SUB-RLS ALGORITHM WITH AN EXTREMELY SIMPLE UPDATE EQUATION

Kensaku Fujii and Juro Ohga

Network Systems Laboratories (L40), FUJITSU LABORATORIES LTD.
4-1-1 Kamikodanaka, Nakahara-ku, Kawasaki, 211-88, Japan
fujiken@flab.fujitsu.co.jp

ABSTRACT

A new type of adaptive algorithm is derived from a first order infinite impulse response (IIR) filter expression of the normalized least mean square (NLMS) algorithm. This new algorithm provides a convergence property similar to that of the recursive least square (RLS) algorithm. Its update equation, however, is extremely simple compared to that of the RLS algorithm. The new algorithm, named 'sub-RLS' algorithm in this paper, can be also derived from the least square (LS) algorithm on an approximation. The prefix 'sub' designates the approximation applied to the LS algorithm for its recursive adaptation. This paper also presents a variation of reducing its processing cost.

1. INTRODUCTION

This paper introduces a new recursive type of adaptive algorithm to estimate the coefficients of finite impulse response (FIR) filter under the least square criterion. This new algorithm is based on a different principle than the 'fast' RLS algorithm [1]-[3], and is formulated by an extremely simple update equation compared to that of the conventional RLS algorithm. However, its convergence property is similar to that of the RLS algorithm, and moreover, its estimation process is performed with numerical stability even in single precision calculation. The new algorithm is derived from a first order IIR filter expression [4] of the NLMS algorithm [5].

The new algorithm, named "sub-RLS" algorithm in this paper, can be also derived from the LS algorithm on the approximation of substituting the coefficients obtained at the last adaptation for those to be evaluated at the current adaptation. The approximation yields a matrix with the null main diagonal, which consists of the auto-correlation coefficients of the reference signal. The null main diagonal forms an incomplete recursive operation not to require calculating the inverse of the auto-correlation matrix for the estimation. Thereby, the operation yields the sub-RLS algorithm with an extremely simple update equation in comparison to the RLS algorithm. The prefix 'sub' designates the incomplete recursive operation applied to the LS algorithm.

This paper finally presents a variation of reducing the processing cost in the sub-RLS algorithm. The sub-RLS algorithm has $I \times I$ divisions to normalize the elements of the auto-correlation matrix, each of which requires more computational cycles than multiplication in digital signal processor (DSP). The division can be removed from the sub-RLS algorithm when the forgetting factor is applied to the estimation process. The forgetting factor keeps the normalization power almost constant, therefore, the divisions can be replaced with the multiplication by the reciprocal of its maximum power. The low processing cost version of the sub-RLS algorithm with no division seems to be similar to the LMS algorithm.

2. A NEW ADAPTIVE ALGORITHM

The new adaptive algorithm is derived from the IIR filter expression [4] shown in Fig. 1. The expression suggests that the m'th coefficient of the adaptive FIR filter with I taps, $H_j(m)$, is estimated as the step response of the low pass filter with the recursive coefficient,

$$\alpha_j(m) = 1 - \mu X_j^2(m) \Big/ \sum_{i=1}^{I} X_j^2(i), \tag{1}$$

where j is sample time index, μ is a step gain,

$$X_j = \left[X_j(1) \quad X_j(2) \quad \cdots \quad X_j(I) \right]^{\mathrm{T}} \tag{2}$$

is the reference signal, and the impulse response of unknown system,

$$h = \left[h(1) \quad h(2) \quad \cdots \quad h(I) \right]^{\mathrm{T}}, \tag{3}$$

is supposed to be stationary.

The expression also reveals that the low convergence rate in the NLMS algorithm is caused by the disturbance,

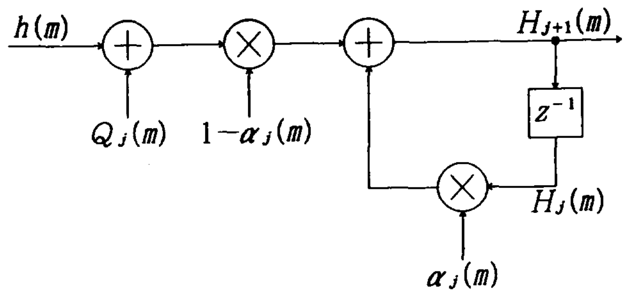

Fig.1 First order IIR filter expression.

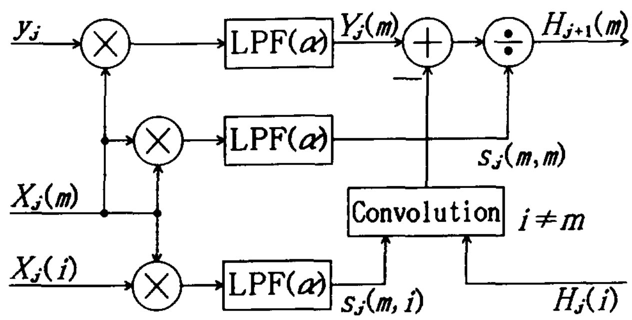

Fig. 2 New adaptive algorithm derived from the IIR filter expression.

$$Q_j(m)=\left[\sum_{i=1}^{I}{}_{i\neq m}\left\{h(i)-H_j(i)\right\}X_j(i)+N_j\right]\Big/X_j(m), \quad (4)$$

remaining in the recursive path. A method of removing the disturbance from the recursive path is to transfer $\sum_{i=1}^{I}{}_{i\neq m}H_j(i)X_j(i)$, which is the components of $Q_j(m)$, to behind that. The substitution of the average,

$$\alpha=\sum_{m=1}^{I}\alpha_j(m)\Big/I=1-\mu/I, \quad (5)$$

for $\alpha_j(m)$ makes the transfer possible. The low path filter input, then, is replaced by

$$y_j=\boldsymbol{h}_j^{\mathrm{T}}\boldsymbol{X}_j+N_j, \quad (6)$$

which is the response of the unknown system including the additive noise, N_j.

The transfer finally yields a new adaptive algorithm [6] whose estimation process is shown in Fig. 2. Here, LPF(α) is the low pass filter with the recursive coefficient α given in (5), and each of those low pass filters calculates,

$$Y_j(m)=\alpha Y_{j-1}(m)+y_jX_j(m) \quad (7)$$

$$s_j(m,m)=\alpha s_{j-1}(m,m)+X_j^2(m) \quad (8)$$

$$s_j(m,i)=\alpha s_{j-1}(m,i)+X_j(m)X_j(i), \quad (9)$$

respectively. In this algorithm, the m'th coefficient $H_j(m)$ is updated as

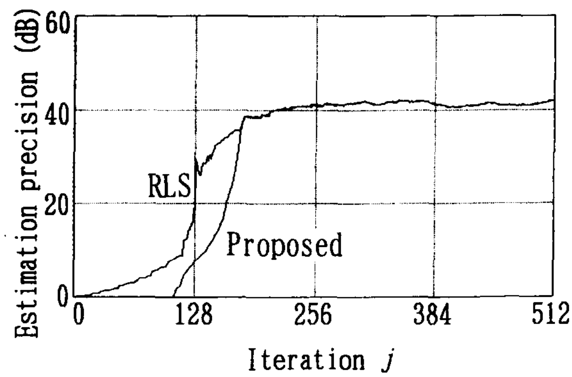

Fig.3 Convergence property of the RLS and the proposed algorithms

$$H_{j+1}(m)=\frac{Y_j(m)-\sum_{i=1}^{I}{}_{i\neq m}H_j(i)s_j(m,i)}{s_j(m,m)}. \quad (10)$$

Actually, this adaptation is implemented by turns from $m=1$ to I, therefore, the set of coefficients used for each adaptation is replaced as follows:

$\{0, H_j(2), H_j(3), \cdots, H_j(I)\}$ for $H_j(1)$,

$\{H_{j+1}(1), 0, H_j(3), \cdots, H_j(I)\}$ for $H_j(2)$, and

$\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots$

$\{H_{j+1}(1), H_{j+1}(2), \cdots, H_{j+1}(I-1), 0\}$ for $H_j(I)$,

after every adaptation. The replacement saves the memories to store those coefficients

3. COMPARISON TO THE RLS ALGORITHM

Figure 3 is an example to confirms that the new adaptive algorithm provides almost the same convergence time as the RLS algorithm, where the coefficient $\alpha=127/128$, which is evaluated for $\mu=1$ and $I=128$, is also used as the forgetting factor for the RLS algorithm. The coefficient α seems to define the forgetting factor.

Equation (10) can be also rewritten as

$$H_{j+1}(m)=Y_j(m)/s_j(m,m)-\sum_{i=1}^{I}{}_{i\neq m}H_j(i)r_j(m,i) \quad (11)$$

which the subtraction, shown in Fig. 2, is transferred to behind the division, where

$$r_j(m,i)=s_j(m,i)/s_j(m,m). \quad (12)$$

Equation (11) is equal to

$$\boldsymbol{H}_{j+1}=\boldsymbol{S}_j\boldsymbol{Y}_j-\boldsymbol{R}_j\boldsymbol{H}_j \quad (13)$$

expressed by the auto-correlation matrix,

$$\boldsymbol{R}_j=\begin{bmatrix}0 & r_j(1,2) & \cdots & r_j(1,I\\ r_j(2,1) & 0 & \cdots & r_j(2,I)\\ \vdots & \vdots & \ddots & \vdots\\ r_j(I,1) & r_j(I,2) & \cdots & 0\end{bmatrix}, \quad (14)$$

the normalization power matrix,

$$S_j = \begin{bmatrix} 1/s_j(1,1) & 0 & \cdots & 0 \\ 0 & 1/s_j(2,2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1/s_j(I,I) \end{bmatrix}, \quad (15)$$

and the cross-correlation vector between the response y_j and the reference signal X_j,

$$Y_j = \begin{bmatrix} Y_j(1) & Y_j(2) & \cdots & Y_j(I) \end{bmatrix}^{\mathrm{T}}. \quad (16)$$

Apparently, this update equation is extremely simple compared to that of the RLS algorithm:

$$H_{j+1} = H_j + E_j P_j X_j / \left(\alpha + X_j^{\mathrm{T}} P_j X_j\right) \quad (17)$$

$$P_{j+1} = \alpha^{-1}\left\{P_j - P_j X_j X_j^{\mathrm{T}} P_j / \left(\alpha + X_j^{\mathrm{T}} P_j X_j\right)\right\}. \quad (18)$$

4. DERIVATION FROM THE LS ALGORITHM

The adaptive algorithm can be also derived from the LS algorithm which is expressed as

$$\sum_{t=1}^{j} X_t X_t^{\mathrm{T}} H_{j+1} = \sum_{t=1}^{j} X_t y_t. \quad (19)$$

The left side of (19) is expanded to

$$\sum_{t=1}^{j} X_t X_t^{\mathrm{T}} H_{j+1} = \begin{bmatrix} s_j(1,1) & s_j(1,2) & \cdots & s_j(1,I) \\ s_j(2,1) & s_j(2,2) & \cdots & s_j(2,I) \\ \vdots & \vdots & \ddots & \vdots \\ s_j(I,1) & s_j(I,2) & \cdots & s_j(I,I) \end{bmatrix} \begin{bmatrix} H_{j+1}(1) \\ H_{j+1}(2) \\ \vdots \\ H_{j+1}(I) \end{bmatrix} \quad (20)$$

whose elements are evaluated by (8) and (9) substituted unity for the coefficient α. The substitution allows to express the right side of (19) as

$$\sum_{t=1}^{j} X_t y_t = Y_j. \quad (21)$$

Another operation is to pre-multiply (19) by the normalization power matrix S_j. The operation for the left side of (19) gives the following expression:

$$\begin{bmatrix} 1 & r_j(1,2) & \cdots & r_j(1,I) \\ r_j(2,1) & 1 & \cdots & r_j(2,I) \\ \vdots & \vdots & \ddots & \vdots \\ r_j(I,1) & r_j(I,2) & \cdots & 1 \end{bmatrix} \begin{bmatrix} H_{j+1}(1) \\ H_{j+1}(2) \\ \vdots \\ H_{j+1}(I) \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} H_{j+1}(1) \\ H_{j+1}(2) \\ \vdots \\ H_{j+1}(I) \end{bmatrix}$$

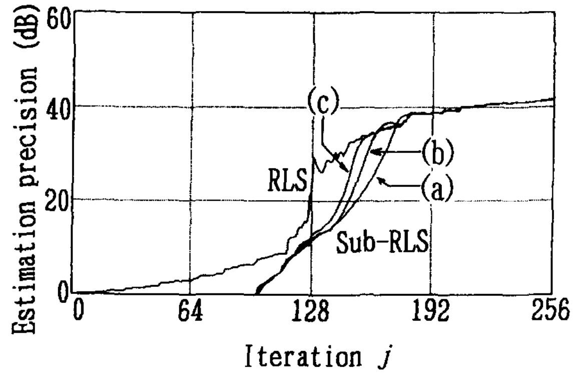

Fig. 4 Effect of repeating the adaptation: (a) no repetitions (b) four repetitions (c)eight repetitions.

$$+ \begin{bmatrix} 0 & r_j(1,2) & \cdots & r_j(1,I) \\ r_j(2,1) & 0 & \cdots & r_j(2,I) \\ \vdots & \vdots & \ddots & \vdots \\ r_j(I,1) & r_j(I,2) & \cdots & 0 \end{bmatrix} \begin{bmatrix} H_{j+1}(1) \\ H_{j+1}(2) \\ \vdots \\ H_{j+1}(I) \end{bmatrix}$$

$$= H_{j+1} + R_j H_{j+1}. \quad (22)$$

The pre-multiplication finally forms

$$H_{j+1} + R_j H_{j+1} = S_j Y_j. \quad (23)$$

As stated above, the set of coefficients used for each adaptation changes as follows:

$\{0, H_j(2), H_j(3), \cdots, H_j(I)\}$ for $H_j(1)$,

$\{H_{j+1}(1), 0, H_j(3), \cdots, H_j(I)\}$ for $H_j(2)$, and

....

$\{H_{j+1}(1), H_{j+1}(2), \cdots, H_{j+1}(I-1), 0\}$ for $H_j(I)$.

These coefficient sets gradually approach the coefficient H_{j+1} every adaptation, still, $H_{j+1} \neq H_j$. Here, the substitution of H_j for H_{j+1} may reduce the convergence rate, however, the approximation enables to transpose $R_j H_{j+1}$ to the right side of (23). The transposition, then, yields (13).

This new adaptive algorithm is regarded as a recursive version of the LS algorithm, although its recursive calculation is approximative. Accordingly, the adaptive algorithm is called "sub-RLS" algorithm in this paper. The prefix "sub" designates the approximation. Apparently, the degree of approximation in the coefficient set,

$\{0, H_{j+1}(2), H_{j+1}(3), \cdots, H_{j+1}(I)\}$,

which is obtained after the last adaptation, is higher than that of the former set,

$\{0, H_j(2), H_j(3), \cdots, H_j(I)\}$.

The repetition of the adaptation is expected to gradually enhance the degree. Figure 4 is an example to confirm the repetition effect.

5. PROCESSING COST REDUCTION

The sub-RLS algorithm has $I \times I$ divisions for the normalization, each of which requires many computational cycles. The divisions can be removed when $\alpha < 1$.

Here, equation (13) is rewritten as

$$H_{j+1} = H_j + \left(S_j Y_j - A_j H_j\right) \tag{24}$$

where

$$A_j = I + R_j$$
$$= \begin{bmatrix} 1 & r_j(1,2) & \cdots & r_j(1,I) \\ r_j(2,1) & 1 & \cdots & r_j(2,I) \\ \vdots & \vdots & \ddots & \vdots \\ r_j(I,1) & r_j(I,2) & \cdots & 1 \end{bmatrix}. \tag{25}$$

The second term of (24), then,

$$S_j Y_j - A_j H_j$$
$$= \begin{bmatrix} 1 & r_j(1,2) & \cdots & r_j(1,I) \\ r_j(2,1) & 1 & \cdots & r_j(2,I) \\ \vdots & \vdots & \ddots & \vdots \\ r_j(I,1) & r_j(I,2) & \cdots & 1 \end{bmatrix} \begin{bmatrix} \Delta_j(1) \\ \Delta_j(2) \\ \vdots \\ \Delta_j(I) \end{bmatrix}, \tag{26}$$

yields the estimation error,

$$\Delta_j = h - H_j, \tag{27}$$

on the average when the noise N_j is negligible. This means that the coefficient is correctly adjusted when the main diagonal of A_j is normalized to unity. However, the coefficient can be also adjusted by the estimation error Δ_j multiplied by

$$\rho \le 1, \tag{28}$$

although the convergence rate may be reduced. This multiplication is substituted by the normalization using larger power than $s_j(m,m)$.

The forgetting factor α keeps the normalization power nearly constant, whose maximum power s_{max} can be evaluated beforehand when the reference signal is stationary. Thus the normalization is also substituted by,

$$\lambda = 1/s_{max}. \tag{29}$$

Introduction of the constant, λ, similar to the step gain of the LMS algorithm, yields an update equation without the division,

$$H_{j+1} = H_j + \lambda\left(Y_j - B_j H_j\right), \tag{30}$$

where,

$$B_j = \begin{bmatrix} s_j(1,1) & s_j(1,2) & \cdots & s_j(1,I) \\ s_j(2,1) & s_j(2,2) & \cdots & s_j(2,I) \\ \vdots & \vdots & \ddots & \vdots \\ s_j(I,1) & s_j(I,2) & \cdots & s_j(I,I) \end{bmatrix}. \tag{31}$$

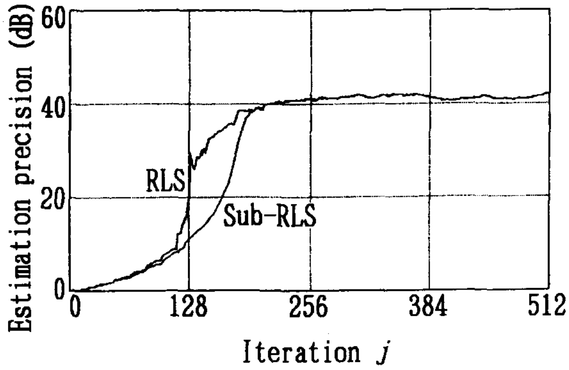

Fig. 5 Convergence property of the sub-RLS algorithm with no divisions.

Figure 5 is a convergence property of the sub-RLS algorithm with $\lambda = 13.0$.

6. CONCLUSION

In this paper, we have presented a new recursive type of adaptive algorithm whose convergence time is nearly equal to that of the RLS algorithm, although its update equation is extremely simple. The new adaptive algorithm, named sub-RLS algorithm, can be also derived from the LS algorithm. The prefix "sub" designates an approximated recursive adaptation. We have also introduced a low processing cost version without the divisions for the normalization.

REFERENCES

[1] G. Carayannis, D. Manolakis and N. Kalouptsidis, "A fast sequential algorithm for least-squares filtering and prediction," IEEE Trans., vol. ASSP-31, no.6, pp.1394-1402, 1983.

[2] D. T. M. Slock and T. Kailath, "Numerically stable fast transversal filters for recursive least squares adaptive filtering," IEEE Trans., vol. ASSP-39, no. 1, pp.92-114, 1991.

[3] P. A. Regalia, "Numerical stability property of a QR-based fast least squares algorithm," IEEE Trans. Vol. SP-41, no. 6, pp.2096-2109, 1993.

[4] K. Fujii, Y. Sakai, H. Masuda and J. Ohga, "A new expression and stability condition of learning identification algorithm," IEICE Trans., vol. J73-A, no. 11, pp. 1769-1775, 1990. [in Japanese]

[5] J. Nagmo and A. Noda, "A learning method for system identification," IEEE Trans., vol.AC-12, no. 3, pp. 282-287, 1967.

[6] K. Fujii and J. Ohga, "A fast adaptive algorithm suitable for acoustic echo canceller," IEICE Trans., vol. E75-A, no. 11, pp.1509-1515, 1992.