A Low-Complexity Complementary Pair Affine Projection Adaptive Filter

Kwang-Hoon KIM\(^{(a)}\), Nonmember, Young-Seok CHOI\(^{(b)}\), Member, Seong-Eun KIM\(^{(c)}\), Nonmember, and Woo-Jin SONG\(^{(d)}\), Member

SUMMARY
We present a low-complexity complementary pair affine projection (CP-AP) adaptive filter which employs the intermittent update of the filter coefficients. To achieve both a fast convergence rate and a small residual error, we use a scheme combining fast and slow AP filters, while significantly reducing the computational complexity. By employing an evolutionary method which automatically determines the update intervals, the update frequencies of the two constituent filters are significantly decreased. Experimental results show that the proposed CP-AP adaptive filter has an advantage over conventional adaptive filters with a parallel structure in that it has a similar convergence performance with a substantial reduction in the total number of updates.

key words: adaptive filter, complementary pair, intermittent update, affine projection (AP), update interval

1. Introduction
Adaptive filters have an inherent trade-off between convergence speed and residual error [1]. To address this, various variable step-size schemes have been presented [2]–[4]. As a different approach, a convex combination of two adaptive filters (i.e., a fast filter and a slow filter) has been recently developed to achieve better performance than that of a single adaptive filter [5], [6]. However, this has a drawback of introducing extra computations as compared with a single adaptive filter. Because of this disadvantage, the use of combinational adaptive filters, especially affine projection (AP) [7] filters, has not received much attention as a solution. Recently, an adaptive mixture of two AP filters with different projection orders, one with a high projection order (fast AP filter) and the other with a low projection order (accurate AP filter), has been proposed [8]. To reduce the computational cost, an alternating selection scheme for parallel AP filters (A-AP) has been devised [9]. This scheme has almost the same amount of computational burden as a single AP filter without degradation of the convergence performance by automatically switching between the speed mode AP filter and the accuracy mode AP filter. However, there is still a computation issue with the AP filter. In this letter, to further reduce the redundant overhead in each adaptation stage, we present a low-complexity scheme consisting of a complementary pair AP (CP-AP) filter which intermittently updates the filter coefficients of each AP filter. The intermittent update is executed in an evolutionary manner, where the update interval is decreased or increased from the previous value by comparing the power of the output error with the threshold associated with the steady-state mean-square error (MSE) [10]. The proposed scheme carries out a status-dependent adaptation of the filter coefficients in each AP filter, leading to a significantly reduced update frequency. This update is based on the distinct MSE with respect to the constituent AP filters. We demonstrate that the proposed CP-AP filter has a similar convergence performance to conventional approaches while dramatically reducing the overall number of updates.

2. A Low-Complexity Complementary Pair Affine Projection (CP-AP) Filter
Consider data \(d(i)\) that originate from the system identification model
\[
d(i) = \mathbf{u}_i^\top \mathbf{w}^o + \nu(i),
\]
where \(\mathbf{w}^o\) is the column vector for the impulse response of an unknown system that we wish to estimate, \(\nu(i)\) accounts for measurement noise, and \(\mathbf{u}_i\) denotes the \(1 \times M\) row input vector given by:
\[
\mathbf{u}_i = [u(i) \; u(i-1) \; \ldots \; u(i-M+1)],
\]
where \(\mathbf{u}_i\) and \(d(i)\) can be expanded in the form of a matrix:
\[
\begin{pmatrix}
\mathbf{u}_i \\
\mathbf{u}_{i-1} \\
\vdots \\
\mathbf{u}_{i-K+1}
\end{pmatrix}
\begin{pmatrix}
d(i) \\
d(i-1) \\
\vdots \\
d(i-K+1)
\end{pmatrix}.
\]
Let \(\hat{\mathbf{w}}_i\) be an estimate for \(\mathbf{w}^o\) at time \(i\), we obtain the update rule of the AP filter such that

---

Copyright © 2014 The Institute of Electronics, Information and Communication Engineers
where $e_i = d_i - U_i w_{i-1}$ and $\mu$ is the step size in the range $0 < \mu < 2$. We mix the outputs of the two AP filters to exploit the merits of the two component filters. As shown in Fig. 1, the outputs of the two AP filters are combined as follows:

$$y(i) = \lambda(i) y_1(i) + [1 - \lambda(i)] y_2(i),$$

(4)

where $y_1(i)$ and $y_2(i)$ are the outputs of the two AP filters at time $i$, i.e., $y_k(i) = u_i w_{k,i-1}, k = 1, 2$. Here, $w_{1,i-1}$ is a fast AP filter and $w_{2,i-1}$ is a slow AP filter, respectively. The $\lambda(i)$ is a mixing parameter and is modified using an auxiliary variable $a(i)$ via sigmoidal function for soft switching between the component filters [5], [6], that is,

$$\lambda(i) = \text{sgm}(a(i)) = (1 + e^{-a(i)})^{-1}.$$

(5)

The adaptation of $a(i)$ is carried out to minimize the squared error $e^2(i)$ by using the gradient descent method via the LMS-type adaptation rule

$$a(i+1) = a(i) + \mu_a e(i)[y_1(i) - y_2(i)] \cdot \lambda(i)[1 - \lambda(i)]$$

$$+ \rho (a(i) - a(i-1)),$$

(6)

where $e(i) = d(i) - u_i w_{i-1}, \mu_a$ is the step-size used in the adaptation of $a(i)$, and $\rho$ is a momentum parameter. The selection of $\mu_a$ and $\rho$ is not affected by the particular scenario in which the filter is being applied. In general, $\mu_a = 100$ is a good choice as shown in [6]. When the outputs of the two filters are similar, the convergence of $a(i)$ will be slow. The momentum parameter is added and set to $\rho = 0.9$ to alleviate this problem [6]. In addition, the convergence performance of the combined filters can be further improved by utilizing the fast convergence behavior of the fast AP filter to increase the convergence rate of the slow AP filter when an abrupt change occurs. This can be carried out by the step by step transfer of a portion of the weight $w_1$ to $w_2$ [6]. Thus, the modified adaptation rule for $w_2$ is obtained as follows:

$$\tilde{w}_{2,i} = \alpha w_{2,i} + (1 - \alpha) w_{1,i-1} \text{ for } \lambda(i) > t,$$

(7)

where $\alpha$ is a parameter close to 1. Following the guidelines in [6], we used $\alpha = 0.9$ and $t = 0.98$ in all cases.

2.1 Selection of Interval Size for Filter Coefficient Update

Here, we present a new scheme combining two complementary AP filters to utilize the characteristic of the AP filter with different update intervals [11]: the lower the update interval, the faster the convergence. The larger the update interval, the lower the steady-state error. The former characteristic is applied to the fast AP filter (with a lower update interval) which achieves fast initial convergence. The latter characteristic is applied to the slow AP filter (with a larger update interval) which provides a low steady-state MSE. In this manner, we present a complementary pair affine projection (CP-AP) filter that intermittently updates the filter coefficients, working toward a state-aware update interval. Therefore, the selection of the update interval is essential to reach our main goal of reducing the heavy computational overhead. The update equations of the AP filters used in the proposed scheme can be written as

$$w_{1,j} = \begin{cases} 
 w_{1,j-1} + \mu_1 U_j^i e_{1,j} & \text{if } i \mod n_1(i) = 0 \\
 w_{1,j-1} & \text{otherwise}
\end{cases}$$

(8)

$$w_{2,j} = \begin{cases} 
 w_{2,j-1} + \mu_2 U_j^i e_{2,j} & \text{if } i \mod n_2(i) = 0 \\
 w_{2,j-1} & \text{otherwise}
\end{cases}$$

(9)

where $U_j^i = U_j^i (U_j^i)^{-1}, e_{1,j} = d_j - U_j^i w_{k,j-1}, k = 1, 2$, and $w_{k,j-1}$ is the $k$th filter coefficient (i.e., $w_{1,j-1}$ is a fast AP filter and $w_{2,j-1}$ is a slow AP filter). The parameters $n_1(i)$ and $n_2(i)$ are adjustable update intervals at the $i$th time instant for the fast and slow AP filters, respectively. The estimate $w_{1,j}$ is kept equal to $w_{1,i-n_1(i)}$ during the time instants $j = i - n_1(i) + 1, i - n_1(i) + 2, \ldots, i - 1$, and $w_{2,j}$ is kept equal to $w_{2,i-n_2(i)}$ during the time instants $j = i - n_2(i) + 1, i - n_2(i) + 2, \ldots, i - 1$. Hence, the filter coefficients of the fast AP filter are updated only once every $n_1(i)$ iterations, and the filter coefficients of the slow AP filter are updated only once every $n_2(i)$ iterations.

To determine the update interval of each AP filter, we propose an evolutionary approach which adjusts the update interval according to the adaptation stage of the CP-AP filter. In this context, the update intervals $n_1(i)$ and $n_2(i)$ can be formulated as follows:

$$n_1(i) = \begin{cases} 
 \max[n_1(i-1) - 1, 1] & \text{if } e^2(i) > \gamma_1 \\
 \min[n_1(i-1) + 1, N_1] & \text{if } \gamma_2 \leq e^2(i) \leq \gamma_1 \\
 \min[n_1(i-1) + 1, N_1] & \text{if } e^2(i) < \gamma_2
\end{cases}$$

(10)

$$n_2(i) = \begin{cases} 
 \max[n_2(i-1) - 1, 1, N_2] & \text{if } e^2(i) > \gamma_1 \\
 \min[n_2(i-1) - 1, 1, N_2] & \text{if } \gamma_2 \leq e^2(i) \leq \gamma_1 \\
 \min[n_2(i-1) - 1, 1, N_2] & \text{if } e^2(i) < \gamma_2
\end{cases}$$

(11)

where

$$e(i) = \lambda(i)e_1(i) + (1 - \lambda(i))e_2(i),$$

(12)

where $e_1(i) = d(i) - u_i w_{k,i-1}, k = 1, 2, \gamma_1$ and $\gamma_2$ denote the error bounds, and $\lambda(i)$ is the mixing parameter. $N_1$ and $N_2$ are the maximum update intervals for the two AP filters. The $\lambda(i)$ is updated to minimize the overall error of the proposed CP-AP filter. As will be discussed in the next subsection, the $\gamma_1$ and $\gamma_2$ are closely related to the criterion of the state transition. In addition, $n_1(i - 1)$ and $n_2(i - 1)$ are the update intervals for the two AP filters at the $(i - 1)$th time instant. In (10), the update interval $n_1(i)$ is adjusted by comparing
the squared output error with the thresholds, \( \gamma_1 \) and \( \gamma_2 \), and satisfies \( 1 \leq n_1(i) \leq N_1 \). To adjust \( n_2(i) \), the same process is executed using (11). For the symmetric framework, we assume that \( N_1 \) and \( N_2 \) are the same and denoted as \( n_{\text{max}} \). When the squared output error is larger than \( \gamma_1 \), we regard the adaptation phase as the initial state. Thus, the update interval \( n_1(i) \) should be reduced by one from \( n_1(i - 1) \) for faster initial convergence. However, the update interval \( n_2(i) \) should be increased by one from \( n_2(i - 1) \) for sparser updates. When the squared output error, i.e., \( e(i) \), is larger than \( \gamma_2 \) and smaller than \( \gamma_1 \), we regard the adaptation phase as the transient state. In this phase, \( n_1(i) \) should be increased by one from \( n_1(i - 1) \) for sparser updates. Simultaneously, the update interval \( n_2(i) \) should be reduced by one from \( n_2(i - 1) \) for smaller transient-state error. Finally, when the squared output error is smaller than \( \gamma_2 \), we regard the adaptation phase as the steady-state. In this case, the update interval for both AP filters should be increased to achieve lower steady-state residual error. This helps greatly reduce the computational burden in the parallel AP filters.

2.2 Determination of the Error Bounds

As suggested by (10) and (11), sparse or dense updating is accomplished by comparing the squared error with the thresholds, i.e., \( \gamma_1 \) and \( \gamma_2 \). Accordingly, each threshold must reflect the state of the AP filter. In [1], the following relation is presented:

\[
\lim_{i \to \infty} E[e^2(i)] = \lim_{i \to \infty} E[e_0^2(i)] + \sigma_v^2. \tag{13}
\]

The expectation of the square of \( e(i) \) as \( i \) tends to infinity, i.e., \( \lim_{i \to \infty} E[e^2(i)] \), is the mean-square error (MSE), and \( \lim_{i \to \infty} E[e_0^2(i)] \) denotes the excess mean-square error (EMSE). The theoretical EMSE of the AP filter was studied in [10]:

\[
J_{x_e}(\infty) = \frac{\mu \sigma_v^2}{2 - \mu} \text{Tr}(R_v) E \left[ \frac{K}{||u||^2} \right]. \tag{14}
\]

From (13) and (14), the MSE is expressed as

\[
\lim_{i \to \infty} E[e^2(i)] = \text{MSE} = \frac{\mu \sigma_v^2}{2 - \mu} \text{Tr}(R_v) E \left[ \frac{K}{||u||^2} \right] + \sigma_v^2, \tag{15}
\]

where \( R_v = E[u_i^* u_i] \) and \( \text{Tr}(\cdot) \) denotes the trace of the matrix. Since it is not feasible to obtain the exact value in (15), we replace it with an instantaneous value. For a large \( M \), the fluctuations in the input signal energy \( ||u||^2 \) from one iteration to the next are sufficiently small to justify the following approximation [1], [12], [13]:

\[
E \left[ 1/||u||^2 \right] \approx 1/E \left[ ||u||^2 \right]. \tag{16}
\]

From (15) and (16), therefore, we set the thresholds in (10) and (11) as follows:

\[
\gamma_1 \approx \mu_1 \sigma_v^2 K/(2 - \mu_1) + \sigma_v^2, \quad \gamma_2 \approx \mu_2 \sigma_v^2 K/(2 - \mu_2) + \sigma_v^2. \tag{17}
\]

The \( \gamma_1 \) and \( \gamma_2 \) are the MSE values of the fast AP and slow AP filters, respectively. The noise variance can be easily estimated during silences [14] or online [15].

3. Experimental Results

We illustrate the performance of the proposed algorithms by carrying out computer simulations in a channel estimation scenario. The unknown channel \( H(z) \) is represented by a moving average model with 16 taps. The adaptive filter and the unknown channel are assumed to have the same number of taps (\( M=16 \)). A signal with a Gaussian distribution is used for the input signal. The input signals are obtained by filtering a white, zero-mean, Gaussian random sequence through a first-order system \( G(z) = 1/(1 - 0.9z^{-1}) \). The signal-to-noise ratio (SNR) is calculated by

\[
\text{SNR} = 10\log_{10} \left( E[y^2(i)]/E[e^2(i)] \right) \tag{19}
\]

where \( y(i) = u_i w^o \). The measurement noise \( e(i) \) is added to \( y(i) \) with SNR=30 dB. Also, we assume that the noise variance, \( \sigma_v^2 \), is known, because it can be easily estimated during silences and online in many practical applications [14], [15]. The mean square deviation (MSD), \( E[||w^o - w_i||^2] \), is taken and averaged over 100 independent trials. The parameter settings chosen for all the simulations are \( \rho = 0.9 \), \( \alpha = 0.9 \), \( t = 0.98 \), and \( \mu_e = 100 \).

Table 1 lists the number of update ratio per 8000 iterations. We can see that the proposed framework has the feature of intermittent updates of each AP filter. As \( n_{\text{max}} \) increases, the ratio of the updates is decreased. It is clear that the total computational complexity of the proposed CP-AP filter is significantly less than that of the conventional combination structure owing to the individual and intermittent updates. Also, the proposed CP-AP filter has a lower computational complexity than the A-AP filter. Furthermore, the proposed CP-AP filter in parallel has roughly half computational burden of the single AP filter when \( n_{\text{max}} = 64 \).

Figure 2 shows a plot of the MSD curves for the AP (\( \mu_1 = 1.0 \), \( \mu_2 = 0.002 \), \( K = 8 \), \( M = 16 \), \( n_{\text{max}} \) = 1.0, \( \alpha = 0.99 \), and \( C = 0.5 \)) [3], A-AP (\( \mu_1 = 1.0 \), \( \mu_2 = 0.002 \), \( K = 8 \), \( M = 16 \), \( \mu = 0.002 \)) [8], and the proposed CP-AP filter. The step-sizes corresponding to the fast and slow AP filters of the proposed CP-AP filter are selected to provide a similar steady-state level attained by the A-AP filter to compare the convergence performance.

<table>
<thead>
<tr>
<th>( n_{\text{max}} )</th>
<th>Fast AP Filter</th>
<th>Slow AP Filter</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>15.64   (%)</td>
<td>62.93   (%)</td>
<td>78.57 (%)</td>
</tr>
<tr>
<td>16</td>
<td>9.85 (%)</td>
<td>59.54 (%)</td>
<td>69.39 (%)</td>
</tr>
<tr>
<td>32</td>
<td>6.67 (%)</td>
<td>54.98 (%)</td>
<td>61.65 (%)</td>
</tr>
<tr>
<td>48</td>
<td>6.22 (%)</td>
<td>52.87 (%)</td>
<td>59.09 (%)</td>
</tr>
<tr>
<td>64</td>
<td>5.81 (%)</td>
<td>50.91 (%)</td>
<td>56.72 (%)</td>
</tr>
</tbody>
</table>
Fig. 2  MSD curves of the AP, VS-AP [10], A-AP [9], C-AP [8], and the proposed CP-AP. \(\mu_1 = 1.0, \mu_2 = 0.002, n_{\text{max}}=8, K = 8, M = 16,\) SNR = 30 dB, Input: Gaussian AR(1), pole at 0.9.

Fig. 3  Averaged evolution of the update intervals \(n_1(i)\) and \(n_2(i)\) for the proposed CP-AP filter.

Fig. 4  Average evolution of the update intervals \(n_1(i)\) and \(n_2(i)\) for various maximum update intervals in the proposed CP-AP filter.

Fig. 5  MSD curves of the proposed CP-AP filter for various maximum update intervals.

of comparison. As can be seen, the proposed CP-AP filter retains the rapid convergence of the fast AP filter and the low steady-state error of the slow AP filter. Note also that the proposed CP-AP filter has a similar convergence performance to the conventional C-AP filter but only involves nearly 80% of the total computational cost for the single AP filter (\(n_{\text{max}} = 8\)).

Figure 3 shows the averaged update interval over 100 independent trials. As expected, the fast AP filter is more frequently updated than the slow AP filter in the initial state. During the transient state, however, the slow AP filter is more frequently updated than the fast AP filter to obtain a low steady-state error. Next, in the steady-state, the update interval of both AP filters is kept large. As demonstrated by the above, the proposed CP-AP filter remarkably reduces the total number of updates enabling the power consumption to be significantly reduced.

Figure 4 shows the average evolution of the update interval \(n_1(i)\) and \(n_2(i)\) for various maximum update intervals in the proposed CP-AP filter. As observed in Fig. 3, the filter coefficients of the fast AP filter are densely updated and the weight vector of the slow AP filter is sparsely updated in the initial state. In the transient state, only the update interval of the fast AP filter increases, resulting in sparser updates. Then, the update intervals of both AP filters increase in the steady-state. Note that the update interval of the fast AP filter increases until it reaches its maximum value, i.e., \(n_{\text{max}}\). The update interval of the slow AP filter, on the other hand, increases until it reaches half of \(n_{\text{max}}\) for the fast AP filter.

Figure 5 shows the convergence behavior of the proposed CP-AP filter with different maximum update intervals. We can see that the convergence behavior is not highly sensitive to the value of \(n_{\text{max}}\).

Figure 6 shows the tracking performance of the proposed CP-AP filter. The unknown channel is changed abruptly from \(w_0\) to \(-w_0\) at the 6000th iteration. It can be seen that the proposed CP-AP works well in the case of a sudden weight change without degrading the convergence performance.

4. Conclusions

In this letter, we have presented a low-complexity complementary pair affine projection (CP-AP) filter. The proposed scheme carries out the weight adaptation by adjusting the update interval for two constituent filters. By optimally balancing the update intervals between the two AP filters, we
Fig. 6 MSD curves of AP, VS-AP, A-AP, C-AP, and the proposed CP-AP filter. The unknown system is suddenly changed from \( w^u \) to \(-w^u\).

exploit the trade-off between a fast convergence and a small steady-state residual error. The resultant intermittent updates of the CP-AP filter substantially reduce the computational burden of the two AP filters in parallel. Experimental results indicate that the CP-AP filter has a similar convergence performance to previous combination schemes while greatly reducing the overall computational burden.

Acknowledgments

This work was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MEST) (2012R1A2A2A01011112), and in part by the MSIP (Ministry of Science, ICT and Future Planning), Korea, under the CITRC (Convergence Information Technology Research Center) support program (NIPA-2014-H0401-14-1001) supervised by the NIPA (National IT Industry Promotion Agency).

References


