TABLE I
MEAN SQUARE ERROR OF PROCESSED PICTURES

Kernel	Condition Number	Eigenvalues Used	Mean Square Error
Data Compression Ratio = 4			
Cubic spline	27.34	64	0.7068
		62	0.7283
		60	0.7567
Modifier cubic spline	1	64	0.658
Hamming window function	10.21	64	1.699
		62	0.8219
		60	0.853
Modified Hamming	1	64	0.695
Data Compression Ratio = 7			
Cubic spline	21.47	48	2.07
		46	2.13
		44	2.2
Modified cubic spline	1	48	2.02
Hamming window function	5.95	48	0.87
Modified Hamming function	1	48	1.46

the same. Thus the modified algorithm may be written as follows.

Step 1: Define the rectangular transform kernel for a given data compression ratio.

Step 2: Decompose the kernel into its eigenvalues and eigenvectors.

Step 3: Select $\lambda_{\max}^{1/2}$ the largest of the eigenvalues.

Step 4: Reconstruct the modified kernel $[A_{\text{mod}}]$ using the equation

$$[A_{\text{mod}}] = \lambda_{\max}^{1/2} \sum_{i=1}^{m} u_i v_i^t.$$

Step 5: Determine the pseudoinverse of $[A_{\text{mod}}]$.

IV. SIMULATION

A diagonally dominant matrix can be considered as an interpolation matrix for which piecewise polynomial functions are very good. Here, three such kernels are derived from cubic B spline functions (CSF) [2], Hamming window functions (HWF) [3], and Gaussian distribution functions. Random test data of length $N = 64$ are processed by these rectangular transforms for a compression ratio of 4 by transform kernels of size $n = 64$ and $m = 16$. The results are given in Figs. 1–3. It is seen that the retransformed data almost agree with the input data. The error reduces initially with the increase in the number of eigenvalues (MM) chosen for calculation of pseudoinverse, but later there is not much difference. The kernels are modified as per the algorithm given in Section III, and processed results are given in Figs. 4–6. In comparison with the processed data through the original transform kernels, mean-square error has been reduced considerably.

The algorithm described in Section II-B is used to process real data, shown in Fig. 7, photo 1, of size 128 X 128 and 8 bits per pixel. This image is transformed to 64 X 64, 8 bit data first and then retransformed to the original size by an inverse using different numbers (MM) of eigenvalues. As in one dimension, the results show that there is considerable improvement in the retransformed image with the increase in the eigenvalues used for pseudoinverse. The signal-to-noise ratios (SNR) of processed images are given in Table I. A blurred image is obtained as the compression ratio is increased. Noise is very large as the data are compressed to 0.5 bits/pixel.

An optimized picture using the Davidon, Fletcher, and Powell method [5] is given in Fig. 8, photo 3. Here the kernel used is derived from the Gaussian distribution function. 40 eigenvalues are selected first and two eigenvalues are optimized by using autocorrelation properties. Only a 4 by 4 autocorrelation matrix of a 32 X 32 data matrix is evaluated instead of the complete data. The picture obtained is reasonably good.

Pictures are processed through modified rectangular transforms having a condition number equal to unity and are shown in Fig. 8. Data of compression of the order of 7 is obtained for a reasonably good picture quality. Out of the three kernels, the one derived from the Hamming window function is the best. For comparison, the processed picture through a transform using the Hanning window function is also shown.

V. CONCLUSIONS

A rectangular transform is used to get data compression in a real image. The results show that a reasonable amount of data compression is possible with fairly simple means. Additional amounts of compression are obtained with modification of the kernels with reduced value for the condition number C.

REFERENCES

[1] A. K. Jain, "Image data compression: A review," *Proc. IEEE*, vol. 69, pp. 349–389, Mar. 1981.
[2] K. Sivaraman and K. P. Rajappan, "Data compression through a rectangular transform," in *Signal Processing*, pp. 377–383, July 1983.
[3] C. de Boor, *A Practical Guide to Splines*. New York: Springer-Verlag, 1978.
[4] A. Antoniou, *Digital Filters: Analysis and Design*. New Delhi, India: Tata McGraw-Hill, 1980.
[5] S. S. Rao, *Optimization Theory and Applications*. New Delhi, India: Wiley, 1978.

Adaptive Realizations of the Maximum Likelihood Processor for Time Delay Estimation

D. H. YOUN AND V. J. MATHEWS

Abstract—**This correspondence introduces an adaptive realization of the maximum likelihood (ML) processor for time delay estimation (TDE). Also presented is a modified ML processor, which requires less computations but still performs better than the other when implemented in an adaptive way. Widrow's least mean square (LMS) adaptive filter algorithm is used to implement the two methods. Simulation results comparing these processors with other existing adaptive TDE algorithms are also presented.**

I. INTRODUCTION

We consider the two-sensor time delay estimation (TDE) model given by

$$x_1(k) = s(k) + w_1(k) \tag{1a}$$

and

$$x_2(k) = s(k - D) + w_2(k) \tag{1b}$$

where $s(k)$ denotes the source signal; $w_1(k)$ and $w_2(k)$ are

Manuscript received June 10, 1983; revised February 24, 1984.
The authors are with the Department of Electrical and Computer Engineering, University of Iowa, Iowa City, IA 52242.

additive noises; $x_1(k)$ and $x_2(k)$ are signals received at two spatially separated sensors; D is the time delay parameter which is assumed to be an integer multiple of the sampling interval; and k is a discrete time index. It is assumed that $s(k)$, $w_1(k)$, and $w_2(k)$ are mutually uncorrelated zero-mean Gaussian random processes.

Most approaches for TDE have been shown to be related through the generalized cross-correlation (GCC) method [1]. It has been shown [1] that if the received signals are stationary and Gaussian, the maximum likelihood (ML) processor, which is one of the GCC methods, is optimum in the minimum mean squared TDE error sense. The time delay estimate $\hat{D}$ for the ML processor is given by the argument $m = \hat{D}$, which maximizes the relevant GCC function given by

$$R_{12}(m) = F^{-1}\left\{\frac{|\gamma_{12}(f)|^2}{1 - |\gamma_{12}(f)|^2} e^{j\theta_{12}(f)}\right\} \tag{2a}$$

where

$$e^{j\theta_{12}(f)} = G_{12}(f)/|G_{12}(f)| = e^{j2\pi fD} \tag{2b}$$

and $|\gamma_{12}(f)|^2$ is the magnitude-squared coherence (MSC) function of $x_1(k)$ and $x_2(k)$ [7]. In (2a)–(2b), $G_{12}(f)$ is the cross power density spectrum (cross-PDS) of $x_1(k)$ and $x_2(k)$, $G_{11}(f)$ and $G_{22}(f)$ are the auto-PDS's of $x_1(k)$ and $x_2(k)$, respectively, and $F^{-1}\{\cdot\}$ denotes the inverse Fourier transform (IFT) of $\{\cdot\}$.

II. Adaptive ML Processor for TDE

The phase functions in (2b) can be reexpressed as

$$e^{j\theta_{12}(f)} = \frac{G_{12}(f)/G_{22}(f)}{|G_{12}(f)/G_{22}(f)|} \tag{3a}$$

while

$$|\gamma_{12}(f)|^2 = \frac{G_{12}(f)}{G_{22}(f)} \cdot \frac{G_{21}(f)}{G_{11}(f)}. \tag{3b}$$

From (3a) and (3b) we can see that the adaptive estimation of $R_{12}(m)$ in (2a) boils down to the adaptive estimation of $G_{12}(f)/G_{22}(f)$ and $G_{21}(f)/G_{11}(f)$.

Now, let us define $h_{ij}(m)$ for $i, j = 1, 2$, or 2, 1 as

$$h_{ij}(m) = F^{-1}\{G_{ij}(f)/G_{jj}(f)\}, \quad |m| \leqslant M. \tag{4}$$

In this correspondence, we use Widrow's LMS algorithm [2] to estimate $h_{ij}(m)$ in a recursive manner, since the relevant time delay estimator can be used to estimate time-varying delay functions [3]–[6]. The LMS algorithm is given by

$$\hat{h}_{ij}(m, k+1) = \hat{h}_{ij}(m, k) + \mu e_j(k)\, x_i(k-m), \quad |m| \leqslant M \tag{5a}$$

where

$$e_j(k) = x_j(k) - \sum_{m=-M}^{M} \hat{h}_{ij}(m, k)\, x_i(k-m). \tag{5b}$$

In (5a), $\hat{h}_{ij}(m, k)$ denotes an estimate of $h_{ij}(m)$ at time k and μ is the convergence parameter [2]. When a time delay estimate is given by $m = \hat{D}$ where $\hat{h}_{ij}(m, k)$ is maximum, the approach has been called the LMSTDE (LMS adaptive filter for TDE) algorithm [3]–[6].

Now, from (4), taking the Fourier transform (FT) of the estimated system function with respect to m yields

$$\hat{H}_{ij}(f, k) = F\{\hat{h}_{ij}(m, k)\} = \widehat{[G_{ij}(f, k)/G_{jj}(f, k)]} \tag{6}$$

where $F\{\cdot\}$ denotes the FT of $\{\cdot\}$ and the carets represent estimated quantities. From (2a)–(6), the ML processor in (2a) can be easily implemented as follows:

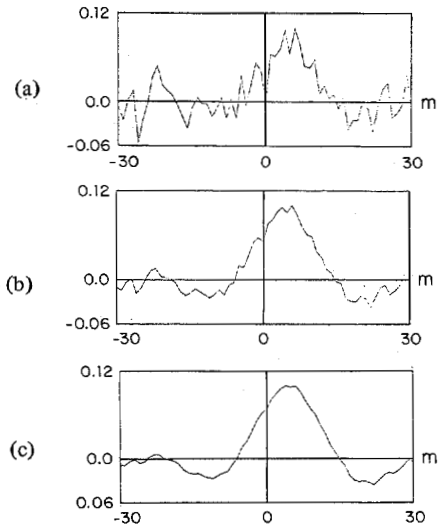

Fig. 1. (a) Estimated GCC function of the LMSTDE algorithm. (b) Estimated GCC function of the AMLTDE-1 algorithm. (c) Estimated GCC function of the AMLTDE-2 algorithm.

$$\hat{R}_{12}^{(1)}(m, k) = F^{-1}\left\{\frac{|\hat{\gamma}_{12}(f, k)|^2}{1 - |\hat{\gamma}_{12}(f, k)|^2} \frac{\hat{H}_{12}(f, k)}{|\hat{H}_{12}(f, k)|}\right\} \tag{7}$$

where $|\hat{\gamma}_{12}(f, k)|^2$ is estimated using a pair of LMS adaptive filters [7]. That is

$$|\hat{\gamma}_{12}(f, k)|^2 = |\hat{H}_{12}(f, k) \cdot \hat{H}_{21}(f, k)|. \tag{8}$$

The above adaptive realization of the ML processor will be referred to as the AMLTDE-1 algorithm.

Now, let us consider the case where $H_{12}(f) = 0$ in some frequency band (i.e., band-limited source signal). Then the phase function in (3a) is not defined in the frequency band where $H_{12}(f) = 0$ or $G_{12}(f) = 0$, and therefore the normalization of $\hat{H}_{12}(f, k)$ with its magnitude results in emphasizing the frequency band where only estimation errors exist. This suggests that, since the time delay information is contained in the phase function in (2b), we may be able to avoid the above problems by not normalizing $\hat{H}_{12}(f, k)$ at all, at the same time retaining all information on the time delay involved. Thus motivated, we define the AMLTDE-2 algorithm as

$$R_{12}^{(2)}(m, k) = F^{-1}\left\{\frac{|\hat{\gamma}_{12}(f, k)|^2}{1 - |\hat{\gamma}_{12}(f, k)|^2} \cdot \hat{H}_{12}(f, k)\right\}. \tag{9}$$

III. Simulation Results

The source signal $s(k)$ was generated by processing a white Gaussian random signal through a 10th-order Butterworth low-pass filter with the cutoff frequency of 100 Hz and the sampling frequency of 2000 Hz. Also, the signal-to-noise ratio of the received signals is given by 1/16, while the delay parameter is $D = 4$ (samples).

In Fig. 1(a), (b), and (c), typical GCC function estimates $\hat{h}_{12}(m, k)$, $\hat{R}_{12}^{(1)}(m, k)$, and $\hat{R}_{12}^{(2)}(m, k)$ for $|m| \leqslant 30$, $\mu = 0.0002$, and $k = 6000$ are displayed. Also, the relevant frequency domain weighting functions and $|\hat{\gamma}_{12}(f, 6000)|^2$, averaged over the 20 trials, are presented in Fig. 2(a)–(d), where 128 points FFT's were used to take the Fourier and inverse Fourier transforms. The mean and variance of the time delay estimates obtained from the 20 sample runs for the three methods (i.e., AMLTDE-1, AMLTDE-2, and LMSTDE) are tabulated in Table I.

From Fig. 1(a)–(c) and Table I, we can observe that the AMLTDE-2 algorithm yields the least noisy GCC function estimate while the AMLTDE-1 algorithm gives less noisy estimates

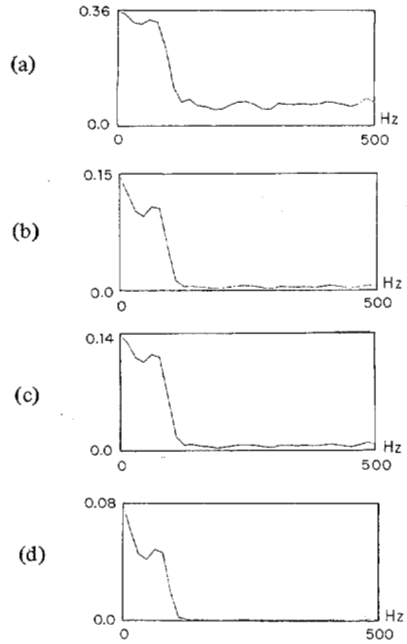

Fig. 2. (a) Averaged weighting function estimate of the LMSTDE algorithm. (b) Averaged MSC function estimate. (c) Averaged weighting function estimate of the AMLTDE-1 algorithm. (d) Averaged weighting function estimate of the AMLTDE-2 algorithm.

TABLE I
MEAN AND VARIANCE OF THE 20 INDEPENDENT TRIALS FOR $D = 4$ (SAMPLES)

Method	Mean, $\bar{D}$	Variance w.r.t. $\bar{D}$, $\sigma_{\bar{D}}^2$
LMSTDE	4.5	2.75
AMLTDE-1	4.55	2.15
AMLTDE-2	3.9	0.69

than the LMSTDE method. These observations may be explained with the results in Fig. 2(a)–(d). Ideally, the weighting functions in the frequency band (100–1000 Hz) where the source signal does not exist should be 0. Weighting function estimation errors in this frequency band introduce GCC function estimation errors. Comparing the amplitudes of the estimated weighting functions for the three processors, we can see that estimation errors in this band are the smallest for the AMLTDE-2 algorithm and the largest for the LMSTDE method.

IV Conclusions

An adaptive implementation of the ML processor for TDE was presented. Also presented was a modified ML processor, which requires less computations, but still performs better than the direct implementation.

The superiority of the AMLTDE-1 and -2 algorithms over the LMSTDE method for band-limited signals was demonstrated through simulation results. Simulation results, with different types of input signals, which were not presented in this correspondence, gave results comparable to the case presented here. In general, between AMLTDE-1 and -2, the latter processor is to be preferred because of its superior performance and computational simplicity.

References

[1] C. H. Knapp and G. C. Carter, "The generalized correlation method for estimation of time delay," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-24, pp. 320–327, Aug. 1976.

[2] B. Widrow *et al.*, "Adaptive noise cancelling: Principles and applications," *Proc. IEEE*, vol. 63, pp. 1692–1716, Dec. 1975.

[3] P. L. Feintuch, N. J. Bershad, and F. A. Reed, "Time delay estimation using the LMS adaptive filter–Dynamic behavior," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-29, pp. 571–576, June 1981.

[4] F. A. Reed, P. L. Feintuch, and N. J. Bershad, "Time delay estimation using the LMS adaptive filter–Static behavior," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-29, pp. 561–571, June 1981.

[5] D. H. Youn, N. Ahmed, and G. C. Carter, "On using the LMS algorithm for time delay estimation," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-30, pp. 798–801, Oct. 1982.

[6] ——, "An adaptive approach for time delay estimation of band-limited signals," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-31, pp. 780–784, June 1983.

[7] ——, "Magnitude-squared coherence function estimation: An adaptive approach," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-31, pp. 137–142, Feb. 1983.