International Journal of Innovative
Computing, Information and Control
Volume **9**, Number **1**, January **2013**

ICIC International ©2013 ISSN 1349-4198
pp. 191–206

FAST EUCLIDEAN DIRECTION SEARCH ALGORITHM IN ADAPTIVE NOISE CANCELLATION AND SYSTEM IDENTIFICATION

Seyed Abolfazl Hosseini[1], Sayed Aref Hadei[2,*]
Mohammad Bagher Menhaj[3] and Paeiz Azmi[2]

[1]Department of Electrical Engineering
Islamic Azad University, Shahr-e-Rey Branch
P.O.Box: 18155-144, Tehran, Iran

[2]Department of Electrical and Computer Engineering
Tarbiat Modares University
Jalal Ale Ahmad Highway, P.O.Box: 14115-111, Tehran, Iran
*Corresponding author: a.hadei@ieee.org; a.hadei@hotmail.com

[3]Department of Electrical Engineering
Amirkabir University of Technology (Tehran Polytechnic)
424 Hafez Avenue, P.O.Box: 15875-4413, Tehran, Iran

Received March 2011; revised November 2011

Abstract. *Over the years, many algorithms have been proposed that can be used for adaptive filtering; least mean square (LMS) and normalized least mean square (NLMS) algorithms have been used in a wide range of signal processing applications because of their simplicity in computation and implementation. On the other hand, the best adaptive filter algorithm, i.e., recursive least square (RLS), converges significantly faster than the LMS and NLMS algorithms but its weakest point is the high computational complexity. The numerical instability problem of RLS algorithm motivates the use of simplified or partial RLS algorithms as a viable alternative to full RLS. In particular, we point out that the fast Euclidean direction search (FEDS) which is a recently introduced algorithm for adaptive filtering can indeed be interpreted as such partial RLS algorithms exhibiting a nice tradeoff between complexity and performance, and numerical robustness. This paper describes a new FEDS algorithm in noise cancellation for speech enhancement, interference cancellation and system identification applications. Furthermore, the performance of the introduced algorithm is investigated by using the energy conservation relation. The simulation results demonstrate the good performance of the FEDS algorithm in attenuating the noise and system identification. We also show that the tracking property of the proposed scheme is better than the RLS algorithm. Likewise, simulations are conduced to corroborate the presented studies and show that the theoretical results agree well with the simulation results.*

Keywords: Adaptive filter, Computational complexity, Convergence rate, Energy conservation relation, Fast Euclidean direction search, Noise cancellation, Speech enhancement, Steady-state mean-square error, System identification

1. **Introduction.** There are many adaptive filter algorithms that are widely used for noise cancellation and system identification applications in the area of communication, radar, sonar, biomedical, electronics and control systems, but they either have a high mean-square error with slow convergence rate (such as, LMS and NLMS algorithms) or a high computational complexity with fast convergence rate and low mean-square error (such as, RLS algorithm) [1-6]. Thus, mean-square error (MSE), convergence rate and

computational complexity are three important points that should be considered in selecting of the adaptive algorithms [7,8]. Recently the conjugate gradient (CG) method has been studied for its feasibility in adaptive filtering applications [9-11]. By using this method, the CG algorithm converges to the optimal solution after searching along N conjugate gradient directions without using the estimation of correlation matrix inverse. Therefore, the CG algorithm is useful in some adaptive applications, though the $O(N^2)$ computational cost is too high for sample-by-sample processing. To address this problem, modified CG algorithms were presented by Joo and Bose, and Chang and Wilson, in [12,13]. These algorithms perform one direction search at each iteration that reduces the computational cost to $O(N)$. As shown in [14], the sample-based CG algorithms can have a comparable performance with RLS type algorithms in various applications. In [15,16], an adaptive algorithm based on the Euclidean direction search algorithm as a single channel 1-D adaptive filter was introduced which we called fast Euclidean direction search (FEDS) algorithm. By considering an initial value for the estimated weight $\mathbf{w}$ and a set of linearly independent directions $\{d_1, \ldots, d_N\}$, direction search (DS) method searches in duration of each direction to find a better estimated weight. Keep in your mind that searching through N directions is called one search cycle and before the next search cycle; directions may or may not be modified, and a new starting estimated weight may be chosen. Obviously, among all the linearly independent set of directions, the Euclidean direction set is the simplest one to perform the line search along a direction. The convergence property of the FEDS algorithm is better than the usual LMS and NLMS algorithms and comparable with the RLS algorithm [17]. Fast Euclidean direction search based adaptive algorithm was presented in [18]. This paper provides the performance analysis of the FEDS algorithm by using the energy conservation arguments [19-21]. We also present a general expression for the steady-state and transient analysis of the FEDS algorithm. Furthermore, we demonstrate the good performance of the introduced algorithm in noise cancellation for speech enhancement and system identification applications by simulation discussions. Finally, the results are compared with the classical LMS, NLMS, and RLS adaptive algorithms. Extensive simulations at the end of the paper show the usefulness of the proposed scheme and confirm the theoretical performance of the FEDS algorithm. The rest of this paper is organized as follows. In the next section, we will have a brief review on some known classical adaptive algorithms. Subsequently, the FEDS algorithm will be introduced in Section 3. The steady-state mean-square error and transient performance of the FEDS algorithm will be analyzed in Section 4. In sequel, Section 5 presents the adaptive noise cancellation and system identification setups. Finally, Section 6 analyzes the performance of the introduced approach by computer simulations and numerical results to demonstrate good accuracy of the derived analytical model on the unknown system identification. Section 7 concludes this paper.

Throughout the paper, the following notations are adopted.

$|\cdot|$ Norm of a scalar.

$\|\cdot\|$ Euclidean norm of a vector.

$\|\boldsymbol{t}\|_A^2$ Weighted norm for some arbitrary vector $\boldsymbol{t}$ which is defined as $\boldsymbol{t}^T A \boldsymbol{t}$.

$(\cdot)^T$ Transpose of a vector or a matrix.

$< \cdot, \cdot >^T$ Inner product of two vectors.

$E\{\cdot\}$ Expectation operator.

$A \otimes B$ Kronecker product of matrices A and B.

$\mathrm{vec}(T)$ Creating an $M^2 \times 1$ column vector $\boldsymbol{t}$ through stacking the columns of the $M \times M$ matrix T.

$\mathrm{vec}(\boldsymbol{t})$ Creating an $M \times M$ matrix T from the $M^2 \times 1$ column vector $\boldsymbol{t}$.

$\mathrm{Tr}(\cdot)$ Trace of a matrix.

Also small boldface letters are used to denote vectors (e.g., $\boldsymbol{u}$), small letters are used to denote scalars (e.g., u) and capital letters are used to denote matrices (e.g., U). The symbol I denotes the identity matrix of appropriate dimensions.

2. **Overview on Some Known Adaptive Algorithms.** It is well known that the filter vector update equation for the LMS algorithm is given by [8]

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) + \mu \boldsymbol{u}(n) e(n) \tag{1}$$

where

$$\boldsymbol{u}(n) = [x(n), x(n-1), \ldots, x(n-M+1)]^T \tag{2}$$

and

$$e(n) = d(n) - \boldsymbol{w}^T(n)\boldsymbol{u}(n). \tag{3}$$

In Equation (1), $\boldsymbol{u}(n)$ and $e(n)$ are the input signal and the output error signal, respectively. Furthermore, the estimated weight vector $\boldsymbol{w}$ is the $M \times 1$ column vector of filter coefficient at time n and μ is the step-size that determines the convergence speed and steady-state mean-square error (MSE).

To increase the convergence speed of the LMS algorithm, the NLMS algorithm was proposed which can be stated as [8]

$$\boldsymbol{w}\,(n+1) = \boldsymbol{w}(n) + \frac{\mu}{\|\boldsymbol{u}(n)\|^2} \boldsymbol{u}(n) e(n). \tag{4}$$

The filter vector update equation in RLS algorithm can be obtained as [8]

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) + C^{-1}(n)\boldsymbol{u}(n)e(n) \tag{5}$$

where $C(n)$ is the estimation of the autocorrelation matrix that it is given by

$$C(n) = \sum_{i=0}^{n} \lambda^{n-i} \boldsymbol{u}(i)\boldsymbol{u}^T(i). \tag{6}$$

The λ parameter is the forgetting factor and $0 \ll \lambda < 1$.

3. **Fast Euclidean Direction Search Algorithm.** In this section, we introduce the fast Euclidean direction search algorithm. Subsection 3.1 presents notation and problem description and Subsection 3.2 presents algorithm development.

3.1. **Notation and problem description.** The error signal, $e(n)$, can be expressed as

$$e(n) = d(n) - \sum_{k=0}^{M-1} w_k(n) x(n-k) \tag{7}$$

Considering the samples $n-L+1, n-L+2, \ldots, n$ where we focus on the situation where $L > M$, (7) can be written as

$$\boldsymbol{e}(n) = \boldsymbol{d}(n) - U(n)\boldsymbol{w}(n) \tag{8}$$

where

$$U(n) = [\boldsymbol{u}_0(n), \boldsymbol{u}_1(n), \ldots, \boldsymbol{u}_{M-1}(n)]. \tag{9}$$

In above equation the input signal $\boldsymbol{u}_j(n)$ and the desired signal samples are defined through

$$\boldsymbol{u}_j(n) = [x(n-j), x(n-j-1), \ldots, x(n-j-L+1)]^T \tag{10}$$

$$\boldsymbol{d}(n) = [d(n), d(n-1), \ldots, d(n-L+1)]^T \tag{11}$$

and $\boldsymbol{e}(n)$ is defined similarly. The adaptive filtering problem can now be formulated as the task of finding the update for $\boldsymbol{w}(n)$, at each time instant n, such that the error is made as small as possible.

In sequel, note that $U(n)\boldsymbol{w}(n)$ can be written as

$$U(n)\boldsymbol{w}(n) = \sum_{k=0}^{M-1} w_k(n)\boldsymbol{u}(n-k) \tag{12}$$

i.e., as a weighted sum of the $U(n)$ columns with the $\boldsymbol{w}(n)$ elements being the weighting factors. A greedy algorithm for successively building (better) approximations to a given vector using linear combinations of vectors from a given set is the matching pursuit (MP) algorithm. Inspired by this algorithm, conceived and developed in another context and with other motivations than those of this paper, we devise a procedure for recursively building an approximation to $\boldsymbol{d}(n)$ using linear combinations of the $U(n)$ columns.

3.2. **Algorithm development.** By assuming an approximation for $\boldsymbol{d}(n-1)$ at time $n-1$ through $U(n-1)\boldsymbol{w}\,(n-1)$, the *a priori* approximation error at time n is given by

$$\boldsymbol{e}_0(n) = \boldsymbol{d}(n) - U(n-1)\boldsymbol{w}(n-1). \tag{13}$$

In building a better approximation through the update of only one coefficient in $\boldsymbol{w}(n-1)$, the new error can be written as

$$\boldsymbol{e}_1(n) = \boldsymbol{d}(n) - \left(U(n)\boldsymbol{w}(n-1) + U(n)w_{j_0(n)}^{\text{update}}(n)\boldsymbol{F}_{j_0(n)}\right) \tag{14}$$

where $j_0(n)$ is the index of the coefficient to be updated in the zero'th P-iteration at time n and $\boldsymbol{F}_j$ is the $M \times 1$ vector with 1 in position j and 0 in all other positions. Intuitively, it would make sense to select $j_0(n)$ as the index corresponding to that column of $U(n)$ that is most similar to the a priori approximation error. Thus, coefficient $j(n)$ has been identified as the one to update. One simple approach in selecting $j(n)$ is incrementing it sequentially by n modulo M. Therefore, after selecting the index $j_0(n)$, the corresponding filter coefficient is given by

$$w_{j_0(n)}(n) = w_{j_0(n)}(n-1) + w_{j_0(n)}^{\text{update}}(n) \tag{15}$$

where $w_{j_0(n)}^{\text{update}}(n)$ is the projection value of the $\boldsymbol{e}_0(n)$ onto the unit vector with direction which is given by $\boldsymbol{u}_{j_0(n)}(n)$, i.e.,

$$w_{j_0(n)}^{\text{update}}(n) = \frac{< \boldsymbol{e}_0(n), \boldsymbol{u}_{j_0(n)}(n) >}{\left\|\boldsymbol{u}_{j_0(n)}(n)\right\|^2} \tag{16}$$

Thus, the zero'th P-iteration updates the filter vector as follows:

$$\boldsymbol{w}^{(0)}(n) = \boldsymbol{w}(n-1) + w_{j_0(n)}^{\text{update}}(n)\boldsymbol{F}_{j_0(n)}. \tag{17}$$

To have control on the convergence speed and stability of the algorithms, we introduce the step-size in the algorithm as follows:

$$\boldsymbol{w}^{(0)}(n) = \boldsymbol{w}(n-1) + \mu w_{j_0(n)}^{\text{update}}(n)\boldsymbol{F}_{j_0(n)}. \tag{18}$$

Substituting (18) into (14), the updated error expression can be written as

$$\boldsymbol{e}_1(n) = \boldsymbol{d}(n) - U(n)\boldsymbol{w}^{(0)}(n). \tag{19}$$

If we want to do more than one P-iteration at time n, the procedure described above starting with finding the maximum projection of $\boldsymbol{e}_0(n)$ onto a column of $U(n)$ can be repeated with $\boldsymbol{e}_1(n)$ and similar (16). This can be repeated as many times as desired, say P times, leading to a sequence of coefficient updates

$$w_{j_0(n)}(n), w_{j_1(n)}(n), \ldots, w_{j_{P-1}(n)}(n). \tag{20}$$

Note that if $P > 2$ it is entirely possible that one particular coefficient is updated more than once at a given time n. The resulting filter coefficient vector after P iterations at

time n is denoted $\boldsymbol{w}^{(P-1)}(n)$, but where there is no risk of ambiguity, we shall refer to this filter vector simply as $\boldsymbol{w}(n)$. It is interesting to note that a slightly different, but equivalent, procedure to the one described above would result if we try to find the least squares solution to the over determined set of equations (remember $L > M$)

$$U(n)\boldsymbol{w}(n) = \boldsymbol{d}(n). \tag{21}$$

Subject to the constraint that, given an initial solution, say $\boldsymbol{w}_0(n)$, we are allowed to adjust only one element of this vector.

From the above, it is evident that the key computations of our adaptive filter algorithm are those of (16). Making use of (13) and (12), it can be found that

$$w_{j_0(n)}^{\text{update}}(n) = \frac{1}{\left\|\boldsymbol{u}_{j_0(n)}(n)\right\|^2}\Big\{ < \boldsymbol{d}(n), \boldsymbol{u}_{j_0(n)}(n) > - \sum_{k=0}^{M-1} w_k(n-1) < \boldsymbol{u}_k(n), \boldsymbol{u}_{j_0(n)}(n) > \Big\} \tag{22}$$

These are the pertinent equations if one coefficient update, i.e., one P-iteration is performed for each new signal sample. Finding the update of (22) involved the very little additional work. It is instructive to explicitly state these equations also for iteration no. $i > 0$ at time n

$$w_{j_i(n)}^{\text{update}}(n) = \frac{1}{\left\|\boldsymbol{u}_{j_i(n)}(n)\right\|^2}\Big\{ < \boldsymbol{d}(n), \boldsymbol{u}_{j_i(n)}(n) > - \sum_{k=0}^{M-1} w_k^{(i-1)}(n-1) < \boldsymbol{u}_k(n), \boldsymbol{u}_{j_i(n)}(n) > \Big\} \tag{23}$$

From these equations it is evident that some terms depend only on n and they need to be computed once for each n and can subsequently be used unchanged for all P-iterations at time n. Other terms depend on both n and the P-iteration index and must consequently be updated for each P-iteration. Since we must associate the update depending only on n with iteration no. 0, this is the computationally most expensive update.

From the above it is evident that the inner products $< \boldsymbol{d}(n), \boldsymbol{u}_j(n) >$ and $< \boldsymbol{u}_k(n), \boldsymbol{u}_j(n) >$ play prominent roles in the computations involved in the algorithm. As formulated up to this point, obvious recursions for these inner products are given by

$$< \boldsymbol{d}(n), \boldsymbol{u}_j(n) >= < \boldsymbol{d}(n-1), \boldsymbol{u}_j(n-1) > + d(n)x(n-j) - d(n-L)x(n-j-L) \tag{24}$$

and

$$< \boldsymbol{u}_k(n), \boldsymbol{u}_j(n) >= < \boldsymbol{u}_k(n-1), \boldsymbol{u}_j(n-1) > + x(n-k)x(n-j) - x(n-k-L)x(n-j-L) \tag{25}$$

4. **Performance Analysis of a FEDS Algorithm.** In this section, the performance analysis of the FEDS algorithm is presented. The filter vector update equation for the FEDS algorithm can be represented as

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) + C^{-1}(n)U(n)\boldsymbol{e}(n) \tag{26}$$

where $C^{-1}(n) = \frac{1}{\left\|\boldsymbol{u}_{j(n)}\right\|^2} i_{j(n)}$. The $i_{j(n)}$ is an $M \times M$ matrix with a 1 in position (j, j) and zeros in all other positions. Besides, the following assumptions are made [7,8].

Assumption **A1:** The noise $v(n)$ is i.i.d and statistically independent of the regression matrix $U(n)$ with variance $\sigma_v^2 = E|v(n)|^2$. Moreover, the random variables $d(n)$, $v(n)$, $U(n)$ have zero mean.

Assumption **A2:** In steady-state condition, $\boldsymbol{w}_t(n)$, $\boldsymbol{w}(n)$ and $\boldsymbol{w}_t(n) - \boldsymbol{w}(n)$ are statistically independent of the input signal.

The learning curve of an adaptive filter algorithm is defined by the evolution of the expected squared a priori error at time n, i.e., as $E\{\boldsymbol{e}_a^2(n)\}$.

To proceed, as done in [8], the *a priori* error can be defined as

$$\boldsymbol{e}_a(n) = \boldsymbol{u}^T(n)[\boldsymbol{w}_t - \boldsymbol{w}(n)] \tag{27}$$

where $\boldsymbol{w}_t(n)$ is an unknown filter vector that we wish to estimate and by defining weight error-vector as $\boldsymbol{\varepsilon}(n) = \boldsymbol{w}_t - \boldsymbol{w}(n)$, (27) can be rewritten as

$$\boldsymbol{e}_a(n) = \boldsymbol{u}^T(n)\boldsymbol{\varepsilon}(n). \tag{28}$$

In sequel, if we evaluate the squared from both sides of (28) and by taking the expectation from both of its sides, we can achieve the following equality

$$E\{\boldsymbol{e}_a^2(n)\} = E\{\boldsymbol{\varepsilon}^T(n)\boldsymbol{u}(n)\boldsymbol{u}^T(n)\boldsymbol{\varepsilon}(n)\} \tag{29}$$

By using the assumptions A1 and A2, we have

$$E\{\boldsymbol{e}_a^2(n)\} = E\{\boldsymbol{\varepsilon}^T(n)R_{uu}\boldsymbol{\varepsilon}(n)\} = E\{\|\boldsymbol{\varepsilon}(n)\|_{R_{uu}}^2\} \tag{30}$$

where again the definition of the autocorrelation matrix $R_{uu} = E\{\boldsymbol{u}(n)\boldsymbol{u}^T(n)\}$ has been used. Thus, to find the learning curve, we need to find $E\{\|\boldsymbol{\varepsilon}(n)\|_{\Sigma}^2\}$, where Σ is some arbitrary squared symmetric matrix with dimension commensurate. To proceed, the desired signal $d(n)$ is modeled by [20]

$$d(n) = \boldsymbol{u}^T(n)\boldsymbol{w}_t + v(n) \tag{31}$$

which we prefer to express as

$$\boldsymbol{d}(n) = U(n)\boldsymbol{w}_t + \boldsymbol{v}(n) \tag{32}$$

where $\boldsymbol{v}(n)$ is measurement noise that assumed to be independent of the input signal matrix $U(n)$ and substituting (32) into (8), the output error can be expressed in terms of $\boldsymbol{e}_a(n)$ as [8]

$$\boldsymbol{e}(n) = \boldsymbol{e}_a(n) + \boldsymbol{v}(n). \tag{33}$$

Furthermore, Equation (26) can be rewritten in terms of weight-error vector as

$$\boldsymbol{\varepsilon}(n+1) = \boldsymbol{\varepsilon}(n) - C^{-1}(n)U(n)\boldsymbol{e}(n). \tag{34}$$

By considering the assumptions A1 and A2, substituting (28) into (29) and applying it instead of $\boldsymbol{e}(n)$ in (34) and evaluating the weighted energies from both sides of it and by taking the expectation from both of its sides, we get

$$E\{\|\boldsymbol{\varepsilon}(n+1)\|_{\Sigma}^2\} = E\{\|\boldsymbol{\varepsilon}(n)\|_{\Sigma'}^2\} + E\{\boldsymbol{v}^T(n)U^{\Sigma}(n)\boldsymbol{v}(n)\} \tag{35}$$

where

$$\begin{aligned}\Sigma' =& \Sigma - \Sigma E\{C^{-1}(n)U(n)U^T(n)\} \\ &- E\{U(n)U^T(n)C^{-T}(n)\}\Sigma + E\{U(n)U^{\Sigma}(n)U^T(n)\}\end{aligned} \tag{36}$$

and

$$U^{\Sigma}(n) = U^T(n)C^{-T}(n)\Sigma C^{-1}(n)U(n). \tag{37}$$

The forms of our expressions exactly match to those presented in [20] for the affine projection algorithm (APA). The difference is only in the definition of the various quantities involved. From this observation, we can rely directly on the results of [20] to establish

$$E\{\|\boldsymbol{\varepsilon}(n+1)\|_{\boldsymbol{\sigma}}^2\} = E\{\|\boldsymbol{\varepsilon}(n)\|_{G\boldsymbol{\sigma}}^2\} + E\{\boldsymbol{v}^T(n)U^{\Sigma}(n)\boldsymbol{v}(n)\} \tag{38}$$

where

$$\boldsymbol{\sigma} = \text{vec}(\Sigma), \quad \Sigma = \text{vec}(\boldsymbol{\sigma}). \tag{39}$$

In above equation (left portion) σ is a vector of column of Σ stacked under each other, and (right portion) Σ is a matrix which is found by taking equal length sub-vectors of $\boldsymbol{\sigma}$ and putting them beside each other. Furthermore, for notational convenience the notation $E\{\|\boldsymbol{\varepsilon}(n+1)\|^2_{\boldsymbol{\sigma}}\}$ to be interpreted as the Σ-weighted norm of $\boldsymbol{\varepsilon}(n+1)$, i.e., as $E\{\|\boldsymbol{\varepsilon}(n+1)\|^2_{\Sigma}\}$. Finally, $M^2 \times M^2$ matrix G is given by

$$G = I - E\{Q(n)\} \otimes I - I \otimes E\{Q(n)\} + E\{Q(n) \otimes Q(n)\} \tag{40}$$

where the identity matrix has dimension $M^2 \times M^2$, the other two identities have dimensions $M \times M$ and $M \times M$ matrix $Q(n)$ is given by

$$Q(n) = X(n)U^T(n)C^{-T}(n). \tag{41}$$

By considering assumption A1, the second term of the right hand side of (38), it can be rewritten as

$$E\{\boldsymbol{v}^T(n)U^{\Sigma}(n)\boldsymbol{v}(n)\} = \sigma_v^2 \boldsymbol{\sigma}^T \boldsymbol{\gamma} \tag{42}$$

where $\sigma_v^2 = E\{v^2(n)\}$ and

$$\boldsymbol{\gamma} = \text{vec}(E\{[C^{-1}(n)U(n)U^T(n)C^{-T}(n)]\}) \tag{43}$$

by using Equation (42), (38) can be rewritten as

$$E\{\|\boldsymbol{\varepsilon}(n+1)\|^2_{\boldsymbol{\sigma}}\} = E\{\|\boldsymbol{\varepsilon}(n)\|^2_{G\boldsymbol{\sigma}}\} + \sigma_v^2 \boldsymbol{\sigma}^T \boldsymbol{\gamma}. \tag{44}$$

By focusing on the learning curve, substituting R_{uu} instead of the Σ and defining $\boldsymbol{r}_{uu} = \text{vec}(R_{uu})$, it can be found that

$$E\{\boldsymbol{e}_a^2(n)\} = E\{\|\boldsymbol{\varepsilon}(n)\|^2_{\boldsymbol{r}_{uu}}\} = E\{\|\boldsymbol{\varepsilon}(0)\|^2_{G^n\boldsymbol{r}_{uu}}\} + \sigma_v^2 \boldsymbol{\gamma}^T\{I + G + \cdots G^{n-1}\}\boldsymbol{r}_{uu}. \tag{45}$$

This expression is easy to compute recursively once we have estimates for G and R_{uu}. Such estimates are easily obtained from a single realization of the signals involved in the adaptive filter. We can modify the algorithm by allowing more than one coefficient update at each time instant giving better performance at the expense of a somewhat higher computational cost. Fortunately, the additional coefficient updates are considerably cheaper than the first [21]. Depending on the number of filter coefficient updates performed at each time instant n, we denote this number by P, and we shall refer the algorithm as FEDS-P. For $P > 1$, we get a somewhat more involved expression for $C^{-1}(n)$.

The mean-square error (MSE) of any adaptive filter is defined as limiting value of $E|\boldsymbol{e}(n)|$; i.e., MSE $= \lim_{n\to\infty} E|\boldsymbol{e}(n)|^2$. Moreover, when the received data $d(n)$ and $x(n)$ satisfies stationary and non-stationary models assumption [8], the MSE is given by

$$\text{MSE} = \sigma_v^2 + \lim_{n\to\infty} E\{|\boldsymbol{e}_a(n)|^2\} \tag{46}$$

that the limiting variance of the *a priori* error $\boldsymbol{e}_a(n)$ is called the excess mean-square error (EMSE) of the estimation, EMSE $= \lim_{n\to\infty} E\{|\boldsymbol{e}_a(n)|^2\}$. Therefore, the MSE and EMSE define each other via MSE = EMSE$+\sigma_v^2$. Thus, considering the Equation (29), EMSE with $n \to \infty$ is given by

$$\text{EMSE} = \sigma_v^2 \boldsymbol{\gamma}^T\{I - G\}^{-1}\boldsymbol{r}_{uu}. \tag{47}$$

Furthermore, mean-square coefficient deviation (MSD) with replace the $\boldsymbol{r}_{uu} = \text{vec}\{I\}$ into (47) is given by

$$\text{MSD} = \sigma_v^2 \boldsymbol{\gamma}^T\{I - G\}^{-1}\text{vec}\{I\} \tag{48}$$

5. Applications.

5.1. **Adaptive noise cancellation.** Figure 1 shows the adaptive noise cancellation (ANC) setup. In this application, the corrupted signal passes through a filter that tends to suppress the noise while leaving the signal unchanged. This process is an adaptive process, which means it cannot require a priori knowledge of signal or noise characteristics. Adaptive noise cancellation algorithms utilize two or more microphones (sensor) in which one microphone is used to measure the speech + noise signal while the other is used to measure the noise signal alone. The technique adaptively adjusts a set of filter coefficients so as to remove the noise from the noisy signal. This technique, however, requires that the noise component in the corrupted signal and the noise in the reference channel have high coherence. Unfortunately this is a limiting factor, as the microphones need to be separated in order to prevent the speech being included in the noise reference and thus being removed. With large separations the coherence of the noise is limited and this limits the effectiveness of this technique. In summary, to realize the adaptive noise cancellation, we use two inputs and an adaptive filter. One input is the signal corrupted by noise (Primary Input, which can be expressed as $s(n)+n_0(n)$). The other input contains noise related in some way to that in the main input but does not contain anything related to the signal (Noise Reference Input which can be expressed as $n_1(n)$). The noise reference input passes through the adaptive filter and output $y(n)$ is produced as close a replica as possible of $n_0(n)$. The filter readjusts itself continuously to minimize the error between $n_0(n)$ and $y(n)$ during this process. Then the output $y(n)$ is subtracted from the primary input to produce the system output $e = s + n_0 - y$, which is the denoised signal. Assume that s, n_0, n_1 and y are statistically stationary and have zero means. Suppose that s is uncorrelated with n_0 and n_1, but n_1 is correlated with n_0. We can get the following equation of expectations

$$E[e^2] = E[s^2] + E[(n_0 - y)^2] \tag{49}$$

when the filter is adjusted so that $E[e^2]$ is minimized, $E[(n_0 - y)^2]$ is also minimized. So the system output can serve as the error signal for the adaptive filter. In Figure 1, we model the signal path from the noise source to primary sensor as an unknown FIR channel $\boldsymbol{w}_{\mathrm{e}}$. Applying the adaptive filter to reference noise at reference sensor, we then employ an adaptive algorithm to train the adaptive filter to match or estimate the characteristics of unknown channel $\boldsymbol{w}_{\mathrm{e}}$. If the estimated characteristics of unknown channel have negligible differences compared with the actual characteristics, we should be able to successfully cancel out the noise component in corrupted signal to obtain the desired signal. Notice that both of the noise signals for this configuration need to be uncorrelated to the signal $s(n)$. In addition, the noise sources must be correlated to each other in some way, preferably equal, to get the best results. Due to the nature of the error signal, the error signal will never become zero. The error signal should converge to the signal $s(n)$, but not converge to the exact signal. In other words, the difference between the signal $s(n)$ and the error signal $e(n)$ will always be greater than zero. The only option is to minimize the difference between those two signals.

5.2. **System identification.** To give an indication of the performance of the proposed algorithm, we present some results obtained when applying the algorithm in a system identification problem. The setup is depicted in Figure 2. The system to be identified is selected as

$$\begin{aligned} P(z) = {} & 0.1483 + 0.2595z^{-1} - 0.0318z^{-2} - 0.1536z^{-3} \\ & - 0.2118z^{-4} + 0.1059z^{-5} + 0.5295z^{-6} + 0.7314z^{-7} \end{aligned} \tag{50}$$

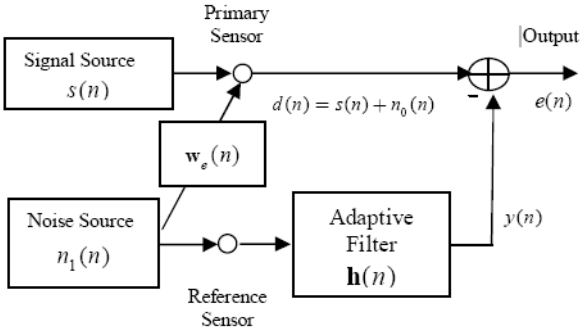


FIGURE 1. Adaptive noise cancellation setup

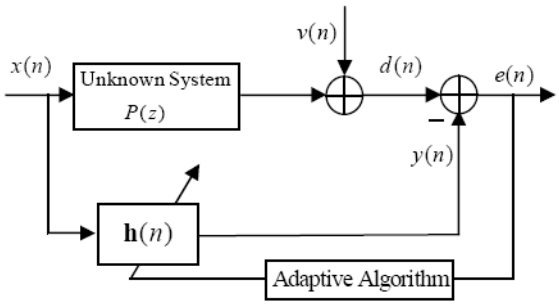


FIGURE 2. System identification setup

The added noise has a variance of $\sigma_v^2 = 10^{-3}$ and we use two different input signals; $u(n)$, in our experiments. The input signals are generated from the first order autoregressive (AR) signal which is defined as

$$u(n) = \rho u(n-1) + w(n) \tag{51}$$

where $w(n)$ is either white Gaussian noise or uniformly distributed input signal between -1 and 1. In the case of colored Gaussian input ρ is set to 0.9 and for the colored uniform input, ρ is set to 0.5.

6. **Simulation Studies.** In this section, we evaluate the performance of each algorithm in noise cancellation and system identification setups as shown in Figures 1 and 2. Extensive computer simulations have been conducted to demonstrate the good performance of the FEDS algorithm in comparison with the other classical adaptive algorithms.

The original, primary and reference signals in noise cancellation are from the reference [22] and shown in Figure 3. The original speech is corrupted with office noise. The signal to noise ratio (SNR) of the primary signal is -10.2180 dB. The order of the filter is set to $M = 8$. The parameter μ is set to 0.002 for the LMS algorithm and 0.005 for the NLMS algorithm. Figure 4(a) shows the filtered output signal and the mean-squared error (learning curve) for the LMS algorithm. The SNR of the filtered signal is calculated for this experiment. The SNR improvement (SNRI) is defined as the final SNR minus the original SNR. The SNRI in the LMS algorithm is 13.5474. Figure 4(b) shows the results of the NLMS algorithm in noise cancellation setup. We can see from Figures 4(a) and 4(b) that the convergence speed of the NLMS algorithm is faster than the LMS algorithm. This fact can be seen from both filtered output and learning curve. Moreover, the SNRI in the NLMS algorithm is 17.1056. Figure 4(c) shows the results of the RLS algorithm

in noise cancellation setup. In this algorithm, the parameter λ is set to 0.99. The results show that the RLS algorithm has faster convergence speed in comparison with the LMS and the NLMS algorithms. The SNRI in this algorithm is 29.7355. In Figure 4(d), the results of the FEDS algorithm are presented and the parameters are set to $L = 25$, $P = 8$, $\mu = 0.002$. The results show that the FEDS algorithm has faster convergence speed and good performance than the LMS and the NLMS algorithms and comparable with the RLS algorithm. The SNRI in this algorithm is 20.9527. Keep in your mind that the computational complexity of the proposed scheme is lower than the RLS algorithm. Table 1 summarizes the SNRI results of the LMS, NLMS, RLS algorithms and the FEDS algorithm.

For the sake of comparison, the filter coefficients evolutions of the LMS, NLMS, RLS and FEDS algorithms are demonstrated in Figure 5. These simulation results show that the performance of the FEDS algorithm is better than the LMS and the NLMS algorithms and comparable with the RLS algorithm. In order to obtain the optimum filter order of the FEDS algorithm, we changed the order of the filter from 1 to 300 and then calculated SNRI for the each order of the filter. Figure 6(a) shows the SNRI versus the order of the filter and can be observed that the FEDS algorithm has the maximum SNRI in $M = 8$. In this simulation, the parameters are set to $L = 25$, $P = 1$, $\mu = 0.002$. Figure 6(b) shows the SNRI versus L and demonstrates that the FEDS algorithm has the maximum SNRI in $L = 25$. In this simulation, the parameters are set to $M = 8$, $P = 1$, $\mu = 0.002$. Figures 6(c) and 6(d) show the SNRI versus μ and P, respectively. The results show that the FEDS algorithm has best performance for $\mu = 0.002$ and $P = 8$.

In sequel, to analyze the convergence and the tracking properties of the FEDS algorithm, we have simulated the FEDS algorithm in the system identification setup. Figure 7 shows the learning curves of the FEDS algorithm. The simulated learning curves are obtained by ensemble averaging over 200 independent trials. In Figures 7(a) and 7(b), we compared the FEDS algorithm with the NLMS and the RLS algorithms for two different input signals. From Figure 7(b) can be observed that by increasing the parameter P, the performance of the FEDS algorithm will be close to the RLS algorithm. In Figure 8, the tracking property of the FEDS algorithm and the RLS algorithm is compared together for two uniform and Gaussian input signals. The unknown signal changes at iteration 500 by multiplying the unknown coefficient with 0.4. From this results can be seen that the FEDS algorithm has good tracking property in comparison with the RLS algorithm. In sequel, we demonstrate the theoretical results by carrying out simulation in the system identification configuration. Figure 9 shows the theoretical and simulated learning curves of the FEDS algorithm for different input signals and different values of P. The theoretical learning curves are obtained from (45) and from these figures can be observed the good agreement between the proposed theoretical results and the simulated learning curves.

TABLE 1. SNR improvement in dB

Algorithm	LMS	NLMS	RLS	FEDS
SNRI (dB)	13.5474	17.1056	29.7355	20.9527

7. **Conclusions.** In this paper, we presented a new adaptive algorithm based on a cyclic Euclidean directional search which is named fast Euclidean direction search algorithm. For the sake of comparison and demonstrating the good performance of our proposed scheme, we compared it with classical adaptive filters, such as the LMS, NLMS and RLS algorithms

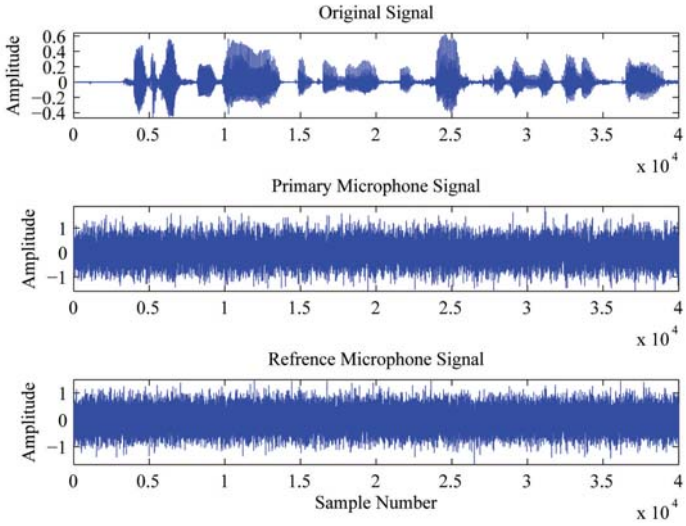


FIGURE 3. Original, primary and reference signals

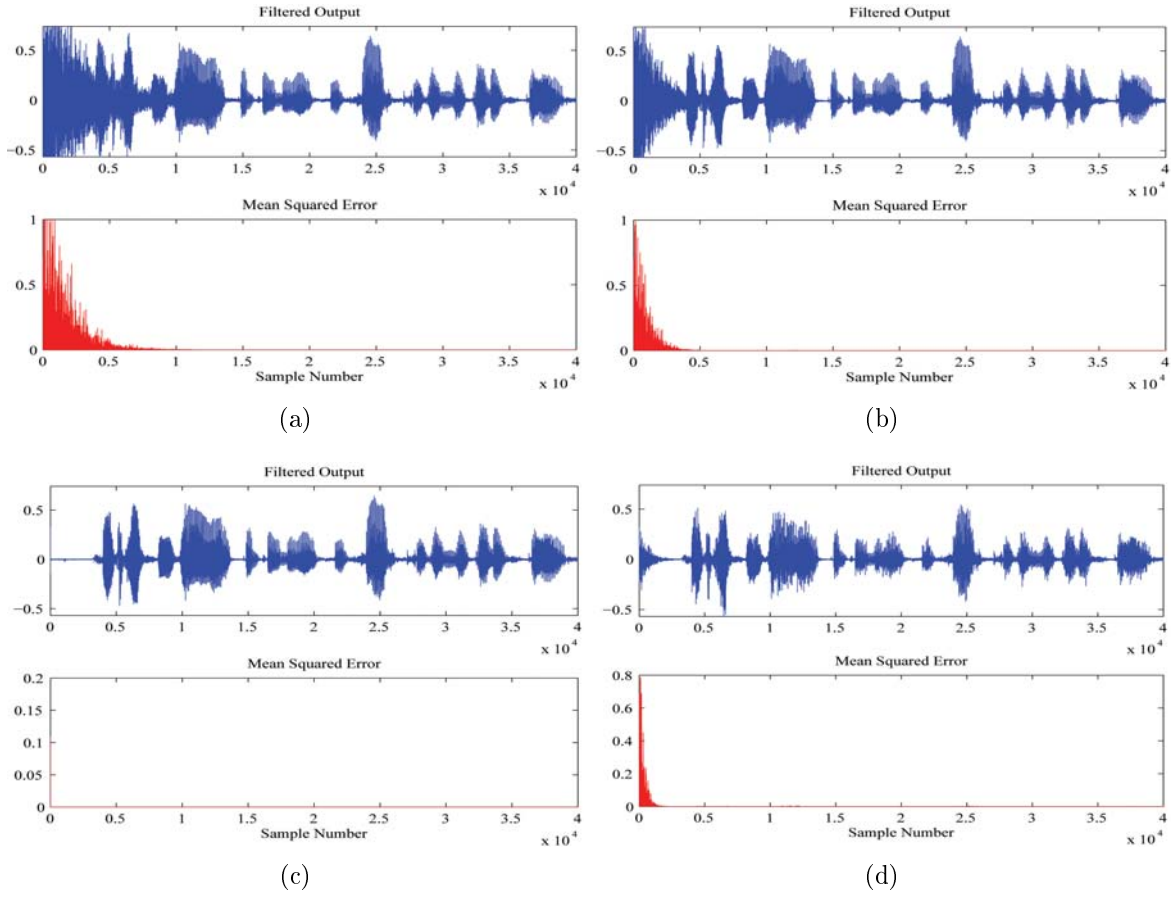


FIGURE 4. Filtered output signal and MSE curves of (a) the LMS algorithm, (b) the NLMS algorithm, (c) the RLS algorithm, (d) the FEDS algorithm

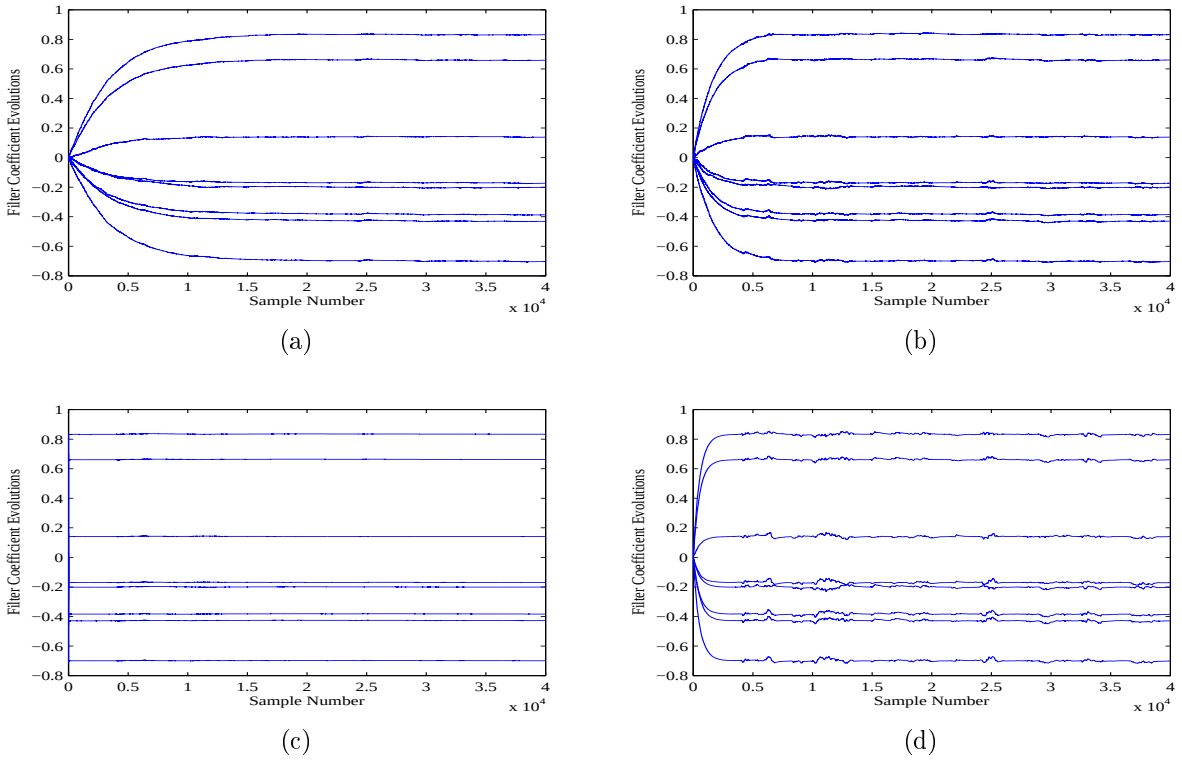


FIGURE 5. Time evolution of the filter taps in ANC through (a) the LMS algorithm, (b) the NLMS algorithm, (c) the RLS algorithm, (d) the FEDS algorithm

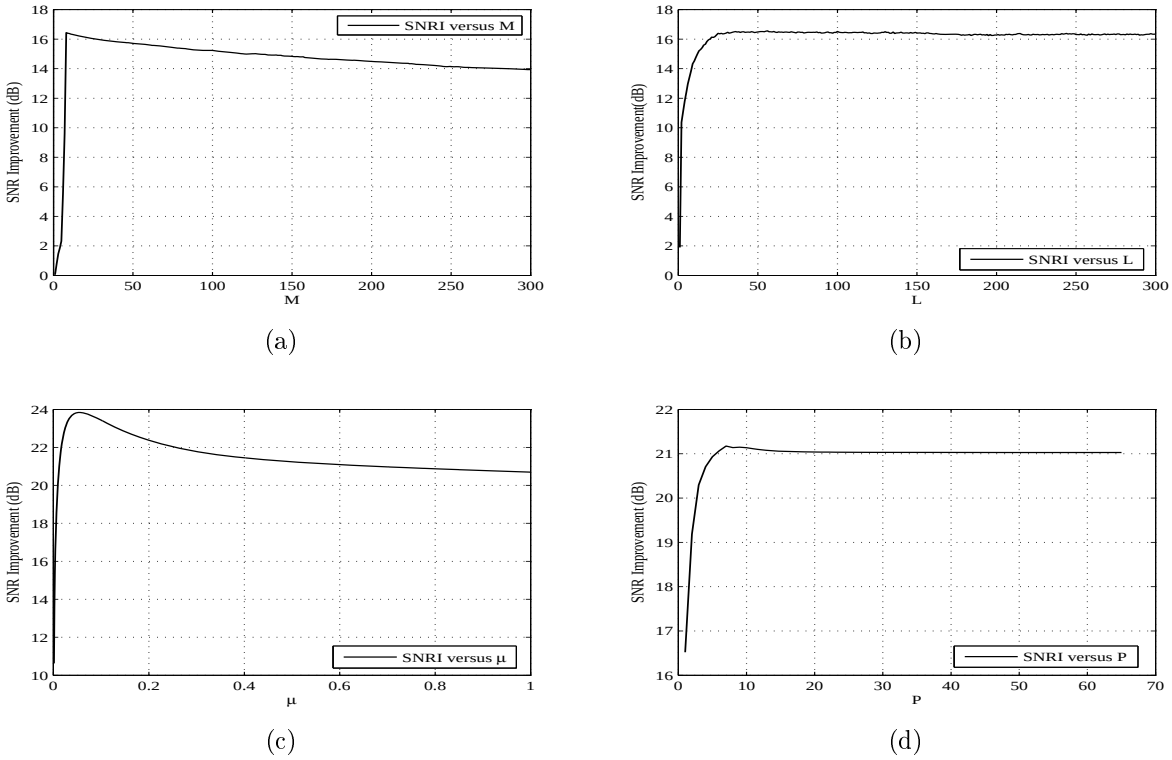
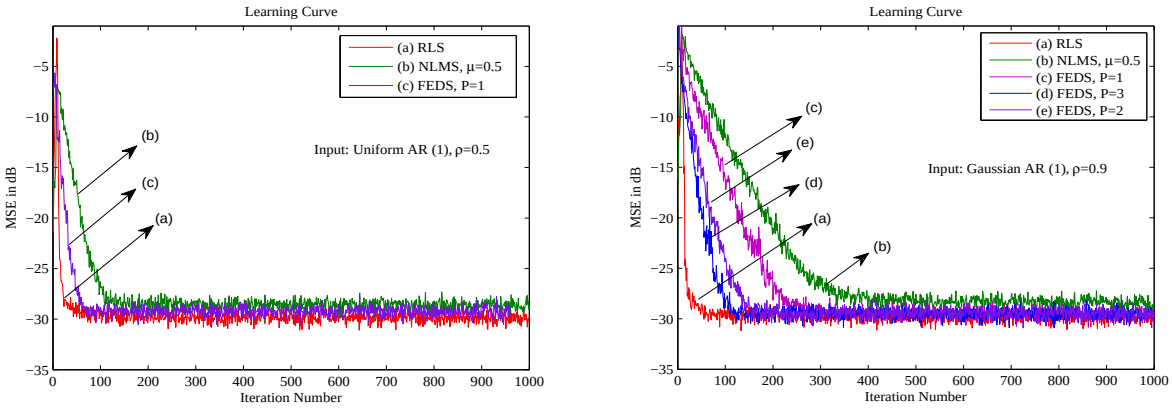
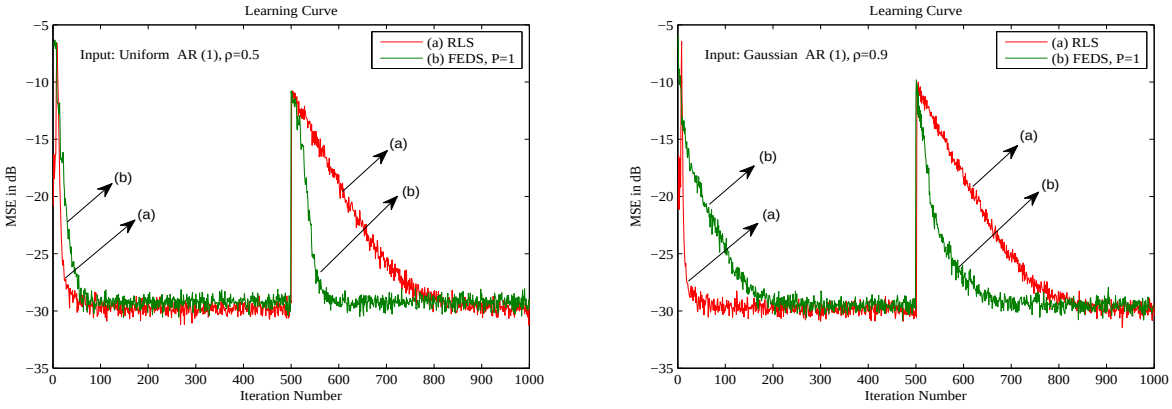


FIGURE 6. SNRI versus M, L, μ and P for the FEDS algorithm



(a) (b)

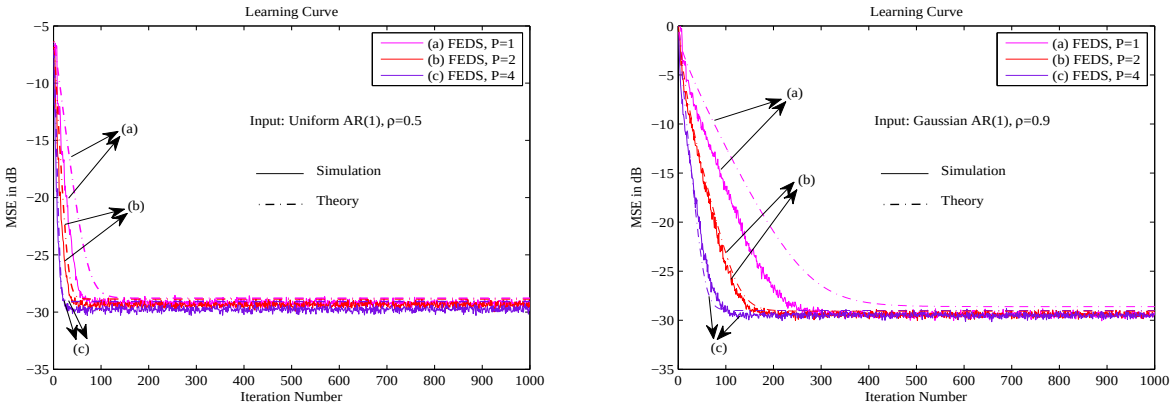
FIGURE 7. Learning curves for the RLS with forgetting factor $\lambda = 0.99$, NLMS with $\mu = 0.5$ and the FEDS with $L = 32$ and $P = 1, 2, 3$ for (a) somewhat colored ($\rho = 0.5$) uniformly distributed input, (b) highly colored ($\rho = 0.9$) Gaussian input



(a) (b)

FIGURE 8. Learning curves for the RLS with forgetting factor $\lambda = 0.99$, NLMS with $\mu = 0.5$ and the FEDS with $L = 32$ and $P = 1$ for (a) somewhat colored ($\rho = 0.5$) uniformly distributed input, (b) highly colored ($\rho = 0.9$) Gaussian input. (In this simulation, at sample no.500 we change the unknown system by multiplying its coefficients by 0.4.)

in noise cancellation for speech enhancement, power line interference cancellation and system identification applications. We demonstrated that the coefficients of the FEDS algorithm are updated in a sequential manner. Therefore, the proposed algorithm has low complexity, good convergence and tracking capabilities and is numerically robust. We also presented a performance analysis of the FEDS based on energy conservation arguments. In the simulation results, we compared the theoretical results with practical simulated results. It was seen that our theoretical results are more accurate and especially represent good agreement with the simulated results for the FEDS algorithm.



(a) (b)

FIGURE 9. Simulated and theoretical learning curves for the FEDS-P for various P values and $L = 32$ (a) somewhat colored ($\rho = 0.5$) uniformly distributed input, (b) highly colored ($\rho = 0.9$) Gaussian input

Acknowledgment. The authors are very grateful to the anonymous reviewers for their very insightful comments. This work was supported in part by the Islamic Azad University, Shahr-e-Rey Branch, Tehran, Iran. The authors gratefully acknowledge the financial and other supports provided by the Islamic Azad University, Shahr-e-Rey Branch, Tehran, Iran.

REFERENCES

[1] W. Harrison, J. Lim and E. Singer, A new application of adaptive noise cancellation, *IEEE Trans. Acoustic Speech Signal Process.*, vol.34, no.1, pp.21-27, 1986.

[2] N. Sonbolestan and S. A. Hadei, A fast affine projection algorithm based on matching pursuit in adaptive noise cancellation for speech enhancement, *Proc. of Intelligent Systems, Modeling and Simulation*, Liverpool, UK, pp.193-198, 2010.

[3] O. T. Inan, M. Etemadi, B. Widrow and G. T. A. Kovacs, Adaptive cancellation of floor vibrations in standing ballistocardiogram measurements using a seismic sensor as a noise reference, *IEEE Trans. Biomed. Eng.*, vol.57, no.3, pp.722-727, 2010.

[4] S. A. Hadei and P. Azmi, A novel adaptive channel equalization method using variable step size partial rank algorithm, *Proc. of Advance International Conference on Telecommunications*, Barcelona, Spain, pp.201-206, 2010.

[5] S. A. Hadei and P. Azmi, Low-complexity adaptive channel estimation over multipath Rayleigh fading non-stationary channels under CFO, *Proc. of IEEE the 18th International Conference on Telecommunications*, Ayia Napa, Cyprus, pp.339-345, 2011.

[6] S. A. Hadei and P. Azmi, Low-complexity adaptive channel equalizers for digital communications, *Proc. of the 19th Iranian Conference on Electrical Engineering*, Tehran, Iran, pp.2964-2969, 2011.

[7] S. Haykin, *Adaptive Filter Theory*, 4th Edition, Prentice Hall, 2002.

[8] A. H. Sayed, *Fundamentals of Adaptive Filtering*, Wiley, 2003.

[9] G. K. Boray and M. D. Srinath, Conjugate gradient techniques for adaptive filtering, *IEEE Trans. Circuits Syst. I.*, vol.39, pp.1-10, 1992.

[10] M. F. Barsoum and W. B. Mikhael, A conjugate-gradient-based block adaptive algorithm, its applications and performance evaluation, *Proc. of the 37th IEEE International Midwest Symp. on Circuits and Syst.*, Lafayette, LA, USA, pp.903-906, 1994.

[11] T. Bose and M. Q. Chen, Conjugate gradient method for adaptive bilinear filtering, *IEEE Trans. Signal Process.*, vol.43, no.6, pp.1503-1508, 1995.

[12] K. S. Joo and T. Bose, A fast conjugate gradient algorithm for 2-D nonlinear adaptive filtering, *Proc. of International Conference on Digital Signal Processing*, pp.314-319, 1995.

[13] P. S. Chang and A. N. Willson, Adaptive filtering using modified conjugate gradient, *Proc. of the 38th IEEE International Midwest Symp. on Circuits and Syst.*, Rio de Janeiro, Brazil, pp.243-246, 1995.
[14] P. S. Chang and A. N. Willson, Adaptive spectral estimation using the conjugate gradient algorithm, *Proc. of IEEE International Conference on Acoust. Speech and Signal Process.*, Atlanta, GA, USA, pp.2979-2982, 1996.
[15] M. Q. Chen and T. Bose, A direction set based algorithm for adaptive filtering, *IEEE Trans. Signal Process.*, vol.47, no.2, pp.535-539, 1999.
[16] G. F. Xu, *Fast Algorithms for Digital Filtering: Theory and Applications*, Ph.D. Thesis, University of Colorado at Boulder, 1999.
[17] J. H. Husoy and M. S. E. Abadi, A comparative study of some simplified RLS type algorithms, *Proc. of ISCCSP*, Hammamet, Tunisia, pp.705-708, 2004.
[18] J. H. Husoy and M. S. E. Abadi, Interpretation and convergence speed of two recently introduced adaptive filters (FEDS/RAMP), *Proc. of IEEE Region 10 Conference on Tencon 2004*, Chiang Mai, Thailand, pp.471-474, 2004.
[19] H. C. Shin and A. H. Sayed, Mean-square performance of a family of affine projection algorithms, *IEEE Trans. Signal Process.*, vol.52, no.1, pp.90-102, 2004.
[20] M. S. E. Abadi, M. B. Menhaj and S. A. Hadei, A fast affine projection algorithm based on matching pursuit with partial parameters adjustment, *Amirkabirl Journal of Science and Technology*, vol.18, no.67-A, pp.11-23, 2008.
[21] J. H. Husoy and M. S. E. Abadi, Unified approach to adaptive filters and their performance, *IET Signal Process.*, vol.2, no.2, pp.97-109, 2008.
[22] *http://www.owlnet.rice.edu/ ryanking/elec431.*

Appendix A. In this appendix, we explain how we can find the general expression for $C(n)$, when $P > 1$. Initializing $\boldsymbol{w}$ at new time instant can be shown as

$$\boldsymbol{w}^{[0]}(n+1) = \boldsymbol{w}^{[p]}(n) \tag{52}$$

Therefore, the output error vector can be stated as

$$\boldsymbol{e}_0(n) = \boldsymbol{d}(n) - U^T(n)\boldsymbol{w}^{[0]}(n+1) \tag{53}$$

when $P > 1$, the following algorithm is performed during each new time instant n

$$\begin{aligned}
&\text{for } p = 0, 1, \ldots, P-1 \\
&\quad j_p(n) = [nP + p] \oplus M \text{ (FEDS)} \\
&\quad \boldsymbol{w}^{[p+1]}(n+1) = \boldsymbol{w}^{[p]}(n+1) + i_{j_p(n)}.\left\|\boldsymbol{u}_{j_p(n)}(n)\right\|^{-2} U(n)\boldsymbol{e}_p(n) \\
&\quad \boldsymbol{e}_{p+1}(n) = \left\{I - U^T(n)C_{p(n)}(n)U(n)\right\}\boldsymbol{e}_p(n) \\
&\text{end}
\end{aligned}$$

where $C_{p(n)}(n)$ in the last row of this algorithm is given by

$$C_{p(n)}(n) = i_{j_p(n)}.\left\|\boldsymbol{u}_{j_p(n)}(n)\right\|^{-2} \tag{54}$$

The detail of the last row in this algorithm can be explained from the following relations

$$\begin{aligned}
\boldsymbol{e}_{p+1}(n) &= \boldsymbol{d}(n) - U^T(n)\boldsymbol{w}^{[p+1]}(n+1) \\
&= \boldsymbol{d}(n) - U^T(n)\left[\boldsymbol{w}^{[p]}(n+1) + C_{p(n)}(n)U(n)\boldsymbol{e}_p(n)\right] \\
&= \left\{I - U^T(n)C_{p(n)}(n)U(n)\right\}\boldsymbol{e}_p(n)
\end{aligned} \tag{55}$$

where $e_p(n) = d(n) - U^T(n)\boldsymbol{w}^{[p]}(n+1)$. From the above expressions, the following relation between iteration $P-1$ and P can be written as

$$\boldsymbol{w}^{[P]}(n+1) = \boldsymbol{w}^{[P-1]}(n+1) + C_{P-1}(n)U(n)\boldsymbol{e}_{P-1}(n) \tag{56}$$

where

$$C_{P-1}(n) = i_{j_{P-1}(n)}.\left\|\boldsymbol{u}_{j_{P-1}(n)}(n)\right\|^{-2} \tag{57}$$

Now by using (56), we find the following relation between iteration P and 0

$$\boldsymbol{w}^{[p]}(n+1) = \boldsymbol{w}^{[0]}(n+1) + \sum_{i=0}^{P-1} C_i(n)U(n)\boldsymbol{e}_i(n) \tag{58}$$

But from (55), $\boldsymbol{e}_i(n)$ can be stated as

$$\boldsymbol{e}_i(n) = \prod_{p=0}^{i-1} \left\{I - U^T(n)C_p(n)U(n)\right\}.\boldsymbol{e}_0(n) \tag{59}$$

Finally, the update equation from iteration $n+1$ to n and with P iteration in each new time instant n is given by

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) + \sum_{i=0}^{P-1} \left\{ C_i(n).\prod_{p=0}^{i-1} \{I - U^T(n)C_p(n)U(n)\} \right\} U(n)\boldsymbol{e}(n) \tag{60}$$

By comparing (60) with (26), we find that to evaluate the performance of the FEDS algorithm, when $P > 1$, we need to modify (60) to find the matrix $C(n)$. Therefore, the matrix $C(n)$ can be given by

$$C(n) = \sum_{i=0}^{P-1} C_i(n).\prod_{p=0}^{i-1} \{I - U(n)C_p(n)U^T(n)\} \tag{61}$$