

AD-A254 009



RL-TR-92-78
Final Technical Report
May 1992



2

ALGORITHMS FOR MULTICHANNEL OPTICAL PROCESSOR

COLSA, Inc.

Stephen T. Welstead

DTIC
SELECTE
AUG 11 1992
S D

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

92-22305



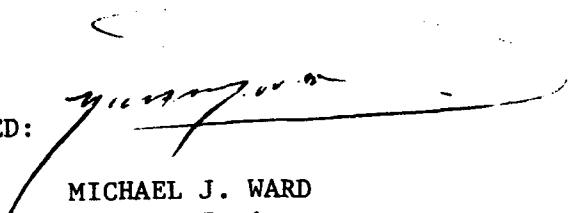
92 8 7 027

Rome Laboratory
Air Force Systems Command
Griffiss Air Force Base, NY 13441-5700

This report has been reviewed by the Rome Laboratory Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

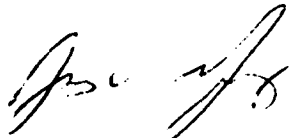
RL-TR-92-78 has been reviewed and is approved for publication.

APPROVED:



MICHAEL J. WARD
Project Engineer

FOR THE COMMANDER:



JAMES W. YOUNGBERG, Lt Col, USAF
Deputy Director
Surveillance & Photonics Directorate

If your address has changed or if you wish to be removed from the Rome Laboratory mailing list, or if the addressee is no longer employed by your organization, please notify RL(OCPA) Griffiss AFB, NY 13441-5700. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave Blank)		2. REPORT DATE May 1992		3. REPORT TYPE AND DATES COVERED Final Jan 91 - Jan 92	
4. TITLE AND SUBTITLE ALGORITHMS FOR MULTICHANNEL OPTICAL PROCESSOR				5. FUNDING NUMBERS C - F30602-88-D-0028, Task P-1-6012 PE - 62702F PR - 4600 TA - P1 WU - PB	
6. AUTHOR(S) Stephen T. Welstead					
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) COLSA, Inc. 6726 Odyssey Drive Huntsville AL 35806				8. PERFORMING ORGANIZATION REPORT NUMBER N/A	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Rome Laboratory (OCPA) Griffiss AFB NY 13441-5700				10. SPONSORING/MONITORING AGENCY REPORT NUMBER RL-TR-92-78	
11. SUPPLEMENTARY NOTES Rome Laboratory Project Engineer: Michael J. Ward, Capt, USAF/OCPA/(315) 330-2944 Prime Contractor: University of Dayton, 300 College Park, Dayton OH 45469-0001					
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited.				12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) This report discusses several alternative algorithms for implementation in an optical processor to perform adaptive interference cancellation. Estimates of the interference signal are formed in the optical processor by weighting and summing delayed copies of signals which are input from multiple auxiliary antennas. The optical processor is limited in the number of weights that can simultaneously be applied. To address this limitation, a new class of algorithms called limited output weight position algorithms is introduced in this report. One particular algorithm which sequentially applies the output weights is shown to provide good cancellation performance in a number of simulated multipath delay scenarios.					
14. SUBJECT TERMS Optical processing, Adaptive algorithms, Adaptive nulling				15. NUMBER OF PAGES 44	
				16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL		

CONTENTS

Section	Page
1. INTRODUCTION	1
2. BACKGROUND	1
2.1 The Application Problem.....	1
2.2 The Optical Processor.....	2
3. THE ALGORITHMS	4
3.1 Steepest Descent.....	4
3.2 Limited Output Weight Position Algorithms.....	6
3.2.1 The Largest Component Selection Method.....	7
3.2.2 Correlation Peak Location Method.....	8
3.2.3 The Sequential Method.....	12
3.2.4 Choosing the Largest Component Values.....	13
3.3 Other Algorithms.....	14
4. ALGORITHM PERFORMANCE.....	15
4.1 Test Case Definitions.....	15
4.2 Simulation Results.....	17
4.3 Conclusions.....	23
5. ANALYSIS OF THE SEQUENTIAL ALGORITHM.....	24
5.1 An Example.....	24
5.2 Analysis.....	29
5.3 Stepsize Selection.....	32
6. ISSUES AND CONCLUSIONS.....	33
REFERENCES.....	35

DTIC QUALITY INSPECTED 8

Accession For	
NTIS GRA&I	<input checked="checked" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

FIGURES

Figure	Page
3.2.1.1 Two delays with unequal amplitudes.....	8
3.2.2.1 The correlation function.....	11
3.2.2.2 A single delay correlation of signals on a carrier.....	11
4.2.1 Correlation for Test Case 1.....	19
4.2.2 Correlation for Test Case 2.....	20
4.2.3 Correlation for Test Case 3.....	21
4.2.4 Correlation for Test Case 4.....	22
5.1.1 Iteration 1.....	25
5.1.2 Iteration 2.....	25
5.1.3 Iteration 3.....	26
5.1.4 Iteration 4.....	26
5.1.5 Iteration 5.....	27
5.1.6 Iteration 6.....	27
5.1.7 Iteration 7.....	28

TABLES

Table	Page
4.2.1 Simulation Results for Test Case 1.....	19
4.2.2 Simulation Results for Test Case 2.....	20
4.2.3 Simulation Results for Test Case 3.....	21
4.2.4 Simulation Results for Test Case 4.....	22

1. INTRODUCTION

This report describes the work performed by Dr. Stephen Welstead, COLSA, Inc., Huntsville, AL, on the project "Algorithms for Multichannel Optical Processor", contract number F30602-88-D-0028, subcontract number RI-74242X. The period of performance covered by this report is 15 January 1991 through 14 January 1992. Laboratory work was performed at the Photonics Center of Rome Laboratory, Griffiss Air Force Base, New York. Analysis work was performed at COLSA headquarters in Huntsville, AL.

This report discusses several alternative algorithms for implementation in an optical signal processor to perform adaptive interference cancellation. A new algorithm is introduced which shows great promise for implementation in the multichannel optical processor. The effort reported here is a continuation of work reported in [1,2,3].

2. BACKGROUND

The application problem and the optical processor architecture have been reported on elsewhere [3]. We include a brief summary here in order to introduce the notation and to establish the advantages and limitations of the optical processing scenario in which the algorithms will be used.

2.1 The Application Problem

The signal processing application problem considered here is that of adaptive cancellation of interference, or jamming, signals. We are primarily interested in applying our results to radar signals. However, the methods discussed here can be applied to other types of signals, both active and passive.

We assume the following scenario, which is standard for adaptive signal processing [4]. A main antenna receives the signal $d(t)$, which is comprised of the desired signal $s(t)$ and an additive interference signal $n(t)$. The characteristics of both $s(t)$ and $n(t)$ are unknown. To determine the characteristics of $n(t)$, one or more omnidirectional auxiliary antennas are used to sample the interference environment. These auxiliary signals are denoted by $n_1(t), \dots, n_M(t)$ (where M is the number of auxiliary antennas). The interference signal $n(t)$ may consist of a linear combination of delayed versions

of some jamming signal $j(t)$. However, for the purpose of cancelling $n(t)$, the only information available to the system designer comes from the auxiliary signals $n_1(t), \dots, n_M(t)$. Thus, in what follows, we will discuss estimates of $n(t)$ only in terms of $n_1(t), \dots, n_M(t)$, and will not specifically refer to the jamming signal $j(t)$.

The problem in general is to determine the linear combination of delayed versions of the signals $n_i(t)$ which forms the best estimate $y(t)$ of the main antenna interference $n(t)$. The best estimate is defined as that $y(t)$ which minimizes the energy of the error signal

$$e(t) = d(t) - y(t). \quad (2.1.1)$$

The algorithms of section 3 are concerned with finding the optimum coefficients and delays to be used in constructing the interference estimate $y(t)$.

2.2 The Optical Processor

The optical processor is described in [3]. For the purpose of algorithm analysis, it is sufficient to think of the processor as consisting of two subsystems. In a multichannel implementation, each of these consists of several parallel channels. The first of these forms a correlation of the error signal $e(t)$, given by (2.1.1), with the auxiliary signal $n_i(t)$. This is done in parallel for each of the M auxiliary inputs. An electronic microprocessor then determines the optimum coefficients and delays, using one of the algorithms which we will discuss in section 3. This information is then fed to the second optical subsystem which applies the coefficients and delays to the auxiliary signals $n_i(t)$ to form the interference estimate $y(t)$.

The first optical subsystem uses a multichannel acousto-optic (AO) cell and a time integrating detector array to perform the correlation function in a fairly standard way. This subsystem works well, providing sufficient dynamic range and spatial resolution to achieve the overall system performance goals. Optical processing enables parallel computation of the correlation function, providing real time performance with essentially unlimited spatial resolution.

The correlation is currently being computed over a 5 μ sec delay range. Delay time is determined by the characteristics of the AO cell. Higher delay times can be achieved using different cells. The

correlation function over this range is sampled in parallel by a 512 element detector array. This number can also be improved by a factor of as much as eight with currently available detector arrays. Due to the parallel nature of the optical processing, there is no real time penalty incurred from computing a larger array of correlation values.

The second optical subsystem also uses a multichannel AO cell to apply time delays to the auxiliary signals $n_i(t)$. This system also requires spatial light modulation to pick out and weight the appropriate delay values along the AO cell. Several candidate spatial light modulators (SLMs) have been considered for this task [5]. Issues such as speed, contrast ratio, and dynamic range have led to the decision to use a steering AO cell as an SLM. The idea here is that by varying the carrier frequency that is applied to the cell, one can vary the angle of the first order diffracted beam. Several different carrier frequencies can be combined in one signal to obtain several distinct angles simultaneously. These different angles correspond to unique tap positions on the delay line.

While this AO SLM approach provides sufficient speed, contrast ratio, and dynamic range for our performance goals, it does impose a limitation on the system. Specifically, we are limited to a small number of optical beams that can be generated at any one time in each channel of the AO cell. This means that there is a limited number of delay positions that can be simultaneously tapped per channel. This limit comes both from limitations of the AO cell itself, as well as from the practical limitations of generating a large number of carrier frequencies simultaneously. The exact upper limit is not known, but for the purpose of algorithm development, we are planning on having no more than eight delay taps per channel available.

It should be kept in mind that these delay taps can be used to select from a continuum of delay values available from the second AO cell. Thus, we have much more than, for example, an eight tap delay line with fixed tap positions. Our configuration enables us to handle up to eight multipath copies per channel of an interference signal, with delay positions being selected from anywhere along a 5 μ sec delay range.

The advantages and limitations of the optical processor have a significant impact on algorithm design. We discuss the algorithms in the next section.

3. THE ALGORITHMS

3.1 Steepest Descent

The traditional approach to adaptive processing is to use some variation of the steepest descent algorithm. While this algorithm has been discussed elsewhere [1], it is instructive to review the key points here, in order to see the relevance of the various quantities involved.

It is assumed that there is a finite number of fixed delay positions x_1, x_2, \dots, x_N available across the total delay range D . Thus we have

$$0 \leq x_1 < x_2 < \dots < x_N \leq D.$$

Typically, the distance between delay positions is fixed, for example at

$$\Delta = D/N$$

so that $x_i = (i - 1) \Delta$. In our optical processor, the number of fixed delay positions would correspond to the spatial resolution of the detector array.

We now introduce the so called *weight vector*

$$\mathbf{W} = (w_1, w_2, \dots, w_N)$$

and form the signal

$$\begin{aligned} y(\mathbf{W}; t) &= w_1 n_1(t - x_1) + w_2 n_1(t - x_2) + \dots + w_N n_1(t - x_N) \\ &= (\mathbf{W}, \mathbf{R}(t)) \end{aligned} \tag{3.1.1}$$

(for ease of notation, we consider only the single channel case here; the theory is the same in the multichannel case). In (3.1.1), the notation $(\ , \)$ indicates vector inner (or scalar) product, and $\mathbf{R}(t)$ is the vector function whose i th component is $n_1(t - x_i)$. The signal

$y(\mathbf{W}; t)$ is the estimate of the interference signal $n(t)$. The problem is to determine the appropriate weight vector \mathbf{W} in order to make this the best estimate.

We emphasize at this point that the weight vector is a consequence of the choice of algorithm. It is not inherently part of the adaptive processing problem. Any weight vector approach will be limited in performance by the number of available tap delay positions. If the true delay falls between two fixed tap positions, performance will suffer. This problem can be alleviated by providing more tap positions, but this places more demand on system resources and may slow down the algorithm. It is possible, although difficult, to approach the problem directly from a multivariable optimization point of view, with the delay positions (chosen anywhere along the delay range) and coefficients forming the higher dimensional optimization space. We discuss this approach in section 3.3.

The goal of the steepest descent algorithm is to determine the weight vector \mathbf{W} which minimizes the error energy:

$$E(\mathbf{W};t) = \int_t^{t+T} (d(s) - y(\mathbf{W};t))^2 ds \quad (3.1.2)$$

This expression is the energy of the error signal given by (2.1.1) over the time interval $[t, t + T]$.

The steepest descent algorithm is an iterative method that finds the minimum of $E(\mathbf{W};t)$ by moving in the direction of the negative gradient of E with respect to \mathbf{W} . Recalling that $y(\mathbf{W};t) = (\mathbf{W}, \mathbf{R}(t))$, the gradient of $E(\mathbf{W};t)$ is easily computed from (3.1.2) as the vector whose i^{th} component is

$$(\nabla E)_i(t) = \int_t^{t+T} e(\mathbf{W};s) n_1(s - x_i) ds \quad (3.1.3)$$

where $e(\mathbf{W};t) = d(t) - y(\mathbf{W};t)$. Note that ∇E can actually be viewed as a correlation function sampled at the points x_i . This is why correlation is significant in the implementation of steepest descent algorithms.

The iterative algorithm for finding the optimum \mathbf{W} can be formulated as

$$\text{new } \mathbf{W} = \text{old } \mathbf{W} + \text{stepsize} \cdot \Delta \mathbf{W} \quad (3.1.4)$$

where *stepsize* is a positive scalar quantity used to control convergence, and $\Delta \mathbf{W} = -1/2 \nabla E$.

The steepest descent algorithm can usually provide an adequate solution, subject to the limitations of fixed tap positions mentioned above. There are, however, two drawbacks. One is that steepest descent is notoriously slow, typically requiring hundreds of iterations for convergence. The other, more serious, drawback is concerned with implementation in our optical processor. While we have no trouble forming the correlation vector $\Delta \mathbf{W}$ with essentially unlimited spatial resolution, we cannot form the interference estimate $y(\mathbf{W}; t)$, given by (3.1.1), for an adequate number of tap delay positions. This is due to the spatial light modulator limitations mentioned in section 2.2.

Because of these limitations, we have been led to consider a new class of weight vector algorithms which are a modification of steepest descent. These algorithms are discussed in the next section.

3.2 Limited Output Weight Position Algorithms

Limited spatial light modulation capability has led us to consider a new class of weight vector algorithms. These algorithms involve using only a limited number of the available tap delay positions to form the estimate $y(\mathbf{W}; t)$, while maintaining full spatial resolution to construct the correlation vector whose components are given by (3.1.3). All tap delay positions are used to form the update weight vector (ie., correlation vector) that is input to the algorithm, while a weight vector with only a limited number of nonzero components is output from the algorithm to form the signal $y(\mathbf{W}; t)$. Thus, we call these algorithms *limited output weight position* algorithms.

Limiting the number of output weight positions does somewhat limit the multipath scenarios which can be handled by these algorithms. However, we believe we can provide enough output positions to handle most foreseeable scenarios. The algorithm should still

provide a significant degree of cancellation performance even for those cases which exceed the capacity of the output weights. A bonus for this type of algorithm is that it is usually orders of magnitude faster than steepest descent with a full output vector. This is reasonable if one considers the fact that if, for example, only 4 weights are needed to achieve cancellation, then it is a burden to have to maintain and update 512 output weight positions.

Several variations of the limited weight position algorithms are discussed here. One is a highly promising algorithm that was able to handle most of the multipath scenarios presented in the simulation tests. This algorithm is recommended over the other two algorithms, which are included here only for historical perspective. The intent is only to record the fact that they were considered and rejected. The version of this algorithm recommended for implementation is the sequential method discussed in section 3.2.3.

3.2.1 The Largest Component Selection Method

This algorithm has been discussed in [2]. It was the first algorithm of the limited output weight position type considered. The idea is simple: pick the largest of the weight vector components for output.

Let N be the total number of tap positions, or weight vector components, available (so, for example, N might be 512). Let $N_0 \ll N$ be the number of nonzero output weight vector positions available (so, for example, N_0 might be 4 or 8). At each iterative step, the correlation update vector is computed, as given by (3.1.3), and the weight vector is updated as in (3.1.4). However, to form the interference estimate, a new weight vector \mathbf{W}_0 is constructed. This vector has N components, all but N_0 of which are 0. The N_0 nonzero positions correspond to the N_0 positions of \mathbf{W} with the largest component values (see Sect. 3.2.4 for an efficient way to choose the N_0 largest values). Note that a cumulative weight vector containing all vector components is maintained in computer memory and the output weight positions are chosen using this weight vector.

This method works well in the case when there is just a single delay to solve for. Oscillatory problems can arise if this delay falls between tap positions, but these can be minimized by providing a high density of tap positions for the input correlation vector. Performance of this algorithm suffers when $N_0 > 1$ and only a single delay is to be found. The algorithm clusters all available weights

near the actual delay position, and then must spend time suppressing the unneeded weights. A more serious problem arises when two or more delays must be found, with different signal amplitudes corresponding to each delay. In this case, the method tends to throw all N_0 of its available output weights at the delay which causes the largest correlation amplitude. The other delays are ignored. This situation is depicted in Fig. 3.2.1.1. The method will not converge in this case.

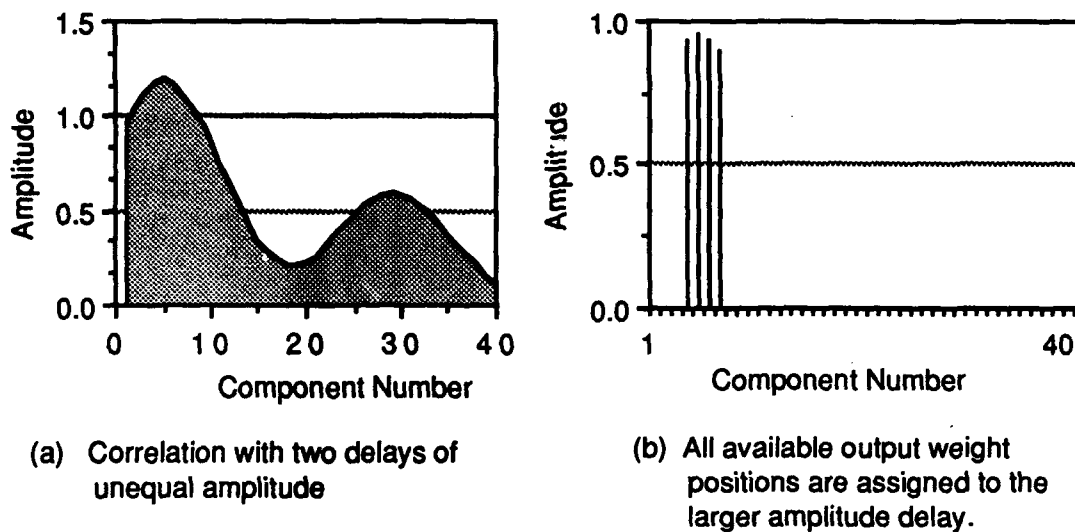


Figure 3.2.1.1 Two delays with unequal amplitudes cause problems for this method.

3.2.2 Correlation Peak Location Method

The correlation vector contains a significant amount of information about the relative delays of the two signals being correlated. The correlation vector, when viewed as a spatial function of the delay variable, will have local maxima at locations corresponding to relative common delays. The relative amplitudes corresponding to these delays are proportional to the amplitudes of the correlation maxima. This is the motivation for considering an algorithm which constructs a weight vector by using information about local peaks in the correlation vector.

Let \mathbf{W}_0 represent the output weight vector (with, as above, only N_0 allowable nonzero components), and let \mathbf{V}_c be the correlation vector whose i^{th} component is given by

$$(V_e)_i(t) = \int_t^{t+T} e(W_o; s) n_1(s - x_i) ds, \quad i = 1, \dots, N. \quad (3.2.2.1)$$

Note that negative values in (3.2.2.1) are allowed, and in fact are essential for changing the locations and values of the output weight components. Since this vector is stored digitally, negative values do not present a problem.

Form the *intermediate weight vector* W_1 :

$$W_1 = W_o + \text{stepsize} \cdot V_e \quad (3.2.2.2)$$

As in the steepest descent algorithm, *stepsize* is a parameter used for controlling convergence (in general, the stepsize here can be larger than that used for steepest descent). Note that W_1 is *not* an accumulation of previous weight vectors, as was used in the largest component selection method. Rather, it consists only of the latest output weight vector added to a correlation vector, which was formed with an error function constructed with that output weight vector.

It should be noted that the cumulative weight vector that is maintained in the previous method is actually of limited usefulness. It is an accumulation of correlation update vectors generated by incorrect interference estimates $y(W;t)$ which have been constructed with limited output weight vectors. This is not a true gradient descent process, and so convergence to the correct solution vector is not guaranteed, or expected.

The intermediate vector W_1 is then searched for local maxima. This is done using a discrete approximation to the first derivative and noting when this expression changes sign. Since we are dealing with a discrete vector, rather than a continuous function, it is feasible to search a small neighborhood near where the first derivative changes sign, and obtain the exact vector component which is a local maximum.

Let N_p be the number of local maxima obtained in this way. Note that N_p is at least 1, since, for example, if the correlation vector is constant, then each component is a local maximum, and if it is monotonic increasing or decreasing, then one of the endpoints will

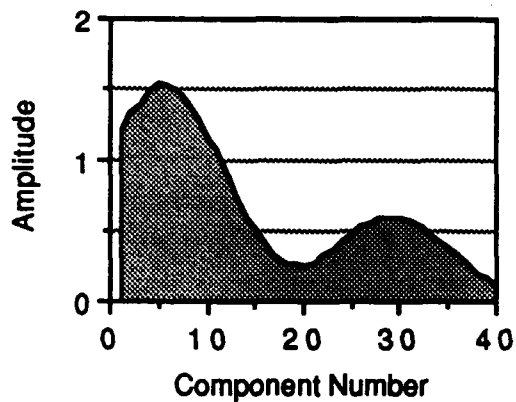
be a local maximum. In general, $N_p > N_0$ so we have to choose the N_0 largest components from those which were selected as local maxima (once again, see Sect. 3.2.4 for an efficient way to do this). The indexes and values of these components are saved.

The new output weight vector is constructed by first setting all components equal to 0, and then setting those components at the saved indexes equal to the saved local maxima values. In order to represent the dynamic range limitations of the optical system, output component values larger than 1.0 are truncated, and negative values are set equal to 0.

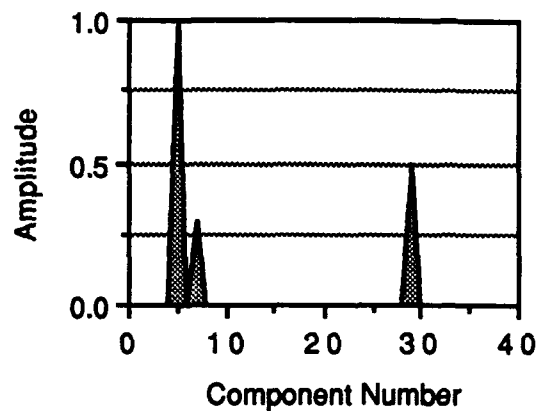
In some cases, this algorithm works very well, as can be seen in the simulation results in the next section. In fact, with an optimum stepsize parameter, it is possible in some cases to obtain a correct solution in one iteration. However, the algorithm is seriously limited by its failure to handle two cases which are quite likely to occur in practice.

The first of these is illustrated in Fig. 3.2.2.1. While this algorithm can easily handle two widely separated delays, such as the case illustrated in Fig. 3.2.1.1, a problem arises when it must identify two closely spaced delays. The larger of the two correlation peaks in Fig. 3.2.2.1 (a) is actually due to two closely spaced delays of different amplitude, as can be seen in Fig. 3.2.2.1 (b). The algorithm can locate one of the two closely spaced delays, but not the other. The iterations in this case either become unstable, or tend to oscillate, rather than converge to a solution.

A second, equally serious, drawback to this algorithm is the fact that it cannot handle signals on a carrier, that is, when the signals are of the form $f_c(t)r(t)$, where $f_c(t)$ is a high frequency sinusoid. The presence of a carrier introduces a large number of local maxima into the correlation function, as shown in Fig. 3.2.2.2. The example shown here corresponds to only a single relative delay between the interference and auxiliary signals. The algorithm, however, will pick out a large number of local maxima. Note that applying the algorithm "twice" won't work either. It is true that applying the local maxima search to the list of local maxima will produce the local maxima of the envelope function. However, some of these are actually local *minima* of the correlation function. This will lead to an incorrect choice of output weight vector.



(a) Correlation Vector



(b) Delay positions and amplitudes

Figure 3.2.2.1 The correlation function shown in (a) was constructed with an interference signal consisting of three delayed copies of the auxiliary signal $n_1(t)$ (a gaussian pulse). The delay positions and amplitudes are shown in (b). The two closely spaced delays produce only a single local maxima in the correlation function.

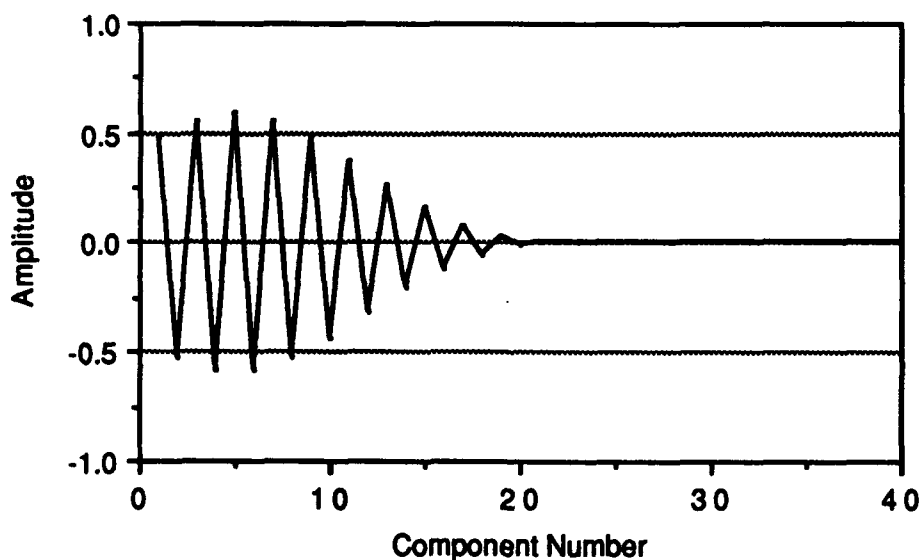


Figure 3.2.2.2 A single delay correlation of signals on a carrier has many local maxima; only one corresponds to the true delay position.

3.2.3 The Sequential Method

This method originally was developed to deal with the largest component selection method's problem of throwing all of its resources at the largest correlation peak. However, it also successfully deals with the shortcomings of the peak location method. The idea behind this method is to use the output weights one at a time. Do the best that you can with one output weight, until no further cancellation is achieved. If the error amplitude exceeds tolerance, then allow two output weights. Continue this process until either the allowable number of output weights has been reached, or the error amplitude has been reduced to below the allowable tolerance.

This approach clearly avoids the problem of applying too many output weights to a single correlation peak. However, this method can also handle closely spaced delays (even those delays corresponding to consecutive weight vector components). It works as well with signals on a carrier as it does with signals with no carrier. Moreover, it is suited to responding to a dynamically changing weight vector.

The key to this algorithm's success is that it uses the intermediate weight vector given by (3.2.2.2) as the basis for the sequential search, rather than the cumulative weight vector of the largest component selection method. This means, for example, that it is not necessary to keep track of components used previously in the output weight vector. These components are present in the first term of the vector sum (3.2.2.2). If these components continue to merit consideration for use in the new output weight vector \mathbf{W}_0 , then they will have a strong presence in the intermediate vector \mathbf{W}_i . If they are no longer needed, then negative values in the corresponding components of the correlation vector \mathbf{V}_c will reduce the influence of these components in \mathbf{W}_i . In this way, the algorithm can quickly respond to a dynamically changing weight vector.

We now summarize the key steps of this algorithm:

(1) Form the correlation vector \mathbf{V}_c , as defined by (3.2.2.1), using the error function $e(\mathbf{W}_0;t)$ formed with the previous output weight vector \mathbf{W}_0 . On the initial iteration, \mathbf{W}_0 is the zero vector. Note that

V_c can have (and must be allowed to have) positive and negative values.

(2) Form the intermediate weight vector W_1 , given by (3.2.2.2). W_1 is allowed to have positive and negative values.

(3) Identify the N_s largest components of W_1 (using the search algorithm described in section 3.2.4 below). N_s is initially 1, and is updated as described below. Save the indexes and values of these vector components.

(4) As in the peak location algorithm, construct the new output weight vector W_o by first setting all components equal to 0, and then setting those components at the saved indexes equal to the saved component values from step 3. To represent the limited dynamic range of the optical system, truncate output components larger than 1, and set negative components equal to 0.

(5) Form the interference estimate $y(W_o;t)$, given by (3.1.1), and the corresponding error function $e(W_o;t)$. If the amplitude of $e(W_o;t)$ is less than the error tolerance, then you are done. (If weight adaptation is performed in the presence of a main signal, then the amplitude (computed as the peak to peak height) of the correlation vector should be used, rather than the amplitude of $e(W_o;t)$.) If the absolute difference between the amplitude of $e(W_o;t)$ on this iteration and the amplitude on the previous iteration is less than some factor times the error tolerance (with the factor typically being 0.1), then increment N_s by 1, otherwise do not change the value of N_s . If $N_s \leq N_o$ (the maximum number of nonzero output components allowed) then go to step 1, otherwise terminate iterations (this is the best you can do with N_o output components).

3.2.4 Choosing the Largest Component Values

Each of the algorithms discussed in sections 3.2.1-3.2.3 requires at some point the selection of the K largest values from a list of L numbers. Here, K and L are integers, and $K < L$ (K and L assume different values in each of the algorithms). Since this selection process is fundamental to each of the algorithms, it is worthwhile to examine an efficient way of accomplishing it. The method described here is what has been used in the simulations of the algorithms reported on below.

If $K = 1$, then it is a simple matter to search the list of L items and determine the maximum component. However, if $K > 1$, then we don't want to have to search the entire list K times. The K largest components can be located in one search of the list. To do this, we need to maintain an auxiliary list of length K . This list will eventually hold our K largest components. To start, load the first K components of the main list into the auxiliary list. Identify the *minimum* element of this auxiliary list, and its index in the auxiliary list. To qualify for membership in the auxiliary list, a main list element must exceed this minimum element.

Begin checking the main list elements, starting with index $K + 1$. If a main list element exceeds the minimum element in the auxiliary list, then replace that minimum element with the new element from the main list. Now, search the *new* auxiliary list for its minimum (which may or may not be the new element that was just added). Note the index and value of the new auxiliary list minimum. Continue checking main list elements against the auxiliary list in this way. When the end of the main list is reached, the auxiliary list will contain the K largest values.

3.3 Other Algorithms

In [2], other algorithmic approaches, including non-weight vector approaches, to this problem were considered. We briefly mention these here for the sake of completeness; the reader is referred to [2] for a more complete discussion.

The *parabolic interpolation* algorithm is a single variable algorithm for finding a local minimum of a nonlinear function. It requires no information other than the function values at given points. It is a robust algorithm that is useful for adjusting single parameters (as suggested, for example, in the final section of this report). However, it becomes unwieldy in the multivariable case, and is probably not suitable for finding multiple delays and amplitudes.

Conjugate gradient and *conjugate direction* methods are weight vector algorithms that require specific information about the problem scenario that is not available in our optical processor. The main advantage of these algorithms is that they guarantee convergence in a finite number of steps. However, in our problem, that finite number is the number of tap positions. This number is

generally quite large, so this finite convergence feature is not likely to present an advantage over the approaches considered here.

Simulated annealing is a multivariable minimization algorithm designed to find the global minimum of a multimodal nonlinear function. However, it is not well suited to real time performance in problems with a large number of variables.

Finally, *neural network* approaches to this problem have also been considered [6]. Neural networks may provide some utility in supplementing other algorithms for performance enhancement, or perhaps may prove useful for data preprocessing tasks. However, the task of training a network which would be adaptable to all possible signal scenarios does not appear to be practical.

4. ALGORITHM PERFORMANCE

This section presents simulation results for the algorithms discussed in sections 3.1 and 3.2. Algorithm performance was investigated for four test cases representing four distinct multipath delay signal scenarios. Each test case was further subdivided into two cases: one with a carrier signal present and one without a carrier signal. The simulation was carried out on a digital computer.

4.1 Test Case Definitions

The four test cases were chosen as representative examples of multipath delay environments that would illuminate the strengths and weaknesses of each algorithm. The sequential method of section 3.2.3 handled these cases better than either of the other algorithms discussed in section 3.2.

The basic signal type considered for each test case is the gaussian pulse:

$$\begin{aligned} r(t) &= \exp(-t^2) & \text{if } |t| < 1.0 \\ &= 0 & \text{otherwise.} \end{aligned} \quad (4.1.1)$$

The algorithms have also been tested with monotone and two-tone sinusoid signals, as well as square pulse signals. The results do not differ significantly from the gaussian pulse case. The gaussian pulse simulates a broadband signal environment and captures all of

the signal characteristics that are significant for our purposes, thus we report only on this case.

While the simulation is essentially unitless, the relationships among the variables are such that one can think of the basic time unit as a μsec , and the basic frequency unit as a MHz.

Each test case assumes one auxiliary antenna which receives the auxiliary signal

$$n_1(t) = 0.5 \cdot r(t - 0.5). \quad (4.1.2)$$

The weight vector algorithms considered here can handle additional auxiliary channels without difficulty. Such additional channels would, however, unduly stress the processing capabilities of the personal computer used for the simulations. For this reason, we consider only the single channel case here.

The interference signal has the form

$$n(t) = \sum_{i=1}^K a_i r(t - d_i) \quad (4.1.3)$$

Thus, K is the number of multipath delays, the d_i are the delay values, and the a_i are the associated amplitudes. The parameter values which define each test case are as follows:

Test Case 1: $K = 1$, $a_1 = 1.0$, $d_1 = 1.0$.

Test Case 2: $K = 2$, $a_1 = 1.0$, $a_2 = 0.5$, $d_1 = 1.0$, $d_2 = 4.0$.

Test Case 3: $K = 3$, $a_1 = 1.0$, $a_2 = 0.5$, $a_3 = 0.3$,
 $d_1 = 1.0$, $d_2 = 4.0$, $d_3 = 1.25$.

Test Case 4: $K = 3$, $a_1 = 1.0$, $a_2 = 0.9$, $a_3 = 0.8$,
 $d_1 = 1.0$, $d_2 = 1.25$, $d_3 = 1.5$. (4.1.4)

Test case 1 thus represents the simplest delay estimation problem, that of determining a single relative delay. Test case 2 is a straightforward multipath problem with two widely separated

delays that will produce two distinct peaks in the correlation function. Test case 3 is a more complicated multipath problem, with three delays, two of which are closely spaced. These three delays produce only two correlation peaks, as illustrated in Fig. 3.2.2.1, and again in Fig. 4.2.3. Test case 4 is also a three delay case, with three closely spaced delays. This test case is an example that stresses the sequential algorithm.

The carrier signal is given by:

$$f_c(t) = \cos((2\pi) 12t). \quad (4.5)$$

Each test case is considered both with and without the presence of this carrier signal. When the carrier signal is present, the basic reference signal $r(t)$ is replaced by $f_c(t)r(t)$.

The simulations were run with weight vectors with $N = 40$ components. Thus, the correlation vectors consisted of 40 components, and 40 positions were available for the output weight vectors. The total delay range is set at $D = 5.0 \mu\text{sec}$. Thus the delay increment is

$$\Delta = 5.0/40 = 0.125 \mu\text{sec}.$$

The relatively small number of weight positions is chosen only to ease the computational burden on the digital simulation. The relative delays d_i are chosen so that an exact solution is possible with this delta. In the actual processor, the number of weight positions N would be much larger, giving a smaller delta and alleviating performance problems due to lack of resolution across the delay range.

The error tolerance (for the amplitude of the error function) was set at 0.01. Thus, cancellation to this tolerance of an interference signal of amplitude 1.0 would correspond to a 100:1, or "40 dB", cancellation ratio.

4.2 Simulation Results

Figures 4.2.1 (a) - 4.2.4 (a) show plots of the correlation vectors for test cases 1 - 4, while figures 4.2.1 (b) - 4.2.4 (b) show the

corresponding correlation vectors in the presence of the carrier signal given by (4.5). (Note that the 40 component correlation vector does not adequately "sample" the true correlation function in the presence of this 12 MHz carrier. Reconstruction of the true correlation function is not necessary for these algorithms to work, however. It is only necessary to know the value of this function at the spatial positions corresponding to the vector components.)

Algorithm performance is summarized in Tables 4.2.1 - 4.2.4. Not all cases are listed for each algorithm, due to the failure of the algorithm to converge to a solution in some cases.

For test case 1, without a carrier signal, the peak location and sequential algorithms work equally well. The steepest descent algorithm eventually converges, but it is very slow. Steepest descent results are included here for comparison only. In the presence of a carrier signal, however, the sequential algorithm is the only algorithm of this group that works at all. Moreover, its performance is not affected by the presence of the carrier signal. The peak location algorithm does not work at all in this case, due to the presence of the many local maxima of the carrier signal. The steepest descent algorithm ought to work in this case, however, the iterations displayed diverging oscillations of the error magnitude. It is believed that these oscillations are due to the nonlinearity introduced by forcing the output vector to have only nonnegative components. The carrier signal introduces negative values into the correlation vector, so that this algorithm version no longer represents the true linear steepest descent algorithm.

The peak location algorithm actually exceeds the performance of the sequential algorithm in test case 2, in the absence of a carrier signal. With the correct stepsize choice, this algorithm actually can obtain a solution in one iteration, even with multiple delays. The sequential algorithm, on the other hand, will always require at least as many iterations as there are delays. This case shows the motivation for considering a peak location type of algorithm. However, once again, in the presence of a carrier signal, this algorithm fails to work at all, while the sequential algorithm remains unaffected.

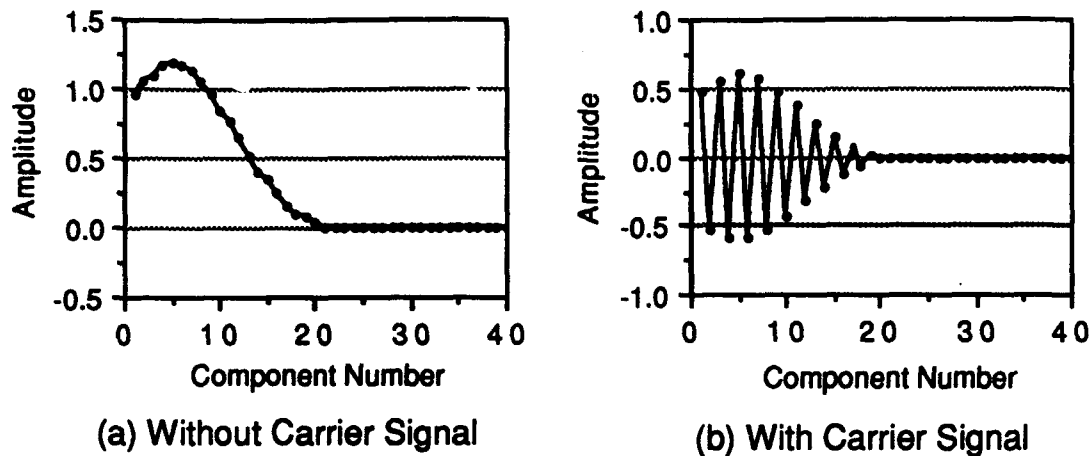


Figure 4.2.1 Correlations for Test Case 1: Gaussian pulse with a single delay

Algorithm	Stepsize	# Iterations	Final Error
Without Carrier Signal			
Sequential	0.5	6	0.004
	0.75	3	0.001
	0.85	1	0.000
	1.00	oscillates*	-
Peak Location	0.5	6	0.004
	0.75	3	0.001
	0.85	1	0.000
	1.00	1	0.000
Steepest Descent	0.125	30+	0.323
	0.15	300 (est.)**	0.25 <
	0.2	oscillates*	-
With Carrier Signal			
Sequential	1.5	3	0.002
	1.75	1	0.000
	2.0	diverges	-

Table 4.2.1 Simulation results for test case 1.

* Error remained bounded, but algorithm produced nonlinear oscillations, failing to converge.

** Iterations continued to converge very slowly. Final number of iterations is estimated; final error is estimated to be smaller than error shown.

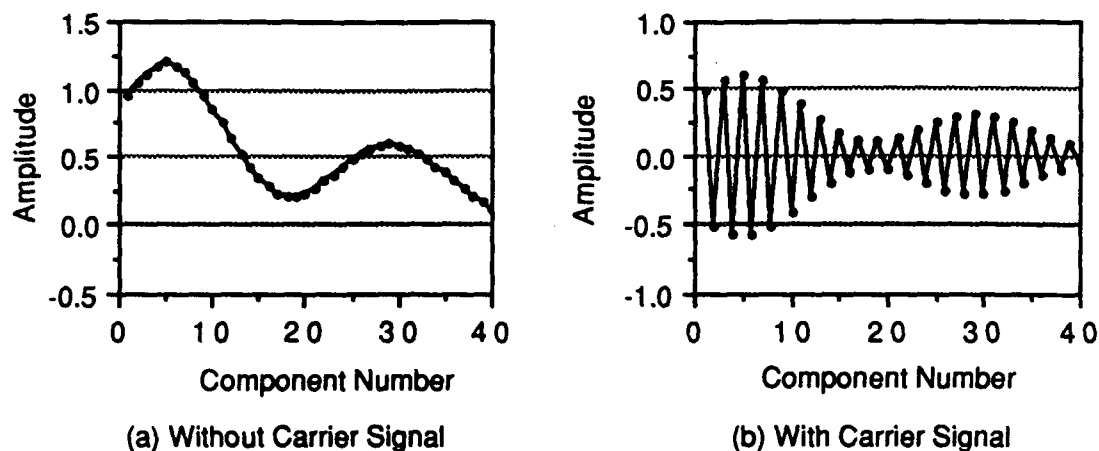


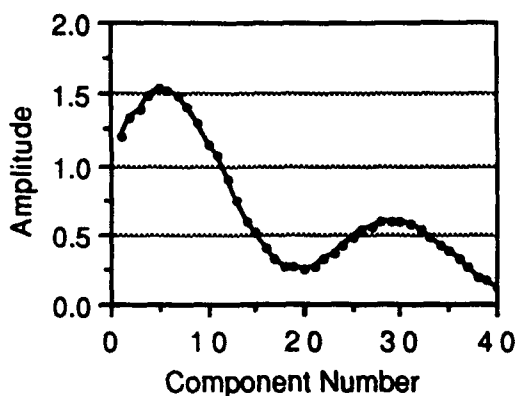
Figure 4.2.2 Correlations for Test Case 2: Gaussian pulse with two delays of unequal amplitude

Algorithm	Stepsize	# Iterations	Final Error
Without Carrier			
Sequential	0.5	7	0.005
	0.75	4	0.005
	0.85	3	0.009
	1.00	oscillates*	-
Peak Location	0.5	6	0.004
	0.85	1	0.009
Steepest Descent	0.15	300 (est.)**	0.25 <
With Carrier			
Sequential	1.75	4	0.002

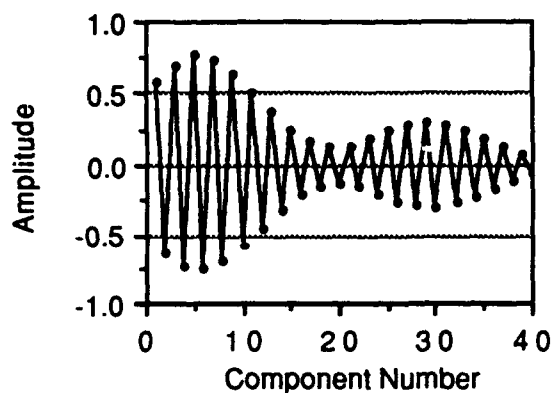
Table 4.2.2 Simulation results for test case 2.

* Error remained bounded, but algorithm produced nonlinear oscillations, failing to converge.

** Iterations continued to converge very slowly. Final number of iterations is estimated; final error is estimated to be smaller than error shown.



(a) Without Carrier Signal



(b) With Carrier Signal

Figure 4.2.3 Correlations for Test Case 3: Gaussian pulse with three delays of unequal amplitudes (two of the delays are closely spaced and produce only one peak in the correlation vector)

Algorithm	Stepsize	# Iterations	Final Error
Without Carrier			
Sequential	0.5	9	0.008
	0.65	7	0.003
	0.75	oscillates*	-
Peak Location	0.5	stuck**	0.298
	0.65	stuck**	0.298
Steepest Descent	0.15	300 (est.)***	0.22 <
With Carrier			
Sequential	1.25	7	0.009

Table 4.2.3 Simulation results for test case 3.

* Error remained bounded, but algorithm produced nonlinear oscillations, failing to converge.

** Further iterations failed to reduce error

*** Iterations continued to converge very slowly. Final number of iterations is estimated; final error is estimated to be smaller than error shown.

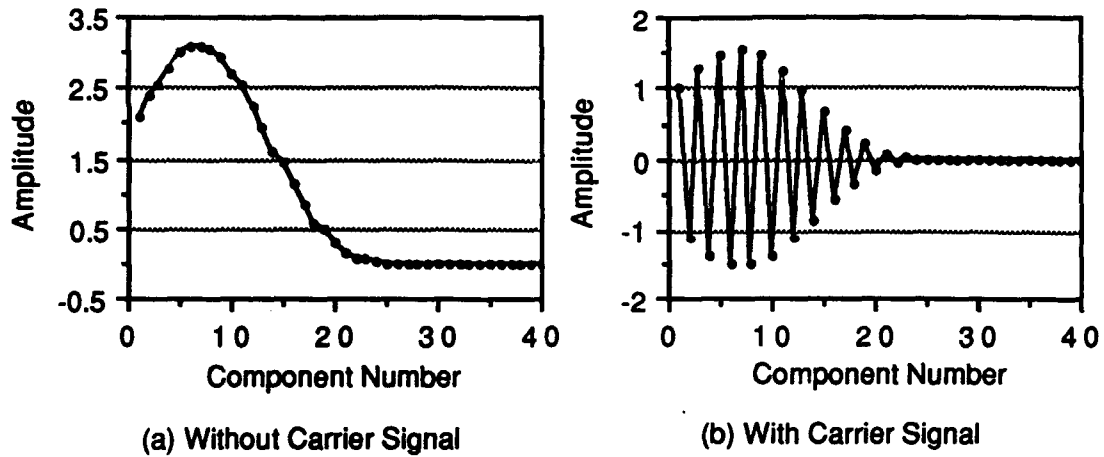


Figure 4.2.4 Correlations for Test Case 4: Gaussian pulse with three closely spaced delays of nearly equal amplitude

Algorithm	Stepsize	# Iterations	Final Error
Without Carrier			
Sequential	0.25	45	0.665
	0.5	50	0.648
Steepest Descent	0.1	120+	0.198**
With Carrier			
Sequential	0.5	48	0.045*
	1.0	27	0.042*
	1.25	20	0.045*
Steepest Descent	0.1	70+	0.339**

Table 4.2.4 Simulation results for test case 4.

* Final error could have been reduced by lowering the error tolerance.

** Iterations continued to converge very slowly. Final number of iterations is estimated; final error is estimated to be smaller than error shown.

Table 4.2.3 shows that the peak location approach can fail even when there is no carrier signal. The difficulty arises here from the fact that the large delay at $d_1 = 1.0 \mu\text{sec}$ dominates the smaller amplitude delay nearby at $d_3 = 1.25 \mu\text{sec}$ (note that these correspond to relative delays of 0.5 and 0.75 μsec , respectively, of the auxiliary signal $n_1(t)$ given by (4.1.2)). The resulting correlation shows only a single peak near the components corresponding to the relative delays 0.5 and 0.75 μsec . The peak location algorithm is never able to resolve the smaller amplitude delay at 0.75 μsec . Note that the steepest descent algorithm eventually outperforms peak location, although it takes many iterations to do so.

The sequential algorithm handles this case both with and without a carrier signal. Further tests showed that this algorithm could handle delays as closely spaced as consecutive weight positions without difficulty.

Finally, Table 4.2.4 shows that the sequential algorithm can fail to provide sufficient interference cancellation in some cases. Reasons for this are put forth in section 5. Closely spaced delay components with similar amplitudes can result in a correlation vector whose maximum components do not correspond to the maximum delay amplitudes. Note, however, that the steepest descent algorithm works in this case, although slowly. It is encouraging to note that the sequential algorithm does perform adequately in this case in the presence of a carrier signal. The reason for this will also be discussed in section 5.

4.3 Conclusions

The sequential algorithm was able to provide a solution for test case 1,2 and 3 with and without a carrier signal, and was able to solve test case 4 in the presence of a carrier signal. The only difficulty for this algorithm occurred in test case 4 without a carrier signal. The algorithm in this case was not able to resolve all of the delays, and as a result provided only partial cancellation. The steepest descent algorithm was able to provide better cancellation in this case, but at the cost of many iterations, as is typical for this algorithm. Given that steepest descent is not an option for implementation on our optical processor, the sequential algorithm is clearly the algorithm of choice. In the next section, we will analyze

this algorithm more closely. Some caveats will be discussed in section 6. However, this appears to be a very promising algorithm for the optical processor.

5 ANALYSIS OF THE SEQUENTIAL ALGORITHM

The results of the previous section show that the sequential algorithm is superior to the other weight vector algorithms in most of the signal scenarios considered. In this section we examine this algorithm more closely in order to gain some insight into its operation.

5.1 An Example

Test case 3 of the previous section provides a good example for showing how the sequential algorithm eventually arrives at the correct solution consisting of three distinct delays with different amplitudes. Figs. 5.1.1 - 5.1.7 show the evolution of the intermediate and output weight vectors through the 7 iterations needed to reach the final solution for this case. For clarity of presentation, we consider only the non-carrier signal case. A stepsize of 0.65 was used (stepsize selection is discussed in the next section). Recall that the interference signal in this case consists of a summation of three copies of the auxiliary signal, with relative delays at 0.5, 0.75 and 3.5 μsec , with amplitudes of 1.0, 0.3 and 0.5, respectively.

Fig. 5.1.1 shows the first iteration. The initial correlation is of the complete interference signal $n(t)$ with the auxiliary signal $n_1(t)$. The output vector \mathbf{W}_0 is initially set equal to the zero vector, so the initial intermediate vector \mathbf{W}_1 , defined by (3.2.2), is just the correlation vector \mathbf{V}_c scaled by the parameter *stepsize*. The number N_s of output components to be selected is initially set at 1. The maximum of this vector is located at component position 5, corresponding to a relative delay of 0.5 μsec . The amplitude of this component is very nearly 1.0. The output weight vector \mathbf{W}_0 thus has a single nonzero component, at position 5, with amplitude close to 1.0.

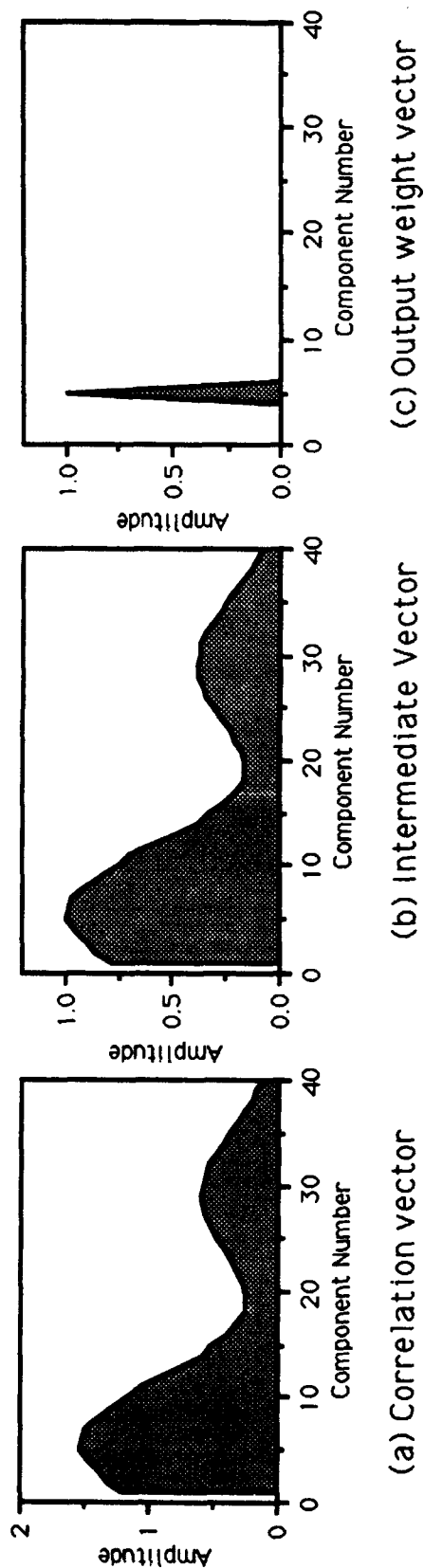


Figure 5.1.1 Iteration 1.1. The dominant correlation component is selected for output.

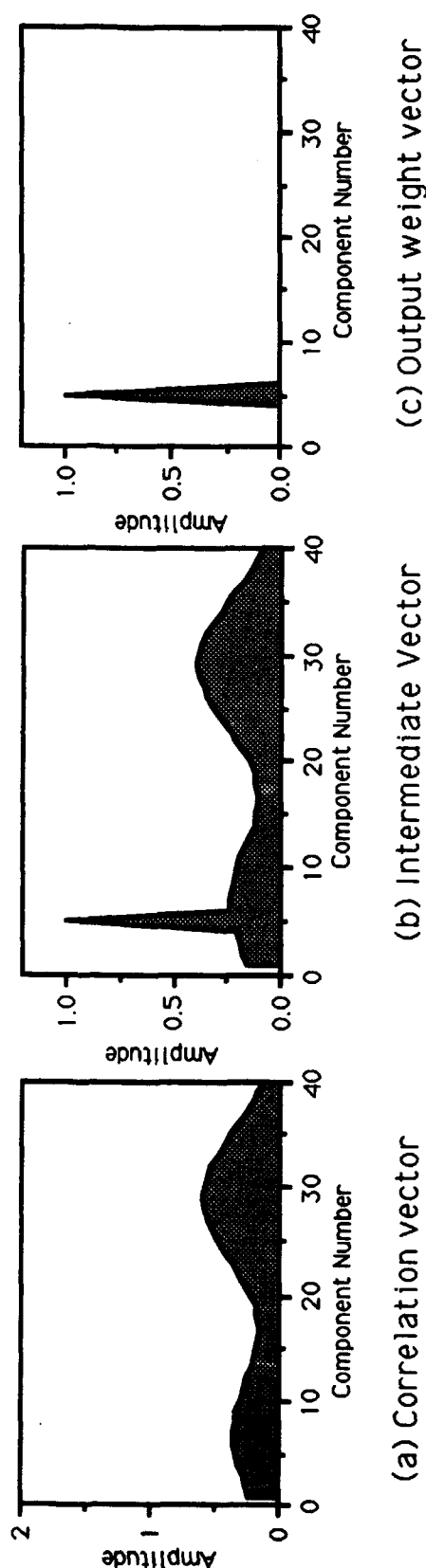


Figure 5.1.2 Iteration 1.2. The correlation vector shows the effect of the removal of the dominant delay component.

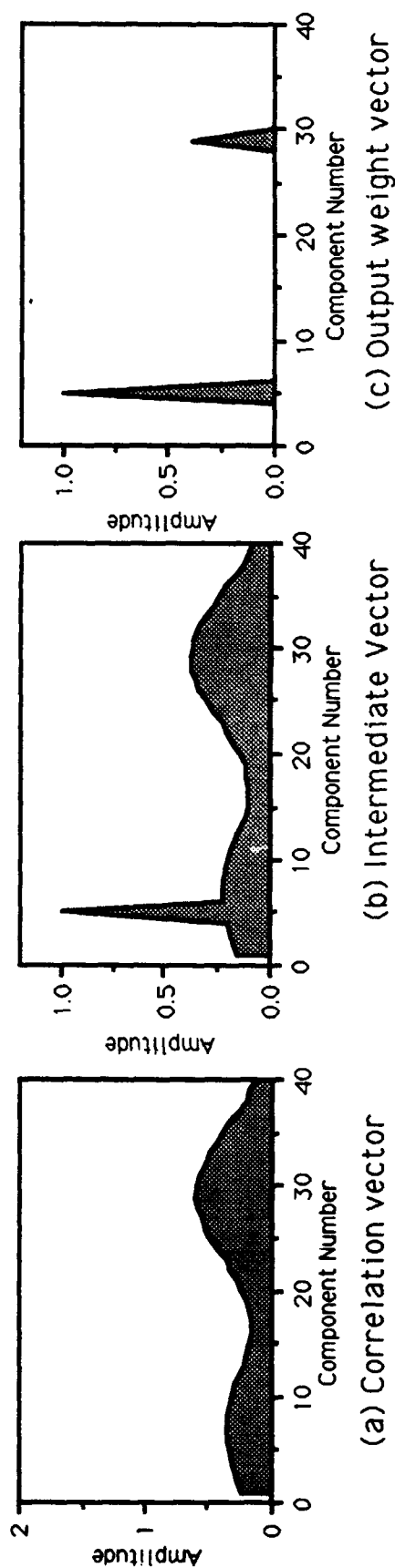


Figure 5.1.3 Iteration 3. A second component is selected from the intermediate vector for output.

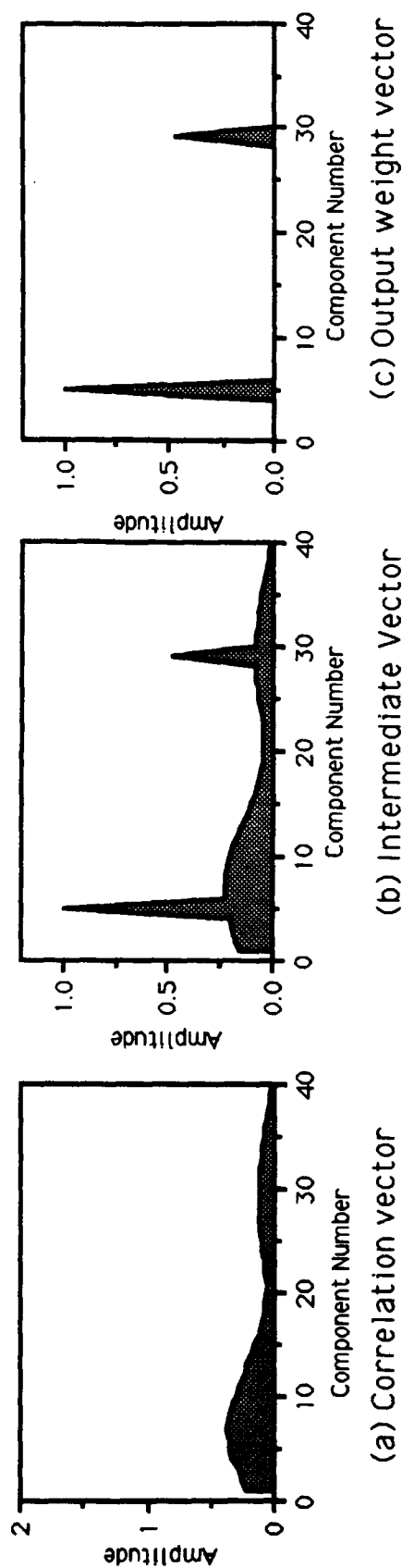


Figure 5.1.4 Iteration 4. The correlation vector amplitude is further decreased as the two largest delay components are cancelled.

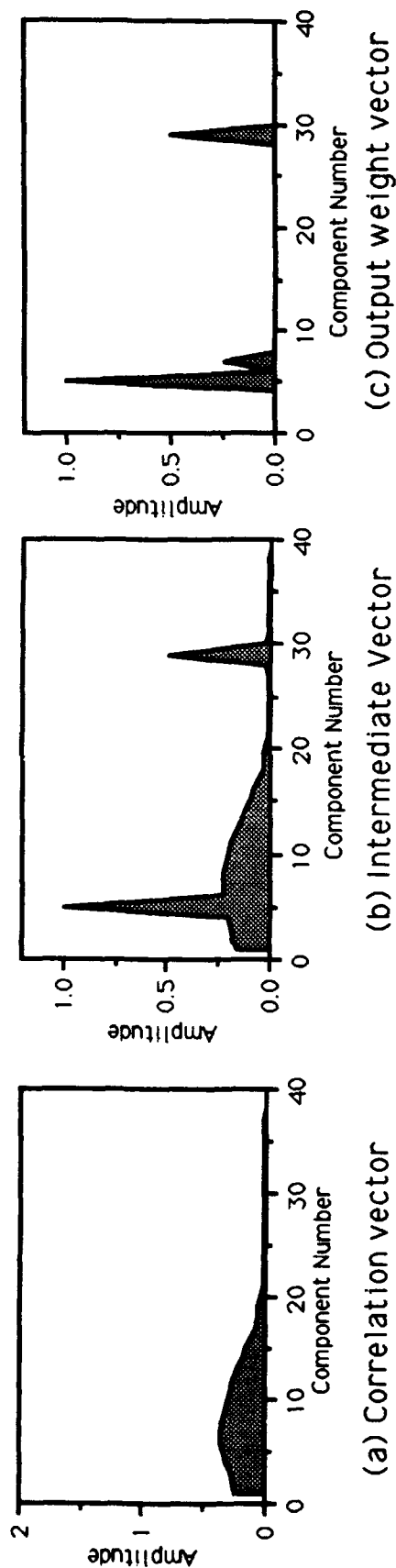


Figure 5.1.5 Iteration 5. A third component is selected from the intermediate vector for output.

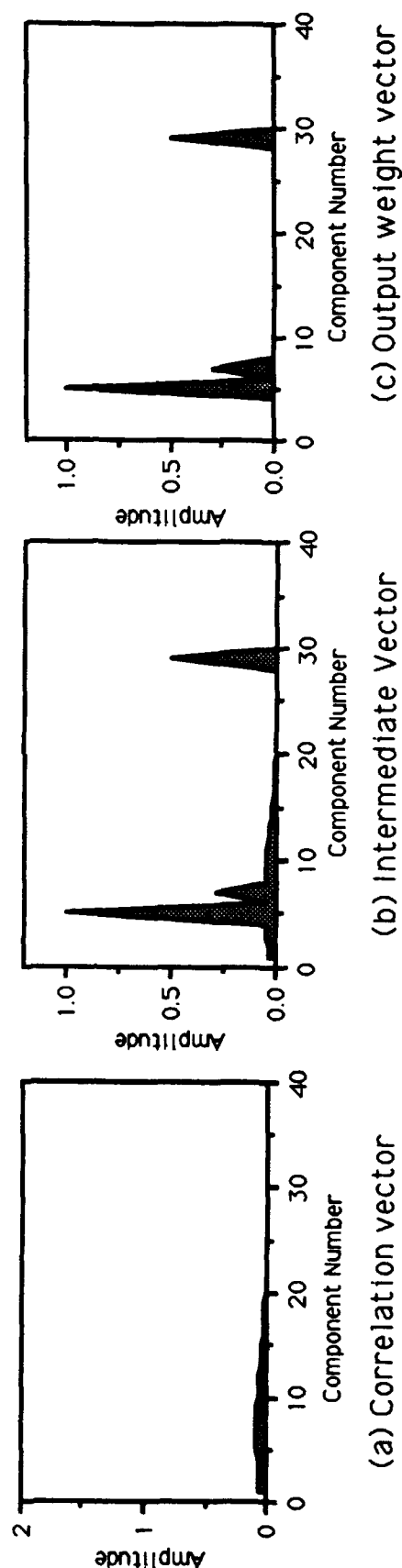


Figure 5.1.6 Iteration 6. The correlation vector is nearly 0 as the three delay components are cancelled.

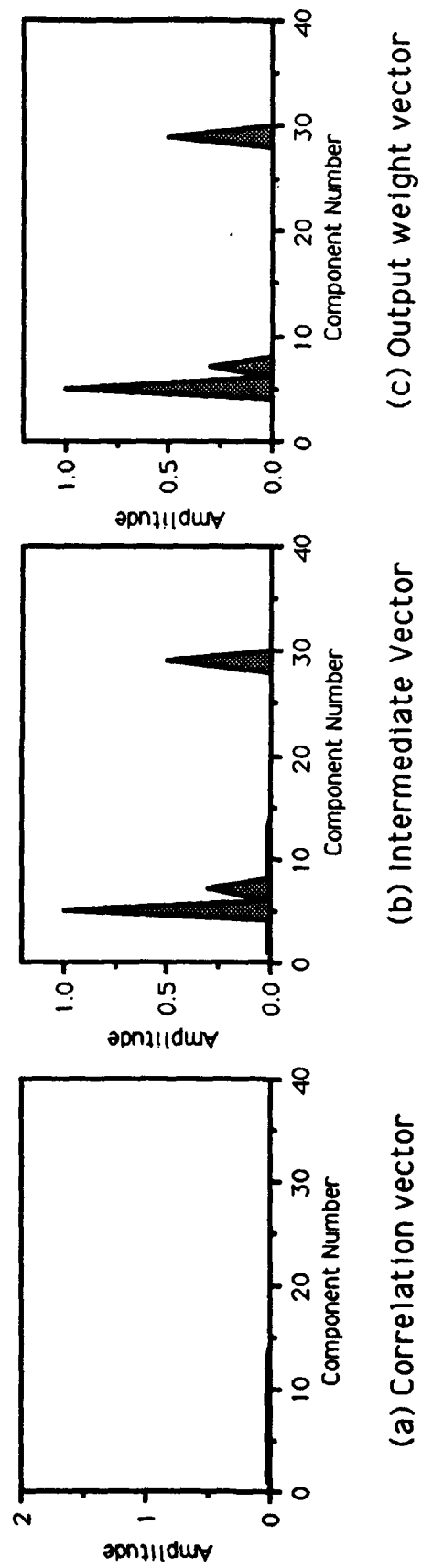


Figure 5.1.7 Iteration 7. The correlation vector shows nearly complete cancellation on the final iteration.

The error signal $e(W_0;t)$ is formed by subtracting the estimate $y(W_0;t)$ from $n(t)$. The resulting correlation of this error signal with $n_1(t)$ is shown in Fig. 5.1.2 (a). The effect of removing the dominant interference component at the relative delay 0.5 can be clearly seen in the reduction in amplitude of the correlation vector. This new correlation vector, scaled by stepsize, is added to the previous output vector W_0 to get the new intermediate vector W_1 . N_s is still 1 at this point, so the maximum of this new intermediate vector is located. This maximum occurs at the same location as in the previous iteration, with little change in amplitude. The new output vector W_0 is thus virtually the same (recall that amplitudes greater than 1.0 are truncated, so the new output amplitude is now exactly 1.0).

Once again, the error signal $e(W_0;t)$ is formed with the new W_0 , and a new correlation vector is computed. The error signal amplitude and the correlation vector amplitude are essentially unchanged since the previous iteration. The effect of this is that the number of output components N_s is incremented to 2. The intermediate vector is formed as before, but now is searched for its 2 largest components, using the search algorithm of section 3.2.4. These occur at component 5, and component 29, which corresponds to a relative delay of 3.5 μ sec. The new output weight vector W_0 displays these 2 nonzero components (Fig. 5.1.3 (c)).

The correlation amplitude can be seen to be further reduced in Fig. 5.1.4 (a) by the introduction of this second output component. Iteration 4 provides additional adjustment to the value of the second output component.

Further refinement of the second output component does not lead to further reduction in the error amplitude or correlation vector amplitude. So, in the fifth iteration shown in Fig. 5.1.5, N_s is incremented to 3, and the three largest components of the intermediate vector are selected. The third component is located at position 7, corresponding to a relative delay of 0.75. The amplitude values are refined in iterations 6 and 7, leading to the final solution. The correlation vector can be seen to be nearly 0 in Fig. 5.1.7 (a).

5.2 Analysis

The sequential algorithm is new, and, as such, is lacking a proof which would delineate conditions under which it will always provide

the correct solution. Clearly, it works for a wide variety of sample problems, as illustrated in the preceding sections. However, the nonlinearity of the algorithm makes analysis difficult. Table 4.2.4 indicates that there are cases where the algorithm fails to provide the optimal solution.

Let us consider the simple situation where the interference $n(t)$ has exactly the form

$$n(t) = \sum_{i=1}^N a_i n_1(t - (i-1)\Delta) \quad (5.2.1)$$

with Δ and N as in section 3.1 (some of the a_i may be 0). The estimate $y(\mathbf{W};t)$ has the form

$$y(\mathbf{W};t) = \sum_{i=1}^N w_i n_1(t - (i-1)\Delta) \quad (5.2.2)$$

for any of the weight vector approaches (some of the w_i may be 0). The correlation component $(\mathbf{V}_c)_k$ (see equation (3.2.2.1)) of $e(\mathbf{W};t)$ with $n_1(t)$ can thus be written as

$$\begin{aligned} (\mathbf{V}_c)_k(t) &= \int_t^{t+T} \left\{ \sum_{i=1}^N (a_i - w_i) n_1(s - (i-1)\Delta) \right\} n_1(s - (k-1)\Delta) ds \\ &= \sum_{i=1}^N (a_i - w_i) \int_t^{t+T} n_1(s - (i-1)\Delta) n_1(s - (k-1)\Delta) ds. \end{aligned} \quad (5.2.3)$$

The correlation vector \mathbf{V}_c can thus be written in matrix-vector form as

$$\mathbf{V}_c(t) = \mathbf{R}(t)(\mathbf{A} - \mathbf{W}) \quad (5.2.4)$$

where $\mathbf{A} = (a_1, \dots, a_N)$, and $\mathbf{R}(t) = (r_{ij}(t))$ is a real symmetric matrix whose ij entry is given by

$$r_{ij}(t) = \int_t^{t+T} n_1(s - (i - 1)\Delta) n_1(s - (j - 1)\Delta) ds. \quad (5.2.5)$$

It can be shown that $r_{ij}(t)$ is a maximum when $i = j$. Also, under appropriate conditions on the range of integration, $r_{ij}(t)$ depends only on $|i - j|$, so that $R(t)$ is a Toeplitz matrix [7] (this fact is not essential, but it does simplify the analysis somewhat). Under these conditions, the diagonal elements have the same value:

$$r(t) = \int_t^{t+T} n_1^2(s) ds \quad (5.2.6)$$

and this value is at least as large as any other entry in $R(t)$.

The sequential algorithm relies heavily on the assumption that the largest amplitude delays will result in the largest correlation components in V_c . Relative to equation (5.2.4), this says that the largest components of $A - W$ should result in the largest components of the product $R(t)(A - W)$. While this assumption is reasonable, and is likely to happen most of the time, unfortunately it is possible to construct fairly simple examples (eg., 3 X 3 matrices) of diagonally dominant Toeplitz matrices $R(t)$ and vectors A in which the maximum components of $R(t)A$ do not correspond to the maximum components of A . With this in mind, it appears unlikely that a convergence proof of this algorithm will be available, except under specialized conditions on the signal $n_1(t)$.

An example of such conditions would be those which would make the matrix $R(t)$ nearly diagonal. This would happen, for example, if the signal $n_1(t)$ were orthogonal to delayed versions of itself over the interval of integration. This would make the integrals equal to 0 in (5.2.5) for $i \neq j$. Such a property is likely to hold for high frequency sinusoid signals. This may be why the sequential algorithm converged in the carrier signal case of test case 4 (Table 4.2.4), while failing in the absence of a carrier signal for the same problem.

The lack of a convergence proof should not detract from the utility of this algorithm. It is well suited to the limited spatial light modulation capabilities of the optical processor for which it was designed, and it works well in the presence of carrier signals, which will be the case in the optical processor.

5.3 Stepsize Selection

The stepsize values used in the simulation results given in section 4 were arrived at through trial and error. It is possible to use the analysis of the previous section to obtain bounds for the stepsize value.

Consider the simple case when all components of \mathbf{A} are 0 except for a single component with value a . Then all components of $\mathbf{R}(t)\mathbf{A}$ are 0 except for a single component with value $r(t)a$. The sequential algorithm will select the corresponding component of the intermediate vector on each iteration. The algorithm is essentially linear in this case with the iterations having the form

$$\mathbf{W}^{(n+1)} = \mathbf{W}^{(n)} + \alpha \mathbf{R}(t)(\mathbf{A} - \mathbf{W}^{(n)}) \quad (5.3.1)$$

where we have denoted the n^{th} iterate of the weight vector by $\mathbf{W}^{(n)}$, and the stepsize by α . In scalar form, the iterations for the one nonzero component of \mathbf{W} are given by

$$\begin{aligned} w^{(n+1)} &= w^{(n)} + \alpha r(t)(a - w^{(n)}) \\ &= (1 - \alpha r(t))w^{(n)} + \alpha r(t) a. \end{aligned} \quad (5.3.2)$$

Once again, the superscript in parentheses indicates iterate. Thus,

$$w^{(n)} = \sum_{j=0}^n (1 - \alpha r(t))^j \alpha a. \quad (5.3.3)$$

If $|1 - \alpha r(t)| < 1$, then the geometric series in (5.3.3) will converge as $n \rightarrow \infty$. Thus, an upper bound on α is

$$\alpha < 1.0/r(t). \quad (5.3.4)$$

This bound will work in the single delay case. A worst case multiple delay scenario might suggest the use of the bound

$$\alpha < 1.0 / \left(\sum_{j=1}^N r_{ij}(t) \right) \quad (5.3.5)$$

The bound in (5.3.5) should be small enough to work in every case, but is probably too small for most cases. In practice, a stepsize somewhere between the bounds given in (5.3.4) and (5.3.5) should be used. Computation of these bounds requires the correlation of $n_1(t)$ with itself (autocorrelation). This will require additional hardware considerations in the optical processor.

6 ISSUES AND CONCLUSIONS

A new algorithm for performing multipath delay estimation and interference cancellation has been introduced in this report. This algorithm has been shaped by the demands and restrictions of implementation in an optical signal processor. The sequential algorithm presented here provides significant performance advantages over traditional adaptive weight vector algorithms such as steepest descent. It can easily be extended to the multichannel case without performance penalty. However, there are some caveats to keep in mind when applying this algorithm.

The sequential algorithm, as with all weight vector approaches, is sensitive to both the resolution of spatial information, and the potential corruption of spatial information. This is particularly true in the presence of a carrier signal. Spatial resolution refers to the density of tap weight positions across the total delay range. Higher spatial resolution results in a smaller Δ . Fortunately, spatial resolution can be increased in the optical processor by using a larger detector array. For best system performance, spatial resolution should be made as high as practical. A large number of tap weight positions will not significantly slow down the sequential algorithm.

Corruption of spatial information refers to a mismatch of spatial information between the first half and the second half of the system. That is, the signal delay associated with a vector component position in the input correlation vector does not equal the signal delay of the corresponding component position in the output vector used to construct $y(W;t)$. The algorithm cannot adapt spatially to correct this mismatch.

This lack of robustness with respect to discrepancies in spatial information can be addressed in two ways. One, of course, is careful calibration of the optical system. This can be done by incorporating a spatial shift parameter into the algorithm, and adjusting its value

while operating the system with signals that differ by known delays. A second approach is to use an algorithm such as parabolic interpolation, mentioned in section 3.3, in conjunction with the sequential algorithm. The idea here is that parabolic interpolation could be used to adapt the single spatial shift parameter, while the sequential algorithm provides the delay and amplitude values. For test purposes, calibration of the system will probably suffice. Given the analog nature of the optical processor, however, it will probably be necessary to include some type of robust spatial adaptivity in any weight vector algorithm.

In addition, we observed in Table 4.2.4 that this algorithm does not converge in every case to an optimal solution. This problem seems to be mitigated in the presence of a carrier signal. Since it is anticipated that the signals encountered by the optical processor will be on carriers, the algorithm should be able to handle most foreseeable signal scenarios.

In conclusion, the sequential algorithm provides a good combination of ease of implementation, real time performance, and ease of application to multiple delay and multichannel scenarios. Steepest descent is not an option due to lack of adequate spatial light modulation. Of the algorithms investigated to date, the sequential algorithm is recommended for implementation in the optical processor for test and prototyping phases.

REFERENCES

- [1] Welstead, S. "Hybrid Electro-Optic Processor", Rome Laboratory Final Technical Report, RL-TR-91-164.
- [2] Welstead, S., "Optical Processor Evaluation", Rome Laboratory Final Technical Report, RL-TR-91-34.
- [3] Ward, M., Keefer, C., Welstead, S., "Adaptive Optical Processor", Rome Laboratory Technical Report, RL-TR-91-270.
- [4] Widrow, B., Stearns, S., *Adaptive Signal Processing*, Prentice Hall, Inc., 1985.
- [5] Ward, M., Keefer, C., Grucza, D., and Welstead, S., "Comparison of spatial light modulators for multipath delay estimation", in *Tech. Digest on Spatial Light Modulators and Applications*, 1990, Opt. Soc. of Amer., Vol. 14, pp. 149-152.
- [6] Welstead, S., Ward, M., Keefer, C., "Neural network approach to multipath delay estimation", *Adaptive Signal Processing*, SPIE Proceedings, Vol. 1565 (1991), pp. 482-491.
- [7] Press, W., Flannery, B., Teukolsky, S., Vetterly, W., *Numerical Recipes*, Cambridge Univ. Press, 1986.

**MISSION
OF
ROME LABORATORY**

Rome Laboratory plans and executes an interdisciplinary program in research, development, test, and technology transition in support of Air Force Command, Control, Communications and Intelligence (C³I) activities for all Air Force platforms. It also executes selected acquisition programs in several areas of expertise. Technical and engineering support within areas of competence is provided to ESD Program Offices (POs) and other ESD elements to perform effective acquisition of C³I systems. In addition, Rome Laboratory's technology supports other AFSC Product Divisions, the Air Force user community, and other DOD and non-DOD agencies. Rome Laboratory maintains technical competence and research programs in areas including, but not limited to, communications, command and control, battle management, intelligence information processing, computational sciences and software producibility, wide area surveillance/sensors, signal processing, solid state sciences, photonics, electromagnetic technology, superconductivity, and electronic reliability/maintainability and testability.