Low Complexity Implementation Method for the Adaptive Filters based on the Gaussian Model

Kai Yokoyama* and Kiyoshi Nishikawa†
* Tokyo Metropolitan University, Tokyo, Japan
E-mail: yokoyama-kai@ed.tmu.ac.jp
† Tokyo Metropolitan University, Tokyo, Japan
E-mail: kiyoshi@tmu.ac.jp

***Abstract*—This paper proposes a method to lower the computational complexity of the adaptive filters based on the Gaussian model (GM-ADF). The conventional GM-ADF is shown to be robust against impulsive noise. However, the price to pay is a higher computational complexity since each coefficient in the GM-ADF has its own Gaussian model, and the parameters (means and variances) of the Gaussian distributions must be computed to estimate them. In this paper, we propose a method which trims down the number of coefficients modeled by the Gaussian distributions, so that the number of the parameter calculated per iteration reduces. Besides, we alter the way to update the coefficients, which would possibly increase the rate of convergence under some conditions. The performance of the proposed method is demonstrated by computer simulations.**

I. Introduction

Adaptive filtering algorithms have been used in various signal processing applications such as echo cancelation, noise cancelation, channel equalization, and system identification [1]-[3]. The most popular adaptive algorithms are the lean-mean-square (LMS) and the normalized-LMS (NLMS) because the LMS-type algorithms have the advantage of easily implemented at a computational complexity of $O(L)$, where L shows the length of the filter [1]. However, in real applications, adaptive filters are suffered from performance degradation under the environments where non-Gaussian noise, such as impulsive noise, exists [10]. To overcome this problem, many robust adaptive algorithms have been developed [4]-[10]. The adaptive filter based on the Gaussian model (GM-ADF) [11] is one of those algorithms.

GM-ADF is derived from the concept of the Gaussian mixture model (GMM), which has been widely used in pattern recognition and signal processing applications, e.g. , for clustering of data or probability density estimation[12]-[15]. In GM-ADF, each coefficient is modeled as a random variable with the Gaussian distribution, and is used to check whether its coefficients should be updated or not. In other words, at each time, a candidate of the value at the next time of each coefficient is computed by an adaptive algorithm and then, the GM-ADF detects the outliers by checking based on the Gaussian distributions to keep the coefficient value to be inside the range of the distribution. Hence, The GM-ADF eliminates the probability of updating the coefficients when the impulsive inference appears. The computational complexity of GM-ADF is higher than other robust LMS-type algorithms since the parameters (means and variances) of the Gaussian distributions for all the coefficients must be estimated. It means that the complexity of the GM-ADF could be heighten for some practical applications.

In this paper, we propose a method to lower the computational complexity of GM-ADF. In the original GM-ADF, each coefficient has its own Gaussian model, and is processed independently. The proposed method, on the other hand, trims down the number of the coefficients modeled as the Gaussian distribution. In the following, we call the proposed method as the adaptive filter based on the Gaussian model trimmed down (TD-GM-ADF). Accordingly, we modify the way to update the coefficients; that is, all coefficients are computed simultaneously using a LMS-type algorithm on the basis of the evaluation with a subset of the coefficients with Gaussian model. We apply the TD-GM-ADF to the NLMS algorithm and the NLMS algorithm using the Gear Shift (GS) strategy [10]. The performance of the proposed method is evaluated by computer simulations. Furthermore, we compare the TD-GM-ADF with the VP-VSS-NLMS algorithm and the GS-VP-VSS-NLMS algorithm [10]. In addition to lower the computational complexity, the simulation results show that the TD-GM-ADF can achieve faster convergence than the conventional GM-ADF.

II. Adaptive algorithm

A. NLMS algorithm

In this paper, we assume the following model for the system identification problem, i.e., the desired signal $d(k)$ is given as

$$d(k) = \mathbf{u}(k)^T \mathbf{h} + v(k) \tag{1}$$

where the superscript T denotes the vector transpose operation, $\mathbf{u}(k) = [u(k), u(k-1), ..., u(k-L+1)]^T$ is a $(L \times 1)$ input signal vector, L stands for the filter length, $\mathbf{h} = [h_0, h_1 ..., h_{L-1}]^T$ is the coefficient vector of an unknown system, and $v(k)$ is the environmental noise which is independent of the input signal.

The purpose of the adaptive filters is to estimate the coefficient vector $\mathbf{h}$ of the unknown system. Let $\hat{h}_i(k)$ be an estimate of the i-th coefficient of $\mathbf{h}$ at time k and $\hat{\mathbf{h}}(k)$ be defined as

$$\hat{\mathbf{h}}(k) = [\hat{h}_0(k), \hat{h}_1(k), ..., \hat{h}_i(k), ..., \hat{h}_{L-1}(k)]^T. \tag{2}$$

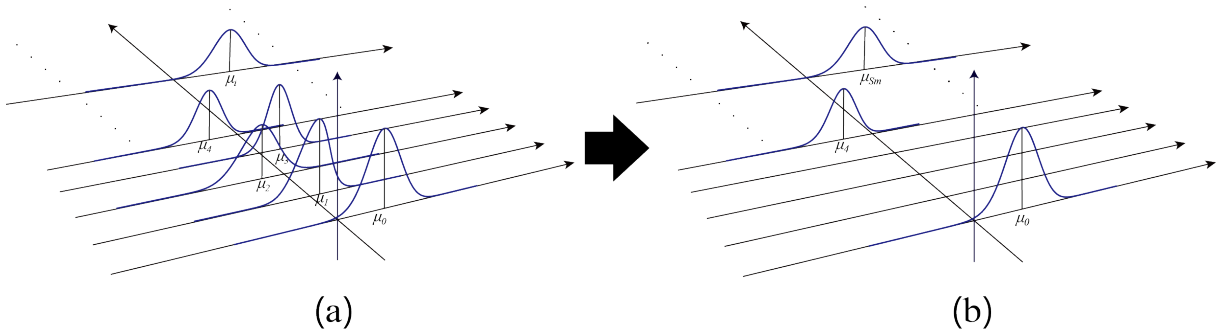

Fig. 1. (a) In the GM-ADF, graphical explanation of the coefficient modeled as the Gaussian RV. All coefficient have their own Gaussian model. (b) In the TD-GM-ADF, graphical explanation of the coefficient modeled as the Gaussian RV. Only part of the coefficients has own Gaussian model.

In this paper, the LMS-type algorithms are assumed to be used, in which coefficients are updated by the equation

$$\hat{\mathbf{h}}(k+1) = \hat{\mathbf{h}}(k) + \lambda(k)\mathbf{u}(k)e(k) \tag{3}$$

where $\lambda(k)$ is the step size at time k and $e(k) = d(k) - \mathbf{u}(k)^T\hat{\mathbf{h}}(k)$. The NLMS algorithm uses the normalized step size $\lambda(k)$ defined as

$$\lambda(k) = \eta/(\mathbf{u}(k)^T\mathbf{u}(k) + \xi) \tag{4}$$

where $0 \leq \eta < 2$ is step size of the NLMS and the small positive constant ξ is used to prevent the division by zero.

B. GM-ADF algorithm

In the GM-ADF, each coefficient is processed independently. Therefore, here, we describe the individual component of (3), i.e., the update formula of the i-th coefficient is given as

$$\hat{h}_i(k+1) = \hat{h}_i(k) + \lambda e(k)u(k-i) \qquad i = 0, 1, ..., L-1. \tag{5}$$

Here, we introduce a new variance $\zeta_i(k)$, as in [11], defined by

$$\zeta_i(k) = \hat{h}_i(k) + \lambda e(k)u(k-i). \tag{6}$$

Because, in the GM-ADF, $\hat{h}_i(k+1)$ of (5) may not used as the coefficient at time $k+1$, we utilize $\zeta_i(k)$ to emphasize this. In addition, GM-ADF assumes that $\hat{h}_i(k)$ can be modeled as a random variable (RV) with the Gaussian distribution. That is, it's probability density $p(\hat{h}_i(k))$ is given by the Gaussian distribution, denoted by $p(\hat{h}_i(k)) = \mathcal{N}(\mu_i, \sigma_i^2)$, where $\mathcal{N}(\mu, \sigma^2)$ shows a Gaussian probability density function (PDF) with the mean μ and the variance σ^2. Besides, the mean value of the distribution is assumed to be the true coefficient h_i, i.e.,

$$\mu_i = h_i. \tag{7}$$

Under these assumptions, the adaptive process of $\hat{h}_i(k)$ using (5) are interpreted as that it seeks the true mean value μ_i of the Gaussian distribution. Fig. 1(a) shows a graphical explanation of the coefficients modeled as an RV with the Gaussian distribution in the GM-ADF. As shown in this figure, each of the coefficients has it's own Gaussian distribution in the GM-ADF.

The motivation that the GM-ADF models the coefficients as the Gaussian RVs is to detect the outliers when the impulsive inference appears. Under the Gaussian model, when $\hat{h}_i(k)$ oscillates within a certain range from μ_i, the coefficients are regarded as the convergence. Therefore, GM-ADF evaluates the following condition after the computation of $\hat{h}_i(k+1)$ using (5):

$$\hat{\mu}_i(k) - \gamma\hat{\sigma}_i(k) < \zeta_i(k) < \hat{\mu}_i(k) + \gamma\hat{\sigma}_i(k) \tag{8}$$

where $\hat{\mu}_i(k)$ and $\hat{\sigma}_i(k)$ are an estimated mean and an estimated variance at time k respectively, and γ is a parameter to determine the width of the acceptable range of the variation. $\hat{h}_i(k+1)$ is checked if it is in the range using condition (7) and determine whether $\hat{h}_i(k)$ should be updated or not. Fig. 2 shows an example of the region where the condition (7) is true. Since we can not use the true information of the distribution $\mathcal{N}(\mu_i, \sigma_i^2)$, it uses the estimated values of μ_i, σ_i. Furthermore, when the condition (7) is true, they are updated as follows

$$\hat{\mu}_i(k+1) = \beta_\mu\hat{\mu}_i(k) + (1-\beta_\mu)\zeta_i(k) \tag{9}$$

$$\hat{\sigma}_i^2(k+1) = \beta_\sigma\hat{\sigma}_i^2(k) + (1-\beta_\sigma)(\zeta_i(k) - \hat{\mu}_i(k))^2 \tag{10}$$

where $0 < \beta_\mu < 1$ and $0 < \beta_\sigma < 1$ are the forgetting factors of the mean and the variance respectively. After that, the i-th coefficient is updated as

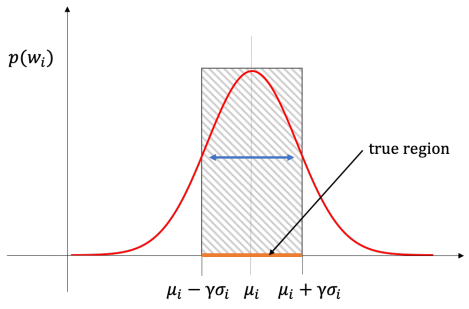

Fig. 2. Graphical explanation of the condition (7). When the $\hat{h}_i(k+1)$ is in true region, the coefficient will be updated.

$$\hat{h}_i(k+1) = \hat{\mu}_i(k+1). \tag{11}$$

On the other hand, when the condition of (7) is false, the coefficient will not be updated (i.e., $\hat{h}_i(k+1) = \hat{h}_i(k)$).

In practice, we need to prepare initial estimates of $\hat{\mu}$ and $\hat{\sigma}^2$. To gain them, they are computed using (8), (9) respectively during time $0 \leq k \leq k^*$, where k^* is a predefined period [15].

C. Computational complexity of the GM-ADF

In the GM-ADF, each coefficient is processed independently. The computations of $\hat{\mu}_i(k+1)$ in (8) and $\hat{\sigma}_i^2(k+1)$ in (9) require 5L multiplications and 3L+2 additions. in order to implement GM-ADF, we need 8L+2 multiplications and 6L+2 additions per iteration.

III. PROPOSED METHOD

A. Preparation

In this section, we propose a novel approach which trims down the number of coefficients modeled as the Gaussian RVs, so as to lower the computational complexity of the GM-ADF, which we call TD-GM-ADF. First, to express the coefficients modeled as the Gaussian RVs, we define the set H_i as follows

$$H_i = \{\hat{h}_i | i = 0, 1, ..., L-1\} \tag{12}$$

Then, we choose the coefficient $\hat{h}_{s_m}$ from H_i, where

$$s_m \subseteq 0, 1, ..., L-1, \tag{13}$$

and

$$m = 0, 1, ..., M-1 \leq L-1 \tag{14}$$

where M shows the number of coefficients modeled as the Gaussian RVs. In the proposed method, therefore, the selected M coefficients are used to detect the outliers. Besides, the set H_{s_m} of $\hat{h}_{s_m}$ is given as

$$H_{s_m} \subseteq H_i. \tag{15}$$

B. TD-GM-ADF algorithm

Here, we describe the proposed procedure. First, as in the conventional GM-ADF, all candidates of coefficients at $k+1$ are simultaneously computed at time k using (3).

Besides, we assume that the coefficients $\hat{h}_{s_m}(k)$ in the subset H_{s_m} can be modeled as the Gaussian RVs. Those coefficients are used to determine whether all coefficients should be updated or not. Fig. 1(b) shows a graphical explanation of the coefficients modeled as the Gaussian RVs in the TD-GM-ADF. As shown in Fig. 1, all coefficients are modeled as the Gaussian RVs in the GM-ADF, whereas a subset of the coefficients is modeled as the Gaussian RVs in the TD-GM-ADF. Hence, the condition (7) becomes

$$\hat{\mu}_{s_m}(k) - \gamma\hat{\sigma}_{s_m}(k) < \hat{h}_{s_m}(k+1) < \hat{\mu}_{s_m}(k) + \gamma\hat{\sigma}_{s_m}(k) \tag{16}$$

where $\hat{\mu}_{s_m}$ and $\hat{\sigma}_{s_m}$ are the estimated mean and variance of the distribution on $\hat{h}_{s_m}$ respectively. Compared with (7), the condition (15) is used only coefficients in the set H_{s_m}. Accordingly, we also rewrite (8), (9) as

$$\hat{\mu}_{s_m}(k+1) = \beta_\mu \hat{\mu}_{s_m}(k) + (1-\beta_\mu)\hat{h}_{s_m}(k+1) \tag{17}$$

$$\hat{\sigma}^2_{s_m}(k+1) = \beta_\sigma \hat{\sigma}^2_{s_m}(k) + (1-\beta_\sigma)(\hat{h}_{s_m}(k+1) - \hat{\mu}_{s_m}(k))^2. \tag{18}$$

In the GM-ADF, each coefficient is checked using (7) independently. Instead, the TD-GM-ADF checks whether or not $\hat{\mathbf{h}}(k+1)$ should be updated using the following condition:

$$|\hat{H}_{s_m}| \geq \lceil H_{s_m}/2 \rceil \tag{19}$$

where $|\cdot|$ shows the number of the member in a set, $\lceil \cdot \rceil$ is the ceiling function, and the subset $\hat{H}_{s_m}$ is

$$\hat{H}_{s_m} = \{\hat{h}_{s_m} \in H_{s_m} | \hat{\mu}_{s_m} - \gamma\hat{\sigma}_{s_m} < \hat{h}_{s_m} < \hat{\mu}_{s_m} + \gamma\hat{\sigma}_{s_m}\}. \tag{20}$$

Namely, when more than half of the selected coefficients satisfy the condition (15), the TD-GM-ADF updates all the coefficients. After that, when condition(17) is false, the update formula of $\hat{\mathbf{h}}(k)$ is expressed as

$$\hat{\mathbf{h}}(k+1) = \hat{\mathbf{h}}(k). \tag{21}$$

The TD-GM-ADF algorithm is summarized in Algorithm 1. In the GM-ADF, each coefficient has its own Gaussian model and is processed independently at time k. By contrast, in the TD-GM-ADF, all the coefficients are simultaneously computed using the NLMS algorithm and then, whether $\hat{\mathbf{h}}(k+1)$ should be updated or not is finally concluded based on the evaluation of the condition (17).

Algorithm 1 : GM-ADF algorithm using proposed method

Parameter setting: β_μ, β_σ, γ, s_m
  Insert $\hat{h}_{s_m}$ into dictionary $\{H_{s_m}\}$
Initialization:
for each time $0 \leq k \leq k^*$, each s_m **do**
  $\hat{\mathbf{h}}(k+1) = \hat{\mathbf{h}}(k) + \lambda\mathbf{u}(k)e(k)$
  $\hat{\mu}_{s_m}(k+1) = \beta_\mu\hat{\mu}_i(k) + (1-\beta_\mu)\hat{h}_i(k+1)$
  $\hat{\sigma}^2_{s_m}(k+1) = \beta_\sigma\hat{\sigma}^2_i(k) + (1-\beta_\sigma)(\hat{h}_i(k+1) - \hat{\mu}_i(k))^2$
end for
Adaption Loop:
for each time $k > k^*$ **do**
  $\hat{\mathbf{h}}(k+1) = \hat{\mathbf{h}}(k) + \lambda\mathbf{u}(k)e(k)$
  for each s_m in $\{H_{s_m}\}$ **do**
    if $\hat{\mu}_{s_m}(k) - \gamma\hat{\sigma}_{s_m}(k) < \hat{h}_{s_m}(k+1) < \hat{\mu}_{s_m}(k) + \gamma\hat{\sigma}_{s_m}(k)$
      Insert $\hat{h}_{s_m}$ into dictionary $\{\hat{H}_{s_m}\}$
    end if
    if $|\hat{H}_{s_m}| \geq \lceil H_{s_m}/2 \rceil$
      $\hat{\mu}_{s_m}(k+1) = \beta_\mu\hat{\mu}_{s_m}(k) + (1-\beta_\mu)\hat{h}_{s_m}(k+1)$
      $\hat{\sigma}^2_{s_m}(k+1) = \beta_\sigma\hat{\sigma}^2_{s_m}(k) + (1-\beta_\sigma)(\hat{h}_{s_m}(k+1) - \hat{\mu}_{s_m}(k))^2$
    else
      $\hat{\mathbf{h}}(k+1) = \hat{\mathbf{h}}(k)$
    end if
  end for
end for

C. Computational complexity of the TD-GM-ADF

The computations of $\hat{\mu}_{s_m}(k+1)$ in (15) and $\hat{\sigma}^2_{s_m}(k+1)$ in (16) require 5M multiplications and 3M+2 additions. To

TABLE I
PARAMETER SETTINGS SUMMARY

Algorithm	η	β_μ	β_μ	γ	k_β	k_e
NLMS	0.7	-	-	-	-	-
GM-ADF(I)	0.7	0.3	0.1	10	-	-
GM-ADF(II)	0.7	0.5	0.1	10	-	-
TD-GM-ADF(I)	0.7	0.3	0.1	10	-	-
TD-GM-ADF(II)	0.7	0.5	0.1	10	-	-
VP-SSS-NLMS(I)	0.7	-	-	-	1000	15
VP-SSS-NLMS(II)	0.7	-	-	-	2000	25
GS-VP-SSS-NLMS	-	-	-	-	2000	15
GS-TD-GM-ADF	-	0.3	0.1	10	-	-

implement the TD-GM-ADF, we need 3L+ 5M+2 multiplications and 3L+3M+2 additions per iteration in total. Note that, when we set $M = L$, the computational complexity of the TD-GM-ADF will be equal to that of GM-ADF. In order to compare the complexity of the TD-GM-ADF and that of the GM-ADF, we define the computational efficiency as

$$\epsilon = M/L. \tag{22}$$

IV. SIMULATION RESULTS

To demonstrate the performance of the proposed method, we applied the TD-GM-ADF algorithm to system identification problems. The unknown system is the FIR filter of $L = 36$ taps, which designed using the Remez algorithm. The input signal is the AR(1) process with the AR parameter a_0 set as $a_0 = 0.95$. The system has the abrupt change at $k = 2500$ to examine the tracking properties of the algorithms.

In addition, the system noise $v(k)$ is a mixture of a white Gaussian noise $s(k)$ with 30 dB signal-to-noise ratio (SNR) and impulsive noise $q(k)$ ($q(k) = 2$) (i.e. $v(k) = s(k)+q(k)$). The $q(k)$ is modeled as $q(k) = X(k)Y(k)$, where $X(k)$ is a Bernoulli process described by the probability $Pr(X(k) = 1) = p$ ($p = 0.01$) and $Pr(X(k) = 0) = 1 - p$, and $Y(k)$ is a zero-mean white Gaussian with the power $\sigma_Y^2 = 1000\sigma_y^2$ (σ_y^2 is the power of the system output $y(k) = \mathbf{u}(k)^T\hat{\mathbf{h}}(k)$) We use the normalized misalignment defined by

$$10log_{10}\frac{\|\hat{\mathbf{h}}(k) - \mathbf{h}\|^2}{\|\mathbf{h}\|^2}. \tag{23}$$

The misalignment curves are gained from 100 independent trials.

A. Comparison with the NLMS and the GM-ADF

First, we compare the performance of the TD-GM-ADF algorithm with those of the NLMS algorithm and the GM-ADF algorithm. The parameter settings are summarized in Table I. Fig. 3, Fig. 4 and Fig. 5 are obtained with $p = 0$, $p = 0.01$ and $p = 0.05$ respectively, and we set $s_m = \{5i|i = 0, 1, 2, ..., 7\}(M = 8)$ in both TD-GM-ADF(I) and TD-GM-ADF(II) . In this case, the computational efficiency becomes $\epsilon = 4.5$. In Fig. 3, It can be seen that the NLMS and the TD-GM-ADF exhibit same behavior and the conventional GM-ADF degrades the performance in terms of the rate of convergence. Considering the results, when the impulsive

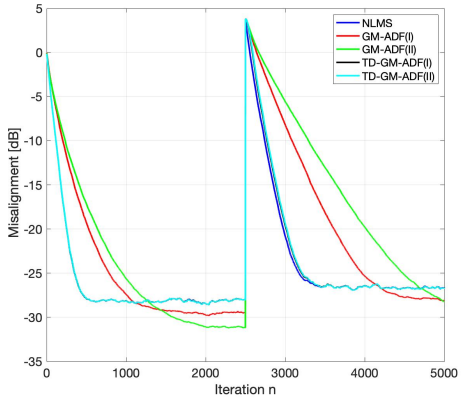

Fig. 3. Misalignment learning curves for the NLMS algorithm, the GM-ADF algorithm and the TD-GM-ADF algorithm. The impulsive interferences occur with probability $p = 0$.

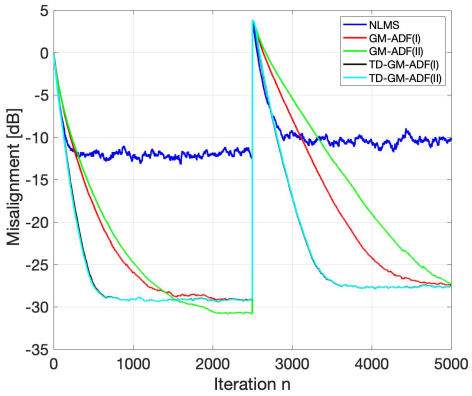

Fig. 4. Misalignment learning curves for the NLMS algorithm, the GM-ADF algorithm and the TD-GM-ADF algorithm. The impulsive interferences occur with probability $p = 0.01$.

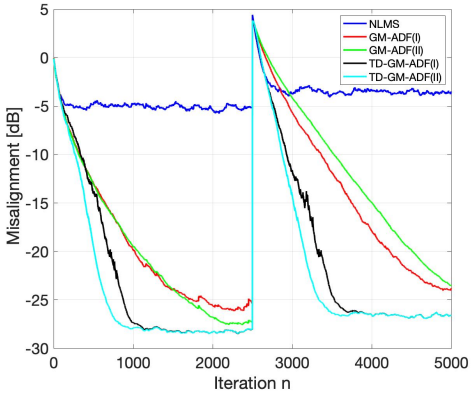

Fig. 5. Misalignment learning curves for the NLMS algorithm, the GM-ADF and the TD-GM-ADF algorithm . The impulsive interferences occur with probability $p = 0.05$.

Algorithm 2 :Robust adaptive algorithms

VP-SSS-NLMS:

$$\hat{\mathbf{h}}(k+1) = \hat{\mathbf{h}}(k) + \lambda s(\beta(k), e(k)/\|u(k)\|)\mathbf{u}(k)e(k)$$

$$A_e = [|e(k)|, |e(k-1)|, ..., |e(k-N_w+1)|$$

$$\bar{e}(k) = \nu\bar{e}(k-1) + (1-\nu)Cmin(A_e(k))$$

$$\beta(k) = \begin{cases} k_\beta e^{-b} & (\bar{e}(k) \geq b/k_e) \\ k_\beta e^{-k_e\bar{e}(k)} & (otherwise) \end{cases}$$

$$s(\beta(k), e(k)/\|u(k)\|) = 1/(1+\beta(k)(e(k)/\|u(k)\|))^2$$

$$\nu = 0.99, N_w = 18, C = 1.483(1+5/(N_w-1)), b = 4.6$$

GS strategy:

$$\eta = \begin{cases} \eta_{large} & (\bar{e}^2(k) \leq \tau) \\ \eta_{small} & (otherwise) \end{cases}$$

$$\eta_{large} = 0.7, \eta_{small} = 0.1, \tau = 10^{-4}$$

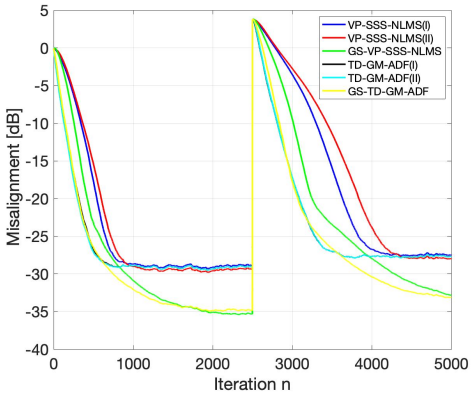

Fig. 6. Misalignment learning curves in system identification. The impulsive interferences occur with probability $p = 0.01$.

noise do not occur, TD-GM-ADF where each coefficient is updated independently could lead to slow convergence. As shown in Fig. 3 and Fig. 4, TD-GM-ADF algorithms behave in a similar manner. As compared to the GM-ADF algorithm, we can see that the rate of convergence of them improves, especially after the system changes at $k = 2500$. In Fig. 5, we can see that both TD-GM-ADF algorithms achieve better performances than other algorithms, in terms of the tracking property and the final misalignment.

B. Comparison with conventional robust algorithms

In the next experiment, the performance of the TD-GM-ADF algorithm and the TD-GM-ADF algorithm using the GS strategy (GS-TD-GM-ADF) algorithm are compared with that of the conventional robust adaptive algorithms, the VP-SSS-NLMS algorithm and the GS-VP-SSS-NLMS algorithm [10], which are summarized in Algorithm 2. For the parameter settings, the setting s_m was same as the previous experiment, and the other parameters used in the simulation are listed in Table I. From Fig. 6, we can see that the performance of the TD-GM-ADF is comparable to those of the conventional robust adaptive algorithms, and the final misalignment of the GS-TD-GM-ADF is better than those of the TD-GM-ADF. As we mentioned earlier, the TD-GM-ADF simultaneously update all coefficients at time k, and it means that it can be easily apply to various adaptive algorithms.

C. Effect of the selection of $s_m$and M

Finally, we examine the effects of the selection of s_m and M on the convergence characteristics, that is; we varied the number of coefficients modeled as the Gaussian RVs. In these simulations, we set $s_m = \{5i|i = 0, 1, \ldots, 7\}(M = 8)$, $s_m = \{10i|i = 0, 1, 2, 3\}(M = 4)$, and $s_m = \{15i|i = 0, 1, 2\}(M = 3)$ in TD-GM-ADF(II). The parameters of TD-GM-ADF(II) were same as Table I. The computational efficiencies ϵ become $\epsilon = 4.5$, $\epsilon = 9$, and $\epsilon = 12$ respectively. The simulation results are shown in Fig. 7 and Fig.8, where

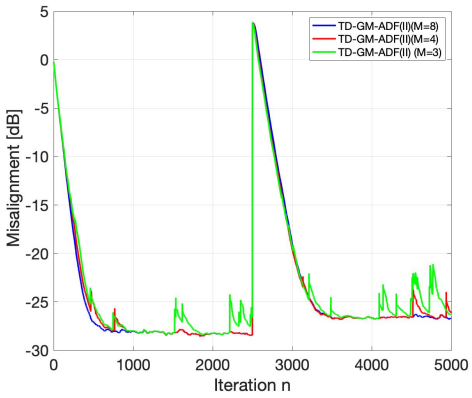

Fig. 7. Misalignment learning curves for the TD-GM-ADF algorithm . In this simulation, we varied M. The impulsive interferences occur with probability $p = 0.01$.

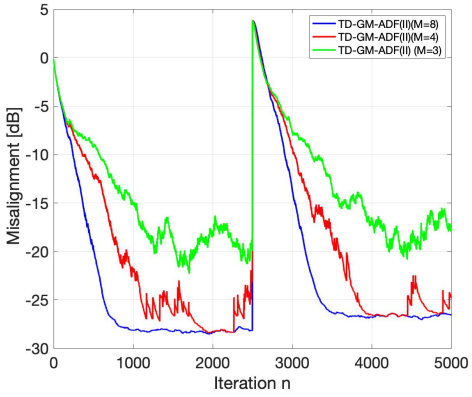

Fig. 8. Misalignment learning curves for the TD-GM-ADF algorithm . In this simulation, we varied M. The impulsive interferences occur with probability $p = 0.05$.

$p = 0.01$ for Fig. 7 and $p = 0.05$ for Fig. 8. In Fig. 7, it can be seen that TD-GM-ADF(II) with $M = 3$ (green) modestly jumps at the points where the impulse noise affect it. In Fig 8, the performances of TD-GM-ADF(II) with $M = 3$ (green) and TD-GM-ADF(II) with $M = 4$ (red) degrade, while that of TD-GM-ADF(II) with $M = 8$ (blue) remains barely unchanged. From these results, we can say that the selections of s_m and M influences the robustness to an impulsive noise and generate the trade-off relation for the computational complexity.

V. Conclusions

In this paper, we proposed a method to lower the computational complexity of the GM-ADF, namely TD-GM-ADF. The subset of its coefficients were modeled as the Gaussian RV and all of its coefficient were simultaneously updated. Consequently, the computational complexity of the TD-GM-ADF was lower as compared to the GM-ADF, in which each coefficient had its own Gaussian model. The simulation results showed that the performance of the TD-GM-ADF algorithm could achieve better converge rate than that of the GM-ADF algorithm depending on the conditions. However, to trim down the number of coefficients as the Gaussian RVs could lead to miss detecting outliners.

References

[1] A. H. Sayed *Fundamentals of Adaptive Filtering*, Wiley-Interscience, 2003.

[2] J. Benesty and Y. Huang, *Adaptive,Signal Processing- Applications to Real World Problems*, Berlin Germany:Springer-Verlag, 2003.

[3] Y. Huang, J. Benesty, and J. Chen, *Acoustic MIMO Signal Processing* Boston, MA, USA: Springer-Verlag, 2006.

[4] Y. Zhang, J. A. Chambers, W. Wang, P. Kendrick, and T. J. Cox, "A new variable step-size lms algorithm with robustness to nonstationary noise," in 2007 *IEEE International Conference on Acoustics, Speech and Signal Processing* - ICASSP '07, vol. 3, April 2007, pp. III–1349–III–1352.

[5] L.Rey, H.Rey, J. Benesty, Senior Member, IEEE, and S. Tressens, "A New Robust Variable Step-Size NLMS Algorithm", *IEEE Trans. Signal Process.*,vol 56,no.5,pp. 1878-1893,May.2008.

[6] J. Yoo, J. Shin and P. Park, "Variable step-size affine projection sign algorithm," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 61, no. 4, pp. 274–278, Apr. 2014.

[7] J. Arenas-Garcia, A. R. Figueiras-Vidal, and A. H. Sayed, "Mean-square performance of a convex combination of two adaptive filters," *IEEE Transactions on Signal Processing*, vol. 54, no. 3, pp. 1078–1090, March 2006.

[8] S. S. Kozat, A. T. Erdogan, A. C. Singer, and A. H. Sayed, "Transient analysis of adaptive affine combinations," *IEEE Transactions on Signal Processing*, vol. 59, no. 12, pp. 6227–6232, Dec 2011.

[9] I. Song, P. Park, and R. W. Newcomb, "A normalized least mean squares algorithm with a step-size scaler against impulsive measurement noise," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 60, no. 7, pp. 442–445, Jul. 2013.

[10] F. Huang, J. Zhang, and S. Zhang, "NLMS Algorithm based on a Variable Parameter Cost Function Robust Against Impulsive Interferences", *IEEE Trans.Signal Process.*,vol 64,no.5,pp. 600-604,May.2017.

[11] K. Nishikawa, " Low Variance Adaptation Method for the LMS-type Adaptive Filter based on the Gaussian Model", *IEEE International Symposium on Circuits and Systems (ISCAS)*, May, 2019

[12] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," *in Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, vol. 2, 1999, p. 252 Vol. 2.

[13] D. A. Reynolds and R. C. Rose, "Robust text-independent speaker identification using gaussian mixture speaker models," *IEEE Transactions on Speech and Audio Processing*, vol. 3, no. 1, pp. 72–83, Jan 1995.

[14] W. M. Campbell, D. E. Sturim, and D. A. Reynolds, "Support vector machines using gmm supervectors for speaker verification," *IEEE Signal Processing Letters*, vol. 13, no. 5, pp. 308–311, May 2006.

[15] K. Nishikawa, Y. Yamashita, T. Yamaguchi, and T. Nishitani, "Consideration on performance improvement of shadow and reflection removal based on gmm," in *2016 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, Dec 2016, pp. 1–7.