SQUARE-ROOT FORM BASED DERIVATION OF A NOVEL NLMS-LIKE ADAPTIVE FILTERING ALGORITHM

Mounir Bhouri

Institut Supérieur d'Informatique et de Mathématiques
email: mounir.bhouri@ieee.org

ABSTRACT

In this paper we derive a new adaptive filtering algorithm. Starting from a general square-root formulation [1], we introduce a normalization transform to the updating scheme of the block-diagonal adaptive algorithm presented in [1]. This algorithm is efficiently implemented with a low complexity, the resulting algorithm is similar to the NLMS one. Simulations, in the context of multichannel adaptive filtering with highly intercorrelated channels, show a fast convergence of our new algorithm.

1. INTRODUCTION

Adaptive filtering has received a considerable attention during last decades because of its application in many fields such as system identification, echo cancellation and equalization.

One of the best known adaptive filter is perhaps the gradient type one using the Least Mean Square (LMS) algorithm, it is extremely robust and simple to implement but it has limited performance. The Recursive Least Squares (RLS) adaptive algorithm is the other well known algorithm, it is more complex but it can yield a very fast convergence.

The square-root form of recursive least squares is based on elementary (rotation) transforms, so it has a better numerical robustness than the standard RLS [5]. Recently, we have generalized this square-root approach (QR-RLS) to other non least squares algorithms [1]. We have also derived other square-root type adaptive algorithms with good performances (in [2] and [3]), but they remain more complex than very low complexity algorithms such as LMS and Normalized LMS (NLMS).

These rotations based algorithms are a very interesting alternative to standard form recursive least squares and the issued fast versions (e.g. FTF). In fact, they combine an inherent numerical stability [5] with the existence of fast versions of these algorithms [4].

However, due to the high order of the filter encountered in some applications (e.g. acoustic echo cancellation), it is important to have low complexity algorithms.

In the following, we will derive a new square-root adaptive filtering algorithm with a first fast (efficient) implementation. Then we will derive another fast algorithm which looks like the NLMS. This later will be simulated for system identification application.

2. DERIVATION OF THE SQUARE-ROOT FORM

We consider the general framework of an adaptive with the general input vector of size N, which includes both monochannel and multichannel adaptive filtering.

The input vector,

$$\mathbf{x}(n) = \begin{pmatrix} x_1(n) & x_2(n) & \cdots & x_N(n) \end{pmatrix}^T$$

The output error is given by

$$e(n) = d(n) - \mathbf{x}^T(n).\mathbf{w}(n)$$

where $d(n)$ is the signal reference.

and $\mathbf{w}(n) = \begin{pmatrix} w_1(n) & w_2(n) & \cdots & w_N(n) \end{pmatrix}^T$ is the parameter vector.

Our new square-root adaptive algorithm will be formulated by an update scheme similar to the Block-diagonal QR (BQR) adaptive algorithm which fits within the general square-root framework [1].

We start from the BQR adaptive algorithm with N unitary-sized blocks

Its update scheme is given by

$$\mathbf{Q}(n).\begin{bmatrix} \mathbf{x}^T(n) & d(n) \\ \lambda\mathbf{R}'(n-1) & \lambda\mathbf{d}^{w\prime}(n-1) \end{bmatrix} = \begin{bmatrix} \mathbf{0}^T & d^e(n) \\ \mathbf{R}(n) & \mathbf{d}^w(n) \end{bmatrix}$$

where the diagonal matrix

$$\mathbf{R}'(n-1) = \mathrm{diag}(r_1(n-1), r_2(n-1), \ldots, r_N(n-1))$$

and $\mathbf{R}(n)$ is an upper triangular matrix decomposed into: a diagonal part ($\mathbf{R}'(n)$) and an off-diagonal part ($\mathbf{B}(n)$).

$$\mathbf{R}(n) = \begin{bmatrix} r_1(n) & \times & \cdots & \times \\ & r_2(n) & \ddots & \vdots \\ & & \ddots & \times \\ \bigcirc & & & r_N(n) \end{bmatrix}$$
$$= \mathbf{R}'(n) + \mathbf{B}(n)$$

and the vector

$$\mathbf{d}^{w\prime}(n-1) = \mathbf{d}^w(n-1) - \mathbf{B}(n-1).\mathbf{w}(n-1)$$

The parameter vector is obtained by solving

$$\mathbf{R}(n).\mathbf{w}(n) = \mathbf{d}^w(n)$$

The orthogonal transform $\mathbf{Q}(n)$ is $(N+1)\times(N+1)$. It can be decomposed into N elementary Givens transform (rotations)

$$\mathbf{Q}(n) = \mathbf{Q}_N(n).\mathbf{Q}_{N-1}(n)\ldots\mathbf{Q}_1(n)$$

where

$$\mathbf{Q}_k(n) = \begin{bmatrix} c_k(n) & 0 & \cdots & 0 & -s_k(n) & & \bigcirc \\ 0 & 1 & & & 0 & & \\ \vdots & & \ddots & & \vdots & & \\ 0 & & & 1 & 0 & & \\ s_k(n) & 0 & \cdots & 0 & c_k(n) & & \\ & & & & & \ddots & \\ \bigcirc & & & & \vdots & & 1 \end{bmatrix}$$
$$(k+1)^{th}$$

$$\begin{aligned} c_k(n) &= \cos\theta_k(n) \\ s_k(n) &= \sin\theta_k(n) \end{aligned}$$

Starting from this algorithm, we will propose a new algorithm. It is based on an N steps ($k = 0, \ldots, N$) update of $\mathbf{R}'(n-1)$ and $\mathbf{d}^{w\prime}(n-1)$ using at each step a normalization transform.

In the rest of this section we will detail the formulation for this new algorithm.

The decomposition of the BQR update scheme into N steps,

for $k = 1, .., N$

$$\mathbf{Q}_k(n).\begin{bmatrix} \mathbf{x}'^T_{k-1}(n) & d'_{k-1}(n) \\ \mathbf{R}_{k-1}(n) & \mathbf{d}^w_{k-1}(n) \end{bmatrix} = \begin{bmatrix} \mathbf{x}'^T_k(n) & d'_k(n) \\ \mathbf{R}_k(n) & \mathbf{d}^w_k(n) \end{bmatrix}$$

where $\mathbf{R}_k(n)$ is equal to

$$\begin{bmatrix} r_1(n) & \times & & & \cdots & & \times \\ & \ddots & & & & & \\ & & r_k(n) & \times & & \cdots & \times \\ & & & \lambda r_{k+1}(n-1) & & & \bigcirc \\ & & & & & \ddots & \\ \bigcirc & & & & & & \lambda r_N(n-1) \end{bmatrix}$$

with

$$\begin{aligned} \mathbf{R}_0(n) &= \lambda.\mathbf{R}'(n-1) \\ \mathbf{d}^w_0(n) &= \lambda.\mathbf{d}^{w\prime}(n-1) \\ \mathbf{R}_N(n) &= \mathbf{R}(n) \\ \mathbf{d}^w_N(n) &= \mathbf{d}^w(n) \end{aligned}$$

and

$$\mathbf{x}'_k(n) = \begin{pmatrix} 0 & \cdots & 0 & x'_{k,k}(n) & \cdots & x'_{N,k}(n) \end{pmatrix}^T$$

with

$$\begin{aligned} \mathbf{x}'_0(n) &= \mathbf{x}(n) \\ d'_0(n) &= d(n) \\ \mathbf{x}'_N(n) &= \mathbf{0} \\ d'_N(n) &= d^e(n) \end{aligned}$$

Every elementary transform $\mathbf{Q}_k(n)$ annihilate one component of the first line-vector $\mathbf{x}'^T_{k-1}(n)$.

It can be easily showed that

$$\mathbf{x}'_k(n) = \gamma_k(n).\mathbf{x}_{k+1,N}(n)$$

where the N-vector,

$$\mathbf{x}_{k+1,N}(n) = \begin{pmatrix} 0 & \cdots & 0 & x_{k+1}(n) & \cdots & x_N(n) \end{pmatrix}^T$$

and

$$\gamma_k(n) = \gamma_{k-1}(n).c_k(n) \quad , \qquad \gamma_0(n) = 1$$

It follows that $\mathbf{Q}_k(n)$ (or $\theta_k(n)$) depends on $\mathbf{Q}_{k-1}(n)$ (or $\theta_{k-1}(n)$).

In order to decouple these matrix, we introduce the normalization transform for $\mathbf{x}'^T_{k-1}(n)$.

Thus, the resulting new algorithm is updated by

for $k = 1, .., N$

$$\mathbf{Q}_k(n).\begin{bmatrix} \mathbf{x}^T_{k,N}(n) & \frac{d'_{k-1}(n)}{c_{k-1}(n)} \\ \mathbf{R}_{k-1}(n) & \mathbf{d}^w_{k-1}(n) \end{bmatrix} = \begin{bmatrix} \mathbf{x}'^T_k(n) & d'_k(n) \\ \mathbf{R}_k(n) & \mathbf{d}^w_k(n) \end{bmatrix} \tag{1}$$

with $d'_0(n) = d(n)$ and $c_0(n) = 1$.

Similarly to [4][3], we expect that our algorithm can be efficiently implemented with a low complexity. However, due to the normalization transform, we reach a more important reduction in complexity than in the other fast QR based adaptive algorithms.

3. EFFICIENT IMPLEMENTATION

In the case of the Fast QR algorithm [4], and since we do not need to explicitly compute the parameter vector, the output error can be efficiently computed with an $O(N)$ complexity.

Similarly, we will derive a new effecient implementation of our previous algorithm (see equation 1)

First, let's define an $(N+1)$ pinning vector

$$p = \begin{pmatrix} 1 & 0 & \cdots & 0 \end{pmatrix}^T$$

Then

$$\mathbf{Q}_k(n).p = \begin{pmatrix} c_k(n) & 0 & \cdots & 0 & s_k(n) & 0 & \cdots & 0 \end{pmatrix}^T \tag{2}$$

If we multiply (1) by the vector $\begin{pmatrix} -\mathbf{w}^T(n) & 1 \end{pmatrix}^T$

$$\begin{aligned} \mathbf{Q}_k(n).&\begin{pmatrix} \frac{d'_{k-1}(n)}{c_{k-1}(n)} - \mathbf{x}^T_{k,N}(n).\mathbf{w}(n) \\ \mathbf{d}^w_{k-1}(n) - \mathbf{R}_{k-1}(n).\mathbf{w}(n) \end{pmatrix} \\ &= \begin{pmatrix} d'_k(n) - \mathbf{x}'^T_k(n).\mathbf{w}(n) \\ \mathbf{d}^w_k(n) - \mathbf{R}_k(n).\mathbf{w}(n) \end{pmatrix} \end{aligned} \tag{3}$$

The scalar product of (2) by (3) gives

$$\frac{d'_{k-1}(n)}{c_{k-1}(n)} - \mathbf{x}^T_{k,N}(n)\mathbf{w}(n) = c_k(n)\left(d'_k(n) - \mathbf{x}'^T_k(n)\mathbf{w}(n)\right) \tag{4}$$

since the k^{th} component of the vector $\mathbf{d}^w_k(n) - \mathbf{R}_k(n)\mathbf{w}(n)$ is equal to zero.

From (3), we verify that $\mathbf{x}'_k(n) = c_k(n).\mathbf{x}_{k+1,N}(n)$

And, by induction, we show that

$$\frac{d'_k(n)}{c_k(n)} = d(n) - \sum_{i=1}^{k} x_i(n).w_i(n-1) \tag{5}$$

Replacing (5) in (4) leads to

$$e_{k-1}(n) = c_k^2(n) . e_k(n) \quad (6)$$

where $e_k(n)$ is the output error computed using the parameter vector $\mathbf{w}_k(n)$,

$$\mathbf{w}_k(n) = \begin{pmatrix} w_1(n-1) \\ \vdots \\ w_k(n-1) \\ w_{k+1}(n) \\ \vdots \\ w_N(n) \end{pmatrix}$$

Then $e_k(n) = d(n) - \mathbf{x}^T(n)\mathbf{w}_k(n)$

Which can be formulated recursively by

$$e_k(n) = e_{k-1}(n) + (w_k(n) - w_k(n-1)) x_k(n) \quad (7)$$

where

$e_N(n)$ denotes the prior output error $e_{prior}(n)$

$e_0(n)$ denotes the posterior output error $e(n)$

Combining (6) and (7),

$$w_k(n) = w_k(n-1) + \frac{s_k^2(n)}{x_k(n)} e_k(n)$$

the value of $s_k(n)$ is then replaced,

$$w_k(n) = w_k(n-1) + \frac{x_k(n)}{r_k^2(n)} e_k(n)$$

where $r_k^2(n) = x_k^2(n) + \lambda^2 . r_k^2(n-1)$

This relation is used as an update equation when k varies from N downto 1.

The resulting new efficient algorithm: ALGO 1,

```
for k = 1,..,N
   σ_k(−1) = 0
end k

for n = 0,..,L
   e_N(n) = d(n) − Σ_{i=1}^{N} x_i(n).w_i(n−1)
   for k = N,..,1
      σ_k(n) = x_k^2(n) + λ^2.σ_k(n−1)
      w_k(n) = w_k(n−1) + (x_k(n)/σ_k(n)) e_k(n)
      e_{k−1}(n) = e_k(n) − (w_k(n) − w_k(n−1)) x_k(n)
   end k
   e(n) = e_0(n)
end n
```

This algorithm has already a low complexity. However, in the case of monochannel adaptive filtering, we will further reduce the complexity by exploiting the shift structure of the input vector.

4. THE NOVEL NLMS-LIKE ADAPTIVE FILTER

Now, we consider the case of monochannel adaptive filter. The result is a simple algorithm. The generalization to multichannel case is quite direct.

In order to get a simple set of update equations, we use the following new definitions for the vectors $\mathbf{x}(n)$ and $\mathbf{w}(n)$

$$\mathbf{x}(n) = \begin{pmatrix} x(n-N+1) & x(n-N+2) & \cdots & x(n) \end{pmatrix}^T$$
$$\mathbf{w}(n) = \begin{pmatrix} w_N(n) & w_{N-1}(n) & \cdots & w_1(n) \end{pmatrix}^T$$

This means that ALGO 1 must be reformulated :

- $w_k(n)$ is replaced by $w_{N+1-k}(n)$
- $e_k(n)$ is replaced by $e_{N+1-k}(n)$

Then we approximate $\sigma_k(n)$ by $\sigma(n+k-N)$ where $\sigma(n) = x^2(n) + \lambda^2 . \sigma(n-1)$.

Finally, we obtain our novel NLMS-like adaptive filtering algorithm: ALGO 2,

```
σ(−1) = 0

for n = 0,..,L
   e_1(n) = d(n) − Σ_{i=1}^{N} x(n+1−i).w_i(n−1)
   σ(n) = x^2(n) + λ^2.σ(n−1)
   α(n) = x(n)/σ(n)
   for k = 1,..,N
      δ_k(n) = α(n+1−k).e_k(n)
      w_k(n) = w_k(n−1) + δ_k(n)
      e_{k+1}(n) = e_k(n) − δ_k(n).x(n+1−k)
   end k
   e(n) = e_{N+1}(n)
end n
```

The complexity of ALGO 2 is equal to $3N+1$ additions, $3N+2$ multiplications and 1 division. It is close to the NLMS complexity.

5. SIMULATIONS

Montecarlo simulations are done over 100 trials.

In next figure, we consider a noisy identification scheme in the case of multichannel adaptive filter with a forgetting factor $\lambda = 0.99$, an $SNR = 20$dB, and two channels with highly intercorrelated speech input signals.

Such scheme is encountered in stereophonic acoustic echo cancellation.

The filter size used here is $N = 32$.

6. CONCLUSION

The generalized square-root approach of adaptive filtering [1] have permitted to us the derivation of new adaptive algorithms computationally efficient.

The last derived one (ALGO 2) has a very low complexity like the NLMS. It has also showed good performances in the case of stéréophonic acoustic echo cancellation.

REFERENCES

[1] M. Bhouri. Generalized square-root adaptive algorithms. In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, June 2000.

[2] M. Bhouri, M. Bonnet, and M. Mboup. A new QRD-based block adaptive algorithm. In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 1497–1500, May 1998.

[3] M. Bhouri, M. Mboup, and M. Bonnet. A fast multi-channel adaptive algorithm. In *Proceeding of European Signal Processing Conference*, September 1998.

[4] J.M. Cioffi. A fast QR frequency-domain RLS adaptive filter. In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 407–410, April 1987.

[5] B. Yang and J.F. Böhme. Rotation-based RLS algorithms: Unified derivations, numerical properties and parallel implementations. *IEEE Trans. SP*, 40:1151–1167, 1992.

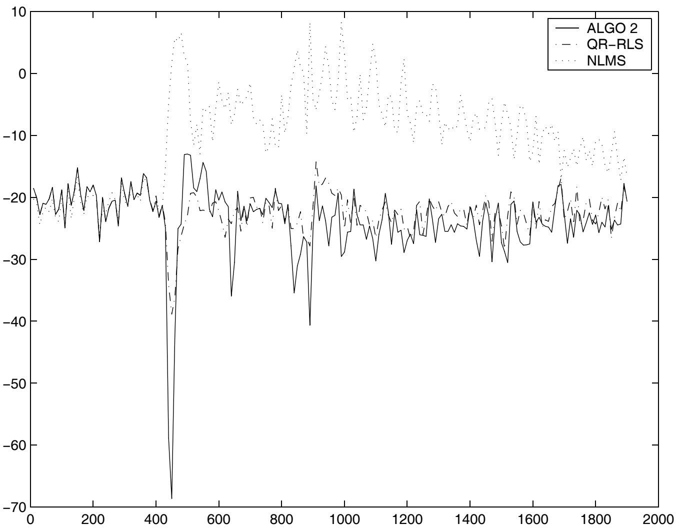

Figure 1: MSE vs. time for ALGO 2, QR-RLS and NLMS (with speech input signal)