205

COMPARISON OF TWO ALGORITHMS FOR TIME DELAY ESTIMATION

by

Sangil Park

B.E., Yonsei University, Korea, 1977

A MASTER'S THESIS

submitted in partial fulfillment of the

requirements for the degree of

MASTER OF SCIENCE

ELECTRICAL ENGINEERING

COLLEGE OF ENGINEERING

KANSAS STATE UNIVERSITY

MANHATTAN, KANSAS

1983

Approved by:

Nasir Ahmed

Major Professor

THIS BOOK CONTAINS NUMEROUS PAGES WITH THE ORIGINAL PRINTING BEING SKEWED DIFFERENTLY FROM THE TOP OF THE PAGE TO THE BOTTOM.

THIS IS AS RECEIVED FROM THE CUSTOMER.

ILLEGIBLE

THE FOLLOWING DOCUMENT (S) IS ILLEGIBLE DUE TO THE PRINTING ON THE ORIGINAL BEING CUT OFF

ILLEGIBLE

TABLE OF CONTENTS

CHAPTER Page

I. INTRODUCTION. 1

II. SOME THEORETICAL ASPECTS. 2

A. LMS algorithm 5

B. ACS algorithm 7

III. SIMULATION DETAILS. 10

IV. SIMULATION RESULTS. 13

A. Fixed Source. 13

B. Moving Source 15

V. CONCLUSION. 18

REFERENCES. 19

APPENDIX

A. Method for Interpolation. 21

B. Method for Generating Time-Varying Singal 25

C. Computer programs 29

ACKNOWLEDGEMENTS

THIS BOOK WAS BOUND WITHOUT PAGE 1 "INTRODUCTION".

THIS IS AS RECEIVED FROM CUSTOMER.

II. SOME THEORETICAL ASPECTS

Consider a simple two-sensor model for estimating time delay, which is

$$x(k) = s(k) + n_1(k)$$
$$y(k) = s[k - D(k)] + n_2(k) \quad (1)$$

where k is the time index; x(k) and y(k) denote the sensor outputs; s(k) is the source signal; $n_1(k)$ and $n_2(k)$ are zero mean, Gaussian and mutually uncorrelated random variables; and D(k) is the time-varying delay function related to the two sensors.

For time delay estimation applications, these two sensor outputs become the inputs to an adaptive time domain filter, as shown in Fig. 1. The time-varying delay function D(k) can be estimated by finding the location of the peak of the adaptive filter coefficient vector [4], i.e., the location of the peak value of the filter impulse response.

In practice the inputs are sampled. Thus the adaptive filter is discrete in time, and has finite length. Thus the peak of the impulse response at a given iteration is determined by interpolating between discrete-time sample points to provide the delay values which are non-integer multiples of the sampling rate [8]. Thus we have

$$h_I(n,k) = \sum_{m=-P_3}^{P_3} \hat{h}(m,k) \frac{\sin 2\pi\beta(n-m)}{2\pi\beta(n-m)} \quad (2)$$

where β denotes normalized bandwidth of the signal and $h_I(n,k)$ denotes the smoothed impulse response value corresponding to the estimated value $\hat{h}(m,k)$ at time n. For adaptive time-varying delay estimation, the location of the peak of $h_I(n,k)$ yields $\hat{D}(k)$.

For simulation purposes we need the time-varying delay signal s[k-D(k)]. It has been shown that this can be obtained via a bank of

THIS BOOK CONTAINS NUMEROUS PAGES WITH DIAGRAMS THAT ARE CROOKED COMPARED TO THE REST OF THE INFORMATION ON THE PAGE.

THIS IS AS RECEIVED FROM CUSTOMER.

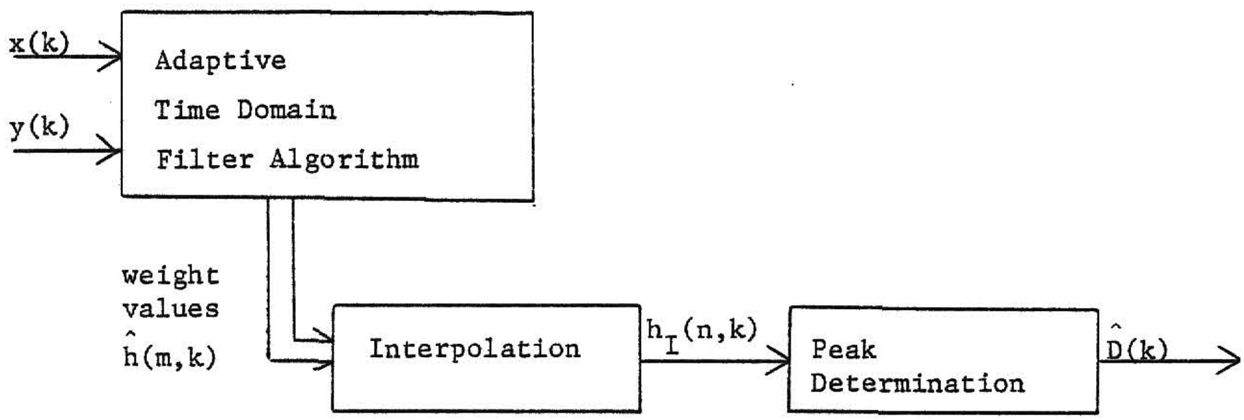

Fig. 1. A block diagram for adaptive time delay estimation.

time-invariant finite impulse response FIR filters, whose k-th filter has the transfer function [7]

$$H(\omega,k) = e^{-j\omega D(k)} \tag{3}$$

where ω is the radian frequency with $|\omega| < \pi$. Using (3) and truncating the filter length, we obtain

$$s[k-D(k)] = \sum_{m=P_1}^{P_1} g(m,k)\ s(k-m) \tag{4}$$

where $g(m,k) \triangleq F^{-1}\{H(\omega,k)\} = \text{sinc}\,[m-D(k)]$, $F^{-1}\{\cdot\}$ denotes the inverse Fourier transform, $\text{sinc}\,(\cdot) = \frac{\sin\pi(\cdot)}{\pi(\cdot)}$, and $(2p_1 + 1)$ is the total number of coefficients $g(m,k)$. Details of these interpolation and time-varying delayed signal are discussed in Appendix.

A. LMS Algorithm

Consider a two-element adaptive filter model which uses a tapped-delay-line structure to provide the adaptive weight adjustment, as shown in Fig. 2. If $H(k)$ is the column vector of filter weights $h(i,j)$, $R_{xx}(k)$ is the correlation matrix of input signals at time k, and $R_{xy}(k)$ is the cross correlation vector between the received reference signal $x(k)$ and the primary signal $y(k)$, then the optimum filter weight vector that minimizes the expected value of $e^2(k)$ in Fig. 2 is given by [1]

$$H(k) = R_{xx}^{-1}(k)R_{xy}(k). \tag{5}$$

Where, $H^T(k) = [h(-p_2,k)\ldots h(0,k)\ldots h(p_2,k)]$.

Thus $H(k)$ is a $2p_2+1$ vector consisting of the filter weights at time k.

The error output, $e(k)$ is then

$$e(k) = y(k) - H^T(k)X(k) \tag{6}$$

where $X(k)$ is input data vector given by

$$X^T(k) = [x(k)\ x(k-1)\ldots\ldots x(k-2p_2)]$$

The LMS algorithm which is an implementation of the steepest descent method [1] updates the weight vector at each k via the relation

$$H(k+1) = H(k) + \mu e(k)X(k) \tag{7}$$

where μ is a parameter that controls the rate of convergence of the algorithm.

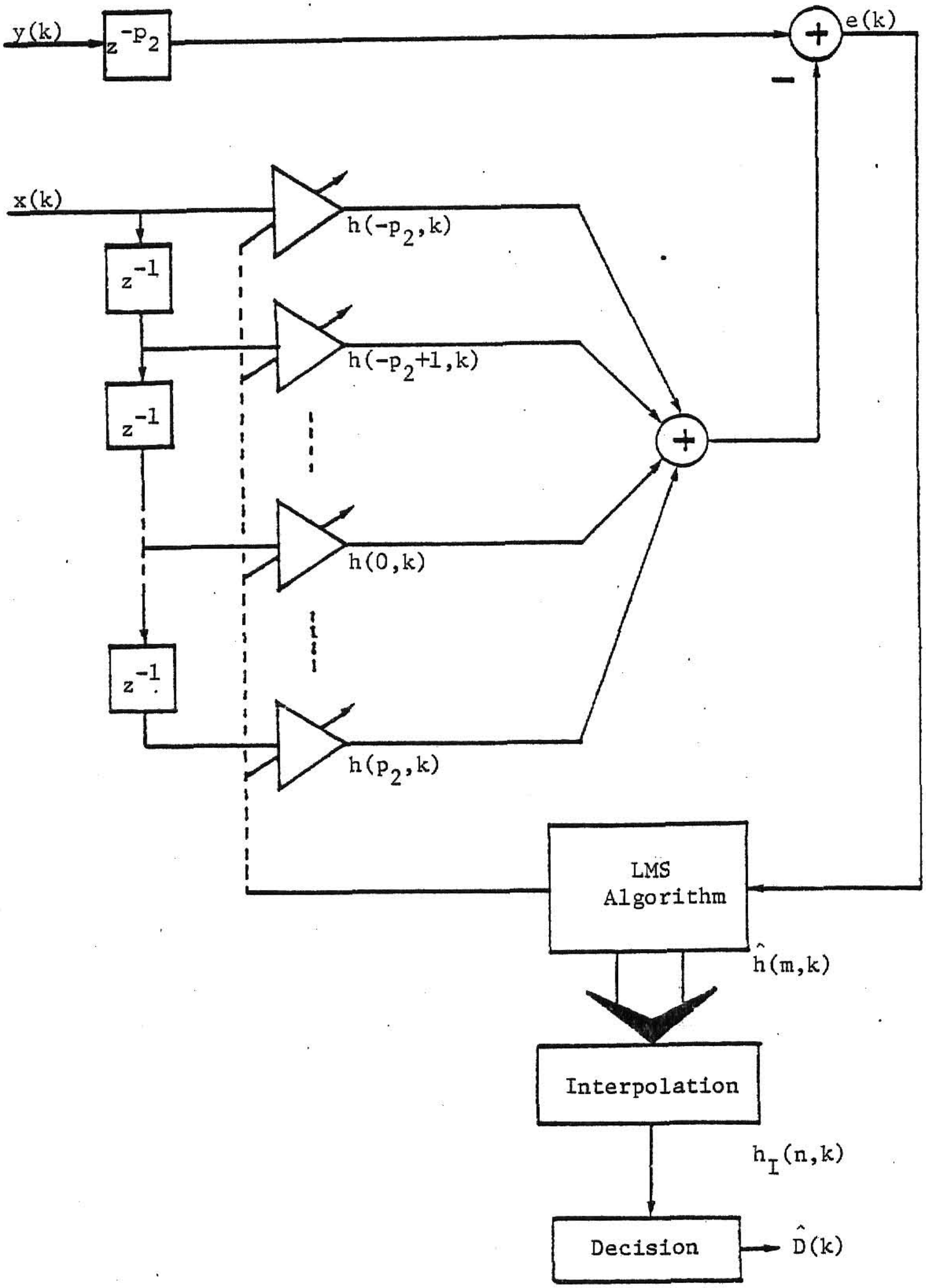

Fig. 2: Structure of the LMS algorithm for time delay estimation

B. ASC Algorithm

Consider the real-valued correlated data sequences x(k) and y(k), where k is the time index. The cross-correlation $R_{xy}(m)$ of x(k) and y(k) is

$$R_{xy}(m) = E\,[x(k)\;y(k-m)]$$
$$= E\,[s(k)\;s(k+D-m)] + E\,[n_1(k)n_2(k-m)] \qquad (8)$$

where E denotes statistically expectation. Since $n_1(k)$ and $n_2(k)$ are independent and zero mean noise sequences, we can show that

$$R_{xy}(m) = R_{ss}(m-D).$$

The auto-correlation, R_{ss}, has the property that [10]

$$R_{ss}(0) \geq |R_{ss}(m)| \;.$$

So R_{xy} will attain a maximum value at m=D.

If x(k) and y(k) are known only over a finite length of time, the estimation of $R_{xy}(m)$ is [5, 10]

$$R_{xy}(m) = \frac{1}{N-m} \sum_{k=0}^{N-m} x(k)\;y(k+m) \quad , \qquad |m| \leq M \qquad (9)$$

where m is the shift index, $R_{xy}(m)$ is the desired estimate, M is the maximum shift and N is the number of points in x(k) and y(k).

The basic idea is to estimate $R_{xy}(m,k)$ using the ASC algorithm which is defined as [5,11]

$$R_{xy}(m,k) = \beta R_{xy}(m,k-1) + (1-\beta)\;x(k)\;y(k-m)$$
$$= \beta[R_{xy}(m,k-1) - x(k)\;y(k-m)] + x(k)y(k-m) \qquad (10)$$

where $0 < \beta < 1$ is a smoothing parameter which determines the convergence rate.

The recursive equation in (10) can be modeled using the structure shown in Fig. 3. It consists of a bank of first-order infinite impulse response (IIR) filters. From (10) we can obtain the transfer function which related x(k), y(k-m) and $R_{xy}(m,k)$ as

$$H(Z) = \frac{1-\beta}{1-\beta z^{-1}}$$

If the unit of time is considered to be the iteration cycle, the related time constant which represents the number of data samples require to attain its steady-state value for each k is given by [5]

$$\tau \simeq \frac{1}{1-\beta} \text{ samples.}$$

If x(k) and y(k) are jointly stationary Gaussian processes, the ASC algorithm is unbiased [5,10], i.e.,

$$\lim_{k\to\infty} E[\hat{R}_{xy}(k,m)] = R_{xy}(m). \tag{11a}$$

And it has been also shown that

$$\lim_{k\to\infty} \left\{\sigma^2_{\hat{R}_{xy}}(k,m)\right\} = \frac{1-\beta}{2}\left[\hat{R}_{xx}(0)\hat{R}_{yy}(0) + R_{xy}^2(m)\right] \tag{11b}$$

From (11b) it is apparent that the related variance goes zero as β close to 1 [5].

The ASC algorithm can be used for time display estimation as depicted in Fig. 4. This is because the time delay function estimate D(k) can be easily determined from the peak value of the impulse response function $\hat{h}(m,k)$ [4].

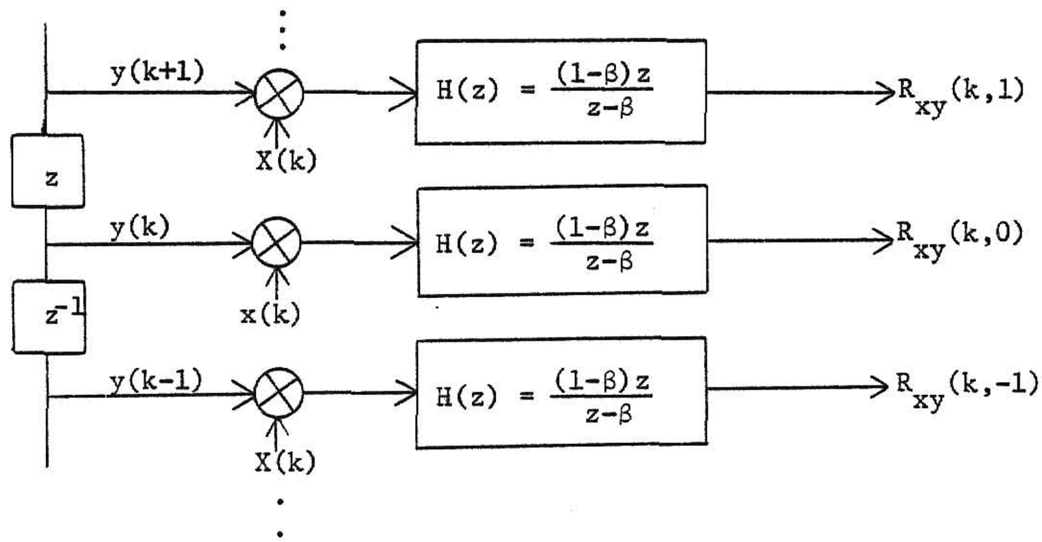

Fig. 3. Structure of the ASC algorithm

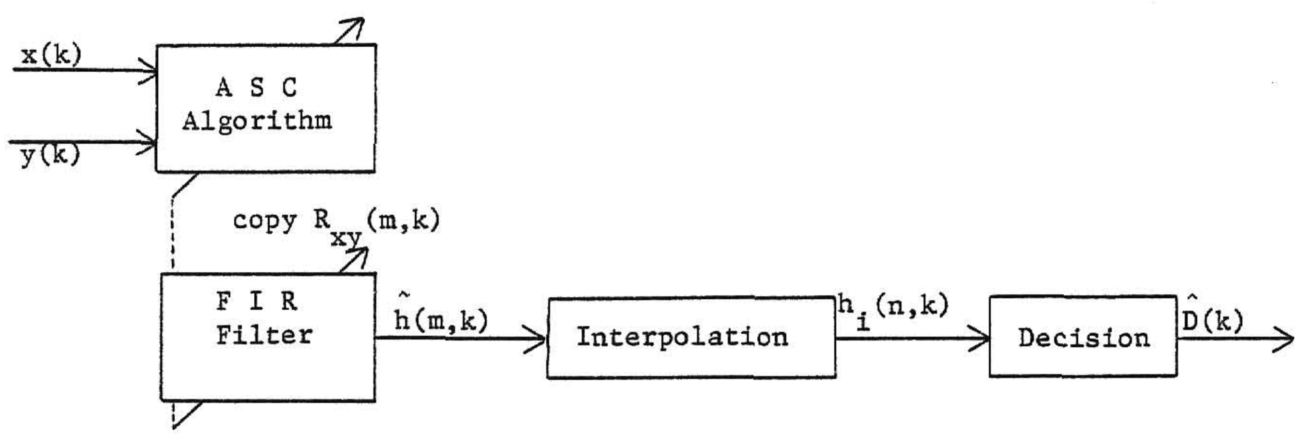

Fig. 4. Block diagram of the ASC algorithm for Time Delay Estimation.

III. SIMULATION DETAILS

The entire computer simulation considered the two sensor model in (1). The block diagram to generate the pertinent signals is depicted in Fig. 5, where $n_0(k)$, $n_1(k)$ and $n_2(k)$ are mutually uncorrelated real white Gaussian noise sequences. The sampling frequency was 2048 samples per second (sps). Two of these sequences were used to represent the additive noises. The first noise sequence, $n_0(k)$, was processed through a pair of tenth-order and second-order Butterworth digital filters to generate the low-passed and the band-passed source signals, respectively which had the following characteristics:

$$|H_1(f)| = \begin{cases} 1, & 0 \leq f \leq 100 \text{ Hz} \\ 0, & \text{elsewhere} \end{cases}$$

and

$$|H_2(f)| = \begin{cases} 1, & 100 \leq f \leq 200 \text{ Hz} \\ 0, & \text{elsewhere} \end{cases}$$

The noise sequences $n_1(k)$ and $n_2(k)$ were added to the resulting low-passed and band-passed signals which were used as input data sequences $x(k)$ and $y(k)$ after having been scaled to realize different signal-to-noise ratios (SNRs). In this simulation SNRs of -3dB, -10dB, and -13dB were considered. Note that the SNR is estimated as the ratio of the total power (variance when mean is zero) of the signal to that of the noise.

In Fig. 5, $D(k)$ is the time-varying delay parameter which has a constant or a linearly varying value. A delayed signal $s[k-D(k)]$ was generated using (4) by truncating the filter length to 61 (i.e., p_1=30) to obtain

$$s[k-D(k)] = \sum_{m=-30}^{30} \text{sinc}\,[m-D(k)]\, s(k-m) \tag{12}$$

The various parameters pertaining to the simulation are summarized in

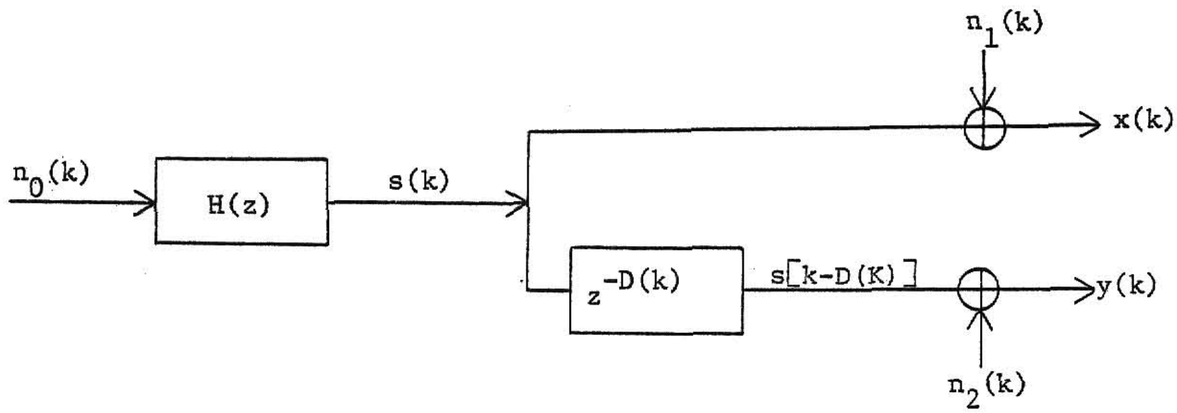

Fig. 5. Block diagram to generate signals

Table 1, where $(2p_1 + 1)$ is the number of filter weights for generating time-varying delayed signal in (4); $(2p_2 + 1)$ is the number of weights used for estimating time delay using the LMS and ASC algorithms; and $(2p_3 + 1)$ denotes the number of interpolation points to estimate the delay which is a non-integer multiple of sampling interval; see (2).

The input data sequences, x(k) and y(k), were processed using the LMS and ASC algorithms. A total of 24 trials were used in each case. Each trial employed 3.9 sec. (8000 data points) to estimate the time delay parameter, D(k), which had constant or a linearly varying value.

	Fixed Source						Moving Source					
Signal	Low Pass			Band Pass			Low Pass			Band Pass		
H(z)	10th order LPF $f_c = 100Hz$ (cutoff frequency)			2nd order BPF $f_1 = 100Hz$ $f_h = 200Hz$			10th order LPF $f_c = 100Hz$			2nd order BPF $f_1 = 100Hz$ $f_h = 200Hz$		
D(k)	12			12			-4 + 0.001 k			-4 + 0.001 k		
(S/N)dB	-3	-10	-13	-3	-10	-13	-3	-10	-13	-3	-10	-13
σ_s^2	0.5	0.1	0.05	0.5	0.1	0.05	0.5	0.1	0.05	0.5	0.1	0.05
$\sigma_{n_1}^2 = \sigma_{n_2}^2$	1	1	1	1	1	1	1	1	1	1	1	1
$\beta = 1 - \mu$	0.9999			0.9999			0.999			0.999		
$2p_1 + 1$	∞			∞			61			61		
$2p_2 + 1$	61			61			21			21		
$2p_3 + 1$	10			10			5			5		

Table 1. Parameters pertaining to the simulation.

IV. SIMULATION RESULTS

A. Fixed Source

Here the time delay function D(k) = 12. A total of 8000 data points were used, and the related parameters that were used are listed in Table 1. The means and variances associated with the delay estimates were obtained using 24 trials from 100 resulting measurements of each trial as follows:

$$\overline{D}_i = \frac{1}{100} \sum_{k=1}^{100} \hat{D}_i\ (50k + 3001)$$

and

$$\sigma^2_{\overline{D}_i} = \frac{1}{100} \sum_{k=1}^{100} [\hat{D}_i\ (50k + 3001) - \overline{D}_i]^2 \tag{13}$$

for the i-th trial, with

$$\overline{D} = \frac{1}{24} \sum_{i=1}^{24} \overline{D}_i$$

and

$$\sigma^2_{\overline{D}} = \frac{1}{24} \sum_{i=1}^{24} (\overline{D}_i - \overline{D})^2 \tag{14}$$

where $\hat{D}_i(k)$ denotes the estimated time delay at k-th iteration of the i-th trial. The corresponding results are summarized in Table 2.

Examination of Table 2 reveals that the mean values of the delay estimates compare very closely for the LMS and ASC algorithms for low-pass and band-pass cases, at each of the three SNRs. However, we observe that the variance of the delay estimate is substantially less for the ASC method, especially for band-pass signals.

SIGNAL		LOW PASS			BAND PASS		
SNR		-3dB	-10dB	-13dB	-3dB	-10dB	-13dB
M E A N	LMS	11.9691	11.9349	11.8743	12.0255	12.0414	11.8744
	ASC	11.9528	11.9060	11.8601	12.0074	12.0092	12.0226
VARIANCE	LMS	0.07629	0.26559	0.69601	0.01249	0.03147	2.70762
	ASC	0.02804	0.18045	0.50993	0.00368	0.01795	0.13169

Table 2. Simulation results for fixed source.

B. Moving Source

Linearly-varying time delay function $D(k) = -4 + 0.001\,k$. The time delay function $D(k)$ was increased linearly from -4 to 3.8 over 8000 data points. Other parameters used for the simulation are listed in Table 1. The desired delay estimate is given by the value of lag which yields the peak value of the impulse response $\hat{h}(m,k)$; see Fig. 1. This aspect is illustrated in Fig. 6 which shows the impulse response function $\hat{h}(m,k)$ and its smoothed function $h_I(n,k)$ by interpolation at 3000, 5000 and 7000 iterations at a SNR of -3dB and a delay function varying at a 0.001 unit/iteration. It is instructive to observe that the peak values of the impulse response functions adapt well with time.

The time-varying delay function $D(k)$ was estimated every 50 sample points from k=3001 to k=8000 for 24 trials. The bias which denotes the time difference between true delay and estimated delay was considered. The bias increases with increases in the changes of $D(k)$, and decreases as the smoothing parameter β is decreased [9]. The bias and its variance were computed as follows:

$$\overline{B}_i = \frac{1}{100} \sum_{k=1}^{100} \left(D(50k + 3001) - \hat{D}_i(50k + 3001)\right)$$

for i-th trial, with

$$\overline{B} = \frac{1}{24} \sum_{i=1}^{24} \overline{B}_i$$

$$\sigma_{\overline{B}} = \frac{1}{24} \sum_{i=1}^{24} (\overline{B}_i - \overline{B})^2$$

where $D(k)$ and $\hat{D}_i(k)$ denote true and estimated delay, respectively. The results for the case of SNR -3, -10, and -13db are summarized in Table 3.

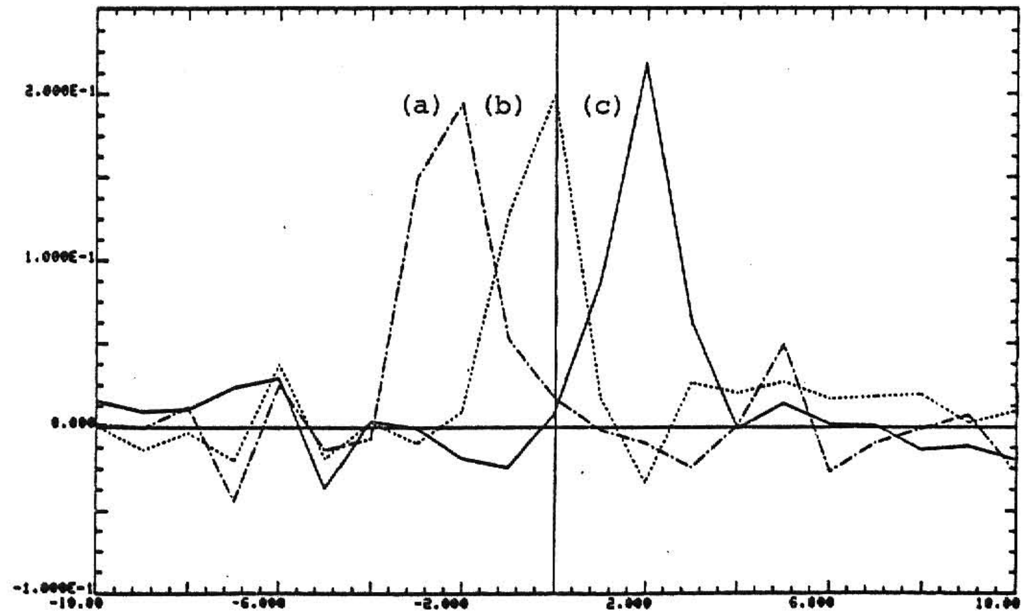

Fig. 6-1. $\hat{h}(m,k)$ without interpolation with 61 weights;

(a) k=3000, (b) k=5000, and (c) k=7000.

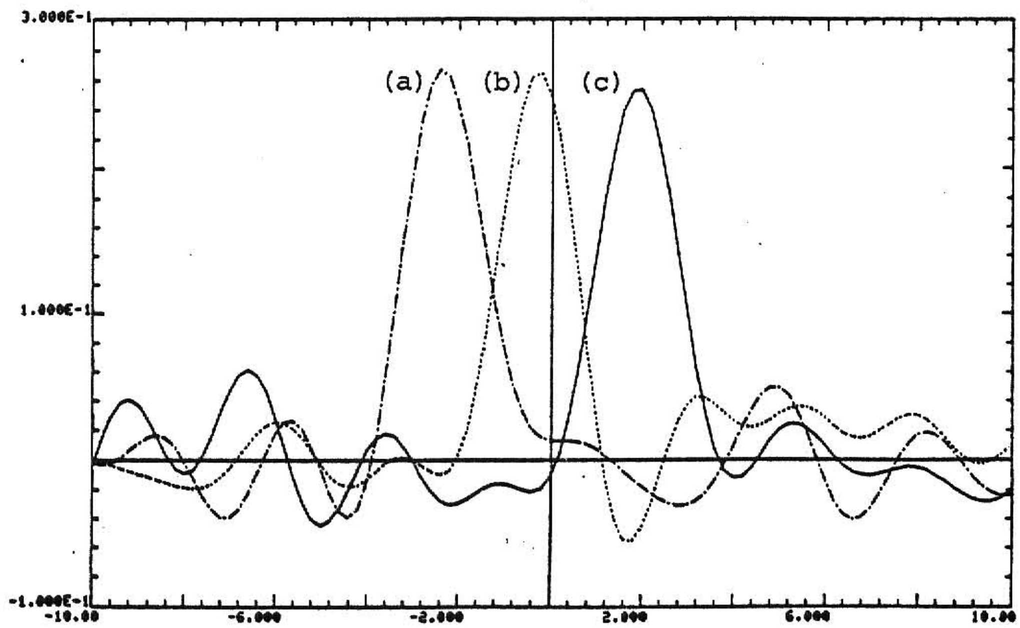

Fig. 6-2. $h_I(n,k)$ with 5 point interpolation with 61 weights;

(a) k=3000, (b) k=5000, and (c) k=7000.

From Table 3, it is apparent that the LMS method is a smaller bias for the delay estimate, especially at -3dB. This difference in bias becomes less significant as the SNR is decreased further. On the other hand, the ASC method yields a variance (of the delay estimate) that is consistently better than that obtained via the LMS method.

SIGNAL		LOW PASS			BAND PASS		
S N R		-3dB	-10dB	-13dB	-3dB	-10dB	-13dB
B I A S	LMS	-0.3234	-0.6159	-0.8266	-0.3041	-0.5623	-0.7762
	ASC	-0.9557	-0.9946	-1.0428	-0.8826	-0.8875	-0.9074
VARIANCE	LMS	0.24094	0.82397	2.14786	0.04479	0.13826	1.40289
	ASC	0.08923	0.47559	1.51040	0.02188	0.08932	0.49422

Table 3. Simulation results for moving source.

V. CONCLUSIONS

We have compared two adaptive methods for estimating time delay functions between a pair of sensor outputs. The ASC algorithm was found to significantly reduce the variance of delay estimate and the computation time for all the cases that were considered. It is straight foreward to determine the total number of arithmetic operations (multiplications and additions) required by LMS and ASC algorithms are 6N and 4N, respectively, at every iteration, where N is the number of weights. Also, the ASC algorithm gave better results for the fixed source case when the signals were band-passed. However, when the source is moving, the bias between the actual and estimated time delays is more than for the LMS algorithm. The main advantages of the ASC algorithm are its computational simplicity and the fact that it consistently yields time delay estimates which have a smaller variance.

REFERENCES

[1] B. Widrow, et al, "Adaptive noise cancelling: Principles and application," Proc. IEEE, Vol. 63, No. 12, pp. 1692-1716, Dec. 1975.

[2] L. J. Griffiths, "A continuously-adaptive filter implemented as a lattice structure," Proc. IEEE Int. Conf. Acoust., Speech, Signal Proc., Hartford, CT, pp. 683-686, May 1977.

[3] N. Ahmed and D. H. Youn, "On a realization and related algorithm for adaptive prediction," IEEE Trans. Acoust., Speech, and Signal Procl, Vol. ASSP-28, No. 5, pp. 493-497, Oct. 1980.

[4] D. H. Youn, "A class of adaptive methods for estimating coherence and time delay functions," Kansas State University, Ph.D. Thesis, 1982.

[5] N. Ahmed and S. Vijayendra, "On an adaptive signal processing algorithm," Midcon/82, Session 3-1, Dallas, Texas, Nov. 30-Dec. 2, 1982.

[6] C. H. Knapp and G. C. Carter, "The generalized correlation method for estimation of time delay," IEEE Trans. Acoust., Speech, and Signal Proc., Vol. ASSP-24, pp. 320-327, August 1976.

[7] D. H. Youn, N. Ahmed, and G. C. Carter, "On using the LMS algorithm for time delay estimation," IEEE Trans. Acoust., Speech, and Signal Proc., Vol. ASSP-30, No. 5, pp. 798-801, Oct. 1982.

[8] F. A. Reed, P. L. Feintuch, and N. J. Bershad, "Time delay estimation using the LMS adaptive filter-static behavior," IEEE Trans. Acoust., Speech, and Signal Proc., Vol. ASSP-29, No. 3, pp. 561-571, June 1981.

[9] P. L. Feintuch, N. J. Bershad, and F. A. Reed, "Time delay estimation using the LMS adaptive filter-dynamic behavior," IEEE Trans. Acoust., Speech, and Signal Proc., Vol. ASSP-29, No. 3, pp. 571-576, June 1981.

[10] D. L. Johnstone and O. L. Frost, "High resolution differential time of arrival and differential doppler estimation," Tech. Report, ARGO System, Inc., Sunnyvale, CA 94086, Jan. 1977.

[11] N. Ahmed and S. Vijayendra, "An algorithm for line enhancement," Proc. IEEE, Vol. 70, No. 12, pp. 1459-1460, Dec. 1982.

APPENDIX

A. A Method for Interpolation [A1, A2]

Restriction on the sample spacing arises if we want to reconstruct the continuous signal x(t) for all time t from its sampled value $x(kT_s)$ for k = 0, 1, 2... Discrete signals which vary slowly enough as functions of the time can be reconstructed uniquely from their samples taken at intervals of T_s. The sampling theorem states that x(t) can be reconstructed precisely for all t is the sampling rate is $f_s = \frac{1}{T_s}$ is at least twice the signal bandwidth B. Such signals are said to be band-limited to B Hz which is less than $\frac{1}{2T_s}$.

The continuous signal x(t) can be obtained by passing its sampled values $x(kT_s)$ through an ideal low-pass filter whose bandwidth is B Hz. A general formula for the interpolated signal can be derived by carrying out the operation depicted in Fig. A-1 where $\bar{x}(t)$ denotes the sampled signal corresponding to the sampled values $x(kT_s)$ -- i.e., [A1]

$$\bar{x}(t) = \sum_{m=-\infty}^{\infty} x(mT_s)\ \delta(t - mT_s) \tag{A.1}$$

where $\delta(t)$ denotes the impulse function.

If $H(\omega)$ is the transfer function of the ideal filter, then

$$\hat{X}(\omega) = H(\omega)\bar{X}(\omega) \tag{A.2}$$

where $\bar{X}(\omega)$ denotes the Fourier transform of $\bar{x}(t)$.

In the time domain, $\hat{x}(t)$ is given by

$$\hat{x}(t) = \int_{-\infty}^{\infty} h(a)\bar{x}(t-a)\ da \tag{A.3}$$

where

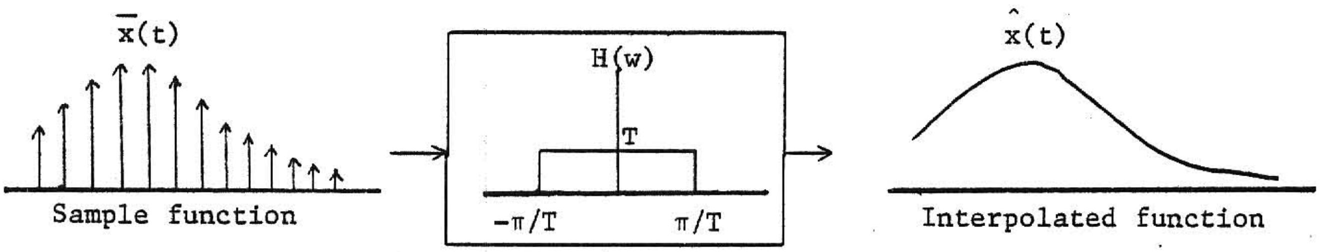

Fig. A-1. An interpolation scheme

$$h(t) = F^{-1}\{H(\omega)\}$$

$$= \int_{-\infty}^{\infty} H(\omega) e^{j\omega t} d\left(\frac{\omega}{2\pi}\right)$$

$$= \frac{T}{2\pi} \int_{-\pi/T}^{\pi/T} e^{j\omega t} \, d\omega$$

$$= \frac{\sin\left(\frac{\pi t}{T}\right)}{\left(\frac{\pi t}{T}\right)} \tag{A.4}$$

Using (A.1) and (A.4) we can obtain

$$\hat{x}(t) = \int_{-\infty}^{\infty} h(a) \sum_{m=-\infty}^{\infty} x(m) \, \delta(t - a - mT_s) \, da$$

$$= \sum_{m=-\infty}^{\infty} x(mT_s) \, h(t - mT_s)$$

$$= \sum_{m=-\infty}^{\infty} x(mT_s) \frac{\sin \pi(t - mT_s)/T}{\pi(t - mT_s)/T} \tag{A.5}$$

which is the desired result.

REFERENCES

[A1] M. Schwartz and L. Shaw, Signal Processing: Discrete special analysis, detection, and estimation, Wiley, 1975.

[A2] S. Stearns, Digital Signal Analysis, Hayden, 1975.

B. A Method for Generating Time-Delayed Signal [B1, B2]

Consider the signal $s(k)$ and its constant delayed version $s(k - D)$. We can obtain $s(k - D)$ by passing $s(k)$ through a time-invariant filter whose transfer function is $H(\omega)$ as

$$F\{s(k - D)\} = S(\omega)H(\omega) \tag{B.1}$$

where $F\{\cdot\}$ denotes the Fourier transform, and w is the radian frequency with $|\omega| < \pi$.

The time shift property of Fourier transfer states that

$$F\{s(k - D)\} = s(\omega)e^{-j\omega D} \tag{B.2}$$

Thus (B.1) and (B.2) yield

$$H(\omega) = e^{-j\omega D} \tag{B.3}$$

which implies that the impulse response function of $H(\omega)$ is

$$\begin{aligned} h(k) &= F^{-1}\{H(\omega)\} \\ &= F^{-1}\{e^{-j\omega D}\} \\ &= \text{sinc }(k - D), \quad |k| < \infty \end{aligned} \tag{B.4}$$

where

$$\text{sinc}(\cdot) \triangleq \frac{\sin \pi(\cdot)}{\pi(\cdot)} .$$

Thus, $s(k - D)$ can be obtained by the filter which has infinite number of weights as

$$s(k - D) = \sum_{m=-\infty}^{\infty} \text{sinc}(m - D)\, s(k - m) \tag{B.5}$$

However, in a practical situation the filter length has to be finite. Since the function $s(m - D)$ in (B.5) approaches zero as $|m|$ increases, truncation can be carried out to obtain

$$\hat{s}(k - D) = \sum_{m=-P}^{P} \text{sinc}(m - D)\, s(k - m) \tag{B.6}$$

Now considering the case when the delay function is time-varying. The the desired delayed signal can be obtained via a bank of time-invariant filters as shown in Fig. B-1. If the time-varying delay function is D(k), then the n-th filter has the transfer function

$$H(\omega,n) = e^{-j\omega D(n)}.$$

From (B.5) we have

$$s[k - D(n)] = \sum_{m=-\infty}^{\infty} h(m,n)\, s(k - m)$$

$$= \sum_{m=-\infty}^{\infty} \mathrm{sinc}[m - D(n)]\, s(k - m) \qquad (B.7)$$

The desired delay signal is obtained by sampled values of $s[k - D(n)]$ at $k = n$ for $|n| < \infty$; see Fig. B-1.

Substituting n=k in (B.7) and using finite filter weights, we obtain

$$\hat{s}[k - D(k) = \sum_{m=-P}^{P} \mathrm{sinc}\left(m - D(k)\right)\, s(k - m) \qquad (B.8)$$

From (B.8), $\mathrm{sinc}\left(m - D(k)\right)$, $|m| < p$ can be interpreted as a weighting function whose maximum value occurs at $m = D(k)$.

Therefore $\hat{s}[k - D(k)]$ in (B.8) can be viewed as being the output of the time-varying filter which is generating a time-varying delay signal. Details of this truncation process are discussed in [B2].

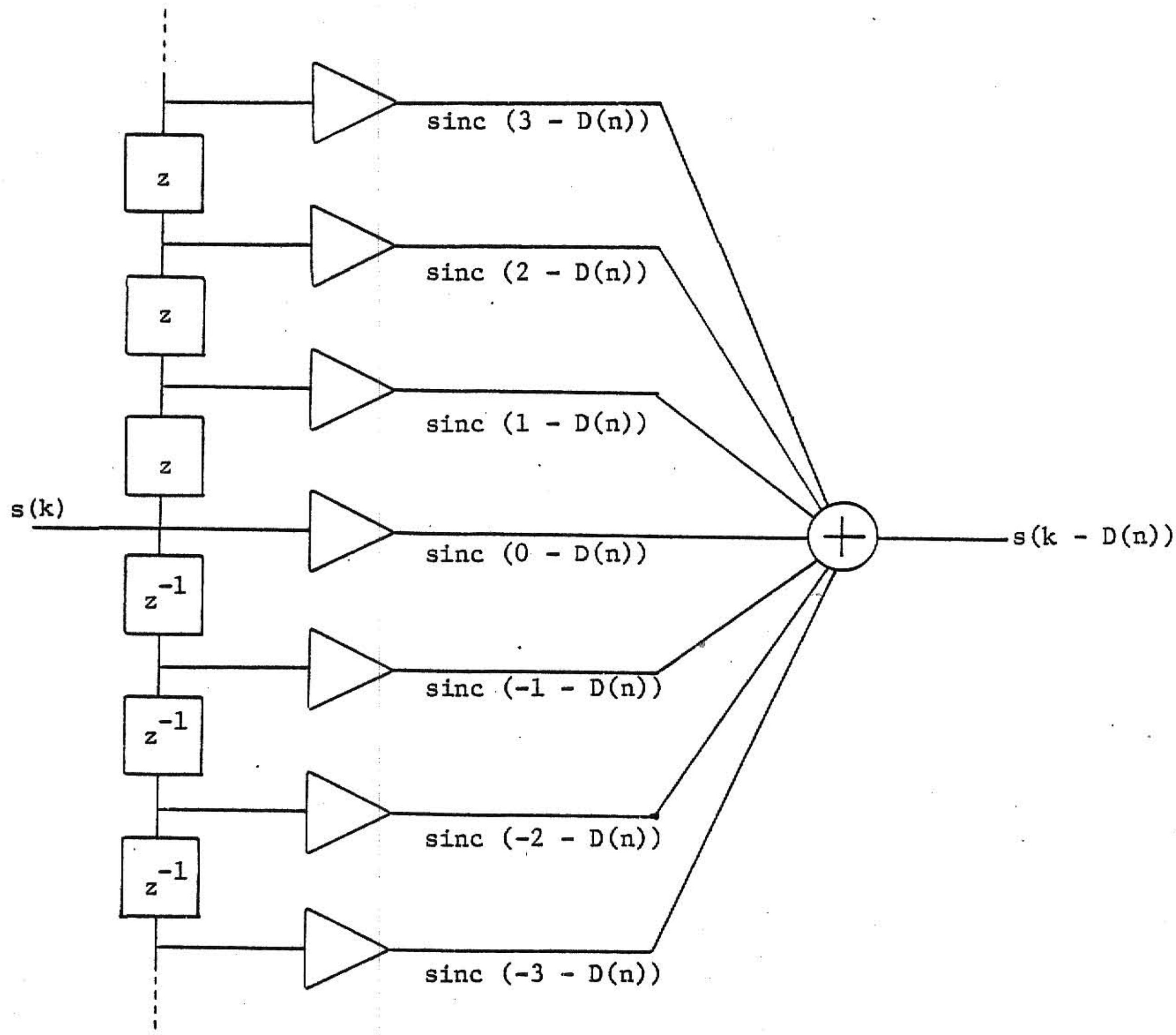

Fig. B-1. Structure for generating delayed signal when the delay is time varying.

REFERENCES

[B1] D. Youn, N. Ahmed, and G. C. Carter, "A method for generating a class of time-delayed signals," in Proc. ICASSP, Atlanta, GA, Mar. 1981, pp. 1257-1260.

[B2] D. Youn, N. Ahmed, and G. C. Carter, "On Using the LMS algorithm for time delay estimation," IEEE Trans. Acoust., Speech and Signal Proc., Vol. ASSP-30, No. 5, pp. 788-801, Oct. 1982.

C. Computer Programs

1. Adaptive least mean square (LMS) filter routine

2. Adaptive short term correlator (ASC) filter routine

3. Time varying delay signal generator routine

4. Interpolation subroutine

```
C*************************************************************************
C
C
C       GENERAL ADAPTIVE LMS-WIDROW FILTER ROUTINE
C
C       SOURCE FILE NAME                        LMSTDE.FR
C
C       PROGRAMMER                              SANGIL PARK
C
C       DATE                                    JUNE 25,1982
C
C
C*************************************************************************
C
C
C
C       PURPOSE
C
C               THIS PROGRAM IMPLEMENTS THE WIDROW LMS ALGOLITHM FOR
C               TIME DELAY ESTIMATION IN FILTER MODE, INTERPOLATES
C               THE DISCRETE DATA, AND OBTAINS THE TRANSIENT MAXIMUM
C               VALUES OF FILTER COEFFICIENTS OR DUMPS COEFFICIENT VALUE.
C
C
C
C       REFERENCE
C
C               WIDROW ET AL.  PROC. IEEE,DEC.1975
C
C               COEF(N+1)=COEF(N)+(ALPHA/VAR)*ER(N)*REF(N)
C
C
C       ROUTINES CALLED BY THIS PROGRAM
C
C
C               OPENW           OPENW
C               QUERY           READR
C               RESET           WRITR
C               INTPL.RB        APPEND
C
C
C*************************************************************************
C
C
C
C       DESCRIPTION OF I/O FILES
C
C
C                       INPUT:  PRIMARY   INPUT FILE
C                               REFERENCE INPUT FILE
C
C                       OUTPUT: MAXIMUM VALUE OUTPUT FILE
C                               COEFFICIENT   OUTPUT FILE
C
C
C
C*************************************************************************
```

```
C
C
        COMMON /OPN/ IFILE(18),NAME(13)
        REAL COEF(256),PRI(-63:1088),REF(-255:512),EST(512),ERR(512)
        REAL RCOEF(1271),VALUE(512)
C
C
        PARAMETER  NSIZE=512
                   RSIZE=FLOAT(NSIZE)
                   NBYTE=NSIZE*4
C
C
C       ** READ IN PARAMETERS **
C
C
1       CALL QUERY("WANT COEFFICIENT [Y] OR MAX.VALUE [N=CR] ? = ",IQC)
        ACCEPT " ENTER THE START POINT = ",DP1
        ACCEPT " ENTER THE   INTERVAL  = ",DP2
        ACCEPT "PRIMARY INPUT DELAY DISTANCE [-64:64]  ? = ",IDELAY
        ACCEPT "# OF COEFFICIENT(WEIGHT) (1-255, ODD #)? = ",NCO
        ACCEPT "THE VALUE OF SMOOTHING PARAMETER: BETA ? = ",BETA
        CALL QUERY("FIXED[Y] OR TIME VARYING[N=CR] VARIANCE? = ",IQV)
        VARIANCE=0.0
        IF (IQV.NE.1)  GOTO 4
C
        ACCEPT "ENTER THE FIXED INPUT VARIANCE VALUE   ? = ",VARIANCE
        GOTO 5
C
4       ACCEPT "ENTER MINIMUM INPUT VARIANCE VALUE:EPS ? = ",EPS
5       ACCEPT "THE NUMBER    OF    ITERATION (DATA)   ? = ",DATA
C
        CALL QUERY("WANT INTERPOLATION(Y/N=CR) ? = ",IQIN)
        IF (IQIN.NE.1)  GOTO 8
        ACCEPT " SAMPLING FREQUENCY ? =          ",SAMP
        ACCEPT " SIGNAL  BAND-WIDTH ? =          ",BAND
        ACCEPT " # OF INTPOL EACH INTERVAL ? = ",NINT
        BN=2.*BAND/SAMP
C
8       ICENTER=(NCO-1)/2
        IBLK=IFIX(DATA/RSIZE)
        IRES=IFIX(DATA-FLOAT(IBLK)*RSIZE)
        ID=IDELAY+ICENTER
        NRECORD=NCO*4
        IF(IQIN.EQ.1) NRECORD=((NCO-1)*NINT+1)*4
C
15      VAR=VARIANCE
C
C
C       ** OPEN I/O FILES **
C
C
        CALL OPENR(1,"PRIMARY   INPUT FILE NAME  = ",NBYTE,SIZE1)
        CALL OPENR(2,"REFERENCE INPUT FILE NAME  = ",NBYTE,SIZE2)
        IF(IQC.EQ.1) CALL OPENW(4,"COEFFS   OUTPUT FILE NAME  = ",NRECORD,SIZ
        IF(IQC.NE.1) CALL OPENW(3,"MAX VAL. OUTPUT FILE NAME  = ",NBYTE,SIZE3
C
C
```

```
C
C       ** MEMORY INITIALIZATION **
C
        DO 41 I=1,NSIZE
          PRI(-64+I)=0.
          PRI(I+64)=0.
          PRI(I+576)=0.
          REF(I)=0.
          EST(I)=0.
          ERR(I)=0.
          VALUE(I)=0.
41      CONTINUE
C
        DO 42 I=1,256
          COEF(I)=0.
          REF(1-I)=0.
42      CONTINUE
C
        N=1
        NN=1
        ALPHA=1.-BETA
        DP3=DP1
        KOUNT=0             ; MAX.VALUE DUMP COUNTER
        COUNT=0.            ; SYSTEM COUNTER
        NCOUNT=0            ; COEF. DUMP COUNTER
C
        TYPE
        TYPE " DATA =",DATA
        TYPE " DELAY=",IDELAY
        TYPE " BETA =",BETA
        TYPE
C
        IF (IBLK) 222,222,333
222     CONTINUE
        LOOP=IRES
        GOTO 25
C
333     CONTINUE
        LOOP=NSIZE
C
25      DO 26 I=0,NCO-1
          REF(-I)=REF(NSIZE-I)
26      CONTINUE
        IF(ID.LE.0) GOTO 100
        DO 27 I=1,ID
          PRI(I)=PRI(NSIZE+I)
27      CONTINUE
C
C
C       *********************
C           MAIN PROGRAM
C       *********************
C
C
100     CALL READR(1,N,PRI(ID+1),2,LCNT,IERR)
        CALL READR(2,N,REF(1),1,LCNT,IERR)
C
```

```
C
C
C        ** MAIN LOOP *
C
C
         DO 1000 I=1,LOOP
C
           COUNT=COUNT+1.0
           EST(I)=0.
           DO 51 II=1,NCO
             EST(I)=EST(I)+COEF(II)*REF(I-II+1)
51         CONTINUE
C
           ERR(I)=PRI(I)-EST(I)
C
C
C          ** UPDATE FILTER COEFFICIENTS **
C
           IF (IQV.EQ.1)  GOTO 55
           VAR=BETA*VAR+ALPHA*REF(I)*REF(I)
           IF (VAR.LE.EPS) VAR=EPS
C
55         DO 44 II=1,NCO
             COEF(II)=COEF(II)+ALPHA*ERR(I)*REF(I-II+1)/VAR
44         CONTINUE
C
           IF(COUNT.NE.DP3) GOTO 1000
           DP3=DP3+DP2
C
C
C          * DUMP COEFFICIENT VALUE *
C
           IF(IQC.NE.1) GOTO 60
           NCOUNT=NCOUNT+1
           IF(IQIN.EQ.1) GOTO 61
           CALL WRITR(4,NCOUNT,COEF,1,IER)
           GOTO 1000
C
61         CALL INTPL(COEF,RCOEF,BN,NINT,NCO)
           CALL WRITR(4,NCOUNT,RCOEF,1,IER)
           GOTO 1000
C
C
C          * DUMP MAX. VALUE *
C
60         KOUNT=KOUNT+1
           IF (IQIN.EQ.1)  GOTO 70
           MAX=1
           FMAX=COEF(1)
           DO 65 I2=2,NCO
             IF(COEF(I2).LE.FMAX) GOTO 65
             MAX=I2
             FMAX=COEF(I2)
65         CONTINUE
           FM=FLOAT(MAX-ICENTER-1)
           GOTO 80
C
```

```
C
C
70        CALL INTPL(COEF,RCOEF,BN,NINT,NCO)
          IMAX=1
          FMAX=RCOEF(1)
          DO 71 I2=2,(NCO-1)*NINT+1
            IF (RCOEF(I2).LE.FMAX) GOTO 71
            IMAX=I2
            FMAX=RCOEF(I2)
71        CONTINUE
          FM=FLOAT(IMAX-1)/FLOAT(NINT)-FLOAT(ICENTER)
C
80        VALUE(KOUNT)=FM
          IF (KOUNT.LT.NSIZE) GOTO 1000
          CALL WRITR(3,NN,VALUE,1,IERR)
          KOUNT=0
          NN=NN+1
C
1000    CONTINUE
C
C
        N=N+1
        IF (IBLK+1-N) 105,102,333
102     IF (IRES.GT.0.) GOTO 222
C
105     IF(KOUNT) 106,106,107
107     CALL CLOSE(3,IERR)
        CALL APPEND(3,NAME,3,4,IER)
        DO 110 II=1,KOUNT
          WRITE BINARY(3) VALUE(II)
110     CONTINUE
C
106     CALL RESET
        CALL QUERY ("<7>RE-EXECUTE LMSTDE(Y/N=CR) ? = ",IQ)
        IF (IQ) 200,200,201
201     CALL QUERY("WANT SAME PARAMETERS (Y/N=CR)? = ",IQ1)
        IF (IQ1) 1,1,15
200     STOP "*  NORMAL TERMINATION OF LMSTDE  *"
        END
```

```
C*************************************************************************
C
C
C       ADAPTIVE SHORT TERM CORRELATOR ROUTINE
C
C
C       SOURCE FILE NAME                    STCTDE.FR
C
C       PROGRAMMER                          SANGIL PARK
C
C       DATE                                JAN. 25,1983
C
C
C*************************************************************************
C
C
C       PURPOSE
C
C               THIS PROGRAM IMPLEMENTS THE SHORT TERM CORRELATOR
C               ALGORITHM FOR TIME DELAY ESTIMATION.  ONE CAN OBTAIN
C               THE IMPULSE RESPONSE FUNCTION(THE COEFFICIENT VALUE)
C               OR THE ESTIMATED DELAY(THE PEAK OF COEFFICIENT) VALUE.
C
C
C       REFERENCE
C
C               N.AHMED ET AL.  82 MIDCON CONFERENCE NOV.30,1982
C
C               RXY(K,L)=BETA1*RXY(K-1,L)+(1-BETA1)*REF(K)*PRI(K+L)/(SX(K)*S)
C
C               SX(K)=BETA2*SX(K-1)+(1-BETA2)*ABS(REF(K))
C
C
C       ROUTINES CALLED BY THIS PROGRAM
C
C               OPENW           OPENW
C               QUERY           READR
C               RESET           WRITR
C               INTPL.RB        APPEND
C
C
C*************************************************************************
C
C
C
C       DESCRIPTION OF I/O FILES
C
C
C
C                       INPUT:  PRIMARY   INPUT FILE
C                               REFERENCE INPUT FILE
C
C                       OUTPUT: MAXIMUM VALUE OUTPUT FILE
C                               COEFFICIENT   OUTPUT FILE
C
C
C*************************************************************************
```

```
C
C
      COMMON /OPN/ IFILE(18),NAME(13)
      REAL PRI(-63:1088),REF(512),VALUE(512)
      REAL RXY(-64:64),RXYINT(1291)
C
C
      PARAMETER       NSIZE=512
                      RSIZE=FLOAT(NSIZE)
                      NBYTE=NSIZE*4
                      EPSILON=0.000001
C
C
1     CALL QUERY("WANT CORRELATION(Y/N)? = ",ICOR)
      IF(ICOR.EQ.1) ACCEPT " AT MULTIPLE OF = ",DUMP
      CALL QUERY("WANT PEAK VALUE(Y/N)? = ",IMAX)
      IF(IMAX.NE.1) GOTO 11
      ACCEPT " START POINT?=   ",DP1
      ACCEPT " INTERVAL?=      ",DP2
C
11    CALL QUERY("WANT NORMALIZATION(Y/N)? = ",NORM)
      IF(NORM.NE.1) GOTO 12
      ACCEPT " SMOOTH.PARA. FOR NORM = ",BETA2
      TYPE
C
12    ACCEPT "PRI.INPUT DELAY UNIT[-64:64]  = ",IDELAY
      ACCEPT "SMOOTH. PARA. FOR CORRELATION = ",BETA1
      ACCEPT "NUMBER OF DATA ITERATION   ?  = ",DATA
      ITER=IFIX(DATA/RSIZE)
      IRES=IFIX(DATA-FLOAT(ITER)*RSIZE)
C
      ACCEPT "NUMBER OF SHIFTS (1 - 64)  ?  = ",LAG
      CALL QUERY("WANT INTERPOLATION(Y/N=CR) ? = ",IQIN)
C
      IF (IQIN.NE.1) GOTO 20
      ACCEPT " SAMPLING FREQUENCY ? =          ",SAMP
      ACCEPT " SIGNAL  BAND-WIDTH ? =          ",BAND
      ACCEPT " # OF INTPOL EACH INTERVAL ? = ",NINT
      BN=2.*BAND/SAMP
C
C
C     ** I/O AND INIT. **
C
C
20    DO 21 I=1,NSIZE
        PRI(I-64)=0.0
        PRI(I+64)=0.0
        PRI(I+576)=0.0
        REF(I)=0.0
        VALUE(I)=0.0
21    CONTINUE
C
      DO 22 I=-LAG,LAG
      RXY(I)=0.0
22    CONTINUE
C
C
```

```
C
C
      SX = 1.0
      SY = 1.0
      LCOEF=2*LAG+1
C
      NB = LCOEF * 4
      IF(IQIN.EQ.1) NB=(2*LAG*NINT+1)*4
      IF(ICOR.EQ.1) SDUMP = DUMP
      IF(IMAX.EQ.1) DP3=DP1
C
      FDUMP = 0.0       ;SYSTEM COUNTER
      KOUNT=0           ;COUNTER FOR MAXIMUM VALUE DUMP
      NCOUNT=0          ;COUNTER FOR CORRELATION FUNCTION DUMP
      JJ=1              ;FOR ITERATION BLOCK COUNTER
      NN=1              ;FOR VALUE BLOCK COUNTER
C
C
C     * OPEN I/O FILES *
C
C
      CALL OPENR(1,"PRIMARY   INPUT FILE NAME  : ",NBYTE,F2)
      CALL OPENR(0,"REFERENCE INPUT FILE NAME  : ",NBYTE,F1)
      IF(ICOR.EQ.1) CALL OPENW(4,"CORR. FUNCTION  FILE NAME  : ",NB,SIZE)
      IF(IMAX.EQ.1) CALL OPENW(3,"CORR.PEAK POINT FILE NAME  : ",NBYTE,SIZE)
C
C
      TYPE
      TYPE "DATA =",DATA
      TYPE "DELAY=",IDELAY
      TYPE "BETA =",BETA1
C
C
      IF(ITER) 999,222,333
222   CONTINUE
      LOOP=IRES
      GOTO 401
C
333   CONTINUE
      LOOP=NSIZE
C
401   DO 402 I=0,LAG+IDELAY
        PRI(IDELAY-I)=PRI(NSIZE+IDELAY-I)
402   CONTINUE
C
C
C     *******************
C     **  MAIN PROGRAM  **
C     *******************
C
C
C     * READ INPUT DATA *
C
C
      CALL READR(0,JJ,REF,1,IER)
      CALL READR(1,JJ,PRI(IDELAY+1),2,IER)
C
```

```
C
C
C        * MAIN LOOP *
C
         DO 1000 I = 1,LOOP
C
           FDUMP = FDUMP + 1.
           IF(NORM.NE.1) GOTO 499
           SX = BETA2*SX + (1.-BETA2)*ABS(REF(I))
           SY = BETA2*SY + (1.-BETA2)*ABS(PRI(I))
           POWER=SX*SY
           IF(POWER.GT.EPSILON) GOTO 500
499        POWER=1.0
500        DO 501 L=-LAG,LAG
             RXY(L)=BETA1*RXY(L)+(1.-BETA1)*REF(I)*PRI(I+L)/POWER
501        CONTINUE
C
C          * DUMP CORRELATION FUNCTION *
C
           IF(ICOR.NE.1.OR.FDUMP.NE.SDUMP) GOTO 600
           NCOUNT=NCOUNT+1
           SDUMP = SDUMP + DUMP
C
           IF(IQIN.EQ.1) GOTO 650
           CALL WRITR(4,NCOUNT,RXY(-LAG),1,IER)
           GOTO 600
C
650        CALL INTPL(RXY(-LAG),RXYINT,BN,NINT,LCOEF)
           CALL WRITR(4,NCOUNT,RXYINT,1,IER)
C
C          * DUMP PEAK POINT OF CORRELATION *
C
600        IF(IMAX.NE.1.OR.FDUMP.NE.DP3) GOTO 1000
           DP3=DP3+DP2
           KOUNT=KOUNT+1
C
           IF(IQIN.EQ.1) GOTO 750
           MAX=1
           FMAX=RXY(-LAG)
           DO 701 I2=-LAG+1,LAG
             IF(RXY(I2).LE.FMAX) GOTO 701
             MAX=I2
             FMAX=RXY(I2)
701        CONTINUE
           FM=FLOAT(MAX)
           GOTO 800
C
750        CALL INTPL(RXY(-LAG),RXYINT,BN,NINT,LCOEF)
           IPEAK=1
           FMAX=RXYINT(1)
           DO 751 I2=2,2*LAG*NINT+1
             IF(RXYINT(I2).LE.FMAX) GOTO 751
             IPEAK=I2
             FMAX=RXYINT(I2)
751        CONTINUE
           FM=FLOAT(IPEAK-1)/FLOAT(NINT)-FLOAT(LAG)
C
```

```
C
C
800       VALUE(KOUNT)=FM
          IF (KOUNT.LT.NSIZE) GOTO 1000
          CALL WRITR(3,NN,VALUE,1,IERR)
          KOUNT=0
          NN=NN+1
C
1000    CONTINUE
C
        JJ=JJ+1
        IF(ITER+1-JJ) 105,102,333
102     IF(IRES.GT.0) GOTO 222
C
105     IF(KOUNT) 106,106,107
107     CALL CLOSE(3,IERR)
        CALL APPEND(3,NAME,3,4,IERR)
        DO 108 II=1,KOUNT
          WRITE BINARY(3) VALUE(II)
108     CONTINUE
C
106     CALL RESET
        CALL QUERY("<7><12>RE-EXECUTE STCTDE (Y/N) : ",IQ)
        IF(IQ.NE.1) GOTO 999
        CALL QUERY(" WANT SAME PARAMETERS (Y/N) :",IQQ)
        IF(IQQ.NE.1) GOTO 1
        GOTO 20
C
999     STOP "** NORMAL TERMINATION OF STCTDE **"
        END
```

```
C****************************************************************************
C
C       TIME VARYING DELAY SIGNAL GENERATION ROUTINE
C
C       SOURCE FILE NAME                        TVFILE.FR
C
C       PROGRAMMER                              SANGIL PARK
C
C       DATE                                    JUNE 25,1982
C
C****************************************************************************
C
C
C       PURPOSE
C               THIS PROGRAM MAKES THE INPUT SIGNAL TIME VARYING
C               BY ARBITRARY DELAY FUNCTION.
C
C
C               INPUT:  DATA  INPUT FILE
C                       DELAY INPUT FILE
C
C               OUTPUT: TIME VARYING DATA OUTPUT FILE
C
C
C****************************************************************************
C
        COMMON  /OPN/ IFILE(18),NAME(18)
        REAL SIG(-63:1024),DSIG(512),DELAY(512)
C
        PARAMETER  NSIZE=512
                   RSIZE=FLOAT(NSIZE)
                   NBYTE=NSIZE*4
                   PI=3.1415927
C
C       ** SET UP PARAMETERS **
C
1       ACCEPT "NUMBER OF COEF( ODD ) ? = ",NCO
        ACCEPT "NUMBER OF ITERATON    ? = ",DATA
        IBLK=IFIX(DATA/RSIZE)
        IRES=IFIX(DATA-FLOAT(IBLK)*RSIZE)
        ICENTER=(NCO-1)/2
C
C       ** OPEN I/O FILES **
C
        CALL OPENR(1,"DATA  INPUT FILE NAME ? = ",NBYTE,SIZE1)
        CALL OPENR(2,"DELAY INPUT FILE NAME ? = ",NBYTE,SIZE2)
        CALL OPENW(3,"DATA OUTPUT FILE NAME ? = ",NBYTE,SIZE3)
C
C       ** INITIALIZATION **
C
        DO 10 I=1,NSIZE
          DSIG(I)=0.0
          DELAY(I)=0.0
10      CONTINUE
        DO 11 I=-63,1024
          SIG(I)=0.0
11      CONTINUE
```

```
C
        N=1
C
        IF(IBLK)222,222,333
222     LOOP=IRES
        GOTO 444
C
333     LOOP=NSIZE
C
444     DO 21 I=0,ICENTER-1
          SIG(-I)=SIG(NSIZE-I)
21      CONTINUE
C
C       *****************
C       ** MAIN PROGRAM **
C       *****************
C
        CALL READR(1,N,SIG(1),2,ICNT,IERR)
        CALL READR(2,N,DELAY,1,ICNT,IERR)
C
        DO 1000 K=1,LOOP
C
          DSIG(K)=0.0
          DO 1001 M=-ICENTER,ICENTER
            IF(M-DELAY(K)) 41,42,41
41          DSIG(K)=DSIG(K)+SIG(K-M)*SIN(PI*(M-DELAY(K)))/(PI*(M-DELAY(K)))
            GOTO 1001
42          DSIG(K)=DSIG(K)+SIG(K-M)
1001      CONTINUE
C
1000    CONTINUE
C
        IF(IBLK-N) 102,101,100
100     CALL WRITR(3,N,DSIG,1,IERR)
        N=N+1
        GOTO 333
C
101     CALL WRITR(3,N,DSIG,1,IERR)
        N=N+1
        IF(IRES) 555,555,222
102     CALL CLOSE(3,IERR)
        CALL APPEND(3,NAME,3,4,IERR)
        DO 103 I=1,IRES
          WRITE BINARY(3) DSIG(I)
103     CONTINUE
C
555     CALL RESET
        CALL QUERY("<7>RE-EXECUTE TVDATA(Y/N=CR) ? = ",IQ)
        IF(IQ)999,999,1
999     STOP "** NORMAL TERMINATION OF TVDATA **"
        END
```

```
C****************************************************************************
C
C       INTERPOLATION SUBROUTINE
C
C       SOURCE FILE NAME                        INTPL.FR
C
C       PROGRAMMER                              SANGIL PARK
C
C       DATE                                    JULY 1,1982
C
C****************************************************************************
C
C
C       PURPOSE
C               THIS PROGRAM GENERATES INTERPOLATING POINTS BETWEEN
C               DISCRETE DATA.
C
C       CALLING SEQUENCE
C
C               CALL INTPL(COEF,RCOEF,BN,NINT,NCO)
C
C       ARGUMENTS REQUIRED
C
C               COEF:   INTPUT DATA ARRAY
C               RCOEF:  OUTPUT DATA ARRAY
C               BN:     BN=2.*(SIGNAL BAND WIDTH)/(SAMPLING FREQ.)
C               NINT:   # OF INTERPOLATING POINTS BTWEEN DISCRETE DATA
C               NCO:    # OF POINTS TO BE INTERPOLATED
C
C
C*************************************************************************
C
C
        SUBROUTINE INTPL(COEF,RCOEF,BN,NINT,ICO)
C
        REAL COEF(128),RCOEF(1271)
C
C
        DO 2 I=1,1271
2         RCOEF(I)=0.0
C
        DO 3 I=1,NINT*(ICO-1)+1
          TAU=FLOAT(I-1)/FLOAT(NINT)+1.
          DO 4 M=1,ICO
            IF(TAU-M) 11,12,11
11              RCOEF(I)=RCOEF(I)+COEF(M)*SIN(3.14*BN*(TAU-M))/(3.14*BN*(TAU-M
                GOTO 4
12              RCOEF(I)=RCOEF(I)+COEF(M)
4         CONTINUE
3       CONTINUE
        RETURN
        END
```

ACKNOWLEDGMENTS

I am indebted to my major professor, Dr. Nasir Ahemd, for making my graduate work here at Kansas State University possible. His generous guidance and valuable suggestions in writing this thesis are greatly appreciated. To the members of my committee, Drs. D. R. Hummels and W. A. Parker, I am very grateful for their contributions. Thanks are also due to Dr. D. H. Youn, University of Iowa for his valuable suggestions.

I wish to express my gratitude to the Department of Electrical Engineering, Kansas State University, Naval Underwater System Center, CT, and Sandia National Laboratories, NM, for the financial support throughout my master's degree program.

Special thanks go to G. Pablo, N. Magotra, S. Vijanyndra, M. Wagdy and all my friends for their encouragement and friendship.

To my mother, grandparents, sister and brother whose love, trust and confidence saw me through while away from home.

To my beloved father, for being my inspiration and strength, this piece of work is heartly dedicated.

COMPARISON OF TWO ALGORITHMS FOR TIME DELAY ESTIMATION

by

Sangil Park

B.E., Yonsei University, Korea 1977

AN ABSTRACT OF A MASTER'S THESIS

submitted in partial fulfillment of the

requirements for the degree of

MASTER OF SCIENCE

ELECTRICAL ENGINEERING

COLLEGE OF ENGINEERING

KANSAS STATE UNIVERSITY

MANHATTAN, KANSAS

1983

ABSTRACT

The purpose of this paper is to compare the performances of two algorithms for adaptive time delay estimation. These are: (1) Widrow's least-mean-square algorithm, and (2) an adaptive short-term correlator algorithm. The desired comparison will be carried out via a digital computer simulation. Band-limited signals which are perturbed by white Gaussian noise and received at two sensors are considered at various bandwidths and signal-to-noise ratios.