International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:1, No:1, 2007

Affine Projection Algorithm with Variable Data-Reuse Factor

ChangWoo Lee, Young Kow Lee, Sung Jun Ban, SungHoo Choi, and Sang Woo Kim

Abstract**—** This paper suggests a new affine projection(AP) algorithm with variable data-reuse factor using the condition number as a decision factor. To reduce computational burden, we adopt a recently reported technique which estimates the condition number of an input data matrix. Several simulations show that the new algorithm has better performance than that of the conventional AP algorithm.

Keywords**—**Affine projection algorithm, variable data-reuse factor, condition number, convergence rate, misalignment.

I. Introduction

AP algorithms are considered as an alternative to increase the convergence rate in adaptive filtering algorithms where input signal is correlated[1][2]. In these algorithms, the idea of variable data-reuse was proposed to lower the overall complexity of the conventional algorithm in the set-membership AP algorithm[3]. This paper proposes a new affine projection algorithm with variable data-reuse factor K (which presents the number of input vector used for update) using the condition number as a decision factor.

It is well known that adaptive filtering algorithms are sensitive to the condition number of input data. Generally, the condition number of a matrix is computed by using L_2-norm. However, it requires much computational burden to calculate the condition number. Recently, a new technique which estimates the condition number of an input data matrix was proposed in the RLS algorithm[4]. This technique used a proper norm instead of L_2-norm to reduce much computation complexity. It is a useful alternative to estimate the condition number when the exact value is not required. Based on this technique, we can easily estimate the condition number of an input data matrix in AP algorithm and use it to exploit a new AP algorithm with variable data-reuse factor. Through this process, this paper explores the relations among data-reuse factor, misalignment and the convergence rate.

As a result, this paper proposes a new algorithm that adaptively adjusts the data-reuse factor K by comparing the condition number of input data matrix at the present step with that at the previous step. This algorithm can lower the arithmetic burden of the conventional AP algorithm by employing a variable data-reuse factor, and moreover it has a lower misalignment and convergence rate similar to that of the conventional AP algorithm. Several simulations show that the performance of the proposed algorithm is better than that of the conventional AP Algorithm. Specifically, the proposed algorithm not only achieves very similar convergence rate, but also reaches a lower misalignment as compared to the conventional AP algorithm.

Notations used in this paper are fairly standard. Boldface symbols are used for vectors (in lowercase letters) and matrices (in uppercase letters). We also have the following notations.
$(\cdot)^T$: Transposition
$(\cdot)^*$: Transposition with complex conjugate
$Tr(\cdot)$: trace
$E(\cdot)$: Expectation
In addition, the symbol $\mathbf{I}$ denotes the identity matrix of appropriate dimensions. All vectors are column vectors except for the input data vector denoted by $\mathbf{u}_i$ which is taken to be a row vector for convenience notation.

II. Review of AP Algorithms

AP Algorithms can be derived from the regularized Newton's recursion[5],

$$\mathbf{w}_i = \mathbf{w}_{i-1} + \mu[\ '\mathbf{I} + \mathbf{R}_\mathbf{u}]^{-1}[\mathbf{R}_\mathbf{du} - \mathbf{R}_\mathbf{u}\mathbf{w}_{i-1}] \qquad (1)$$

where $\mathbf{w}_i$ is a weight vector at time instant i, μ is a fixed step size, $'$ is a fixed regularization parameter, $\mathbf{R}_\mathbf{u} = E[\mathbf{u}^*\mathbf{u}]$ and $\mathbf{R}_\mathbf{du} = E[\mathbf{du}^*]$.

We replace $\{\mathbf{R}_\mathbf{u}, \mathbf{R}_\mathbf{du}\}$ by the following instantaneous approximations[5]:

$$\widehat{\mathbf{R}}_\mathbf{u} = \frac{1}{K}\sum_{j=i-K+1}^{i}\mathbf{u}_j^*\mathbf{u}_j\ ,\ \widehat{\mathbf{R}}_\mathbf{du} = \frac{1}{K}\sum_{j=i-K+1}^{i} d(j)\mathbf{u}_j^* \qquad (2)$$

where K is the number of input vectors used for parameter update and is called the data reuse factor.

If the K × M block data matrix

$$\mathbf{U}_i = \begin{pmatrix} \mathbf{u}_i \\ \mathbf{u}_{i-1} \\ \vdots \\ \mathbf{u}_{i-K+1} \end{pmatrix}, \qquad (3)$$

ChangWoo Lee, Young Kow Lee, Sung Jun Ban and SungHoo Choi are with Electrical and Computer Engineering Division, Pohang University of Science and Technology, Pohang, Kyungbuk, 790-784, Korea (e-mail: caprix@postech.edu, dudry@postech.edu, bansjkr@postech.edu, csh425@postech.edu).

Sang Woo Kim is with Electrical and Computer Engineering Division, Pohang University of Science and Technology, Pohang, Kyungbuk, 790-784, Korea (corresponding author phone: 82-54-279-2237; fax:82-54-279-2903; e-mail:swkim@postech.edu).

and the K × 1 data vector

$$\mathbf{d}_i = \begin{pmatrix} d(i) \\ d(i-1) \\ \vdots \\ d(i-K+1) \end{pmatrix} \quad (4)$$

are introduced, then we can express $\{\widehat{\mathbf{R}}_{\mathbf{u}}, \widehat{\mathbf{R}}_{\mathbf{du}}\}$ simply as[5]

$$\widehat{\mathbf{R}}_{\mathbf{u}} = \frac{1}{K}\mathbf{U}_i^*\mathbf{U}_i \quad , \quad \widehat{\mathbf{R}}_{\mathbf{du}} = \frac{1}{K}\mathbf{U}_i^*\mathbf{d}_i. \quad (5)$$

By (5), the Newton's recursion (1) becomes

$$\mathbf{w}_i = \mathbf{w}_{i-1} + \mu(\ \mathbf{I} + \mathbf{U}_i^*\mathbf{U}_i)^{-1}\mathbf{U}_i^*[\mathbf{d}_i - \mathbf{U}_i\mathbf{w}_{i-1}]. \quad (6)$$

Using the inversion formula in [5]

$$(\ \mathbf{I} + \mathbf{U}_i^*\mathbf{U}_i)^{-1}\mathbf{U}_i^* = \mathbf{U}_i^*(\ \mathbf{I} + \mathbf{U}_i\mathbf{U}_i^*)^{-1}, \quad (7)$$

(6) becomes

$$\mathbf{w}_i = \mathbf{w}_{i-1} + \mu\mathbf{U}_i^*(\ \mathbf{I} + \mathbf{U}_i\mathbf{U}_i^*)^{-1}[\mathbf{d}_i - \mathbf{U}_i\mathbf{w}_{i-1}]. \quad (8)$$

If we choose $\quad > 0$ for all i, then the inversion term $(\ \mathbf{I} + \mathbf{U}_i\mathbf{U}_i^*)^{-1}$ is positive definite[6], and the choice $\quad = 0$ results in the standard AP algorithm[7]

$$\mathbf{w}_i = \mathbf{w}_{i-1} + \mu\mathbf{U}_i^*(\mathbf{U}_i\mathbf{U}_i^*)^{-1}[\mathbf{d}_i - \mathbf{U}_i\mathbf{w}_{i-1}]. \quad (9)$$

III. How to Estimate the Condition Number of an Input Data Matrix in AP Algorithm

Recently, the procedure which estimates the condition number of an input signal covariance matrix using a proper norm was presented in the context of RLS equation[4].

In [4], the means to estimate the condition number of a matrix $\mathbf{R}$ was presented as follows:

$$\ {}_E^2[\mathbf{R}^{1/2}] = \| \mathbf{R}^{1/2} \|_E^2 \| \mathbf{R}^{-1/2} \|_E^2 \quad (10)$$

which has the below property(11) and expressions (12), (13),

$$\frac{1}{L^2}\ {}_2[\mathbf{R}] \leq \ {}_E^2[\mathbf{R}^{\frac{1}{2}}] \leq \ {}_2[\mathbf{R}] \quad (11)$$

$$\| \mathbf{R}^{1/2} \|_E = \left\{ \frac{1}{L} Tr[\mathbf{R}] \right\}^{1/2} \quad (12)$$

$$\| \mathbf{R}^{-1/2} \|_E = \left\{ \frac{1}{L} Tr[\mathbf{R}^{-1}] \right\}^{1/2}, \quad (13)$$

where L is the size of a matrix $\mathbf{R}$ and subscript E is used to distinguish the different condition numbers.

To avoid the confusion of square in the formula (10), it is defined as $\ {}_P[\mathbf{R}] \equiv \ {}_E^2[\mathbf{R}^{\frac{1}{2}}]$.

Instead of concentrating on an input signal correlation matrix $\mathbf{U}_i^*\mathbf{U}_i$, this paper concentrates on $\mathbf{U}_i\mathbf{U}_i^*$ because the performance in the recursion formula (9) is directly related to the condition number of $\mathbf{U}_i\mathbf{U}_i^*$. The condition number of $\mathbf{U}_i\mathbf{U}_i^*$ becomes as follow:

$${}_P(\mathbf{U}_i\mathbf{U}_i^*) = \frac{1}{K^2} Tr(\mathbf{U}_i\mathbf{U}_i^*) Tr((\mathbf{U}_i\mathbf{U}_i^*)^{-1}), \quad (14)$$

where K is the number of an input vector used for updates or the size of $\mathbf{U}_i\mathbf{U}_i^*$.

From this equation, the condition number can be easily estimated without significantly increasing the arithmetic complexity, because the inversion term has already been calculated in (9) and it is a general assumption that $\mathbf{U}_i\mathbf{U}_i^*$ is learned before the calculation of the inversion. Specifically, the additional computation burden is shown in the Table *I*.

Figure 1 shows the comparison of (a) ${}_2(\mathbf{U}_i\mathbf{U}_i^*)$ and (b) ${}_P(\mathbf{U}_i\mathbf{U}_i^*)$. Figure 2 shows the comparison of (a) ${}_P(\mathbf{U}_i\mathbf{U}_i^*)$ and (b) $\frac{1}{K^2}\ {}_2(\mathbf{U}_i\mathbf{U}_i^*)$. From figure 1, it appears that the graphs are different and the difference between these values is very large. However, with careful observation of figure 1 and figure 2, it is evident that the pattern of the condition number using E-norm is similar to that of the condition number using L_2-norm. In some special cases like in this paper, the accurate condition number is not required since the pattern of the condition number is enough for the proposed algorithm in the next section. From the above property, this paper will use the estimated condition number to adaptively adjust the data-reuse factor.

IV. The New AP Algorithm with Variable Data-Reuse Factor

This section shows a new AP algorithm with variable data-reuse factor. Table *II* presents the proposed algorithm.

In this algorithm, $\mathbf{U}$-matrix and $\mathbf{d}$-vector are updated as the conventional AP algorithm. However, K is adaptively assigned using the condition number as a decision factor and the weight parameter vector is updated under this data-reuse factor K.

The proposed algorithm is based on the following two relations. The first one is the relations among the data-reuse factor, misalignment and convergence rate. The second one is the relations among the condition number, misalignment and convergence rate. From these relations, the condition number can be used to adjust the data-reuse factor. In the rest of this section, these relations are discussed in details.

In the first relation, it is well known that the convergence rate becomes fast as data-reuse factor K increases. It was proven through numerical formulas and simulations on [8]. The relation between misalignment and data-reuse factor K was proved from the following equation [7]:

$$EMSE = \frac{\mu\ {}_v^2}{(2-\mu)} Tr(\mathbf{R}_u) E[\frac{K}{\|\mathbf{u}_i\|^2}] \quad (15)$$

where $\ {}_v^2$ is the noise variance. From this equation, we can see that misalignment increases as K increases.

Some papers have explored the second relation. That is, misalignment increases and convergence rate becomes slightly slow as the condition number of an input correlation matrix increases. Generally, the measurement noise contributes to the misalignment. The amount of the contribution depends on $(\mathbf{U}_i\mathbf{U}_i^*)^{-1}$, which is related to the condition number of $\mathbf{U}_i\mathbf{U}_i^*$ [9]. As a result, misalignment becomes lower as the condition number of $\mathbf{U}_i\mathbf{U}_i^*$ decreases. The other reference reveals that the convergence rate of AP algorithm are slightly affected

by the condition number[10]. Specifically, it is known that convergence rate of AP algorithm can become slightly fast if the condition number decreases.

From the above relations, in case P increases, the proposed algorithm decreases K by 1 to reduce misalignment. In other words, the decreased K lowers misalignment, although the convergence rate becomes slow because the condition number of a matrix tends to increase with the matrix size[11]. It improves bad misalignment through sacrificing the convergent speed. In the other case, we increase K by 1, and it improves the convergence rate through sacrificing the good misalignment. It means that this paper can use the condition number and data-reuse factor to meet the conflicting requirements of fast convergence and low misalignment.

The proposed algorithm does not decrease K continually whenever P increases because the convergence rate of AP algorithm is related to K as mentioned before. If K decreases in every step, the convergence rate becomes very slow, even though the misalignment becomes lower. To overcome this situation, the resetting technique is adopted whenever K becomes less than 2. If K is 1, the condition number always becomes 1. In this case, K can never be larger than 1 under any circumstances in this algorithm. Therefore, the proposed algorithm resets the data-reuse factor whenever K becomes 1.

V. Simulation Results

In this section, some simulation results are presented to demonstrate the performance of the proposed algorithm. In all cases, we consider a system identification problem that is trying to identify the weights of 16-tap FIR filter. The input signal is correlated and the output noise variance (which corresponds to the minimum MSE) is -30dB. The regularization factor is 10^{-7}, the step size is 1 in all cases (NLMS, AP algorithm and the proposed algorithm) and the maximum K is 6. The results are obtained by an ensemble averaging over 500 independent trials or one iteration.

Figure 3 shows that the convergence rate of the proposed algorithm with variable data-reuse factor is very similar to that of the conventional AP algorithm and misadjustment of the proposed algorithm is smaller than that of the conventional AP algorithm. Figure 4 shows an example of the percentage of data-reuse factor for one iteration. Figure 5 is the ensemble average of K.

Figure 6 shows the performance of the proposed algorithm in details. Here, the normalized least square method shows a very slow convergence rate as mentioned before. In this situation, it is shown that AP algorithm can be used to improve the convergence rate. The relations among the data-reuse factor, convergence rate and misalignment can be clearly shown in this figure. As mentioned in section IV, the convergence rate becomes fast and misalignment becomes large as the data-reuse factor increases. The proposed algorithm achieves low misalignment, which is similar to the level of small data-reuse factor (K=2) keeping the fast convergence rate similar to that of large one (K=6). Additionally, the proposed algorithm requires computation complexity similar to that of middle one (K=4). In other words, the proposed algorithm shows better performance without much additional computation burden through adjusting the data-reuse factor.

VI. Conclusion

This paper proposed a new AP algorithm with variable data-reuse factor since the introduction of a variable data-reuse factor K allows a significant reduction in the overall complexity as compared to a fixed K. The proposed algorithm adjusts the data-reuse factor using the condition number as a decision factor to meet the conflicting requirements of fast convergence and low misalignment in AP algorithm.

To easily estimate the condition number in AP algorithm with low computation, a proper norm was used in this paper. It could be used to adjust the data-reuse factor through analyzing misalignment and convergence rates that are related to this condition number. As a result, the new algorithm was implemented to adaptively adjust the data-reuse factor K comparing the condition number at the present step with that at the previous step.

Simulation results showed that the proposed algorithm could outperform the conventional AP Algorithm in terms of computation complexity and misalignment.

References

[1] K.Ozeki and T. Umeda, “An adaptive filtering algorithm using an orthogonal projection to an affine subspace and its properties,” *Electron. Commun. Jpn.*, vol 67-A, no. 5,pp. 19-27, 1984.

[2] Paulo S. R. Diniz, *ADAPIVE FILTERING Algorithms and Practical Implemenration 2nd*, Kluwer Academic Publishers, 2002.

[3] Stefan Werner, Paulo S. R. Diniz, and José E. W. Moreira, “Set-membership affine projection algorithm with variable data-reuse factor”, *Proc. 2006 IEEE, Intern. Symposium on circuits and systems, Island of Kos, Greece*, pp.-, May. 2006.

[4] Jacob Benesty and Tomas Gänsler, “A RECURSIVE ESTIMATION OF THE CONDITION NUMBER IN THE RLS ALGORITHM”,*Acoustics, Speech, and Signal Processing, 2005, Proceedings. IEEE ICASSP’05* ,Volume 4, March 2005, pp 18-23.

[5] A.H. Sayed, *Fundamentals of Adaptive Filtering* , New York: Wiley, 2003

[6] Hernán G. Ray, Leonardo Rey Vega, Sara Tressens and Bruno Cernuschi Frias,“Analysis of explicit regulariztion in affine projection algorithm: robustness and optimal choice,” *EUSIPCO 2004*,pp.1809-1812, Sep. 2004.

[7] Hyun-Chool Shin and Ali H. Sayed,“Mean-Sqaure Performance of a Family of Affine Projection Algorithms”, *Signal Processing,IEEE Trans.*, Vol 52, No 1, January 2004, pp 90-102.

[8] Sundar G. Sankaran and A.A.(Louis) Beex,“Convergence Behavior of Affine Projection Algorithms”, *Signal Processing,IEEE Trans.*, vol 48, Issue 4, April 2000, pp 1086-1096.

[9] Dr. Sundar G. Sankaran ,“On Ways to Improve Adaptive Filter Performance”,Ph.D. Dissertation, 1999.

[10] P. Eneroth, T. Gaensler, J. Benesty, and S. L. Gay, “State of the art of stereophonic acoustic echo cancellation”, in Proc. RVK99, Sweden, June 1999.

[11] J.H. Wilkinson, “The algebraic eigenvalue problem”, Oxford University Press, 1965.

TABLE I
The Added Computation Burden

	×	+	/
The Proposed algorithm	2	2(K-1)	1

TABLE II
THE PROPOSED ALGORITHM

Initialize
$\mathbf{w} = [0, ..., 0]^T$
K = maxK
Choose μ in the range $0 < \mu \leq 2$
Do for i ≥ 0
if condN(i-1) $>$ condN(i)
  if K $<$ maxK
    K = K + 1
  end
else
  if K ≥ 3
    K = K - 1
  else
    K = maxK
  end
end
$\overline{\mathbf{U}}_{i+1} = \mathbf{U}_{i+1}(1:K,:)$
$\overline{\mathbf{d}}_{i+1} = \mathbf{d}_{i+1}(1:K,:)$
$\overline{\mathbf{e}}_{i+1} = \overline{\mathbf{d}}_{i+1} - \overline{\mathbf{U}}_{i+1}\mathbf{w}_i$
$\mathbf{w}_{i+1} = \mathbf{w}_i + \mu\overline{\mathbf{U}}_{i+1}(\overline{\mathbf{U}}_{i+1}\overline{\mathbf{U}}_{i+1}^*)^{-1}[\overline{\mathbf{d}}_{i+1} - \overline{\mathbf{U}}_{i+1}\mathbf{w}_i]$.
condN(i+1) = $\chi_p(\overline{\mathbf{U}}_{i+1}\overline{\mathbf{U}}_{i+1}^*)$

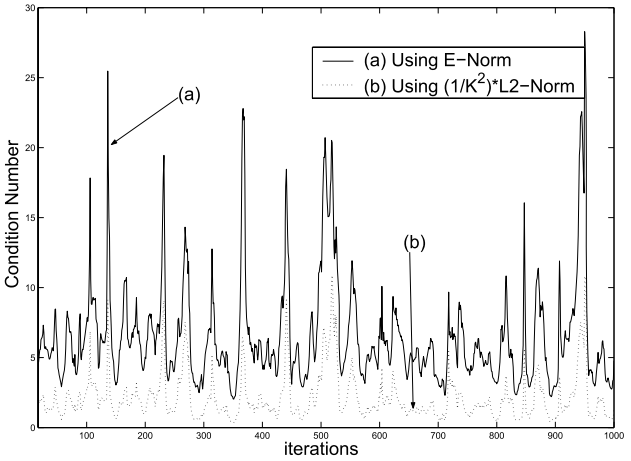

Fig. 2. Comparison of condition numbers using (a)E-norm($\chi_P(\mathbf{U}_i\mathbf{U}_i^*)$) and (b)$L_2$-norm($\frac{1}{K^2}\chi_2(\mathbf{U}_i\mathbf{U}_i^*)$) for one iteration

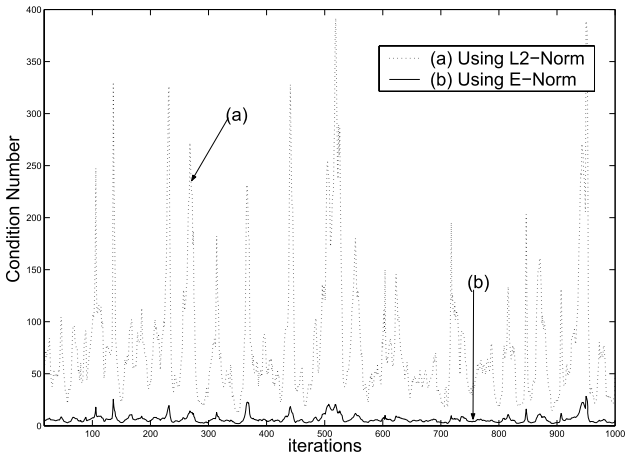

Fig. 1. Comparison of condition numbers using (a)L_2-norm($\chi_2(\mathbf{U}_i\mathbf{U}_i^*)$) and (b)E-norm($\chi_p(\mathbf{U}_i\mathbf{U}_i^*)$) for one iteration

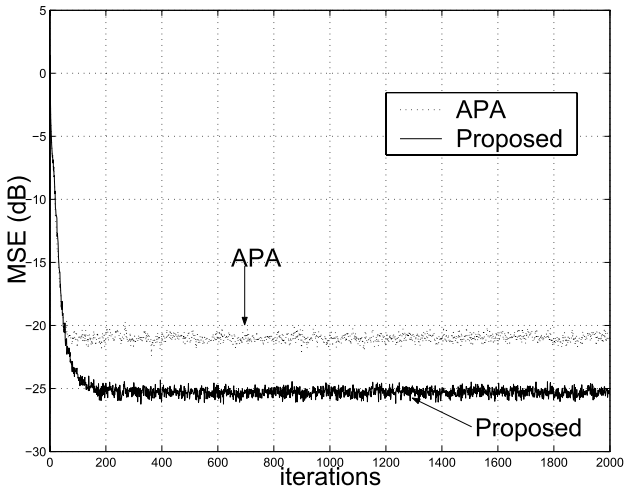

Fig. 3. Comparison of learning curves for the proposed algorithm and conventional AP algorithm with fixed K

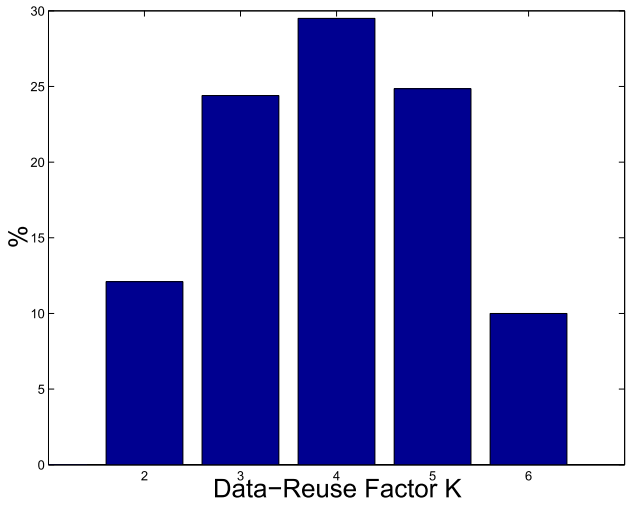

Fig. 4. Percentage of Variation of K for one iteration

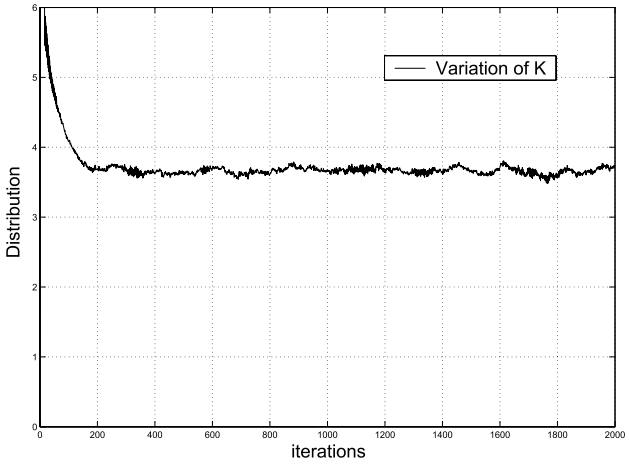

Fig. 5. Variation of K for 500 ensemble average

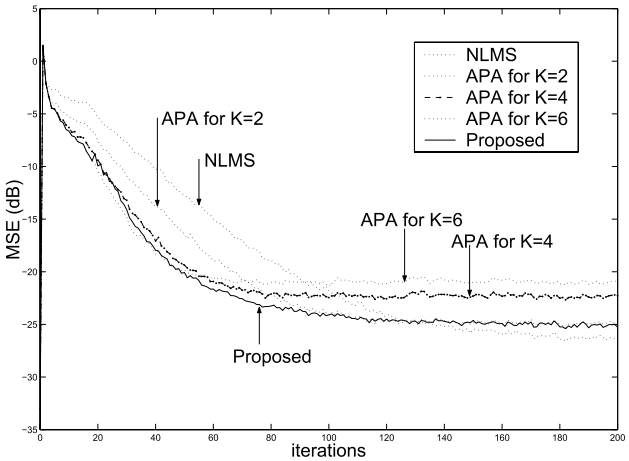

Fig. 6. Comparison of learing curves for NLMS, AP algorithm for K=2, AP algorithm for K=4, AP algorithm for K=6 and the proposed algorithm