LETTER *Special Section on Digital Signal Processing*

A Stable Least Square Algorithm Based on Predictors and Its Application to Fast Newton Transversal Filters

Youhua WANG† ***and*** **Kenji NAKAYAMA**†, ***Members***

SUMMARY In this letter, we introduce a predictor based least square (PLS) algorithm. By involving both order- and time-update recursions, the PLS algorithm is found to have a more stable performance compared with the stable version (Version II) of the RLS algorithm shown in Ref. [1]. Nevertheless, the computational requirement is about 50% of that of the RLS algorithm. As an application, the PLS algorithm can be applied to the fast newton transversal filters (FNTF) [2]. The FNTF algorithms suffer from the numerical instability problem if the quantities used for extending the gain vector are computed by using the fast RLS algorithms. By combing the PLS and the FNTF algorithms, we obtain a much more stable performance and a simple algorithm formulation.

key words: *adaptive filters, RLS algorithm, fast RLS algorithm, fast newton transversal filters, stability*

1. Introduction

The fast RLS algorithms, which use the relation between the forward and the backward predictions for computing the gain vector, combine four transversal filters implemented in a parallel form [1]. The four filters include two predictors, one filter for a gain vector and the other filter for a tap-weight vector of the adaptive filter. Each filter has the same order that equals to the adaptive filter length N. The method proposed in Ref. [2] shows, however, that if the input signal can be adequately modeled by an Mth-order autoregressive model, denoted $AR(M)$, and M is possible to be selected much smaller than N, then the gain vector can be extended from M to N based on the Mth-order predictors without sacrificing the performance. According to this method, the white noise can be considered as an $AR(0)$ sequence, and the speech signal can be modeled from $AR(15)$ to $AR(20)$. Therefore, computational saving is significant in some applications, in which N is usually much greater than M. An echo canceler is included in this category.

Like any other fast version of the RLS algorithm, the FNTF algorithms also suffer from the numerical instability problem, if the predictor used for extending the gain vector is computed by using the fast RLS algorithm. In order to overcome this problem, it was suggested to use the RLS algorithm for the predictor part [2]. However, the extension of the gain vector needs not only the predictor but also other quantities like prediction error, conversion factor, etc. Therefore, additional computation is necessary and the formulation of the algorithm becomes quite complicated.

In order to solve these problems, in this letter, we introduce a predictor based least square (PLS) algorithm. Unlike the fast RLS algorithm whose recursion is based on a fixed order, the PLS algorithm involves both order- and time-update recursions. So only one predictor, forward or backward, is needed. Although the computational complexity of the PLS algorithm increases with M^2, the numerical property is greatly improved compared with the fast RLS algorithm. Simulation shows that the PLS algorithm has a more stable performance than the stable version (Version II) of the RLS algorithm shown in Ref. [1]. Nevertheless, the computational requirement is about 50% of that of the RLS algorithm. We notice that there is no reported investigations concerning the PLS algorithm.

As an application, the PLS algorithm is suited for applying to the FNTF algorithms, in which the PLS algorithm can be used for the prediction part and the FNTF algorithms used for the extending part. The combined algorithms are shown to have a much more stable performance and a simple algorithm formulation.

2. Predictor Based Least Square Algorithm

The PLS algorithm can be derived from any version of the fast RLS algorithm. Since only one predictor, forward or backward, is needed, we can write two versions of the PLS algorithm as follows

Condition: For both versions, when time $n = 1, 2, 3, \ldots$ compute the order updates in the following sequence: $m = 1, 2, \ldots M$, where M is the final order of the predictor.

Algorithm 1. PLS Algorithm Using Forward Predictor (FPLS)

$$\eta_m(n) = u(n) + \mathbf{a}_m^T(n-1)\mathbf{u}_m(n-1) \quad (1)$$

$$F_m(n) = \lambda F_m(n-1) + \gamma_m(n-1)\eta_m^2(n) \quad (2)$$

$$\gamma_{m+1}(n) = \frac{\lambda F_m(n-1)}{F_m(n)}\gamma_m(n-1) \quad (3)$$

Manuscript received January 5, 1995.
Manuscript revised March 8, 1995.

†The authors are with the Graduate School of Natural Sci. & Tech., Kanazawa University, Kanazawa-shi, 920 Japan.

$$\tilde{\mathbf{k}}_{m+1}(n) = \begin{bmatrix} 0 \\ \tilde{\mathbf{k}}_m(n-1) \end{bmatrix} + \frac{\eta_m(n)}{\lambda F_m(n-1)} \begin{bmatrix} 1 \\ \mathbf{a}_m(n-1) \end{bmatrix} \quad (4)$$

$$\mathbf{a}_m(n) = \mathbf{a}_m(n-1) - \gamma_m(n-1)\eta_m(n)\tilde{\mathbf{k}}_m(n-1) \quad (5)$$

where $\eta_m(n)$ is the *forward a priori prediction error*, $F_m(n)$ is the *minimum power of the forward prediction error*, $\gamma_m(n)$ is the *conversion factor*, $\tilde{\mathbf{k}}_m(n)$ is the *normalized gain vector*, $\mathbf{a}_m(n)$ is the *tap-weight vector of the forward predictor*, $\mathbf{u}_m(n)$ is the *tap input vector*.

Algorithm 2. PLS Algorithm Using Only Backward Predictor (BPLS)

$$\psi_m(n) = \mathbf{c}_m^T(n-1)\mathbf{u}_m(n) + u(n-m) \quad (6)$$

$$B_m(n) = \lambda B_m(n-1) + \gamma_m(n)\psi_m^2(n) \quad (7)$$

$$\gamma_{m+1}(n) = \frac{\lambda B_m(n-1)}{B_m(n)}\gamma_m(n) \quad (8)$$

$$\tilde{\mathbf{k}}_{m+1}(n) = \begin{bmatrix} \tilde{\mathbf{k}}_m(n) \\ 0 \end{bmatrix} + \frac{\psi_m(n)}{\lambda B_m(n-1)} \begin{bmatrix} \mathbf{c}_m(n-1) \\ 1 \end{bmatrix} \quad (9)$$

$$\mathbf{c}_m(n) = \mathbf{c}_m(n-1) - \gamma_m(n)\psi_m(n)\tilde{\mathbf{k}}_m(n) \quad (10)$$

where $\psi_m(n)$ is the *backward a priori prediction error*, $B_m(n)$ is the *minimum power of the backward prediction error*, $\mathbf{c}_m(n)$ is the *tap-weight vector of the backward predictor*.

The filtering part is common for both the FPLS and the BPLS algorithms.

$$\alpha_M(n) = d(n) - \hat{\mathbf{w}}_M^T(n-1)\mathbf{u}_M(n) \quad (11)$$

$$\hat{\mathbf{w}}_M(n) = \hat{\mathbf{w}}_M(n-1) + \tilde{\mathbf{k}}_M(n)\gamma_M(n)\alpha_M(n) \quad (12)$$

where $\alpha_M(n)$ is the *a priori estimation error*, $d(n)$ is the *desired signal*, $\hat{\mathbf{w}}_M(n)$ is the *tap-weight vector of the adaptive filter*.

Please refer [1] for more detailed meaning of the symbols used in Eqs. (1)–(12).

To initialize the PLS algorithm at time $n = 0$, set

$$\mathbf{a}_m(0) = \mathbf{c}_m(0) = \mathbf{0}_m \quad (13)$$

$$F_m(0) = B_m(0) = \delta \quad (14)$$

$$\tilde{\mathbf{k}}_m(0) = \mathbf{0}_m \quad (15)$$

$$\gamma_m(0) = 1 \quad (16)$$

where $m = 1, 2, ..., M$. δ is a small positive constant.

At each iteration $n \geqq 1$, generate the first-order variables as follows

$$\tilde{\mathbf{k}}_1(n) = \frac{1}{\lambda}\Phi_1^{-1}(n-1)u(n) = \frac{u(n)}{\lambda\Phi_1(n-1)} \quad (17)$$

$$\gamma_1(n) = \frac{1}{1+\tilde{\mathbf{k}}_1(n)u(n)} = \frac{\lambda\Phi_1(n-1)}{\Phi_1(n)} \quad (18)$$

where $\Phi_1(n)$ is the first-order of the input correlation matrix and satisfies

$$\Phi_1(n) = \lambda\Phi_1(n-1) + u^2(n) \qquad (\Phi_1(0) = \delta) \quad (19)$$

3. Stability Analysis of PLS Algorithm

It is known that the instability of the fast RLS algorithm is mainly caused by the accumulation of the round-off errors in the recursive process [3]–[5]. In this section, we will prove that the accumulation of the round-off errors in the PLS algorithm can be greatly reduced compared with the fast RLS algorithm.

In the fast RLS algorithms, we know that the minimum powers of the forward and the backward predictor errors most probably accumulate the round-off errors. So we want to analize the error propagation properties of these powers in the PLS algorithm and compare with that of the fast RLS algorithm.

Suppose at time n and order m, some round-off errors are introduced in the forward predictor of the FPLS algorithm and make

$$\eta_m(n) = \eta_m^*(n) + \epsilon_\eta(n) \quad (20)$$

where $(\cdot)^*$ denotes the variable without numerical error. Then the minimum power of the forward prediction error indicated by Eq. (2) can be written as

$$\begin{aligned} F_m(n) &= \lambda F_m(n-1) + \gamma_m(n-1)\eta_m^2(n) \\ &= \lambda F_m(n-1) + \gamma_m(n-1)\eta_m^{*2}(n) \\ &\quad + \gamma_m(n-1)\epsilon_F(n) \\ &= F_m^*(n) + \gamma_m(n-1)\epsilon_F(n) \end{aligned} \quad (21)$$

where

$$\epsilon_F(n) = 2\eta_m(n)\epsilon_\eta(n) + \epsilon_\eta^2(n) \quad (22)$$

According to Eq. (3), we have

$$\begin{aligned} \gamma_m(n-1) &= \frac{\lambda F_{m-1}(n-2)}{F_{m-1}(n-1)}\gamma_{m-1}(n-2) \\ &= \frac{\lambda F_{m-1}(n-2)}{F_{m-1}(n-1)} \cdot \frac{\lambda F_{m-2}(n-3)}{F_{m-2}(n-2)} \\ &\quad \cdots \frac{\lambda F_1(n-m)}{F_1(n-m+1)}\gamma_1(n-m) \\ &= \prod_{i=1}^{m-1} \frac{\lambda F_{m-i}(n-i-1)}{F_{m-i}(n-i)}\gamma_1(n-m) \end{aligned} \quad (23)$$

Note that

$$\frac{\lambda F_{m-i}(n-i-1)}{F_{m-i}(n-i)} < 1, \qquad i = 1 \ldots m-1 \quad (24)$$

and the first-order of the conversion factor shown in Eq. (18) satisfies

$$\gamma_1(n-m) = \frac{\lambda\Phi_1(n-m-1)}{\lambda\Phi_1(n-m-1) + u^2(n-m)} \leqq 1 \quad (25)$$

So we get

$$\gamma_m(n-1) < \gamma_1(n-m) \leq 1 \tag{26}$$

Correspondingly, the propagation of the numerical errors is therefore

$$\gamma_m(n-1)\epsilon_F(n) < \epsilon_F(n) \tag{27}$$

This means that the round-off error $\epsilon_F(n)$ produced at n can be reduced as m and n increase.

For the same reason, we can write the minimum power of the backward prediction error indicated by Eq. (7) as

$$\begin{aligned} B_m(n) &= \lambda B_m(n-1)+\gamma_m(n)\psi_m^{*2}(n)+\gamma_m(n)\epsilon_B(n) \\ &= B_m^*(n) + \gamma_m(n)\epsilon_B(n) \end{aligned} \tag{28}$$

From Eq. (8), we have

$$\begin{aligned} \gamma_m(n) &= \frac{\lambda B_m(n-1)}{B_m(n)}\gamma_{m-1}(n) \\ &= \frac{\lambda B_m(n-1)}{B_m(n)} \cdot \frac{\lambda B_{m-1}(n-1)}{B_{m-1}(n)} \\ &\quad \cdots \frac{\lambda B_1(n-1)}{B_1(n)}\gamma_1(n) \\ &= \prod_{i=0}^{m-1} \frac{\lambda B_{m-i}(n-1)}{B_{m-i}(n)}\gamma_1(n) \end{aligned} \tag{29}$$

Note that

$$\frac{\lambda B_{m-i}(n-1)}{B_{m-i}(n)} < 1 \qquad i = 0 \ldots m-1 \tag{30}$$

and

$$\gamma_1(n) = \frac{\lambda \Phi_1(n-1)}{\lambda \Phi_1(n-1) + u^2(n)} \leq 1 \tag{31}$$

So we can readily deduce that

$$\gamma_m(n)\epsilon_B(n) < \epsilon_B(n) \tag{32}$$

In the fast RLS algorithms, a fixed order of the the minimum power of the forward prediction error is given by

$$F_M(n) = \lambda F_M(n-1) + \gamma_M(n-1)\eta_M^2(n) \tag{33}$$

Using Eqs. (3) and (8) ($m = M$), we can write

$$\begin{aligned} \gamma_M(n-1) &= \frac{\lambda F_M(n-2)}{F_M(n-1)} \cdot \frac{B_M(n-1)}{\lambda B_M(n-2)}\gamma_M(n-2) \\ &= \frac{\lambda F_M(n-2)}{F_M(n-1)} \cdot \frac{B_M(n-1)}{\lambda B_M(n-2)} \\ &\quad \cdots \frac{\lambda F_M(0)}{F_M(1)} \cdot \frac{B_M(1)}{\lambda B_M(0)}\gamma_M(0) \\ &= \prod_{i=1}^{n-1} \frac{\lambda F_M(n-i-1)}{F_M(n-i)} \\ &\quad \cdot \frac{B_M(n-i)}{\lambda B_M(n-i-1)} \end{aligned} \tag{34}$$

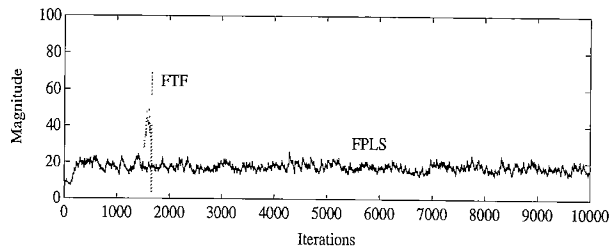

Fig. 1 Numerical behavior of PLS and FTF. Minimum power of forward prediction error $F_M(n)$ computed under the following conditions: white noise input, $M = 100$, $\lambda = 0.98$ and $\delta = 1$.

where $\gamma_M(0) = 1$ is the initial value. Since the following is not always guaranteed

$$\frac{\lambda F_M(n-i-1)}{F_M(n-i)} \cdot \frac{B_M(n-i)}{\lambda B_M(n-i-1)} < 1 \qquad i = 1 \ldots n-1 \tag{35}$$

the accumulation of the round-off errors may occur and eventually lead to the divergence of the fast RLS algorithms.

Figure 1 shows the numerical performances of the minimum power of the forward prediction error $F_M(n)$ computed by using the FPLS and the fast transversal filter (FTF) algorithms. The FTF algorithm diverges after 1500 iterations, while the FPLS algorithm is stable.

4. Comparison Between PLS and RLS Algorithms

Even though both the PLS and the RLS algorithms provide the same exact least square solution, the basis for these two algorithms are entirely different. The RLS algorithm is derived from the matrix inversion lemma. The PLS algorithm is, however, based on the relationship between the predictor and the gain vector. So it can be expected that they will exhibit different numerical properties.

In the RLS algorithm, it is known that the divergence occurs when the inverse of the input correlation matrix loses its symmetry. This source of instability can be overcome by computing the gain vector exactly as it arises in the formulation of the RLS algorithm instead of exploiting the symmetry property of the inverse of the input correlation matrix. Another source of instability is that when the forgetting factor λ reduces, the inverse of the input correlation matrix tends to be singular and may lose its positive definiteness [6]–[8].

Simulation shows, however, that these two sources of instabilities do not exist in both the FPLS and the BPLS algorithms. The reason for the numerical stability of the PLS algorithm is under further investigation. Here we want to show the stable performance through computer simulation. The simulation is done under the following conditions: $M = 100$, $\lambda = 0.8$ and $\delta = 0.01$. A white noise sequence with the power normalized to unity is used as the input. The computational precision

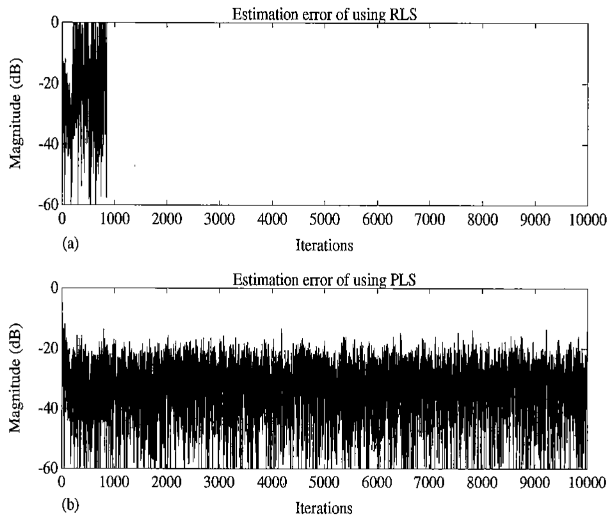

Fig. 2 Numerical behavior of PLS and RLS, (a) Estimation error of using RLS when $\lambda = 0.8$, (b) Estimation error of using PLS when $\lambda = 0.8$.

Table 1 Comutational Complexity of RLS and PLS.

Computation / Algorithm	Multiplications	Divisions	Additions & Subtractions
RLS	$2M^2+7M+5$	M^2+4M+3	$2M^2+6M+4$
FPLS / BPLS	$\frac{3}{2}M^2+4M+1$	M	M^2+2M+1

is 32-bit floating-point arithmetic.

Figure 2 shows the numerical performance of the RLS and the PLS algorithms. As shown in Fig. 2 (a), when $\lambda = 0.8$, the RLS algorithm becomes unstable after 150 iterations. Furthermore, after 850 iterations, the inverse of the input correlation matrix becomes singular. No such instability occurs in the PLS algorithm, as shown in Fig. 2 (b).

Another advantage of the PLS algorithm is that about 50% computations are required compared with the RLS algorithm as shown in Table 1. This is because the symmetry property of the inverse of the input correlation matrix is exploited in the PLS algorithm. It has been shown that the explosive divergence will occur if this property is used in the RLS algorithm [7].

5. Application to FNTF Algorithms

There are three versions of the FNTF algorithms for extending the gain vector [2]. Version 1 requires both the forward and backward predictors. Version 2 requires only the forward, and Version 3 requires only the backward predictor. The FPLS and the BPLS algorithms can readily be used in Version 2 and 3. For convenience, we write Version 2 in this section.

Version 2. Extending Gain Vector Based on FPLS Algorithm

Define:

$$\mathbf{S}_{M+1}(n) = \frac{\eta_M(n)}{\lambda F_M(n-1)} \begin{bmatrix} 1 \\ \mathbf{a}_M(n-1) \end{bmatrix} \quad (36)$$

$$n^c = n - N + M \quad (37)$$

set $\mathbf{G}_N(0) = \mathbf{0}$, $g_N(0) = 0$. From the FPLS algorithm, we get $\mathbf{S}_{M+1}(n)$, $\mathbf{S}_{M+1}(n^c)$, $\eta_M(n)$, $\eta_M(n^c)$, $\tilde{\mathbf{k}}_M(n^c)$, $\gamma_M(n^c)$, then the normalized gain vector $\tilde{\mathbf{k}}_N(n)$ can be calculated as

$$\begin{bmatrix} \mathbf{G}_N(n) \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{G}_N(n-1) \end{bmatrix} + \begin{bmatrix} \mathbf{S}_{M+1}(n) \\ \mathbf{0}_{N-M} \end{bmatrix} - \begin{bmatrix} \mathbf{0}_{N-M} \\ \mathbf{S}_{M+1}(n^c) \end{bmatrix} \quad (38)$$

$$\tilde{\mathbf{k}}_N(n) = \begin{bmatrix} \tilde{\mathbf{k}}_M(n^c) \\ \mathbf{0}_{N-M} \end{bmatrix} + \mathbf{G}_N(n) \quad (39)$$

$$g_N(n) = g_N(n-1) + S^1_{M+1}(n)\eta_M(n) - S^1_{M+1}(n^c)\eta_M(n^c) \quad (40)$$

$$\beta_N(n) = \frac{1}{\gamma_M(n^c)} + g_N(n) \quad (41)$$

where $S^1_{M+1}(n)$ denotes the first element of $\mathbf{S}_{M+1}(n)$.

The filtering part is given by

$$\alpha_N(n) = d(n) - \hat{\mathbf{w}}_N^T(n-1)\mathbf{u}_N(n) \quad (42)$$

$$\hat{\mathbf{w}}_N(n) = \hat{\mathbf{w}}_N(n-1) + \tilde{\mathbf{k}}_N(n)\alpha_N(n)/\beta_N(n) \quad (43)$$

Following the same procedure, we can write Version 3, which combines the BPLS and the FNTF algorithms [9].

In Version 2, the quantities of $\mathbf{S}_{M+1}(n)$ are calculated in the FPLS algorithm. So only 3M additions are necessary for computing the gain vector $\tilde{\mathbf{k}}_N(n)$ from $\tilde{\mathbf{k}}_M(n)$. Thus, the computational complexity of the combination of the PLS and the FNTF algorithms is about $\frac{3}{2}M^2 + 3M + 2N$ (multiplications and divisions per iteration). This saving of computation is significant especially in the case of $N \gg M$, which is satisfied in acoustic echo canceler application.

The numerical property of the combined algorithm depends on the gain vector and the predictor of order M. Since both of them are calculated by the PLS algorithm, a good numerical performance for the gain vector of order N can be expected.

6. Simulation Results

In this section, some simulations for an echo cancellation problem by using the combination of the PLS and the FNTF algorithms are carried out.

The echo path is tested inside a car environment. Figure 3 (a) shows two impulse responses for no person and three persons in the car, respectively. From this figure, the power of the impulse responses is concentrated within about 250 samples. So we choose $N = 250$. A speech signal shown in Fig. 3 (b) with power normalized to unity is used as the far-end signal, and the near-end signal is supposed to be zero. Since suitable initial values are important to the FNTF algorithms[2], we choose $\delta = 1$ and $\lambda = 0.98$. In order to make comparisons for the convergence rate as well as the tracking speed, the echo path is changed from one (no passenger) to the other (three passengers) at a 1500 sampling point.

Figure 4 shows the mean square residual echo for (1) $N = 250$, $M = 20$, implemented by using the combination of the PLS and the FNTF algorithms and the combination of the FTF and the FNTF algorithms. (2) $N = M = 250$, implemented by using the RLS algorithm. The simulation results demonstrate first that the FNTF algorithm becomes unstable after 1180 iterations when the fast RLS algorithm is used for computing the prediction part. The use of the PLS algorithm for the prediction part provides a stable performance. Second,

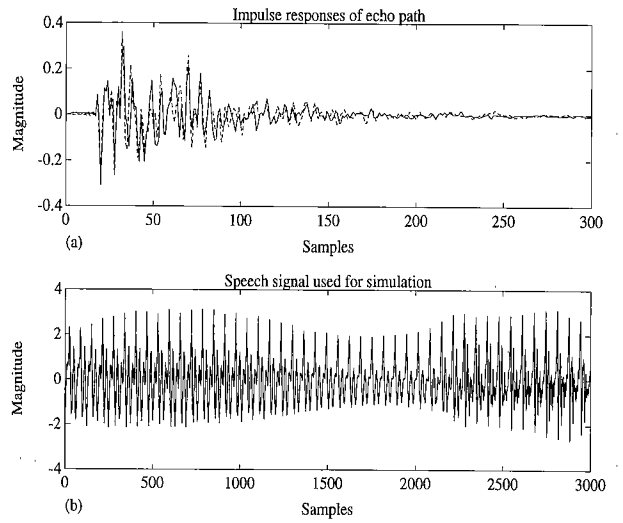

Fig. 3 Simulation conditions, (a) Impulse responses of the echo path, (b) Speech signal used for simulation.

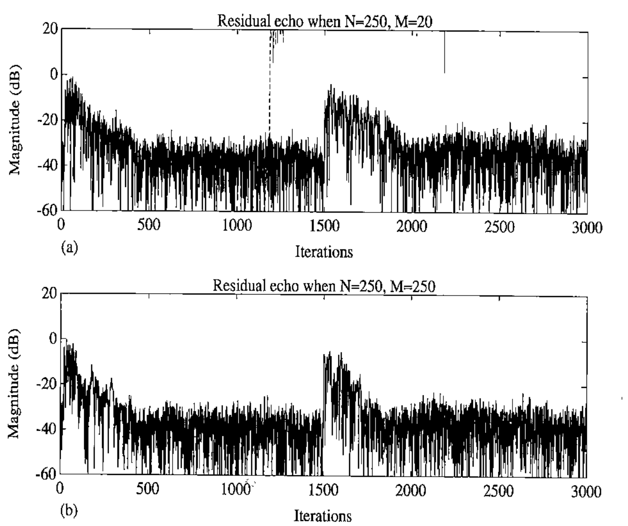

Fig. 4 Simulation results, (a) Combination of PLS and FNTF (solid line) and combination of FTF and FNTF (dashed line) when $N = 250$, $M = 20$, (b) RLS when $N = 250$, $M = 250$.

the speech signal we used can be modeled by an $AR(20)$. Therefore, even though the order of the adaptive filter is $N = 250$, the choice of the order $M = 20$ for the predictor gives the performance that is comparable to the optimum least square solution of $M = 250$. However, the computations (multiplications and divisions per iteration) are about 1160 required for the combination of the PLS and the FNTF algorithms, which are much fewer than 190,250 required for the RLS algorithm.

7. Conclusion

A predictor based least square algorithm has been introduced. It is shown that the PLS algorithm behaves more stable and needs less computation compared with the RLS algorithm. Furthermore, the PLS algorithm is very suited for applying to the FNTF algorithms, in which the required quantities for extending the gain vector are also computed in the PLS algorithm. Thus, the combination of these two algorithms provides not only a stable performance but also a simplified algorithm formulation. The simulation results for echo cancellation problem demonstrate the practical usefulness of applying this combined algorithm to real problems. The numerical property of the PLS algorithm will be further studied.

Acknowledgements

The authors wish to thank Dr. Z. Ma for kind help.

References

[1] Haykin, S., *Adaptive Filter Theory*. Second edition, Prentice-Hall, 1991.

[2] Moustakides, G.V. and Theodoridis, S., "Fast newton transversal filters – a new class of adaptive estimation algorithms," *IEEE Trans. on Signal Processing*, vol.39, no.10, pp.2184–2193, Oct. 1991.

[3] Botto, J.L. and Moustakides, G.V., "Stabilization of fast Kalman algorithms," *IEEE Trans. Acoust., Speech, Signal Processing*, vol.ASSP-37, pp.1342–1348, Sep. 1989.

[4] Slock, D.T.M. and Kailath T., "Numerically stable fast transversal filters for recursive least squares adaptive filtering," *IEEE Trans. on Signal Processing*, vol.39, pp.92–114, Jan. 1991.

[5] Benallal, A. and Gilloire, A., "A new method to stabilize fast RLS algorithms based on a first-order model of propagation of numerical errors," *Proc. ICASSP '88*, pp.1373–1376, 1988.

[6] Verhaegen, M.H., "Round-off error propagation in four generally-applicable, recursive, least-squares estimation schemes," *Automatica*, vol.25, no.3, pp.437–444, May 1989.

[7] Ljung, S. and Ljung, L., "Error propagation properties of recursive least-squares adaptation algorithms," *Automatica*, vol.21, pp.157–167, 1985.

[8] Bottomley, G.E. and Alexander, S.T., "A theoretical basis for the divergence of conventional recursive least squares filters," *Proc. ICASSP '89*, pp.908–911, 1989.