FIXED-WINDOW CONSTANT MODULUS ALGORITHMS:
ADAPTIVE IMPLEMENTATIONS

Xiang-yang Zhuang and A. Lee Swindlehurst
Department of Electrical and Computer Engineering
Brigham Young University, Provo, UT 84602
email: {jeffz, swindle}@ee.byu.edu

ABSTRACT
Adaptive implementations of two recently proposed block-
iterative algorithms, referred to as fixed-window constant
modulus algorithms (FWCMA), are developed. Their tran-
sient behavior, in relation to the delay at which the user
sequence is recovered, is discussed and compared with
other existing variations of CMA. The adaptive FWCMA-
2 appears to be particularly promising, attaining conver-
gence with little data. It retains the merits of its batch
counterpart, namely, fast convergence, robustness to ill-
conditioning, no required step-size tuning, and relatively
low complexity. The built-in control over the converged delay
translates into more reliable and predictable transient be-
behavior. Equalizer calculation can be initiated from any ini-
tialization, and at any time instant, to attain convergence
instantaneously without storing any received data. Thus,
different initializations can be used to easily obtain multiple
equalizers recovering the signal at different delays or recov-
ering distinct users under a near-far situation.

1. INTRODUCTION
In [1], we recently proposed two fixed-window constant
modulus algorithms (FWCMA-1 and FWCMA-2) in which
a fixed block of samples is iteratively reused. These fixed
window CM algorithms are data efficient, computationally
inexpensive and require no step-size tuning. The number
of iterations needed for the FWCMA to attain conver-
gence is small due to its proven super-fast convergence.

The FWCMA are also robust to channel ill-conditioning
and can even be applied to the single-channel equalization
case where the channel matrix has more columns than rows.
While other versions of CMA demonstrate unpredictable
transient behavior and very different steady-state MSE per-
formance from trial to trial with a near-far situation.

However, the FWCMA proposed in [1] are batch imple-
mentations. Usually an adaptive implementation is desired
due to the non-stationarity of rapidly varying wireless chan-
nels and other drawbacks. For example, in batch process-
ing, symbol decisions must wait until the whole data block is
collected and processed. The processing load is unbalanced
between the data collection and processing phase. More-
ever, batch processing requires more storage than stream
processing. In this paper, after introducing the batch al-
gorithms, we develop adaptive update rules based on both
FWCMA-1 and FWCMA-2. Connections to other CMA
variations and convergence behavior under noisy observa-
tions are discussed, from which a modification of the adap-
tive FWCMA-1 is introduced. The extension of the adap-
tive FWCMA-2 to multi-user cases is also proposed. The
simulation results are presented last.

2. DATA MODEL
To begin, we consider a single user transmitting through
multiple channels which are obtained by either oversam-
pling in time or using an antenna array. For the single user
case, the data model can be written as

\[ x(n) = [x_{MP}^T(n) \cdots x_{MP}^T(n-E+1)]^T \]

where \( M \) is the number of sensors, \( P \) is the oversampling
factor, \( x_{MP}(n) \) is a vector containing samples from all \( MP \)
channels at time instant \( n \), \( L \) is the effective channel length
in symbol periods, and \( E \) is the number of taps in the tem-
poral filter implemented after each sub-channel (i.e., the
'smoothing' factor). Also, \( h_{l} \) is an \( MP \)-tuple vector con-
taining the coefficients of all the sub-channels at time instant \( i \),
\( s(n) = [s(n) \cdots s(n-E-L+2)]^T \) is the signal vector and
\( n(n) = [n_{l}^T(p(n)) \cdots n_{l}^T(p(n-E+1))]^T \) is the noise vector.

With \( d \) users, the model becomes

\[ x(n) = [H_{1} \cdots H_{d}] \begin{bmatrix} s_{1}(n) \\ \vdots \\ s_{d}(n) \end{bmatrix} + n(n) = \mathcal{H}s + n(n) \]

where \( \mathcal{H} \) is a complex matrix of dimension \( m \times q \) with
\( m \equiv MPE \) and \( q \equiv dE - d + \sum_{i=1}^{d} L_{i} \). A full column

Authorized licensed use limited to: IEEE Editors in Chief. Downloaded on August 22, 2009 at 23:48 from IEEE Xplore. Restrictions apply.
rank $\mathcal{H}$ is usually assumed, though we mention here that the FWCMA still applicable when $\mathcal{H}$ is rank-deficient or when $\mathcal{H}$ is "wide" as encountered in single-channel equalization. We assume an i.i.d. data sequence and mutual independence between multiple users as well.

3. ALGORITHM DEVELOPMENT

Denoting the equalizer output as $y = w^H x$ where $(\cdot)^H$ is the conjugate transpose, the CM(2,2) cost function is:

$$J(w) = E(1 - |y|^2)^2 = E(1 - w^H xx^H w)^2.$$  (3)

FWCMA-1 is based on the following approximation:

$$J(w_{k+1}) \approx E(1 - w_{k+1}^H xx^H w_k)^2$$  (4)

where the subscript "k" denotes the block iteration number. The underlying idea of such an approximation is that we seek the weight change making the post-estimation output $y_{k+1}$ of unity modulus.

The batch FWCMA-1 is derived as the Wiener solution to (4) followed by a normalization. Denoting $z_k = xx^H w_k = y_k^* x$, we summarize it as follows:

$$w_{k+1} = \frac{E(z_k z_k^H)}{E(z_k)} = R_z^{-1} R_{xz} w_k$$

$$w_{k+1} = w_{k+1} = \frac{w_{k+1}}{1 + R_{xx}^{-1} R_{xz} w_{k+1}}$$  (5)

The batch FWCMA-1 can also be obtained from minimizing (3) (not its approximation) via the steepest-descent approach, i.e., taking $\mu = \frac{1}{2} R_{xx}^{-1}$ in the following:

$$w_{k+1} = w_k - \frac{1}{2} \mu \nabla J,$$  (6)

where $\nabla J$ is the gradient of (3) with respect to $w$:

$$\nabla J = 4E\{ |y|^2 - 1 \} xx^H w = 4R_{zz} w - 4R_{xz} w.$$  (7)

The batch FWCMA-2 is obtained from (6) with a different choice of $\mu$, namely $\mu = \frac{1}{4} R_{xx}^{-1}$:

$$w_{k+1} = 2w_k - R_{xx}^{-1} R_{xz} w_k$$

$$w_{k+1} = w_{k+1} = \frac{w_{k+1}}{1 + R_{xx}^{-1} R_{xz} w_{k+1}}.$$  (8)

Theorem: If $\mathcal{H}$ is full column rank and the data sequence is i.i.d., then in the absence of noise the block FWCMA-1 of (5) and FWCMA-2 (8) will converge to the global solution in which the global response $g^H \triangleq w^H \mathcal{H}$ has only one non-zero component. If the $i^{th}$ entry of $|g_0|$ is the maximum element of the initial global response $g_0$, then $|g_0|$ will converge to 1 (hence, the equalizer will converge to delay $i-1$).

Proof: See [1].

Starting from some initialization $w_0$, $R_{zz}$ is re-computed at each iteration using the stored data block in both FWCMA-1 (which actually requires $R_z^{-1}$) and FWCMA-2. $R_{xx}$ is rank-deficient in the noiseless case, and a pseudoinverse should be used which requires a singular value decomposition with proper rank detection. Alternatively, direct inversion can be used after diagonal loading.

A common method of turning a block recursion into a sequential process is to treat the iteration index $k$ as the symbol index. Based on this approach, we first develop an adaptive version of FWCMA-1. We treat the non-stationary process $z_k = xx^H w_k$ in (4) as some "received" signal which is known at time instant $k + 1$, and treat "1" as the training data. An RLS-type symbol-by-symbol adaptation rule can then be obtained as follows:

$$y_{n} = w_{n-1}^H x_n; \quad z_n = y_{n} x_n$$

$$K_n = \frac{\lambda^{-1} P_{n-1} z_n}{1 + \lambda^{-1} z_n^H P_{n-1} z_n}$$

$$P_n = \lambda^{-1} P_{n-1} - \lambda^{-1} P_{n-1} z_n^H P_{n-1}$$

$$w_n = w_{n-1} + K_n (1 - |y_n|^2)$$  (9)

where $P_0$ and $w_0$ are suitably initialized and $\lambda$ is the exponential weighting factor for tracking time-varying channels. The symbol index is $n = 1, 2, \ldots$. We mention that the approximation introduced in (4) and the design of the adaptation rule somewhat resemble those used in the subspace tracking algorithm PAST (Projection Approximate Subspace Tracking) [2].

Alternatively, we can start from (6) to obtain the adaptive FWCMA-1. Still thinking of "k" in (6) as the symbol index, approximating the expectation in $VJ$ by its instantaneous value, and taking $\mu = \frac{1}{2} R_{xx}^{-1}$ results in

$$w_n = w_{n-1} + R_{xz}^{-1} x_n y_n (1 - |y_n|^2)$$  (10)

where $R_{xz}$ at time $n$ is denoted as $R_{x,n}$ and is updated via $R_{x,n} = \lambda R_{x,n-1} + z_n x_n^H$. It is not difficult to observe that the above adaptation is actually equivalent, though in a more concise form, to the previously derived adaptive FWCMA-1 in (9) where we apply the standard RLS method to the approximated cost function (4).

From the same steepest-descent rule and approximation, a plain LMS-type (or normalized) CMA can be obtained from a fixed (or normalized) scalar step-size. Similarly, a simple-minded adaptive implementation of FWCMA-2 can be developed by just taking $\mu = \frac{1}{2} R_{xx}^{-1}$ in (6) which results in

$$w_n = w_{n-1} + R_{xz}^{-1} x_n y_n (1 - |y_n|^2)$$  (11)

where $R_{x,n} = \lambda R_{x,n-1} + z_n x_n^H$. This can be compared in form with the Least-Squares CMA (LSCMA) [3], which is

$$w_n = w_{n-1} + R_{xz}^{-1} x_n (y_n^0 - y_n).$$  (12)

Note that LSCMA was originally derived from an RLS-type adaptation with the training symbols replaced by a projection onto the unit circle $f(y_n) = \frac{y_n}{|y_n|}$. Of course, other non-linear functions or tentative decisions can be used as training data. We can see that (11) and (12) are almost identical except the step-size of (12) is reduced by a factor of $\frac{1}{2} R_{xx}^{-1}$. We see that the sequential rules developed above are closely connected with existing methods. They all can be derived from (6) and more importantly, introduce approximations which may destroy the monotonic convergence that we have shown for block FWCMA-1 and FWCMA-2. The
loss of this feature could make the convergence behavior unpredictable as we will discuss in the next section. We desire an implementation that makes no approximation and possesses all the merits of the block algorithm, but at the same time, does not store all the data like the block methods. In the following, such an adaptive algorithm based on FWCM-2 is proposed.

Let us examine \( R_{s2} = E(xx^H|w^Hx|^2) \). Noticing that \( xx^H = \text{unvec}(x \otimes x) \) and \( |w^Hx|^2 = (x \otimes x)^H(w \otimes w) \), we can write \( R_{s2} \) as

\[
R_{s2} = \text{unvec}\{E[(x \otimes x^*)(x \otimes x^*)^H](w \otimes w^*)}\]  

(13)

where “\( \text{unvec} \)” is the inverse operation of row stacking, “\( \otimes \)” denotes the Kronecker product and “\( * \)” denotes conjugation. FWCM-2 requires only the statistics \( R_{s2} \) and \( R_{s2} \), but not the data itself. \( R_{s2} \), whose re-computation originally required stored the data, can now be obtained as above using the fourth moment matrix defined as \( G \equiv E[(x \otimes x^*)(x \otimes x^*)^H] \), which can be accumulated adaptively. \( R_{s2} \) is updated as in the standard RLS algorithm. Therefore, the iterations defined in (8) can now be easily started at any time instant during data accumulation. We can check whether convergence has been reached after each iteration by comparing the change between two g’s obtained in two consecutive iterations with a pre-set threshold \( \eta \). Alternatively, we can check if their inner product is close enough to “1”. As soon as enough data is processed so that the assumed i.i.d. property holds reasonably well and the accumulated statistics are reasonably accurate, the convergence should be achieved instantaneously within a small number of iterations, just like the batch method. After convergence, channel tracking usually takes only one iteration. Since the assumed i.i.d. property does not hold well for small amounts of data, we may just accumulate the statistics for the first \( N_0 \) samples without doing any iterations. If the statistics are still not good enough after \( N_0 \) samples, iterations may produce meaningless results which are even farther from the global solution than the initialization \( w_0 \). In this case, if iterations for the next symbol are initialized from that result, convergence can be slowed down. Furthermore, the delay to which the algorithm converges may not be the one expected from the initialization. So we simply choose to still start from the original \( w_0 \) at the next symbol if convergence is not declared after some number \( (D) \) iterations. Actually, since we can start the iterations from \( w_0 \) and at \( any \) time we want, there is significant flexibility in how the algorithm is implemented in addition to the way mentioned above. Also, we can start from several \( w_0 \) in parallel to get several equalizers in order to recover the same user at different delays or to recover distinct users. The same \( R_{s2} \) and \( G \) matrix can be used after they are accumulated once. One detailed implementation is described as follows:

**Initialization:**

\( P_0 = \delta^{-1} I \), \( \delta \) = small positive constant
\( R_0 = 0 \), covariance matrix
\( G_0 = 0 \), fourth moment matrix
For each time instant, \( n = 1, 2, ... \), compute

\[
R_n = \lambda R_{n-1} + x_n x_n^H
\]

\[
G_n = \lambda^2 G_{n-1} + (x_n \otimes x_n')(x_n \otimes x_n')^H
\]

\[
K_n = \frac{\lambda^{-1} P_{n-1} - x_n x_n^H}{1 + \lambda^{-2} R_{n-1} x_n x_n^H} P_{n-1}
\]

\[
P_n = \lambda^{-1} P_{n-1} - \lambda^{-2} K_n x_n x_n^H P_{n-1}
\]

If \( n > N_0 \) (iteration after \( N_0 \) samples)

\[
w_n = \frac{2w_{n-1} - P_n \text{unvec}(G_n(w_n \otimes w_n'))w_n}{\sqrt{w_n^H(R_n/n)w_n}}
\]

While \( (w_n - w_{n-1})^T R_n(w_n - w_{n-1}) > \eta \) \& \( I \leq D \)

\[
w_n = w_{n-1}; \quad I = I + 1
\]

\[
w_n = 2w_n - P_n \text{unvec}(G_n(w_n \otimes w_n'))w_n
\]

\[
w_n = w_n/\sqrt{w_n^H(R_n/n)w_n}
\]

End (While Loop)

*If \( I > D \), \( w_n = w_0/\sqrt{w_0^H(R_0/n)w_0} \)*

End (If “\( n > N_0 \)”)

The main computational burden of adaptive FWCM-2 is the accumulation of the fourth moment matrix \( G \) which requires \( m^4 \) complex flops at each instant. On the other hand, in the batch version, \( R_n \) can be obtained without accumulating \( G \), so the average complexity per symbol for the batch FWCM-2 may be even lower than its adaptive version. However, once \( G \) is obtained, it can be used to get multiple equalizers with little extra cost while the batch FWCM-2 must compute \( R_n \) for each equalizer and each iteration. Moreover, \( G \) is Hermitian and each row is a vectorized Hermitian matrix, a property that can be effectively exploited to reduce the complexity by a factor of about four. The computation may be further reduced if we consider the time-shift relationship introduced by the temporal filter “smoothing” in the data vector \( x \).

**4. CONVERGENCE BEHAVIOR, DELAY CONTROL AND MULTI-USER CASES**

Convergence speed is generally fast for the adaptive FWCM-1. But failures to achieve global convergence or slow convergence have been observed, especially under low SNR conditions and initializations that are far away from the global minima. Not only are there many local minima when noise is present, but the equalizer solution is not unique; i.e., there can be several global minima corresponding to different delays. Since the statistical expectation is approximated by an instantaneous estimate, the monotonic characteristic may lost during convergence. Adaptation may jump between basins of different global minima, which can cause unreliable transient behavior. The delay factor determines the steady-state MSE performance and could significantly affect convergence speed. The output SNR for delay \( i - 1 \) is determined by the eigenstructure of the channel matrix as follows (see derivation in [1]):

\[
SNR_i = \frac{E[w^H \mathcal{H} w]}{E[w^H n^2]} = \frac{1}{\sigma_i^2 (|\lambda_1|^4 + \ldots + |\lambda_q|^4)}
\]

where the SVD form \( \mathcal{H} = U_{m \times q} \Sigma_{m \times q} V_{q \times q} \) is assumed for \( \mathcal{H} \in \mathbb{C}^{m \times q} \), \( \lambda_i = \Sigma_{ii} \) (\( i = 1 \ldots q \)) and \( v_{ij} \) is the \((ij)\)-th element of \( V \).
In the adaptive FWCMA-1 of (9) or (10), the step-size gets smaller as more data are collected, since $R_{\mu,n}^1$ in (10) decreases in proportion to $1/n$ when $\lambda = 1$. This also partially explains why the RLS algorithm does not have good tracking performance. The step-size is even more important to blind methods. For example, in LSCMA where $R_{\mu,n}^1$ in (12) also decreases in proportion to $1/n$, the more influential first few tentative symbol decisions may well be incorrect, but are used as the training data. On the other hand, the normalized CMA (NCMA) [4] is more capable of escaping from local minima because of its non-decreasing step-size which depends only on $y_n$, though its bigger step-size also causes larger steady-state error. The performance of the normalized sliding window CMA (NSWCMA) [5] fits in between LSCMA and NCMA. Based on these observations, we modify the step-size of the adaptive FWCMA-1 to make it decrease more slowly, for example, by replacing $R_{\mu,n}^1$ with $\alpha\sqrt{n}R_{\mu,n}^1$, where $\alpha$ is suitably chosen to ensure stability and convergence. A similar adjustment can be applied to LSCMA. The rate at which the step-size decreases is now reduced to $1/\sqrt{n}$. This adjustment was used for both the adaptive FWCMA-1 and LSCMA in the simulations of the next section.

When the instantaneous gradient approximation is introduced, CM algorithms can not reliably control the delay to which they converge. But the adaptive FWCMA-2 preserves the delay control characteristics of its batch version, a property that is very much desired in multi-user cases. If users are received with approximately equal power, different spike initializations may be enough to result in the recovery of most of the users. If a near-far problem exists, however, a deflation approach can be used to recover weak users after the stronger ones have been extracted. After each iteration $k$ at time instant $n$, $g$ is projected onto a subspace orthogonal to the span of the previously obtained $g$’s and then the projection is re-normalized, i.e.,

$$w_{p+1} = (I - (R/n)W_p W_p^H)w_{p+1}$$

$$w_{k+1} = w_{k+1}^n/\sqrt{(w_{k+1}^n)^H (R/n)w_{k+1}^n}$$

where the columns of $W_p$ are the coefficients of the $p$ previously estimated equalizers. The initialization $w_0$ should also be projected. To prevent errors in the previous $w$’s from affecting the next one, we can relax the subspace confinement by omitting the projection step after several iterations, if $w$ is already out of the basins of the previous solutions. Note that our approach for multi-user case is different from that in [6] where an extra term measuring the output correlation is added to the cost function to penalize convergence to the same sequence.

5. SIMULATIONS

We simulate a three ray channel with a delay spread of 4T where T is the symbol period. Our parameters are $M = 2$ (two-sensor ULA), $P = 2$ (oversampling by a factor of two) and $E = 2$ (a two-tap filter following each of the 4 sub-channels). Thus $H$ is an 8 x 5 matrix. We use an arbitrary spike initialization $w_0$ which is far from the global solution and which should converge to delay 2 according to the Theorem. The channel output SNR is only 8dB. Our performance measure is the residual ISI defined as $ISI = \frac{\sum_{i=1}^{n} (g_i, e_i)^2}{\max_{i} |g_i|^2}$. Averaged over 200 Monte-Carlo runs, the result of adaptive FWCMA-1 is shown in Figure 1, in comparison with NCMA and LSCMA. The step-size of both FWCMA-1 and LSCMA have been adjusted as mentioned in Section 4 with the choice of $\alpha = 0.3$. For NCMA, $\mu = 0.25$ is also suitably chosen to optimize stability and convergence speed. We also plot, as a reference, the ISI for a Wiener filter also recovering data at delay 2. It is computed using all 800 transmitted symbols as training data. A typical run of the adaptive FWCMA-2 under the same setting is plotted in Figure 2. The first $N_0 = 20$ samples are used for accumulating statistics only and we set $D = 3$ and $\eta = 0.2$. In this simulation, we can see that convergence is reached when about 60 samples are processed. The behavior between sample $N_0$ and the convergence point is due to the fact that the i.i.d. assumption does not hold well for an insufficient number of symbols. In 200 trials, the average number of samples required for convergence was 62.6 with a minimum of 20, a maximum of 147 and a standard deviation of 23.4. The reliable convergence behavior of the adaptive FWCMA-2 is further reflected by the probability of converging to delay 2 in the Monte-Carlo trials, which is 0.95, compared with 0.46 for NCMA, 0.80 for LSCMA, and 0.73 for the adaptive FWCMA-1.

Figure 1: Convergence comparison of the adaptive FWCMA-1, LSCMA and NCMA. (SNR=8dB, QSPK)

Figure 2: Convergence of the adaptive FWCMA-2. (SNR=8dB, QSPK)

6. REFERENCES


