix}
\right/ (\lambda z(n-1)) \quad \text{(A10)}
\]

\[
C(n) = [I_N \ B_N(n-1)]L(n) \quad \text{(A11)}
\]

\[
U(n) = [U_1 \ u_1] \quad \text{(A12)}
\]

\[
U_1 = 2(\gamma_N(n))^2 \{[\bar{\epsilon}(n) + \bar{r}(n)A_N^N(n-1)]X^T_N(n-1) - \bar{\epsilon}(n)\bar{r}(n)q^T_N\} / (\lambda z(n-1)) \quad \text{(A13)}
\]

\[
u_1 = \lambda \gamma_N(n)(1 - \rho_N(n))/\theta(n) + 2(\gamma_N(n))^2\bar{\epsilon}(n)\bar{r}(n)A_N^N(n-1)/(z(n-1)) \quad \text{(A14)}
\]

\[
\theta(n) = 1 + \bar{r}(n)\gamma_{N+1}(n)K^N_{N+1}(n) \quad \text{(A15)}
\]

\[
\rho_N(n) = \lambda \bar{z}(n-1)/z(n) \quad \text{(A16)}
\]

\[
M^r(n) = \begin{bmatrix}
0 & 0 & \cdots & 0 & B_N^1(n-1) \\
1 & 0 & \cdots & 0 & B_N^2(n-1) \\
0 & 1 & \cdots & 0 & B_N^3(n-1) \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \cdots & 1 & B_N^N(n-1)
\end{bmatrix} \quad \text{(A17)}
\]

\[
M^r(n) = \begin{bmatrix}
-\bar{r}(n)\gamma_N(n)I_N & -\bar{r}(n)K_N(n) \\
\theta_N & (\bar{r}(n))^2
\end{bmatrix} \quad \text{(A18)}
\]

\[
H(n) = \lambda \beta(n-1)\gamma_N(n) \begin{bmatrix}
-K_N(n) \\
2\bar{r}(n)
\end{bmatrix} \quad \text{(A19)}
\]

APPENDIX B

The sub-matrices of the \(F^{NS}(n)\) calculated for NS-FRLS are:

\[
F_{11}^{NS}(n) = F_{11}(n) \quad \text{(B1)}
\]
\[ F_{12}^{\text{NS}}(n) = F_{12}(n) \] (B2)

\[ F_{13}^{\text{NS}}(n) = F_{13}(n) \] (B3)

\[ F_{21}^{\text{NS}}(n) = \begin{bmatrix} C(n) \\ \mathbf{U}^{\text{NS}}(n) \end{bmatrix} \] (B4)

\[ F_{22}^{\text{NS}}(n) = \begin{bmatrix} \mathbf{M}^e(n) & \mathbf{0}_N^T \\ 2\bar{\rho}(n)(\gamma(n))^2 \mathbf{q}_N^T & (\rho_4(n)/\theta(n))^2 - \mu_z^N + 2(n-1)(\gamma(n)\tilde{K}_{N+1}^{N+1}(n))^2 \end{bmatrix} \] (B5)

\[ F_{23}^{\text{NS}}(n) = \begin{bmatrix} \tilde{K}_{N+1}^{N+1}(n)\mathbf{I}_N & \mathbf{0}_N^T \\ 2(\gamma(n))^2 \tilde{K}_{N+1}^{N+1}(n)\mathbf{X}_N^T(n) & (\mu_3 - 1)\lambda(\gamma(n)\tilde{K}_{N+1}^{N+1}(n)) \end{bmatrix} \] (B6)

\[ F_{31}^{\text{NS}}(n) = \mathbf{M}^{\text{NS}}(n)F_{21}^{\text{NS}}(n) + \mathbf{M}^{\text{NS}}(n) + \mathbf{H}^{\text{NS}}(n)\mathbf{q}_N^T \mathbf{L}(n) \] (B7)

\[ F_{32}^{\text{NS}}(n) = \mathbf{M}^{\text{NS}}(n)F_{22}^{\text{NS}}(n) + \mathbf{M}^{\text{NS}}(n) + \mathbf{H}^{\text{NS}}(n)[\mathbf{q}_N^T \mathbf{0}] \] (B8)

\[ F_{33}^{\text{NS}}(n) = \begin{bmatrix} \theta^{-1}(n)\mathbf{I}_N + 2\theta^{-1}(n)\gamma(n)\tilde{K}_N(n)\mathbf{X}_N^T(n) & (\mu_3 - 1)\lambda\theta^{-1}(n)\gamma(n)\tilde{K}_{N+1}^{N+1}(n)\tilde{K}_N(n) \\ 2(1 - \theta^{-1}(n)\bar{\rho}(n)\gamma(n)\mathbf{X}_N^T(n) & \lambda(1 + (1 - \mu_3)(1 - \theta^{-2}(n))) \end{bmatrix} \] (B9)

\[ \mathbf{U}^{\text{NS}}(n) = [\mathbf{U}_1^{\text{NS}} \ u_1^{\text{NS}}] + 2\bar{\rho}(n)(\gamma(n))^2 \mathbf{q}_N^T \mathbf{I}(n) \] (B10)

\[ \mathbf{U}_1^{\text{NS}} = 2(\gamma(n))^2 \bar{\rho}(n)\mathbf{X}_N^T(n)(n-1)/\lambda(n-1) \] (B11)

\[ u_1^{\text{NS}} = \gamma(n)(1 - \rho_4(n)/(\lambda(n-1)\theta(n))) - \mu_z^N(\gamma(n)\tilde{K}_{N+1}^{N+1}(n))^2 \] (B12)

\[ \theta(n) = \theta^{\text{NS}}(n) = 1 + \bar{\rho}(n)\gamma(n)\tilde{K}_{N+1}^{N+1}(n) \] (B13)

\[ \mathbf{M}^{\text{NS}}(n) = \begin{bmatrix} -\bar{\rho}(n)\gamma(n)\tilde{K}_N(n) & -\bar{\rho}(n)\tilde{K}_N(n) \\ \mathbf{0}_N^{-\gamma} & (\bar{\rho}(n))^2 \end{bmatrix} \] (B14)
ADAPTIVE NOISE REDUCTION USING NUMERICALLY STABLE FRLS ALGORITHM

\[ H^{\text{NS}}(n) = \gamma_N(n)\{(1 - \mu_s)\beta(n) + \mu_s\lambda^{-1}\beta(n)\} \]

\[ M^{\text{NS}}(n) = \begin{bmatrix} 0_N & -\mu_s\gamma_N(n)\lambda^{-1}\beta(n) & 2\mu_s\gamma_N(n)\lambda^{-1}\beta(n) \end{bmatrix} \]

\[ M^{\text{NS}}(n) = \begin{bmatrix} 0_N & -2\mu_s\gamma_N(n)\lambda^{-1}\beta(n) & 2\mu_s\gamma_N(n)\lambda^{-1}\beta(n) \end{bmatrix} \]

REFERENCES