Pseudo Affine Projection Algorithms for Multichannel Active Noise Control

Felix Albu, Martin Bouchard, and Yurii Zakharov, Members, IEEE

Abstract—For feedforward multichannel active noise control (ANC) systems, the use of adaptive finite impulse response (FIR) filters is a popular solution, and the multichannel filtered-x least-mean-square (FX-LMS) algorithm is the most commonly used algorithm. The drawback of the FX-LMS is the slow convergence speed, especially for broadband multichannel systems. Recently, some fast affine projection algorithms have been introduced for multichannel ANC, as an interesting alternative to the FX-LMS algorithm. They can provide a significantly improved convergence speed at a reasonable additional computational cost. Yet, the additional computational cost or the potential numerical instability in some of the recently proposed algorithms can prevent the use of those algorithms for some applications. In this paper, we propose two pseudo affine projection algorithms for multichannel ANC: one based on the Gauss-Seidel method and one based on dichotomous coordinate descent (DCD) iterations. It is shown that the proposed algorithms typically have a lower complexity than the previously published fast affine projection algorithms for ANC, with very similar good convergence properties and good numerical stability. Thus the proposed algorithms are an interesting alternative to the standard FX-LMS algorithm for ANC, providing an improved performance for a computational load of the same order.

Index Terms—multichannel active noise control, adaptive filtering, fast affine projection algorithms, Gauss-Seidel solving scheme, dichotomous coordinate descent

I. INTRODUCTION

ACTIVE noise control (ANC) systems are being increasingly researched and developed [1]. These systems work on the principle of destructive interference between an original primary disturbance sound field measured at the location of \( K \) error sensors (typically microphones), and a secondary sound field that is generated by \( J \) control actuators (typically loudspeakers). In ANC systems (and also in active vibration control systems), a common approach is to use FIR filters as adaptive controllers, in either feedforward or feedback control configurations. The feedforward approach relies on the use of \( I \) reference sensors providing information on the signals to be attenuated at the error sensors. Defining \( L \) as the length of the FIR filters and \( M \) as the length of the plant impulse response (also assumed to be FIR), Fig.1 shows a multichannel feedforward active noise control system using adaptive FIR filters, with the different dimensions \( I, J, K, L \), and \( M \). The rest of this paper assumes a feedforward ANC structure, although it is straightforward to adapt the algorithms introduced in the paper to a feedback ANC structure. The plant impulse responses shown in Fig. 1 each have a delay, corresponding to the propagation delay between an actuator and an error sensor. In control theory, it is well known that delays are a nuisance for control applications, and the effect of the delay on the performance of standard adaptive FIR filtering algorithms for ANC have been well documented before [1]. Basically, the larger the delay, the slower the adaptation of the algorithms used to train the adaptive FIR filters in the ANC system, because a smaller step size has to be used in the algorithms. This means that a large delay can reduce both the initial convergence speed of the ANC system and the ability of the system to quickly adapt to varying conditions (i.e. tracking). To avoid this problem, the delay compensated or modified filtered-x structure for active noise control systems using FIR adaptive filtering was introduced in [2], and it is presented in Fig. 2. Intuitively, since the plant normally includes a propagation delay, there is a delay before the effect of a change in the adaptive FIR filter coefficients becomes effective in the error sensor signals. This is the cause of the requirement for a slower adaptation of the FIR filter coefficients by the training algorithm (i.e. smaller step size). The structure in Fig. 2 eliminates the plant delay by computing an estimate of the primary field signals, which are unaffected by the changes of the adaptive FIR filter coefficients. This is done by using the knowledge of the actuator signals and the plant models in Fig. 2. By filtering the actuator signals with the plant models, estimates of the actuators contributions at the location of the error sensors are obtained. By subtracting these contributions from the measured error sensors signals, estimates of the primary field signals are obtained. The delay compensated modified
Recently, some fast affine projection (FAP) algorithms have been introduced for multichannel ANC [3]-[5], as an interesting alternative to the FX-LMS and MFX-LMS algorithms. They can provide a significantly improved convergence performance at a lower cost [4],[5]. However, the new MFX-GSPAP algorithm has an inherent better numerical stability than the MFX-FAP-RLS, because it avoids the RLS recursion. However, the MFX-GSPAP still requires at least one inverse matrix computation. This can be very complex for large matrices and prone to numerical instability. Therefore, in Section III, a second new algorithm for multichannel ANC systems, termed MFX-GSPAP algorithm, is proposed. It is based on the dichotomous coordinate descent method used for solving linear systems [10-11], such as the one that needs to be solved in the PAP algorithm (see eq.(8) in Section II). The DCD method and the resulting new MFX-DCDPAP algorithm do not require any matrix inversion, thus reducing the complexity. The new MFX-DCDPAP also exhibits a convergence speed similar to the previously published MFX-GSFAP and MFX-FAP-RLS algorithms, with a better numerical stability than the MFX-FAP-RLS (no RLS recursion), and a lower computational complexity than both previously published algorithms. The computational complexity of the proposed algorithms is evaluated in Section IV and it is compared with some previously published algorithms. Results of simulations comparing the new proposed algorithms with some previously published algorithms are presented in Section V. Section VI concludes this work.

II. MULTICHANNEL MODIFIED FILTERED-X GAUSS SEIDEL PSEUDO AFFINE PROJECTION ALGORITHM

An In the context of ANC systems, a monochannel
feedforward system using an adaptive FIR filter with a modified filtered-x structure as in Fig. 2 and with filter weights adapted with a classical affine projection (AP) algorithm [12] can be described by the following equations (1)-(5) [3]-[4]:

\[ y(n) = w^T(n)x(n) \quad (1) \]

\[ v(n) = h^T x'(n) \quad (2) \]

\[ \hat{d}(n) = e(n) - h^T y(n) \quad (3) \]

\[ \hat{e}_N^T(n) = \hat{d}_N^T(n) + A(n)w(n) \quad (4) \]

\[ w(n+1) = w(n) - \mu A^T(n)\left( A(n)A^T(n) + \delta I \right)^{-1}\hat{e}_N^T(n) \quad (5) \]

The variable \( n \) refers to the discrete time. The column vectors \( x(n) = [x(n), \ldots, x(n-L+1)]^T \) and \( x'(n) = [x(n), \ldots, x(n-M+1)]^T \) consist of the last \( L \) and \( M \) samples of the reference sensor signal \( x(n) \), respectively (refer to Fig. 2). The coefficients of the adaptive FIR filter are represented by the column vector \( w(n) = [w_1(n), \ldots, w_L(n)]^T \). The coefficients of the (fixed) FIR filter modeling the plant between signals \( y(n) \) and \( e(n) \) are described by the column vector \( h = [h_1, \ldots, h_M]^T \). The column vector \( y(n) = [y(n), \ldots, y(n-M+1)]^T \) consists of the last \( M \) samples of the actuator signal \( y(n) \). \( e(n) \) is the error-sensor signal. The samples of the filtered reference signal \( v(n) \) are collected in the column vector \( v(n) = [v(n), \ldots, v(n-L+1)]^T \) and the \( N \times L \) matrix

\[ A(n) = [v(n) \cdots v(n-N+1)]^T \]

where \( N \) is the affine projection order [12]. The row vectors \( \hat{d}_N(n) = [\hat{d}(n) \cdots \hat{d}(n-N+1)] \) and \( \hat{e}_N(n) = [\hat{e}(n) \cdots \hat{e}(n-N+1)] \) consist of estimates \( \hat{d}(n) \) of the primary sound field \( d(n) \) and of alternative error signal samples \( \hat{e}(n) \), both computed in delay-compensated modified filtered-x structures as in Fig. 2. Finally, \( \mu \) is a normalized convergence gain \( 0 \leq \mu \leq 1 \), \( I \) is an identity matrix of size \( N \times N \) and \( \delta \) is a regularization factor that may be used to help with eventual numerical instability.

In the adaptive filtering literature, it is well known that when the input signal of an adaptive filter trained with an Affine Projection algorithm is a time series (such as \( v(n) \)), which is the actual input signal of the adaptive filter in ANC structures as shown in Fig. 2), then the redundancy found in the input signal or the data matrix \( A(n) \) can be exploited. This results in a family of algorithms called the Fast Affine Projection (FAP) algorithms [13]-[14]. For an adaptive FIR filter of \( L \) coefficients, FAP algorithms include a set of \( N \) linear equations to be solved (with typically \( N << L \)), and this set of equation is often solved by the use of a built-in recursive least-squares (RLS) algorithm [15] or some more efficient built-in fast-RLS algorithm [15] inside the FAP algorithm. In the context of ANC algorithms, a FAP algorithm with a built-in fast-RLS algorithm was first introduced for monochannel ANC systems in [3]. More recently, an AP algorithm and a FAP algorithm were introduced in [4] for the more general case of multichannel ANC. The FAP algorithm in [4] used a built-in RLS algorithm instead of a built-in fast-RLS algorithm as in [3], because the numerical instability of the fast-RLS algorithm has proven to be more problematic in the multichannel case. Thus, the FAP algorithm in [4] used the name FAP-RLS, or more precisely modified filtered-x FAP-RLS (MFX-FAP-RLS), since it also made use of the structure of Fig. 2.

In recent years, other schemes have been investigated to replace the built-in RLS or fast-RLS algorithms inside the FAP algorithm, to improve the numerical stability of the algorithms and also possibly reduce the computational load. The Gauss-Seidel inversion scheme [7] is one of those schemes that were successfully applied to the FAP algorithm [16]. The adaptation of the Gauss-Seidel FAP algorithm (GSPAP) to multichannel ANC systems was recently published in [5] as the MFX-GSPAP algorithm, exhibiting a lower complexity and a better numerical stability than the MFX-FAP-RLS, for the same convergence speed. As an attempt for further complexity reduction, a Pseudo Affine Projection (PAP) algorithm was recently derived from the original AP algorithm by applying a Levinson-Durbin recursion [6]. By replacing the Levinson-Durbin recursion with the Gauss-Seidel method [7], a simpler algorithm was derived in [8], called the Gauss-Seidel Pseudo Affine Projection (GSPAP) algorithm. For a similar convergence performance, the complexity of the GSPAP algorithm is typically lower than for the FAP-RLS and GSPAP algorithms, as will be shown later in this paper. This section mostly presents the extension of the GSPAP algorithm to the case of multichannel ANC systems: the MFX-GSPAP algorithm.

The derivation of the GSPAP algorithm, in the context of monochannel ANC using the structure of Fig. 2, is first presented. Let's consider the linear prediction with order \( N - 1 \) (prediction filter with coefficients \( f_i \), \( 1 \leq i \leq N - 1 \)) of the signal \( v(n) \), based on the last \( L \) samples of \( v(n) \) and producing a residual prediction signal \( u(n) \):

\[ u(n) = f^T v_N(n) \quad (6) \]
with the column vectors
\[ \mathbf{v}_N(n) = [v(n), \cdots, v(n-N+1)]^T \]
and
\[ \mathbf{f} = [1, f_1, \cdots, f_{N-1}]^T. \]
The coefficients \( f_i \) could be computed from the last \( L \) samples of \( v(n) \) by solving the following Yule-Walker equation [17]:
\[ \mathbf{R}(n) \mathbf{f} = [\mathbf{v}^T(n) \mathbf{u}(n), 0, \cdots, 0]^T \tag{7} \]
with \( \mathbf{u}(n) = [u(n), u(n-1), \cdots, u(n-L+1)]^T \) and
\[ \mathbf{R}(n) = \mathbf{A}(n) \mathbf{A}^T(n). \]
Defining \( \mathbf{p}(n) \) as
\[ \mathbf{p}(n) = \left[ v^T(n) \mathbf{u}(n) / v^T(n) \mathbf{u}(n), \cdots, v^T(n) \mathbf{u}(n) \right]^T, \]
the previous formula can be re-arranged as:
\[ \mathbf{R}(n) \mathbf{p}(n) = [1, 0, \cdots, 0]^T \tag{8}. \]
One single iteration from the Gauss-Seidel scheme [7] can be used to compute \( \mathbf{p}(n) \), using as an initial condition for \( \mathbf{p}(n-1) \) from the previous iteration of the GSPAP algorithm. More details on the implementation of the Gauss-Seidel scheme are provided later in this section. With the knowledge of \( \mathbf{p}(n) \), the prediction residual signal \( \mathbf{u}(n) \) can be computed by:
\[ \mathbf{u}(n) = \mathbf{f}^T \mathbf{v}_N(n) = \mathbf{v}_N^T(n) \mathbf{f} \]
\[ = \mathbf{v}_N^T(n) \mathbf{p}(n) \left( \mathbf{v}^T(n) \mathbf{u}(n) \right) = \mathbf{v}_N^T(n) \mathbf{p}(n) \overline{p}^{-1}(n) \tag{9} \]
where \( \overline{p}(n) \) is the first component of \( \mathbf{p}(n) \) and \( \overline{p}^{-1}(n) \) is its inverse value.

For the case with \( \mu = 1 \) (referred to in the literature as case without relaxation), the previous equation (5) for the AP algorithm can be re-written in terms of the prediction residual signal \( \mathbf{u}(n) \) instead of the original input signal \( \mathbf{v}(n) \) [6], with the following simplified equation:
\[ \hat{\mathbf{e}}(n) = \hat{\mathbf{d}}(n) + \mathbf{v}^T(n) \mathbf{w}(n) \tag{10} \]
where \( \hat{\mathbf{e}}(n) \) is computed as a subset of (4):
\[ \hat{\mathbf{e}}(n) = \hat{\mathbf{d}}(n) + \mathbf{v}^T(n) \mathbf{w}(n) \tag{11} \]
A more efficient implementation can be achieved by further re-arranging the equations. First, the direct computation of \( \mathbf{R}(n) = \mathbf{A}(n) \mathbf{A}^T(n) \) should be avoided and it can be evaluated instead by considering the special structure of \( \mathbf{R}(n) : \mathbf{R}(n) = \begin{bmatrix} \overline{r}(n) & \mathbf{r}(n) \\ \mathbf{r}^T(n) & \overline{\mathbf{R}}(n-1) \end{bmatrix} \), with \( \overline{\mathbf{R}}(n) \) defined as
\[ \overline{\mathbf{R}}(n) = \begin{bmatrix} \overline{r}(n) & \mathbf{r}(n) \\ \mathbf{r}^T(n) & \overline{\mathbf{R}}(n-1) \end{bmatrix}, \]
the top left \( (N-1) \times (N-1) \) values of \( \mathbf{R}(n) \). \( \mathbf{r}(n) \) and \( \overline{\mathbf{r}}(n) \) can then be computed by:
\[ \mathbf{r}(n) = \mathbf{r}(n-1) + \mathbf{v}_N(n) v(n) - \mathbf{v}_N(n-L) v(n-L) \tag{12} \]
\[ \overline{\mathbf{r}}(n) = \overline{\mathbf{r}}(n-1) + v(n) v(n) - v(n-L) v(n-L) \tag{13} \]
where \( \mathbf{v}_N(n) \) corresponds to the last \( N-1 \) rows of \( \mathbf{v}(n) \) (i.e. \( \mathbf{v}_N(n) = [v(n-1), \cdots, v(n-N+1)]^T \)). By introducing a scalar \( m(n) \) computed as:
\[ m(n) = m(n-1) + u(n) v(n) - u(n-L) v(n-L) \tag{14} \]
then (10) can be re-written as:
\[ \mathbf{w}(n+1) = \mathbf{w}(n) - \frac{1}{m(n) + \delta} \mathbf{u}(n) \hat{\mathbf{e}}(n). \tag{15} \]
Re-introducing a relaxation factor \( \mu (0 \leq \mu \leq 1) \) can reduce the error signal \( \hat{\mathbf{e}}(n) \) in some situations, but since the resulting algorithm with the \( \mu \) relaxation factor has not been formally derived with \( \mu \), the result is then called a Pseudo Affine Projection algorithm (PAP) instead of a FAP algorithm:
\[ \mathbf{w}(n+1) = \mathbf{w}(n) - \mu \frac{1}{m(n) + \delta} \mathbf{u}(n) \hat{\mathbf{e}}(n). \tag{16} \]

To summarize, the equations describing the GSPAP for monochannel ANC are given by (1), (2), (3), (8) through a Gauss-Seidel iteration, (9), (11), (12), (13), (14) and (16).

Those ten equations can be extended to the case of multichannel ANC systems, still using the structure of Fig. 2 (i.e. delay-compensated modified filtered-x structure). This results in the multichannel modified filtered-x Gauss-Seidel Pseudo Affine Projection (MFx-GSPAP) algorithm described by the following equations:
\[ y_j(n) = \sum_{i=1}^{I} w_{i,j}(n) x_i(n) \tag{17} \]
\[ v_{i,j,k}(n) = h_{j,k}^T x_i^T(n) \tag{18} \]
\[ \hat{d}_{j,k}(n) = \epsilon(n) - \sum_{j=1}^{I} h_{j,k}^T y_j(n) \tag{19} \]
\[ \hat{\mathbf{e}}^T(n) = \hat{\mathbf{d}}^T(n) + \mathbf{V}^T(n) \mathbf{w}(n) \tag{20} \]
\[ \mathbf{R}_0(n) = \mathbf{R}_0(n-1) + \mathbf{A}_0(n) \mathbf{V}_0(n) \]
\[ - \mathbf{A}_0(n-L) \mathbf{V}_0(n-L) \tag{21} \]
\[ \overline{\mathbf{R}}_0(n) = \overline{\mathbf{R}}_0(n-1) + \mathbf{V}_0^T(n) \mathbf{V}_0(n) \]
\[ - \mathbf{V}_0^T(n-L) \mathbf{V}_0(n-L) \tag{22} \]
\[ \mathbf{R}(n) \mathbf{P}(n) = \mathbf{C} \] (to solve with Gauss-Seidel method)
\[ U_0(n) = A_0^T(n) \mathbf{P}(n) \overline{\mathbf{P}}^{-1}(n) \tag{23} \]
\[ M(n) = M(n-1) + U_0^T(n) \mathbf{V}_0(n) \]
\[ - U_0^T(n-L) \mathbf{V}_0(n-L) \tag{24} \]
\[ w(n+1) = w(n) - \mu U(n)M^{-1}(n)\hat{e}^T(n) \]  
(26)

where the following additional notations are defined: the variables \( i = 1, \ldots, I \), \( j = 1, \ldots, J \), and \( k = 1, \ldots, K \) refer to the different reference sensors, actuators, and error sensors, respectively. \( R(n) \) is now a \( KN \times KN \) auto-correlation matrix, initialized as an identity matrix multiplied by a regularization factor \( \delta_1 \),

\[
R(n) = \begin{bmatrix}
\overline{R}_0(n) & R^0(n) \\
R_0(n) & \overline{R}(n-1)
\end{bmatrix},
\]

where \( \overline{R}(n) \) is the top left \( K(N-1) \times K(N-1) \) block of \( R(n) \), while \( R_0(n) \) and \( \overline{R}_0(n) \) are, respectively, \( K(N-1) \times K \) and \( K \times K \) correlation matrices initialized with zeros. \( P(n) \) is an inverse \( KN \times KN \) correlation matrix, while \( \overline{P}(n) \) is the top \( K \times K \) block of \( P(n) \). \( C \) is a \( KN \times KN \) constant matrix whose elements are zeros, except for the top \( K \times K \) block set to an identity matrix. The \( L \times K \) matrix \( U(n) = [U_{ij}(n) \ldots U_{ij}(n-L+1)]^T \) consists of decorrelated reference signal matrices \( U_{ij}(n) \) of size \( L \times K \). \( M(n) \) is a \( K \times K \) inverse matrix initialized with an identity matrix multiplied by a regularization factor \( \delta_2 \). The vectors \( x_i = [x_i(n), \ldots, x_i(n-L+1)]^T \) and \( x'_i = [x'_i(n), \ldots, x_i(n-M+1)]^T \) consist of the last \( L \) and \( M \) samples of the reference signal \( x_i(n) \), respectively. The vector \( y_j = [y_j(n), \ldots, y_j(n-M+1)]^T \) consists of the last \( M \) samples of the actuator signal \( y_j(n) \). The samples of the filtered reference signal \( v_{i,j,k}(n) \) are collected in the \( L \times K \), \( L \times L \) and \( KN \times I \) matrices:

\[
V_0(n) = \begin{bmatrix}
v_{1,1,1}(n) & \ldots & v_{1,1,K}(n) \\
\vdots & \ddots & \vdots \\
v_{1,J,1}(n) & \ldots & v_{1,J,K}(n)
\end{bmatrix},
\]

\[
V(n) = \begin{bmatrix}
V_0^T(n) & \cdots & V_0^T(n-L+1)
\end{bmatrix}^T,
\]

\[
A_0(n) = \begin{bmatrix}
V_0(n) & \cdots & V_0(n-N+1)
\end{bmatrix}^T,
\]

while the matrix \( A_0(n) \) consists of the last \( K(N-1) \) rows of \( A_0(n) \). The \( 1 \times K \) vectors \( \hat{d}(n) = [\hat{d}_1(n), \hat{d}_2(n), \cdots, \hat{d}_K(n)] \) and \( \hat{e}(n) = [\hat{e}_1(n), \hat{e}_2(n), \cdots, \hat{e}_K(n)] \) consist of estimates \( \hat{d}_k(n) \) of the primary sound field \( d_k(n) \) and of alternative error signals samples \( \hat{e}_k(n) \), both computed in delay-compensated modified filtered-x structures, as mentioned earlier. The \( M \times 1 \) vector \( h_{j,k,m} = [h_{j,k,1}, \cdots, h_{j,k,M}]^T \) consists of taps \( h_{j,k,m} \) of the (fixed) FIR filter modeling the plant between signals \( y_j(n) \) and \( e_k(n) \). The \( L \times 1 \) vector \( w(n) = [w_{i,j,1}(n) \cdots w_{i,j,K}(n)] \) consists of the coefficients from all the adaptive FIR filters linking the signals \( x_i(n) \) and \( y_j(n) \) (\( \forall i, j \)). Finally, \( e_k(n) \) is the \( k \)th error sensor signal and \( \mu \) is again a normalized gain, \( 0 \leq \mu \leq 1 \).

To determine \( P(n) \) needed in (24), equation (23) is solved using the Gauss-Seidel method [7] for each column of \( P(n) \) and \( C \). To do this, equation (23) is transformed in \( K \) independent equations, \( k = 1, \cdots, K \), where \( p_k(n) \) is the \( k \)th column of \( P(n) \) and \( c_k \) is the \( k \)th column of \( C \). Since the filter length \( L \) is usually significantly bigger than the affine projection order \( N \), i.e. \( L >> N \), the correlation matrix \( R(n) \) is slowly varying in time, as is the solution of the system (23). Therefore, assuming that we have already obtained an accurate estimate of the vector \( p_k(n-1) \) for the time sample \( (n-1) \), the vector \( p_k(n-1) \) can be used as an initial condition in the Gauss-Seidel method. This is equivalent to solving the system

\[
R(n)p_k(n) = c_k
\]

with one GS iteration:

\[
[p_k(n)] = \frac{1}{[R(n)]_{i,j}} \left( [c_k]_{i,j} - \sum_{j=1}^{K} [R(n)]_{i,j} [p_k(n-1)]_{j} \right)
\]

\[
- \sum_{j=i+1}^{K} [R(n)]_{i,j} [p_k(n)]_{j},
\]

where \([c_k]_{i,j}\) is the \( i \)th element of the vector \( c_k \), \([p_k(n)]_{i,j}\) is the \( i \)th element of the vector \( p_k(n) \), \([R(n)]_{i,j}\) is the \( (i,j) \)th element of the matrix \( R(n) \), and \( i = 1, \cdots, KN \). In [5], it was found that one GS iteration per sample was enough for the MFX-GSFAP algorithm to achieve approximately the same performance as that of the theoretically more accurate MFX-FAP-RLS algorithm.

Even though \( P(n) \) is computed recurrently by using Gauss-Seidel iterations, the MFX-GSPAP (and also the MFX-GSFAP in [5]) computes \( P(n) \) directly from the correlation matrix \( R(n) \), unlike the MFX-FAP-RLS [4] or other RLS-based algorithms. Therefore, it has the potential for an inherently better numerical stability (no RLS recursion). Moreover, in the MFX-GSPAP and MFX-GSFAP algorithms, it may not be required to invert \( R(n) \) for each iteration of the algorithm (i.e. \( R(n) \) has to be always updated but its inverse...
does not necessarily have to be computed for each iteration). This is not the case in the MFX-FAP-RLS or purely RLS-based algorithms, because the recurrent scheme for inverting \( R(n) \) cannot miss any update without having undesirable effects caused by discontinuities. For a proper initialization of the MFX-GSPAP algorithm, at the first iteration of the algorithm \( v(n) \) should be non-zero and \( v(n-1) \ldots v(n-L+1) \) (or \( v(n-1) \ldots v(n-N+1) \)) should all be zero.

The proposed MFX-GSPAP algorithm directly computes the adaptive filter coefficients \( w \) in (26), used for the filtering between the reference sensors and the actuators in (1). This is unlike the previously published FAP algorithms for active noise control [3]-[5], which instead compute what is referred to as "auxiliary coefficients". The direct computation of the vector \( w \) can be useful for a few reasons. For example, the only computation that must absolutely be done in real time in ANC applications is the computation of the actuator values in (1). The other computations could possibly be done offline, at a reduced rate, using recorded blocks of data (at the cost of having reduced tracking capabilities). Also, the direct access to the time domain coefficients can be interesting because it provides more physical insights into the control system, for example to observe its causality and the number of required coefficients, etc.

III. MULTICHANNEL MODIFIED FILTERED-X DICHOTOMOUS COORDINATE DESCENT PSEUDO AFFINE PROJECTION ALGORITHM

The MFX-GSPAP algorithm introduced in the previous section provides both a direct estimation of the adaptive coefficients \( w \) and a computation complexity typically lower than the previous FAP algorithms for multichannel ANC [4]-[5]. However, the MFX-GSPAP algorithm still requires at least one inverse matrix computation. This can be very complex for large matrices and prone to numerical instability. Therefore, in this section, a second new pseudo affine projection algorithm called the modified filtered-x Dichotomous Coordinate Descent Pseudo Affine Projection (MFX-DCDPAP) algorithm is introduced. It uses the first seven equations (17)-(23) of the MFX-GSPAP algorithm. However, for solving the linear system in (23), the DCD method [10]-[11] is used; this multiplication-free and division-less procedure is presented below.

Let a system of equations to be solved be \( R(n)p_k(n) = c_k \). The DCD algorithm is based on a binary representation of elements of the solution vector with \( M_b \) bits within an amplitude range \([-H, H]\). The iterative approximation of the solution vector \( p_k(n) \) starts by updating the most significant bit of its elements and proceeds to less significant bits. If a bit update happens (such an iteration is called "successful"), the vector \( c_k \) is also updated. The complexity of the method is mainly due to updates of the vector \( c_k \), i.e. due to the "successful" iterations. The parameter \( N_{upd} \), that represents the maximum number of "successful" iterations, limits the algorithm complexity. Denote \( r_i(n) \) the \( i \)th column of the matrix \( R(n) \). The DCD algorithm can be described as follows.

Initialisation: \( p_k(n) = 0, d = H, q = 0 \).

for \( m = 1 \): \( M_b \)

\[ d = d / 2 \]

(a) \( \text{flag} = 0 \)

for \( i = 1 \): \( KN \)

if \( \left[ \left[ c_k \right] \right] > (d / 2)[R(n)]_j \), then

\[ \text{flag} = 1, q = q + 1 \]

\[ p_k(n) = [p_k(n)] + \text{sgn}([c_k]) \cdot d \]

\[ c_k = c_k - \text{sgn}([c_k]) \cdot d \cdot r_i(n) \]

if \( q > N_{upd} \), then the algorithm stops

end of the \( i \)-loop

if \( \text{flag} = 1 \), then go to (a)

end of the \( m \)-loop

It can be seen from the algorithm description that if \( H \) is a power of two, then only multiplications by factors of power of two are used; these can be replaced by bit shifts [10]. Thus, the DCD algorithm can be implemented without explicit multiplications and divisions (well known as potential sources of numerical instability). The peak complexity of the DCD algorithm for given \( M_b \) and \( N_{upd} \) is

\[ KN (2 N_{upd} + M_b) \]

shift-accumulation (SACs) operations

To complete the MFX-DCDPAP algorithm, in addition to equations (17)-(23) and the DCD algorithm to solve the system (23), the modification proposed in [18] for the GSPAP algorithm is adapted to multichannel ANC systems. If the \( p(n) \) estimate is exact and there is no regularization [18], the equation (10) with \( \mu \) included can be replaced by

\[ w(n+1) = w(n) - \frac{\mu A^T(n)p(n)}{p^T(n)A(n)v(n)} \]

\[ = w(n) - \frac{\mu A^T(n)p(n)}{p^T(n)A(n)v(n)} \]

Therefore, for the multichannel ANC systems the following computations are performed:

\[ \hat{U}_o(n) = A^T(n)P(n) \]

(30)

\[ w(n+1) = w(n) - \mu \hat{U}(n)\hat{e}(n) \]

(31)

where \( \hat{U}(n) = \left[ \hat{U}_o(n), \ldots, \hat{U}_o(n-L+1) \right]^T \) and therefore is
updated in a similar way to $U(n)$. The remaining two equations of the MFX-DCDPAP algorithm are (30) and (31). The advantage of this new algorithm is that it avoids the matrix inverse, a known source of numerical instability. Also, it has a reduced complexity as will be shown in Section IV. It will also be verified in section V that the modifications proposed for the MFX-DCDPAP do not alter significantly the convergence and steady-state properties of the MFX-DCDPAP algorithm, compared to the MFX-GSPAP algorithm, when ideal plant models are used.

Just like the MFX-GSPAP algorithm introduced in the previous section, the MFX-DCDPAP directly computes the adaptive filter coefficients $w$ in (31), used for the filtering between the reference sensors and the actuators in (1). It thus presents the same advantages as the ones described at the end of Section II for the MFX-GSPAP algorithm.

IV. COMPUTATIONAL COMPLEXITY

The computational complexity of the proposed algorithms was estimated by the number of multiplications required per iteration. Matrix inversions were assumed to be performed with standard LU decomposition that requires $O\left(X^3/2\right)$ multiplications, where $X$ is the size of a square matrix. As a reference for comparison, the number of multiplications per MFX-GSFAP algorithm iteration is [5]:

$$LJK(2L + M + 2(N - 1)(K + J) + K + 1) + K^2(2N - 1) + J(KM + K(N - 1) + IL) + KN + K^3N^2/p$$

where $p$ is the update period of the GS algorithm [5]. The number of multiplications per MFX-GSPAP algorithm iteration is:

$$LJK(M + 2L + 3KN + 2K + 1) + JL + JKM + K^3\left(\frac{1}{2} + \frac{N^2}{p}\right) + K^2$$

and the number of multiplications per MFX-DCDPAP algorithm iteration is:

$$LJK(2M + 2L + 3KN) + JL + JKM$$

It can be seen that the MFX-GSPAP algorithm has $LJK(2K + 1) + K^3\left(\frac{1}{2} + \frac{N^2}{p}\right) + K^2$ more multiplications per iteration than the MFX-DCDPAP algorithm. This increase of the numerical complexity is approximately proportional with $K^2$ and can be reduced by updating less frequently the GS part of the algorithm. For the MFX-GSPAP and MFX-GSFAP algorithms, updating less frequently the GS part produces a reduction of about $\frac{p - 1}{p}K^3N^2$ multiplications and therefore can be important, especially for high values of $K$ and $N$. The performance of those two algorithms is only slightly reduced when the update of the solution for the linear system in (23) is not done at the sample rate [5], [9]. If a proper regularization factor is used, a value of $p$ up to $p = 10$ can be used safely for updating the solution of (23), without having signs of instability and with an average loss of less than 1 dB in convergence. Updating less frequently the solution of (23) in the MFX-DCDPAP algorithm doesn’t change its complexity given by (34) in terms of multiplications, since this part is computed only with additions and shift operations.

Table 1 evaluates the complexity of the two new introduced PAP algorithms, compared with the complexity of previously published LMS and FAP based algorithms for multichannel ANC systems [4], [5], [19]. The values between accolades correspond to an update period $p = 10$. It can be seen that all the affine projection derived algorithms are only slightly more complex than the benchmark MFX-LMS algorithm. For the chosen parameters $(I, J, K, L, M, N)$ the complexity of the MFX-DCDPAP algorithm is lower than that of the MFX-GSPAP algorithm updated at the sample rate ($p = 1$), both of them being less complex than the previously published MFX-GSFAP algorithm. This is the typical situation in most cases, but for some particular parameter values, especially for low projection orders $N$ and high updating factors $p$, the MFX-GSFAP algorithm can be slightly less complex than the new MFX-GSPAP algorithm. If $I, J, K, L, M$ have fixed values, then the MFX-DCDPAP algorithm complexity is only proportional with $N$ (in terms of multiplies), while the complexity of the MFX-GSPAP and MFX-GSFAP algorithms is proportional with $N^2$. It can be seen from Fig. 3 that the MFX-DCDPAP algorithm is the most efficient, particularly for high values of the projection order $N$. Similar conclusions can be obtained if $I, J, L, M, N$ are fixed and $K$ is variable.

The last two columns of Table 1 give a performance/cost ratio obtained from the attenuation achieved by the algorithms after 50,000 iterations (averaged over the last 5000 iterations) for ideal and noisy plant models, divided by the number of multiplications per iteration. It can be seen that the proposed MFX-DCDPAP and MFX-GSPAP algorithms provide the best performance/cost ratio for the considered ANC system parameters. It can also be seen from Table 1 that the performance/cost ratios decrease when noisy plant models are used, although the two new proposed algorithms still produce the best performance. It should also be mentioned that other algorithms for multichannel ANC such as RLS-based algorithms produce much weaker performance/cost ratios, as found in [5]. Thus the new proposed algorithms are an interesting option for real implementations.

V. SIMULATIONS

The new MFX-GSPAP and MFX-DCDPAP algorithms were simulated and compared to the previously published multichannel modified filtered-x LMS algorithm (MFX-LMS, [19]) and the multichannel modified filtered-x GS-FAP algorithm (MFX-GSFAP, [5]). We used in our simulation $I = 1, J = 3, K = 2$ and the reference signal was a white noise with zero mean and variance one. The simulations were performed with acoustic transfer functions experimentally measured in a duct. The impulse responses used for the multichannel acoustic plant had 64 samples each ($M = 64$), while the adaptive filters had 100 coefficients.
each \( L = 100 \). For all the affine projection algorithms, a value of 0.9 was used for the step size \( \mu \) and the regularization factors were \( \delta = \delta_2 = 2 \cdot 10^3 \) for the ideal case (clean plant models) and \( \delta = 10^4 \), \( \delta_2 = 2 \cdot 10^4 \) for plant models with a signal to noise (SNR) ratio of 10 dB. The step size \( \mu \) for the MFX-LMS algorithm was \( 2 \cdot 10^{-5} \) and the parameter \( H \) of the DCD algorithm was set to 1/128. The convergence performance has been averaged over 200 simulations. The performance of the algorithms was measured by

\[
\text{Attenuation (dB)} = 10 \cdot \log_{10} \left( \sum_{i} \mathbb{E}[e_i^2(n)] \right) - \sum_{i} \mathbb{E}[d_i^2(n)] \tag{35}
\]

In [4], the effect of the projection order \( N \) on the convergence performance of the multichannel MFX-FAPRPS algorithm for the considered experimental setup (i.e. acoustic transfer functions) was evaluated. Note that the convergence performance of the MFX-FAPRPS was also found in [5] to be identical to the MFX-GSFAP (except for potential numerical stability issues). A value of \( N = 1 \) for the multichannel MFX-FAPRPS or MFX-GSFAP algorithms corresponds to a normalized version of the multichannel MFX-LMS algorithm, and it only provides a marginal convergence improvement over that standard algorithm [4]. A value of \( N = 5 \) was found in [4] to provide a significant convergence gain compared to the case with \( N = 1 \), with over 10 dB of extra convergence between 20000 and 50000 iterations completed by the algorithms. Values of \( N = 10 \) and \( N = 100 \) can provide further convergence gain, especially in the early stages of convergence (in the first 20000 iterations), however the difference in convergence compared to the case \( N = 5 \) becomes quite small as the number of iterations increases: almost no gain for \( N = 10 \) and only 2-3 dB of gain for \( N = 100 \). Therefore, for the considered experimental setup, the extra convergence performance provided by using \( N = 10 \) or \( N = 100 \) is not worth the extra computational complexity that they would bring over the case \( N = 5 \). Thus a value of \( N = 5 \) was selected for our simulations, as in [4], [5].

Fig. 4 shows that the implementation using 12 DCD iterations and 16 bits provides an almost identical performance with the method using the ideal matrix inverse (which corresponds to the flat horizontal 0dB line in Fig. 4). In this case, the theoretical peak complexity of the DCD algorithm is 200 SACs. Also, it can be seen in the same figure that if an average loss of about 1 dB is allowed, the number of bits can be reduced to 8 and the peak DCD complexity to 160 SACs. However, the average DCD complexity is around 60% of the theoretical peak complexity in both cases (125 and 90, respectively). The DCD part increases the number of additions, but it has no divisions or multiplications. Therefore, \( N_{\text{udp}} = 12 \) and \( M_b = 16 \) were used in the simulations of the MFX-DCDPAP algorithm.

Fig. 5 compares the performance of the selected algorithms averaged over 200 simulations, with ideal plant models, for a multichannel ANC system obtained from Matlab™ implementations of the algorithms (double precision 64 bits floating point format). It can be seen that the MFX-GSPAP and MFX-DCDPAP algorithms have almost the same performance as the previously published MFX-GSFAP algorithm. As expected, their convergence performance is also better than that of the LMS-based algorithm.

Fig. 6 shows the performance when plant models with a 10 dB SNR were used, again averaged over 200 simulations. Noise was added on a frequency by frequency basis to the ideal plant models, i.e. in the frequency response a random complex value with a magnitude of 10 dB less that the original magnitude was added to each frequency, to generate the noisy plant models. It can be seen from Fig. 6 that, when 10 dB SNR plant models were used, the MFX-GSPAP algorithm outperformed the other algorithms. Therefore, the approximation used in deriving the MFX-DCDPAP algorithm reduces a bit its robustness and performance with noisy plant models in comparison with the MFX-GSFAP algorithm. However, it should also be mentioned that other algorithms for multichannel ANC such as RLS-based algorithms produce a much weaker convergence performance when noisy plant models are used, as found in [4],[5]. Thus the new proposed algorithms are again an interesting option for real implementations.

VI. CONCLUSIONS

The multichannel MFX-GSPAP and MFX-DCDPAP algorithms were introduced for practical active noise control systems using FIR adaptive filters. Both algorithms provide a significant improvement of the convergence speed over the MFX-LMS algorithm, with a similar computational complexity. The proposed algorithms have either a typically lower computational complexity or a better numerical stability than the previously published fast affine projection algorithms for multichannel ANC. The proposed algorithms provide an excellent performance/cost ratio and are thus good candidates for practical real-time implementations. When choosing between the two proposed algorithms, the advantage of the MFX-DCDPAP algorithm is its reduced numerical complexity (in terms of multiplies), especially for high projection orders. On the other hand, the MFX-GSPAP has shown to produce a better performance when used with noisy plant models, thus it is potentially more robust to inaccuracies of the plant model.

VII. ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their useful comments.

REFERENCES


## TABLE I

Comparison of the computational load of the MFX-GSPAP and MFX-DCDPAP algorithms with other multichannel delay-compensated modified filtered-X algorithms for multichannel ANC systems, and evaluation of a performance/cost ratio for \( p = 1 \) (the values between accolades corresponds to \( p = 10 \)).

<table>
<thead>
<tr>
<th>Algorithm for multichannel ANC systems, ( L=100, M=64, N=5 )</th>
<th>Multiplications per iteration for ( I=1, J=1, K=1 )</th>
<th>Multiplications per iteration for ( I=1, J=3, K=2 )</th>
<th>Performance/cost ratio after 50000 iterations, for ( I=1, J=3, K=2, L=100, M=64 ) and ideal plant models (dB/multiplication)</th>
<th>Performance/cost ratio after 50000 iterations, for ( I=1, J=3, K=2, L=100, M=64, 10 ) dB SNR plant models (dB/multiplication)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MFX-LMS [19]</td>
<td>428</td>
<td>2,268</td>
<td>-4.0E-03</td>
<td>-3.4E-03</td>
</tr>
<tr>
<td>MFX-DCDPAP</td>
<td>443</td>
<td>2,448</td>
<td>-6.9E-03</td>
<td>-4.9E-03</td>
</tr>
<tr>
<td>MFX-GSPAP</td>
<td>473 (451)</td>
<td>2,686 (2,506)</td>
<td>-6.3E-03 (-6.6E-03)</td>
<td>-4.7E-03 (-4.9E-03)</td>
</tr>
<tr>
<td>MFX-GSFAP [5]</td>
<td>479 (457)</td>
<td>2,796 (2,616)</td>
<td>-6.1E-03 (-6.3E-03)</td>
<td>-4.1E-03 (-4.2E-03)</td>
</tr>
</tbody>
</table>

Fig.1. A multichannel feedforward active noise control system with adaptive FIR filters.
Fig. 2. A delay compensated or modified filtered-x structure for active noise control.

Fig. 3. The number of multiplications per algorithm iteration for the MFX-GSFAP, MFX-GSPAP and MFX-DCDPAP algorithms and for variable projection orders with two situations:

a) $I = 1, J = 1, K = 1, L = 100, M = 64, p = 1$

b) $I = 1, J = 3, K = 2, L = 100, M = 64, p = 1$.

Fig. 4. The attenuation difference over 50,000 iterations between the convergence curves of the MFX-DCDPAP algorithm using the ideal matrix inverse and the algorithm using different numbers of DCD iterations and bits.

Fig. 5. Convergence curves of multichannel delay-compensated modified filtered-x algorithms for ANC, with ideal plant models, averaged over 200 simulations.

Fig. 6. Convergence curves of multichannel delay-compensated modified filtered-x algorithms for adaptive noise control, with 10 dB SNR noisy plant models, averaged over 200 simulations.