

2

NAVAL POSTGRADUATE SCHOOL Monterey, California

AD-A261 781



THESIS

DTIC
ELECTE
MAR 25 1993

D

**GEOLOCATION OF AN ELECTROMAGNETIC EMITTER
USING A CYCLOSTATIONARY
TIME DIFFERENCE OF ARRIVAL TECHNIQUE**

by

LT Timothy A. Benson

17 December 1992

Thesis Advisor:

Herschel H. Loomis, Jr.

Distribution ~~limited~~ ^{with} to U.S. Government Agencies and their Contractors: Critical Technology; December 1992. Other requests for this document must be referred to Superintendent, Code 043, Naval Postgraduate School, Monterey, CA. 93943-5000 ~~via the Defense Technical Information Center, Cameron Station, Alexandria, VA. 22304-6146.~~

DESTRUCTION NOTICE: Destroy by any method that will prevent disclosure of contents or reconstruction of the document.

93-05972



98 3 23 017

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED		1b. RESTRICTIVE MARKINGS	
2a. SECURITY CLASSIFICATION AUTHORITY		3. DISTRIBUTION/AVAILABILITY OF REPORT Distribution limited to U.S. Government Agencies and their Contractors	
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE		4. PERFORMING ORGANIZATION REPORT NUMBER(S)	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)		5. MONITORING ORGANIZATION REPORT NUMBER(S)	
6a. NAME OF PERFORMING ORGANIZATION Electrical and Computer Engineering	6b. OFFICE SYMBOL (if applicable) EC	7a. NAME OF MONITORING ORGANIZATION Naval Postgraduate School	
6c. ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000		7b. ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000	
8a. NAME OF FUNDING/SPONSORING ORGANIZATION	8b. OFFICE SYMBOL (if applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER	
8c. ADDRESS (City, State, and ZIP Code)		10. SOURCE OF FUNDING NUMBERS	
		PROGRAM ELEMENT NO.	PROJECT NO.
		TASK NO.	WORK UNIT ACCESSION NO.
11. TITLE (Include Security Classification) GEOLOCATION OF AN ELECTROMAGNETIC EMITTER USING A CYCLOSTATIONARY TIME DIFFERENCE OF ARRIVAL TECHNIQUE (U)			
12. PERSONAL AUTHOR(S) Timothy A. Benson			
13a. TYPE OF REPORT Master's Thesis	13b. TIME COVERED FROM _____ TO _____	14. DATE OF REPORT (Year, Month, Day) 1992 December 17	15. PAGE COUNT 88
16. SUPPLEMENTARY NOTATION The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the United States Government.			
17. COSATI CODES		18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)	
FIELD	GROUP	GEOLOCATION;CYCLOSTATIONARY;TDOA;SPECTRAL CORRELATION;CYCLE FREQUENCY;BPSK.SPREAD SPECTRUM	
19. ABSTRACT (Continue on reverse if necessary and identify by block number) Many applications of signal processing require that the location of an electromagnetic emitter be determined. This thesis implements an algorithm developed by Dr. William A. Gardner in 1987, that geolocates such an emitter. Special features of the algorithm not only allow the detection of cyclostationary signals in strong negative signal to noise ratio environments, but also discriminate against unwanted signals not of interest that may either occupy the same frequency band or have similar signal characteristics as the signal of interest. This is accomplished by determining spectral correlation densities for cycle frequencies equal to various exploitable features of the bpsk direct sequence spread spectrum signal.			
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT <input checked="" type="checkbox"/> UNCLASSIFIED/UNLIMITED <input type="checkbox"/> SAME AS RPT. <input type="checkbox"/> DTIC USERS		21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED	
22a. NAME OF RESPONSIBLE INDIVIDUAL Herschel H. Loomis, Jr.		22b. TELEPHONE (Include Area Code) (408) 656-3214	22c. OFFICE SYMBOL EC/LM

Distribution ^{Auth} limited to U.S. Government Agencies and their Contractors: Critical Technology; December 1992. Other requests for this document must be referred to Superintendent, Code 043, Naval Postgraduate School, Monterey, CA 93943-5000 ~~via the Defense Technical Information Center, Cameron Station, Alexandria, VA 2304-6145.~~

Geolocation of an Electromagnetic Signal Using a
Cyclostationary Time Difference of Arrival Technique

by

Timothy A. Benson
Lieutenant, United States Navy
B.S.E.E., University of New Mexico, 1986

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING
(with a concentration in Space Systems Engineering)

from the

NAVAL POSTGRADUATE SCHOOL

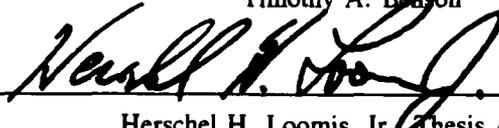
December 1992

Author:

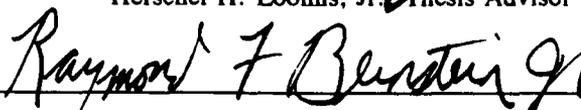


Timothy A. Benson

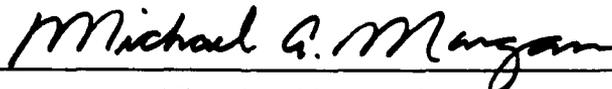
Approved by:



Herschel H. Loomis, Jr., Thesis Advisor



Raymond F. Bernstein, Jr., Second Reader



Michael A. Morgan, Chairman

Department of Electrical and Computer Engineering

ABSTRACT

Many applications of signal processing require that the location of an electromagnetic emitter be determined. This thesis applies an algorithm developed by Dr. William A. Gardner in 1987, that geolocates such an emitter. Special features of the algorithm not only allow the detection of cyclostationary signals in strong negative signal to noise ratio environments, but also discriminate against unwanted signals not of interest that may either occupy the same frequency band or have similar signal characteristics as the signal of interest. This is accomplished by determining spectral correlation densities for cycle frequencies equal to various exploitable features of the bpsk direct sequence spread spectrum signal.

Accession For	
NTIS	CRA&I <input type="checkbox"/>
DTIC	TAB <input checked="" type="checkbox"/>
Unannounced	
Justification	
By	
Distribution /	
Availability Codes	
Dist	Avail and/or Special
C-2	

DTIC QUALITY INSPECTED 1

TABLE OF CONTENTS

I. INTRODUCTION	1
A. BACKGROUND	1
B. OVERVIEW	2
II. SIGNAL CHARACTERISTICS	5
A. SPREAD SPECTRUM	5
B. BINARY PHASE SHIFT KEYING (BPSK)	8
C. BPSK-DSSS SIGNAL	10
D. SUMMARY	10
III. SPECTRAL CORRELATION	12
A. DETECTING HIDDEN PERIODICITY	12
B. BIFREQUENCY PLANE	15
C. APPLICATIONS	17
IV. TDOA	19
A. OBTAINING A FIX	19
B. GEOLOCATION	20

V. SPECCOA ALGORITHM	22
A. INTRODUCTION	22
B. THEORETICAL DEVELOPMENT	24
C. ALGORITHM TESTING	26
1. SSPI Software Package	26
2. Testing Overview	26
3. α_0 Equals Chip Rate	28
4. α_0 Equals Twice the Carrier Frequency	30
5. α_0 Equals Twice the Carrier Frequency Plus the Chip Rate	30
6. Interference of Other Signals	31
VI. CONCLUSIONS	40
APPENDIX A. HYPERBOLA DEVELOPMENT	42
APPENDIX B. TEST CASE GRAPHS: α_0 EQUALS CHIP FEATURE	45
APPENDIX C. TEST CASE GRAPHS: α_0 EQUALS TWICE CARRIER	
FREQ.	52
APPENDIX D. TEST CASE GRAPHS: α_0 EQUALS TWICE CARRIER FREQ.	
PLUS CHIP FEATURE	58

APPENDIX E. COMPUTER PROGRAMS	64
A. SIGNAL GENERATION SOFTWARE	64
B. GENERATING SPECTRAL CORRELATION DENSITIES	65
LIST OF REFERENCES	78
INITIAL DISTRIBUTION LIST	80

I. INTRODUCTION

A. BACKGROUND

Many applications of signal processing require that the direction or the location of the source of a signal of interest be determined. This requirement could be for a variety of reasons. Active systems such as radar, transmit a signal and evaluate the reflected energy to determine first if a contact exists, and then the contact's bearing and range. From this information contact heading and speed can be elicited, and the contact tracked. Passive systems, on the other hand, do not transmit, but rather receive signals generated entirely from other sources. Submarines, for example, use passive sonar techniques to process noise levels from all directions to determine their exact bearing, and then, through various maneuvers of the ship, an exact position of a particular signal of interest. Ascertaining the location of various signals of interest is necessary for collision avoidance, primarily, and tracking and targeting, secondly. This type of system uses an array to determine the signal's direction, or bearing relative to the ship. An output is produced from each element of the array in sequence as the signal's wavefront passes over it. The time delay from the first element's output to the last element's output is electronically calculated and a beam produced in the direction of the signal.

The system described in this paper is also a passive system but receives electromagnetic energy rather than sound pressure energy as in sonar. And like the sonar system characterized, a time delay is determined and converted to a Line of Position.

Obtaining two or more Lines of Position pinpoints the origin of the signal of interest. This is called Geolocation when the source of the signal of interest is on the Earth.

Applications for this type of system are many. Locating a distress beacon on the ground from a downed commercial airplane in rugged territory, rescuing a pilot shot down over enemy territory, finding violators of communication regulations, communication surveillance of illegal activities such as drug smuggling operations or of hostile forces during armed conflict or even during peacetime, and of course, military signal intelligence gathering [Ref. 3].

B. OVERVIEW

Figure 1-1 gives a general idea of the problem addressed in this paper. An emitter transmits an electromagnetic signal that is received by a pair of receivers, the platform receiver pair. The emitter could be either a previously unknown signal source, or a known source that has changed position. In either case, the problem is first to detect the signal and then geolocate the source of its emission.

Detecting the signal is not necessarily a simple task. Other signals not of interest and noise interfere with and mask the signal of interest, as shown in Figure 1-2. The signals not of interest

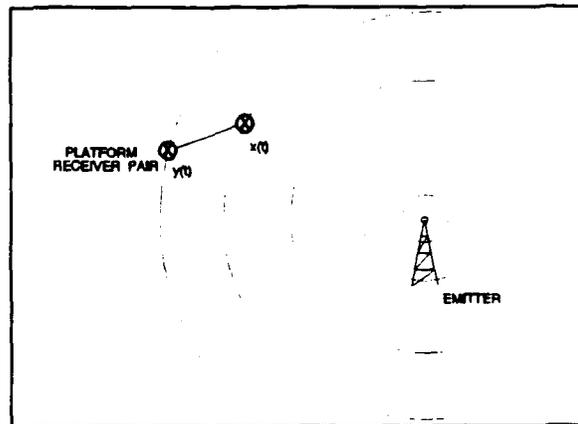


Figure 1-1 General scenario of geolocation problem.

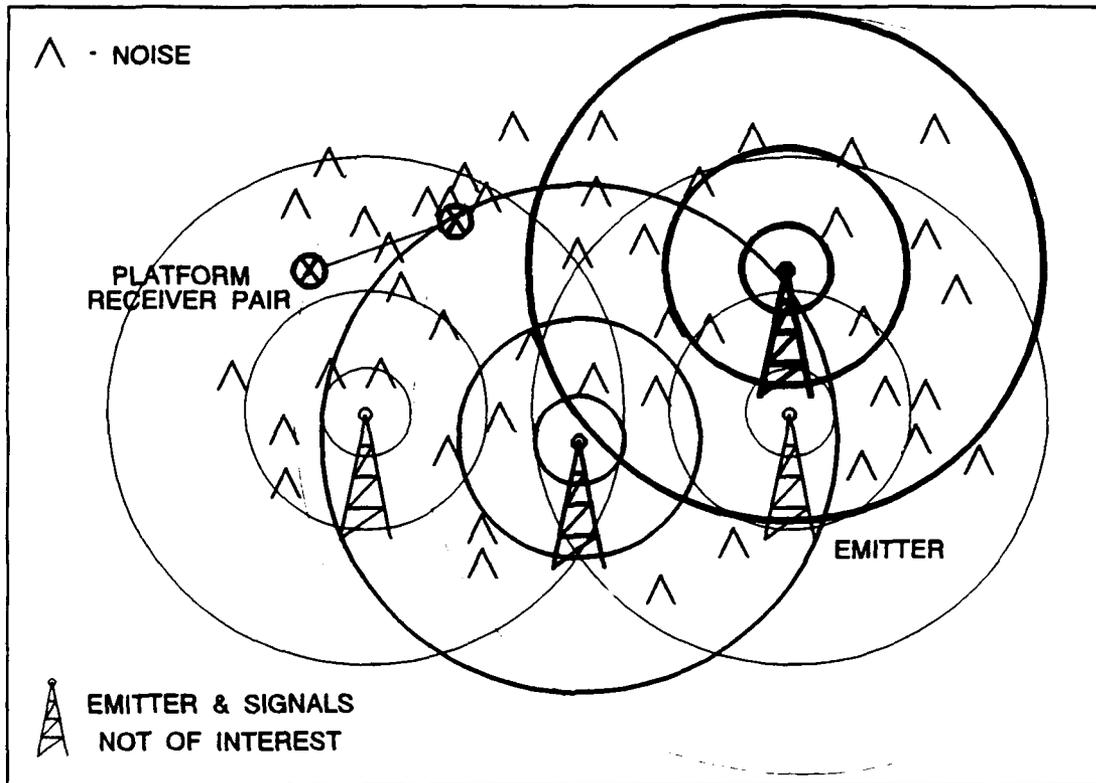


Figure 1-2 Real-world scenario showing interfering noise and signals not of interest.

might occupy the same frequency band as the signal of interest and might also be at a higher power level. In addition, noise further hides the signal of interest by "cluttering" the electromagnetic spectrum, making it more difficult to detect and extract the desired signal. Since the type of signals that are of interest in this paper are Direct Sequence Spread Spectrum (DSSS) signals, and are usually imbedded in noise, special techniques must be employed to recover these signals from the noise.

In the following chapters specific problems of detecting DSSS signals are discussed and special methods that are used to overcome them introduced. Then, having established the groundwork for detecting DSSS signals, a method to geolocate the signal

of interest, the Spectral Coherence Alignment (SPEC COA) algorithm, is described that is especially tolerant to noise and interfering signals not of interest. Results of testing the SPEC COA algorithm using computer generated test signals are presented. Varying degrees of noise are used ranging from positive signal to noise ratios (SNR) to establish a baseline reference, to negative signal to noise ratios, which means that the noise level is greater than the signal. In addition, interfering signals are inserted to test the algorithm's tolerance to signals not of interest.

II. SIGNAL CHARACTERISTICS

This chapter gives a brief description of the characteristics of the type of signal studied in this paper, a binary-phase shift key (BPSK), direct sequence spread spectrum (DSSS) signal. Some knowledge of the type of signal being studied and its characteristics is necessary to facilitate a better understanding of the information presented in later chapters.

A. SPREAD SPECTRUM

Spread Spectrum (SS) is defined as a signal whose *transmitted* bandwidth is substantially wider than the bandwidth of the information itself. In addition, the increased bandwidth must be caused by a spreading signal or code called the modulation which must also be known by the recipient in order to despread the signal. SS technology has been in existence for over thirty years. A SS signal has many features that make its use desirable for communications in general and military applications in particular. It provides jamming resistance, interference rejection, multiple-access capability, and low probability of intercept (LPI) capability. [Ref 6: p. 404][Ref 8: p. 2-1]

Using SS techniques, the energy from a signal is spread over a wide band of frequencies by means of a spreading code. The spreading before transmission and subsequent despreading at the receiver provides a spread spectrum signal with its

desirable characteristics. Orthogonal spreading codes allow numerous signals to be transmitted simultaneously on the same frequency band. A potential receiver selects the desired signal by despreading with the correct code while other signals, because of their unique codes, are left undisturbed. This is code division multiple access (CDMA). Interference with narrow-band signals is minimized because the SS signal's power is spread over such a large bandwidth that the in-band narrow-band signal to spread spectrum signal ratio is very large. This explains the other desirable features of SS signals. Since the power level is very low at each frequency over which the signal is spread, the signal has a low probability of detection and/or intercept (LPD/LPI), and an interfering or potential jamming signal is spread during the despreading process of recovering the desired signal. This gives the signal of interest a high processing gain which necessitates a much stronger interfering or potential jamming signal in order to prevent reception of the signal of interest.

Some categories of Spread Spectrum are:

- Frequency Hop (FH)
- Stacked Carrier
- Direct Sequence (DS)
- Hybrid

Frequency Hop Spread Spectrum transmits communication signals by rapidly hopping between several carrier frequencies, numbering from just a few to many thousands. These systems can be either a fast or a slow hopper, depending on whether several data bits are transmitted per hop frequency (slow hopper) or several hop

frequencies are used per data bit (fast hopper). This method makes the signal more jam resistant. In addition, it is difficult to collect the entire signal, giving it a low probability of intercept. [Ref. 8]

Stacked carrier signals provide extremely reliable communications by transmitting simultaneously on two or more frequencies. This increases anti-jamming resistance and lowers the error rate of transmission. Two drawbacks of using stacked carrier are its high signal to noise ratio making it susceptible to interception and its inefficient use of the frequency spectrum.

Direct Sequence Spread Spectrum uses a pseudo-random spreading sequence at a chip frequency, f_c , to modulate the carrier (frequency f_o), and spread a signal over a wide frequency band. See Figure 2-1.

The degree of spreading is a function of the chip frequency and can be determined by dividing the chip frequency by one-half of the original bandwidth, or spreading factor = $2f_c/BW_o$, where BW_o is the bandwidth of the original signal.

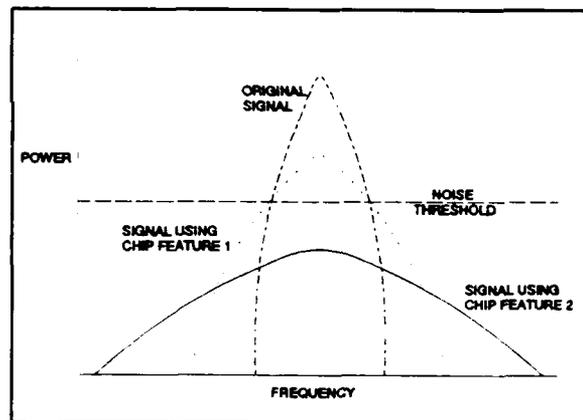


Figure 2-1 Illustration of chip features spreading original signal and lowering its peak power.

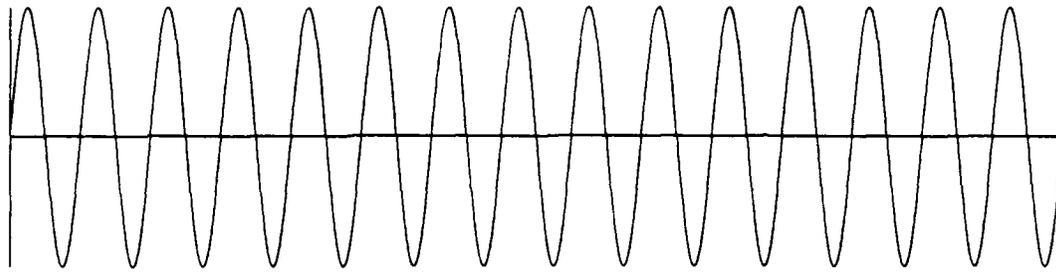
Note also that the power is spread over a wide frequency band and the peak power is lowered, potentially to below that of the noise threshold. Thus, the spreading can be to such a degree that any background noise masks the final transmitted signal. Using this method makes detection of the signal more difficult because an ordinary receiver

perceives only noise. In addition, the signal is harder to jam since a jammer would have to jam numerous frequencies and the process of despreading the signal tends to spread the jamming signal as described previously.

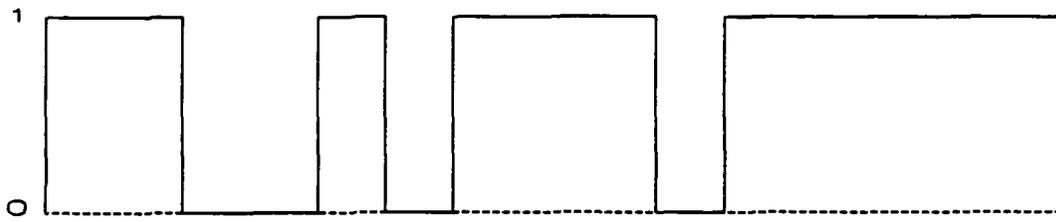
The Hybrid spread spectrum signal uses a combination of other spread sequence techniques, the most common of which is the frequency hop/direct sequence hybrid. Frequency hopping is used to achieve the same large bandwidth of a direct sequence signal but with a lower chip rate. This is desirable since it is more difficult to achieve the large bandwidth using direct sequence methods alone. [Ref. 8]

B. BINARY PHASE SHIFT KEYING (BPSK)

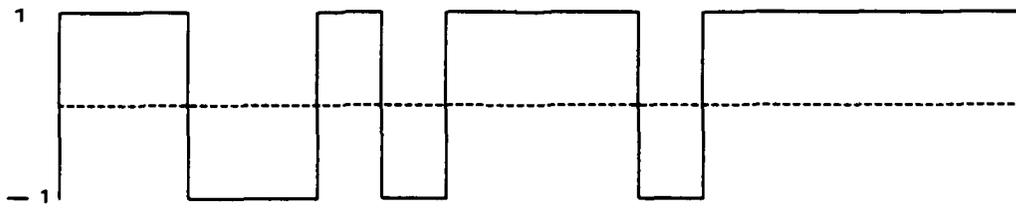
A BPSK signal is produced by shifting the phase of a sinusoidal carrier 180° with a binary data sequence. See Figure 2-2. Shown is the sinusoidal carrier signal, $\cos(\omega_c t)$, and the unipolar data sequence, $d(t)$, which is shifted to a bipolar signal, $m(t)$, to modulate the carrier, resulting in the signal to be transmitted, $m(t)\cos(\omega_c t)$. The effect of the modulating signal is to shift the phase of the carrier by 180° each time the data bit changes from a binary 1 to a binary 0, or vice versa. [Ref. 6: pp. 332-336] The number of data bits per second is the data rate or bit frequency, f_b , of the signal. Also of note is the carrier frequency of the signal, $f_c (= \omega_c / 2\pi)$. The bandwidth of the transmitted signal is then $[(f_c + f_b) - (f_c - f_b)]$ or $2f_b$. In the next section it is shown how BPSK and DSSS are combined to produce a signal with a substantially increased bandwidth.



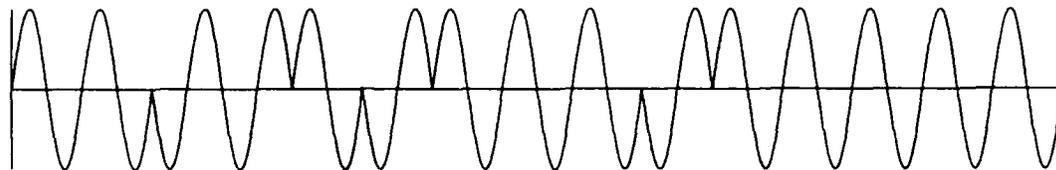
carrier signal, $\cos(\omega_c t)$



binary data sequence, $d(t)$



shifted binary data sequence, $m(t)$.



modulated carrier signal, $m(t)\cos(\omega_c t)$

Figure 2-2 BPSK signal production.

C. BPSK-DSSS SIGNAL

Important features of a BPSK-DSSS signal are the carrier frequency, f_o , the chip rate feature or spreading code rate, f_c , and the bandwidth of the transmitted signal. A spreading code similar in appearance to that of the binary data sequence shown in Figure 2-2, is modulo-2 added to the data sequence and shifted to obtain a bipolar binary modulating signal for the carrier frequency.¹ Differences between the data sequence and the spreading code are the frequency (f_c is substantially greater than f_o) and the randomness of the binary sequence - the best spreading codes are those that are as near random as possible, and of course the data sequence is not random since it contains the information to be transmitted.

The result is a transmitted signal with the same carrier frequency as before, f_o , but with a significantly increased bandwidth equal to $[(f_o + f_c) - (f_o - f_c)]$ or $2f_c$. The ratio of the new bandwidth to the original bandwidth is f_c/f_o which is called the processing gain. There is also a reduction of the peak power of the BPSK-DSSS signal as compared to the original signal (see Figure 2-1, above) by the same factor. It is called the processing gain since it is the increase in power that the signal gains during the recovery process.

D. SUMMARY

This chapter describes the various features associated with the signal of interest addressed in this paper. Special intercept techniques are needed to detect signals such

¹Modulo-2 is XOR addition: $0 \oplus 0 = 0$, $0 \oplus 1 = 1$, $1 \oplus 0 = 1$, $1 \oplus 1 = 0$.

as these that are below the noise threshold. W. A. Gardner in a good survey article [Ref. 3] discusses the exploitation of spectral redundancy in cyclostationary signals that allow processing signals that are deeply imbedded in noise. This property, which is inherent to manmade communication signals, enables a sifting through of extraneous noise and interfering signals to piece together and recover the desired signal. [Ref. 4] introduces several methods that use cyclostationarity to geolocate a signal of interest. These items are discussed in more detail in Chapter 3.

III. SPECTRAL CORRELATION

Cyclostationarity is a spectral redundancy feature that allows the detection of signals deeply imbedded in noise, especially those signals that are Direct Sequence Spread Spectrum signals. This chapter covers the theory associated with this property and how it is used to uncover hidden signals.

A. DETECTING HIDDEN PERIODICITY

This theoretical development is taken from W. A. Gardner's *Statistical Spectral Analysis, A Nonprobabilistic Theory*, [Ref. 7, pp. 359-362], which can be studied for a more detailed treatment of the subject. Spectral lines can be generated by putting a signal through a quadratic time-invariant (QTI) transformation, e.g., $H[s(t)s(t - \tau)]$.

Starting with a signal $x(t)$ which contains a *finite-strength additive sinewave component* (an ac component) with frequency α

$$a \cos(2\pi\alpha t + \theta), \quad \alpha \neq 0, \quad (3.1)$$

if the Fourier coefficient

$$M_x^\alpha = \lim_{T \rightarrow \infty} \frac{1}{T} \int_{-T/2}^{T/2} x(t) e^{-j2\pi\alpha t} dt \quad (3.2)$$

exists and is not zero, then

$$M_x^\alpha = \frac{1}{2}ae^{i\theta}. \quad (3.3)$$

In this instance, the power spectral density of $x(t)$ includes a spectral line at frequency $f = \alpha$ and its image $f = -\alpha$. In other words the power spectral density contains the additive term

$$|M_x^\alpha|^2 [\delta(f - \alpha) + \delta(f + \alpha)] \quad (3.4)$$

where $\delta(\bullet)$ is the impulse function. See Figure 3-1. The spectral lines of $x(t)$ at $f = \pm\alpha$ are due to the finite additive sine wave component. [Ref. 8][Ref. 3, pp. 16,17]

Rewriting $x(t)$ in terms of (3.1) and lumping all else that $x(t)$ may contain into $n(t)$ gives

$$x(t) = a \cos(2\pi\alpha t + \theta) + n(t) \quad (3.5)$$

where $n(t)$ is random. The signal $x(t)$ can be said to have hidden periodicity if $n(t)$ is much greater than the sine-wave component making it not readily discernable with a perfunctory observation. However, due to the spectral lines at $f = \pm\alpha$ the hidden periodicity can be detected by examining

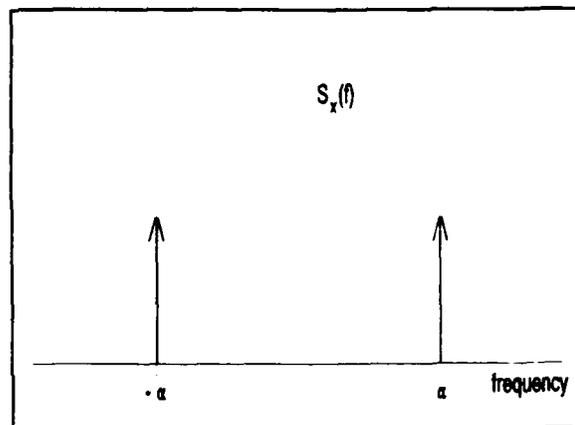


Figure 3-1 Graph of the power spectral density with spectral lines at $\pm\alpha$.

the signal more carefully. [Ref. 3, pp. 16, 17]

This signal has first-order periodicity with frequency α and its spectral components can be detected without any special transformation of the original signal. Many man-made signals, however, contain hidden, higher-order periodicities that do not generate spectral lines at the frequencies of the hidden periodicities. Using a simple squarer quadratic transformation, i.e. $x^2(t) = x(t)x(t)$, some signals can be changed into a first-order periodicity exposing their hidden spectral components. For other signals, the simple squarer transformation does not work and a better transformation is one that uses a time delay such as

$$y(t) = x(t)x(t - \tau) \quad (3.6)$$

for various nonzero delays τ [Ref. 3, p. 18]. This delayed transformation gives rise to the cyclic autocorrelation function. It can be shown [Ref. 7, chap. 10] that the signal $x(t)$ contains second-order periodicity at frequency α if

$$R_x^\alpha(\tau) = \lim_{T \rightarrow \infty} \int_{-T/2}^{T/2} x(t + \frac{\tau}{2})x(t - \frac{\tau}{2})e^{-j2\pi\alpha t} dt \quad (3.7)$$

exists and is not zero for some non-zero α . R_x^α is defined as the cyclic autocorrelation of $x(t)$ for cycle frequency parameter α . The definition of this function comes from the Fourier coefficient of the additive sinewave component with cycle frequency α . Substituting (3.6) into (3.2) gives

$$M_y^\alpha = \langle y(t) e^{-j2\pi\alpha t} \rangle = \langle x(t)x(t - \tau) e^{-j2\pi\alpha t} \rangle, \quad (3.8)$$

where $\langle \bullet \rangle$ is the time average

$$\lim_{T \rightarrow \infty} \frac{1}{T} \int_{-\tau/2}^{\tau/2} (\bullet) dt. \quad (3.9)$$

Adjusting the arguments for symmetry yields

$$M_y^\alpha = \langle y(t) e^{-j2\pi\alpha t} \rangle = \langle x(t + \frac{\tau}{2})x(t - \frac{\tau}{2}) e^{-j2\pi\alpha t} \rangle, \quad (3.10)$$

which leads to the definition of $R_x^\alpha(\tau)$ shown in (3.7). Taking the cyclic autocorrelation function one step further leads to the spectral correlation density and the development of the bifrequency plane.

B. BIFREQUENCY PLANE

From statistical processing it is known that the power spectral density and the autocorrelation function are fourier transform pairs, i.e.,

$$S_x(f) = \int_{-\infty}^{\infty} R_x(\tau) e^{-j2\pi f\tau} d\tau \quad (3.11)$$

and

$$R_x(\tau) = \int_{-\infty}^{\infty} S_x(f) e^{j2\pi f\tau} df \quad (3.12)$$

Similarly, it can be shown [Ref. 7, p. 389] that the cyclic spectral correlation density and the cyclic autocorrelation function are also Fourier transform pairs,

$$S_x^\alpha(f) = \int_{-\infty}^{\infty} R_x^\alpha(\tau) e^{-j2\pi f\tau} d\tau \quad (3.13)$$

and

$$R_x^\alpha(\tau) = \int_{-\infty}^{\infty} S_x^\alpha(f) e^{j2\pi f\tau} df \quad (3.14)$$

where α is the cycle frequency. Using this concept a correlation can be taken in the frequency domain of a signal $x(t)$. The correlation parameter is called the cycle frequency α , and a plot of the frequency vs. the cycle frequency produces the bifrequency plane. Figures 3-2 and 3-3 show how this plane is developed. Starting with the frequency domain signal shown in Figure 3-2, a correlation is taken between frequencies $(f+\alpha/2)$ and $(f-\alpha/2)$ as α , the cycle frequency, varies from $-f_s$ to $+f_s$, where f_s is the sample frequency. Mathematically the product $X(f + \alpha/2)X^*(f - \alpha/2)$ is taken and the magnitude plotted on the bifrequency plane shown in Figure 3-3, at coordinates (f, α) . This unsmoothed bifrequency plane is called the cyclic periodogram. The

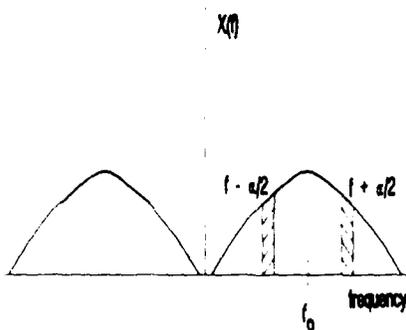


Figure 3-2 Frequency domain signal showing correlation being taken at $(f \pm \alpha/2)$.

unsmoothed bifrequency plane may be smoothed either in time or frequency to minimize variance effects and noise [Ref. 8: Ch. 13]. Figure 3-4 is a plot of a typical bifrequency plane of a BPSK signal with carrier frequency f_0 . Important features on this plot are the power spectrum shown for

cycle frequency equal to zero, the spreading code or chip rate features (the four smaller peaks), and the carrier feature, the large peak occurring at cycle frequency equal to twice the carrier frequency, or $\alpha = 2f_c$.

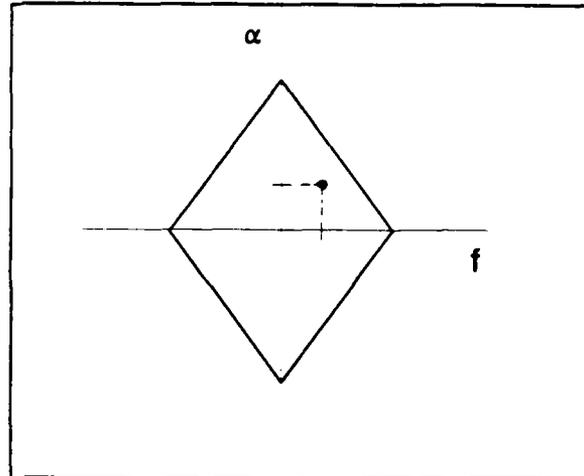


Figure 3-3 General shape of the bifrequency plane. Magnitudes of the correlation are plotted at coordinates (f, α) as f varies over the signal's bandwidth, and α varies from $-2f_s$ to $+2f_s$.

C. APPLICATIONS

There are numerous applications of

spectral correlation. Some of these [Ref. 8: pp. 30-34] are:

- Detection and classification of signals.
- Parameter estimation such as carrier frequency and keying rate.
- Spatial filtering.
- Direction finding.
- Frequency shift filtering for signal extraction.
- Frequency-shift prediction.
- Time difference of arrival.

[Ref. 8] gives further information on specific applications. The last item, time difference of arrival, is the subject of this thesis. Using the spectral correlation densities for the cycle frequency not equal to zero, as shown in Figure 3-4, lines of position are developed and the location of the emitter of the signal determined. Determining a line

of position is discussed next in Chapter IV, and the specific algorithm used to calculate it is presented in Chapter V.

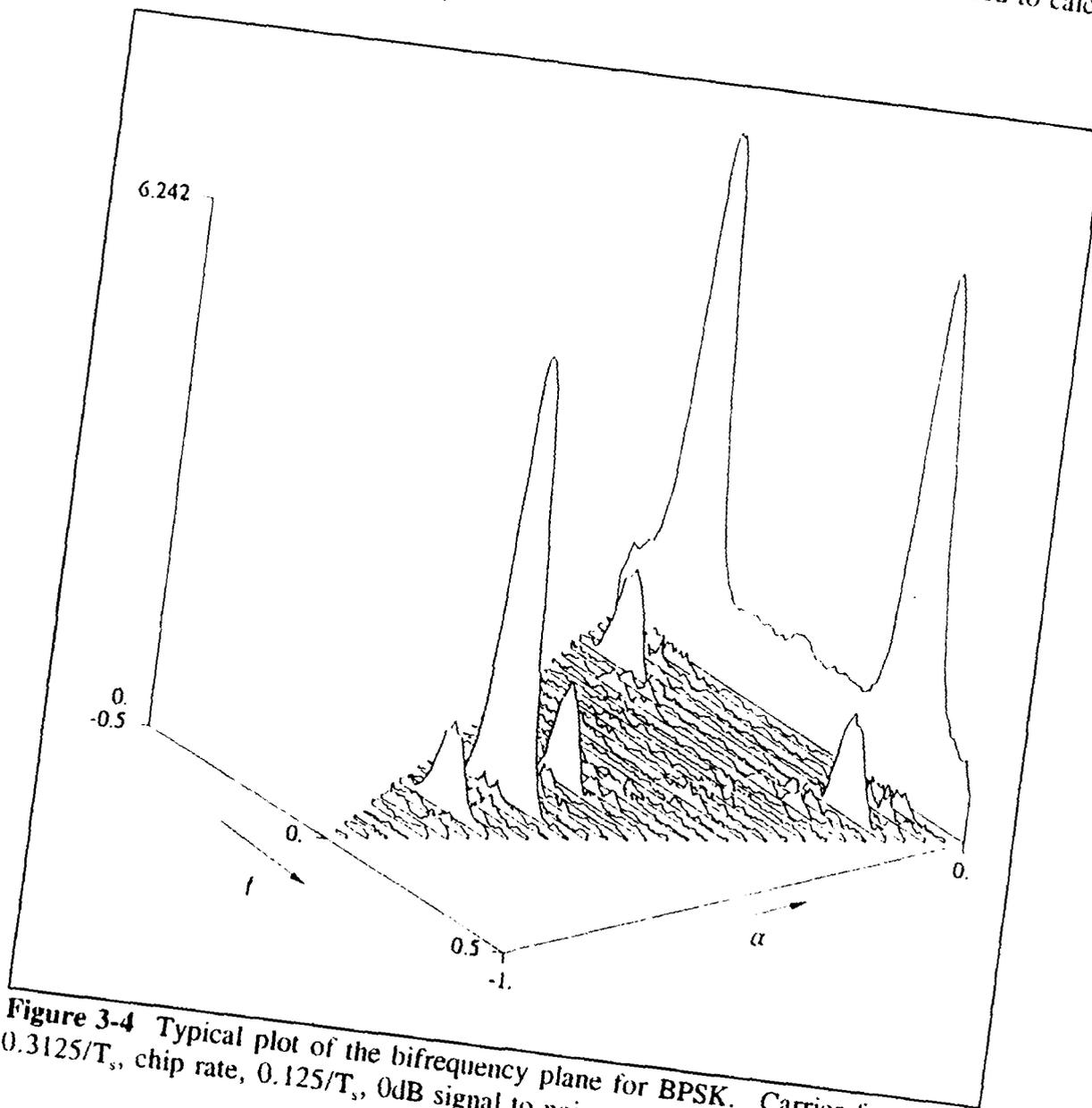


Figure 3-4 Typical plot of the bifrequency plane for BPSK. Carrier frequency, $0.3125/T_s$, chip rate, $0.125/T_s$, 0dB signal to noise ratio.

IV. TDOA

A. OBTAINING A FIX

In Naval terms, to obtain a fix means to determine a ship's position. Several methods exist that are used. One such system developed during World War II, is the LORAN electronic navigation system and is briefly mentioned here because its method of fixing a ship's position is similar to the method used in this thesis to geolocate an emitter. LORAN, from long range navigation, produces hyperbolic lines of position based on the time-delay *difference* between signals received from a pair of land-based transmitting stations [Ref 1: pp. 345-348]. Figure 4-1 depicts this situation. A pair of

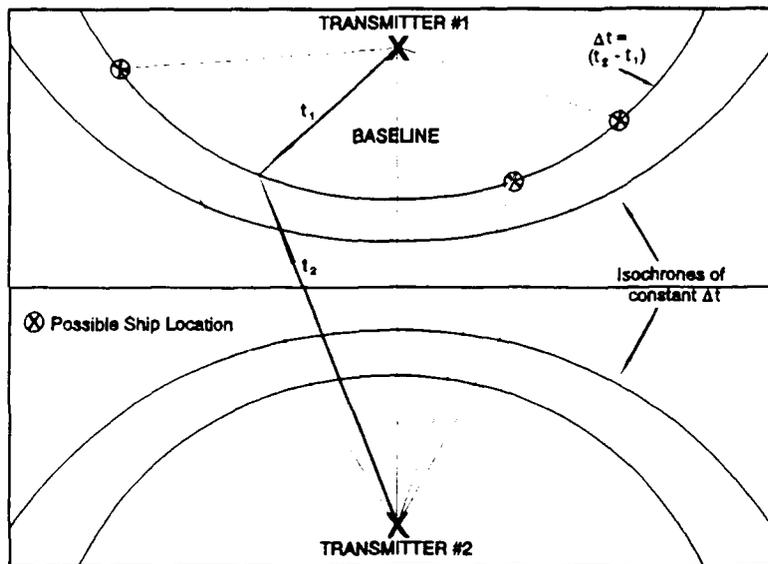


Figure 4-1 Depiction of hyperbolic isochrones of constant Δt .

fixed transmitters separated by a known distance along the baseline transmits signals that produce hyperbolic¹ isochrones. Isochrones are plotted on special navigation charts and the correct isochrone is determined based on the

¹A hyperbola is a mathematical term defined as the locus of points whose difference in distance from two fixed points is constant.

transmitter pair's unique identification number and the time of arrival difference between the received signals, i.e., the Δt or difference between t_2 and t_1 .² A ship receives these signals in an attempt to fix its position. Possible ship positions could be anywhere along the indicated hyperbola; three possible locations are marked. This establishes a single line of position. Repeating the procedure for a second pair of transmitters gives a second line of position. The intersection of the two hyperbolic isochrone fixes the ship's position, as in Figure 4-2.

B. GEOLOCATION

In the above scenario, the location of the transmitters is already known since their positions are printed on navigation charts. However, for the problem addressed in this thesis, the situation is reversed, that is to say, the signal is a single point origin, rather than a pair of transmitters, and the receivers are a pair

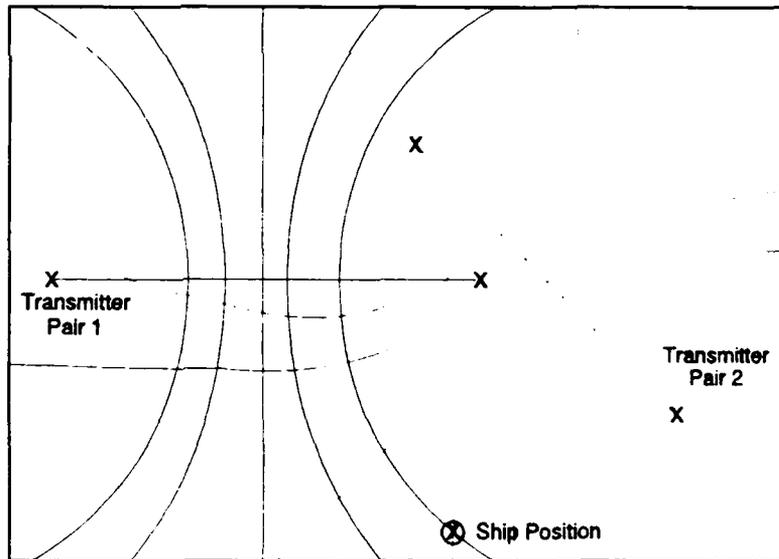


Figure 4-2 Intersection of two Isochrones to get a Fix.

²The two transmitters are designated Master and Slave since the Slave transmitter does not transmit until the Master station transmission reaches the Slave station. This avoids position ambiguities near the line exactly between the two transmitters. If both stations transmitted simultaneously, there would be zero time-delay difference exactly on the center line, and if the ship was not exactly on the center line, ambiguity would exist as to which side of the line the ship was actually located, and a position error would result.

of antenna platforms separated by a known distance. See Figure 4-3. In this situation, each platform receives the same signal from an unidentified source but at a different time. At time t_0 , if the signal at receiver platform one is $x(t)$, and the signal at receiver platform two is $y(t)$, then $y(t) = x(t - d)$ where d is the delay or difference of times between the signal arriving at platform one, $x(t)$, and platform two, $y(t)$. The difference yields a hyperbola of possible emitter locations, the hyperbolic line of position at time t_0 . A second hyperbolic line of position is obtained a short time later at time t_1 as the platform receiver pair moves a short distance from its original position. The intersection of the two hyperbolas geolocates the emitter source.

Appendix A provides a mathematical proof to show that the above described scenario does indeed produce a hyperbola. Next, Chapter V describes the algorithm that produces the time delay to obtain the hyperbolic lines of position just discussed.

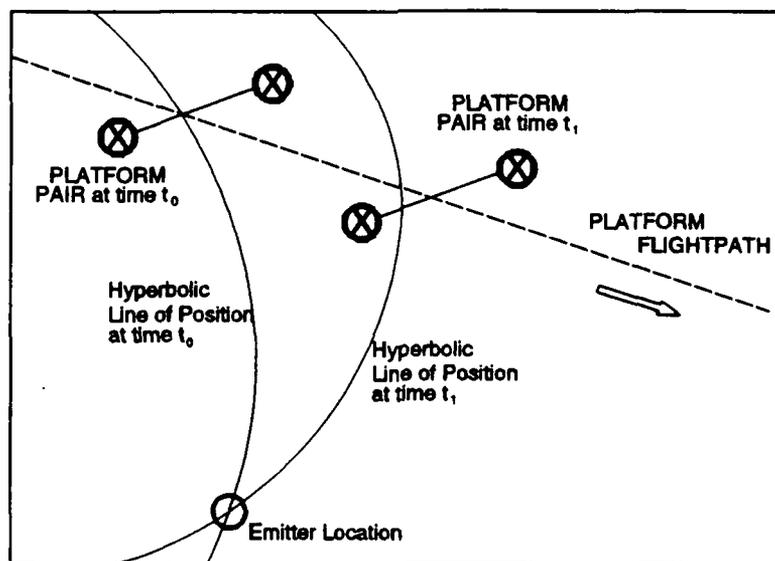


Figure 4-3 Moving antenna platforms Geolocating an emitter.

V. SPECCOA ALGORITHM

A. INTRODUCTION

In Chapter III, it is shown how a spectral redundancy feature called cyclostationarity is used to detect signals that are deeply imbedded in noise, i.e., signals with a negative signal-to-noise ratio (SNR). Once the signal is detected by the platform receiver pair it can be further processed by Time Difference of Arrival (TDOA) techniques to geolocate the new signal. Using the cyclostationary properties of the signal of interest, the auto-spectral correlation density for the signal at the first receiver,

$S_x^\alpha(f)$, and the cross-spectral correlation density for the signals at both receivers,

$S_{yx}^\alpha(f)$, for a specific cycle frequency, α_o , are obtained. These three elements are the

main ingredients needed for use in the SPEcTral COherence Alignment (SPECCOA)

TDOA algorithm which is the implementation of the following equation:

$$D = \underset{\tau}{\text{arg max}} \operatorname{Re} \left\{ \int_{-\pi/2}^{\pi/2} S_{yx}^{\alpha_o}(f) S_x^{\alpha_o}(f)^* e^{j2\pi f\tau} df e^{j\pi\alpha_o\tau} \right\} \quad (5.1)$$

Here $S_x^{\alpha_o}(f)^*$, denotes the *conjugate* of the auto-spectral correlation of the signal x .

Describing the SPECCOA algorithm in words, the product of the two spectral correlation densities is inverse Fourier transformed. The result is multiplied by a correction factor, $e^{j\pi\alpha_0\tau}$, the last term of equation (5.1), to ensure the maximum value of the algorithm occurs at the time delay, d . Finally, the maximum value of the real part is taken. This value is the output of the algorithm and is the delay between when the signal of interest arrives at receiver one, $x(t)$, and when it arrives at receiver two, $y(t)$. i.e., the delay, d , in $y(t) = x(t - d)$. See Figure 5-1 below. Depicted is a typical plot

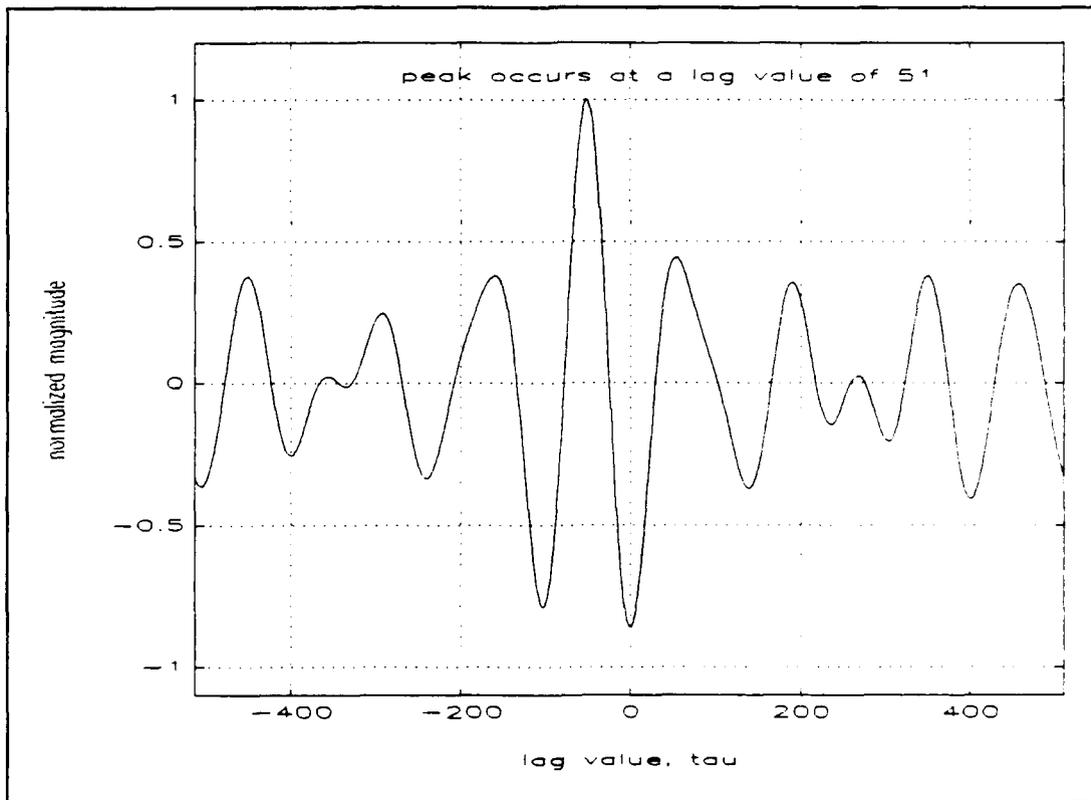


Figure 5-1 Typical plot of SPECCOA algorithm output.

of equation (5.1). A negative lag is shown; a positive lag indicates that the signal first arrived at the other receiver. A lag of 51, meaning 51 samples since the signal is

digitally processed, is shown for this example. The maximum lag value is converted to an actual time delay by multiplying the lag value by the sampling period. Using the time delay, a hyperbola is generated representing the line of position as described in Chapter IV. A second time delay from the algorithm generated a short time later produces a second hyperbolic line of position. The intersection of these two lines of position geolocates the emitter of the signal of interest.

B. THEORETICAL DEVELOPMENT

This section shows that the output of the SPECCOA algorithm of equation (5.1) is indeed the desired time delay, d , between the signals $x(t)$ and $y(t) = x(t - d)$. The proof begins by starting with an equivalent expression for the cross-spectral correlation density [Ref. 8: ch. 7], $S_{yx}^{\alpha_o}(f)$, and making appropriate substitutions.

$$S_{yx}^{\alpha_o}(f) = Y_{1/\Delta f}(t_o, f + \alpha_o / 2) X_{1/\Delta f}^*(t_o, f - \alpha_o / 2) \quad (5.2)$$

where $1/\Delta f$ is the size of the FFT and indicates frequency smoothing. Since $y(t) = x(t - d)$, and using f' to distinguish from the argument f in (5.2), then

$$Y_{1/\Delta f}(t_o, f') = X(t_o, f') e^{-j2\pi f' d} \quad (5.3)$$

which is a property of the Fourier transform. Substituting (5.3) into (5.2) gives

$$S_{yx}^{\alpha_0}(f) = X_{1/\Delta f}(t_0 f + \alpha_0/2) X_{1/\Delta f}^*(t_0 f - \alpha_0/2) e^{-j2\pi(f + \alpha_0/2)d} \quad (5.4)$$

or

$$S_{yx}^{\alpha_0}(f) = S_{xx}^{\alpha_0}(f) e^{-j2\pi(f + \alpha_0/2)d} \quad (5.5)$$

since by definition, $S_x = X \cdot X^*$, but here it is for a specific alpha, α_0 .

Using equation (5.5), the inverse Fourier Transform of the cross-spectral and auto-spectral correlations is taken.

$$\begin{aligned} & \mathcal{F}^{-1} \{ S_{yx}^{\alpha_0}(f) S_x^{\alpha_0}(f) \} \\ &= \int_{-\infty}^{\infty} S_x^{\alpha_0}(f) S_x^{\alpha_0}(f) e^{-j2\pi(f + \alpha_0/2)d} e^{j2\pi f \tau} df \end{aligned} \quad (5.6a)$$

$$= \left[\int_{-\infty}^{\infty} |S_x^{\alpha_0}(f)|^2 e^{-j2\pi f d} e^{j2\pi f \tau} df \right] e^{-j\pi \alpha_0 d} \quad (5.6b)$$

The bracketed term, an inverse Fourier transform, has a maximum value when the variable factor, $e^{-j2\pi f d} e^{j2\pi f \tau} = e^{j2\pi f(\tau - d)}$, is maximum. This occurs when the lag, τ , is equal to the delay, d . Finally, multiplying the last term of (5.6b) by $e^{j\pi \alpha_0 \tau}$ produces the desired SPECCOA equation. This last product produces a second variable term, $e^{j\pi \alpha_0(\tau - d)}$. It is also maximum when the lag is equal to the delay. Thus, the output of

the SPECCOA algorithm is the delay between signals $x(t)$ and $y(t)$. [Ref 4: pp. 1174, 1177]

C. ALGORITHM TESTING

1. SSPI Software Package

The SPECCOA algorithm was tested using simulated signals generated by a software package from Statistical Signal Processing, Inc. (SSPI). This is a comprehensive package that generates a variety of simulated signals. Input values allow setting of sample size, cycle frequency, carrier frequency, desired signal delay, signal power, noise power, and several other items not listed here.¹ The simulated signals are processed by the software package to obtain spectral and cross-spectral correlation densities for the user specified cycle frequency. The results are stored in data files for further processing at the users discretion or for plotting using the SSPI *sxaf_plot* command. For the application of this paper, the data for the spectral and cross-spectral correlation densities was input into the SPECCOA algorithm.

2. Testing Overview

Testing was performed on the exploitable features shown in Figure 5-2: the spreading frequency or chip rate feature, and the carrier frequency feature. These features are exploited by setting the cycle frequency, α_c , equal to the specific feature to be exploited. Thus, to exploit the chip rate, the four smaller peaks, α_c is set to f_c or to

¹ The complete list of items is given in Appendix E.

$2f_0 \pm f_c$, and to exploit the carrier frequency, f_0 , the larger peak, α_0 , is set to $2f_0$. Since $f_0 + f_c$ and $f_0 - f_c$ are essentially the same, a total of three sets of tests were performed; $\alpha_0 = f_c$, $\alpha_0 = 2f_0$, and $\alpha_0 = f_0 + f_c$. Each set and their results is described next. One note that is pertinent to each set of tests is the SNR input. Generally, the SNR referred to is the ratio, in

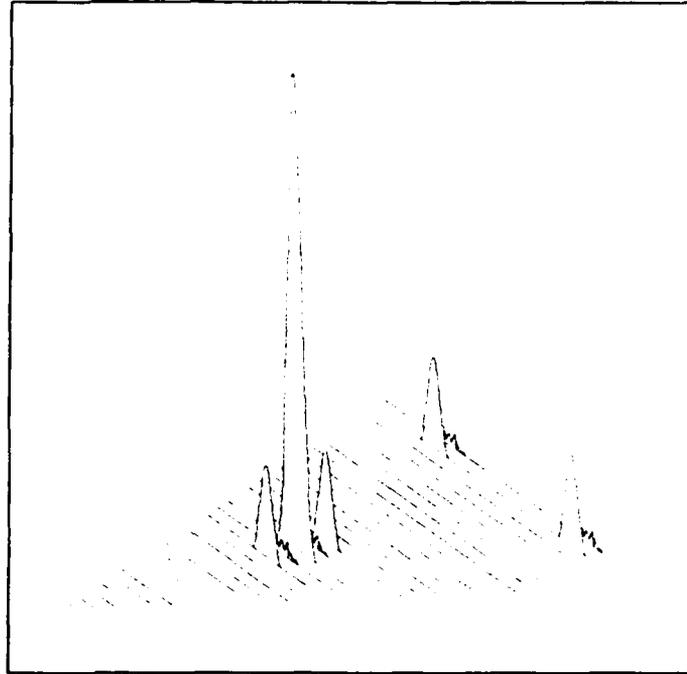


Figure 5-2 Bifrequency plot of exploitable cycle frequency features.

dB, of the signal input power to the total noise power which is a distinct input in generating the test signal as described in Appendix E. The final SNR is actually somewhat larger since the broadband noise is essentially filtered out when the SPECCOA algorithm calculates a delay, and only the inband-noise affects the final SNR. The final SNR can be determined by calculating the signal power to inband noise ratio: $SNR_{final} = (P_s/P_n) \div f_c$, where P_s and P_n are the input signal and noise power inserted into the signal generation program, and f_c is the digital spreading code rate or chip feature. Thus for an input SNR of -15dB and a f_c of 0.0625 the actual in-band SNR is: $(-15dB) - 0.0625$ or $\left(\frac{1}{31.6}\right)(16) = 0.506$ or -3dB which means that the inband SNR is 12dB greater than

the input total SNR for a chip feature of 0.0625. A summary of the increase in SNR over the input broadband SNR for all the chip frequencies used is shown in Table 5-1.

TABLE 5-1
SNR INCREASE OF INBAND SNR OVER
INPUT TOTAL SNR FOR ALL CHIP RATES
USED IN THE TEST SIMULATIONS

f_c^1	SNR INCREASE	f_c^1	SNR INCREASE
0.015625	18dB	0.2625	5.8dB
0.03125	15	0.032	4.9
0.0625	12	0.3225	4.91
0.125	9	0.38	4.2
0.20	7	0.3825	4.17
0.26	5.9	0.4025	4.0

$^1x_x/T_s$

3. α_0 Equals Chip Rate

Table 5-2 shows the variables input into the signal generation program along with a sample of typical values used. Other variables, shown in the complete list of Appendix E, were not changed. Testing consisted of varying the delay, percent smoothing, chip feature and the broadband SNR. The sample length is listed in Table 5-2 even though it was not a variable tested, because it was varied in response to changing the SNR in order to obtain an acceptable output. Complete results of this test set are presented in Appendix B.

Table 5-2 INPUT VARIABLES INTO SIGNAL GENERATION PROGRAM

input delay	% smoothing ¹	no. of freq samples ²	chip feature, f_c^3	carrier frequency, f_o^3	SNR. dB
101	0.01	2048	0.0625	0.16	-5

¹ $\Delta t \Delta f = (\% \text{ smooth}/100)(\text{no. of freq samples})$

²input is power of two; samples going to *spec* program is (pwr of 2 input)(1 - α_n)

³ f_c/f_s

One particular result of note is the percent smoothing variation.² In order to first detect a signal for further examination, a significant amount of smoothing is required, on the order of 5%. However, test results show that very little smoothing is required of the signal for insertion of the spectral correlation density into the SPECCOA algorithm. Figures 5-4 through 5-7 show the effects of 5% and 0.01% smoothing ($\Delta t \Delta f = 1$ and 410, respectively) for input total SNR's ranging from 0dB to -15dB. Clearly, to detect a signal and determine if any features are available for further exploitation a high degree of smoothing is required. Examining the 0.01% smoothing plots would not reveal whether anything was exploitable. But, for input into the SPECCOA algorithm, the 0.01% smoothing is more desirable. Figure 5-8 shows the effect on the algorithm's performance for increasing smoothing. Even smoothing as little as 0.39% causes erroneous results. Inband SNR for Figures 5-4 through 5-8 is 12dB higher than the input SNRs shown as explained earlier.

$$\% \text{ smoothing} = \frac{\Delta f}{f_s} \cdot 100$$

General conclusions for results of this first test set are that the algorithm worked well for varying parameters of delay, chip feature and carrier frequency for input SNRs down to -15dB corresponding to an inband SNR of -3dB for a chip feature of $0.0625/T_s$. Lower SNRs can be achieved by increasing the sample length, see Figure 5-9, but of course, the time to obtain an output from the algorithm increases due to the amount of calculations necessary. Thus, the cycle frequency equal to the chip rate feature is an exploitable feature.

4. α_c Equals Twice the Carrier Frequency

The results of this test set closely resemble those of set number one. One major difference is the overall smoother appearance of the output graphs. This is due to a greater degree of zero-padding required to reach the next higher power of two before inverse Fourier transforming. Complete results of this test set are presented in Appendix C. General conclusions for this test set are the same as for set number one: cycle frequency equal to twice the carrier frequency is an exploitable feature for BPSK signals, but because of the extra smoothing due to zero-padding, results are more accurate.

5. α_c Equals Twice the Carrier Frequency Plus the Chip Rate

The results of this test set are presented in Appendix D. General results are similar to those described above, however, a different output appearance is evident. This is due to truncation down to the next lower power of two before inverse Fourier transforming, and, as with the second test case, the results are generally more accurate

than test case one. Overall conclusion: cycle frequency equal to $2f_0 + f_c$ is suitable for exploitation of BPSK signals.

6. Interference of Other Signals

The testing of the algorithm's ability to discriminate against signals not of interest was done using four types of interference: Multiple signal interference, wide-band signal interference, coband interference, and narrow-band interference. Figure 5-3 shows how the spectral correlation is able to distinguish between different signals. Shown is the wide-band interferer case. Compare with Figure 5-2. The parameters of the interfering signals are shown in Table 5-3, and Figure 5-9 shows the output results. The algorithm was able to sift through extraneous signals not of interest and correctly determine the delay.

**TABLE 5-3 SIGNAL PARAMETERS
FOR SIGNAL INTERFERENCE TEST CASE**

	signal type	signal power	carrier frequency ¹	chip rate ¹	input delay
MULTIPLE INTERFERENCE	S1	0dB	0.201	0.0196	700
	S2	0dB	0.121	N/A	216
	S3	0dB	0.312	N/A	351
WIDE-BAND INTERFERENCE	bpsk	0dB	0.21875	0.10	700
COBAND INTERFERENCE	bpsk	0dB	0.16	N/A	700
NARROW-BAND INTERFERENCE	bpsk	0dB	0.128	0.016	700
SIGNAL OF INTEREST	bpsk	0dB	0.16	0.0625	700

¹ f_x/f_s

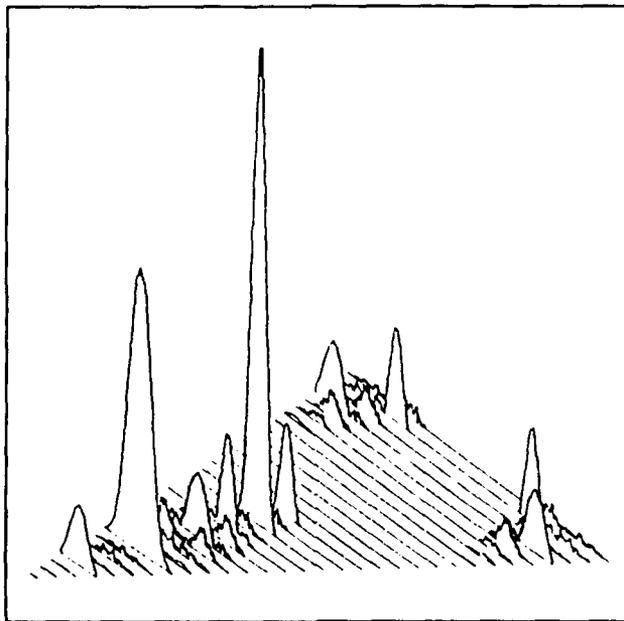


Figure 5-3 Bifrequency plot showing how spectral features of two similar signals are separated using spectral redundancy techniques.

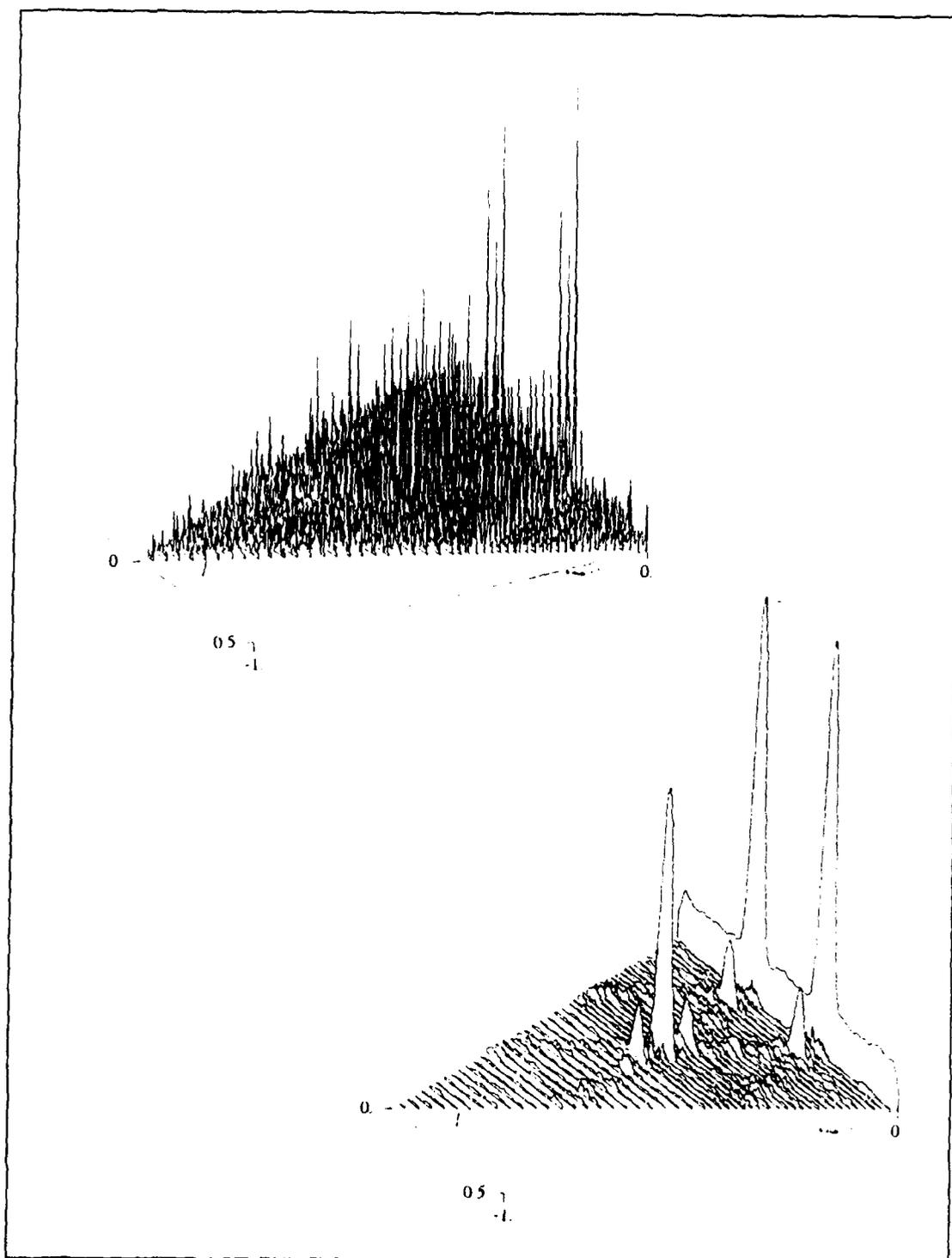


Figure 5-4 Effect of smoothing on bifrequency plane output and detection of exploitable cycle frequency features: a) .01% smoothing, b) 5% smoothing. 0dB SNR input, +12dB in-band SNR.

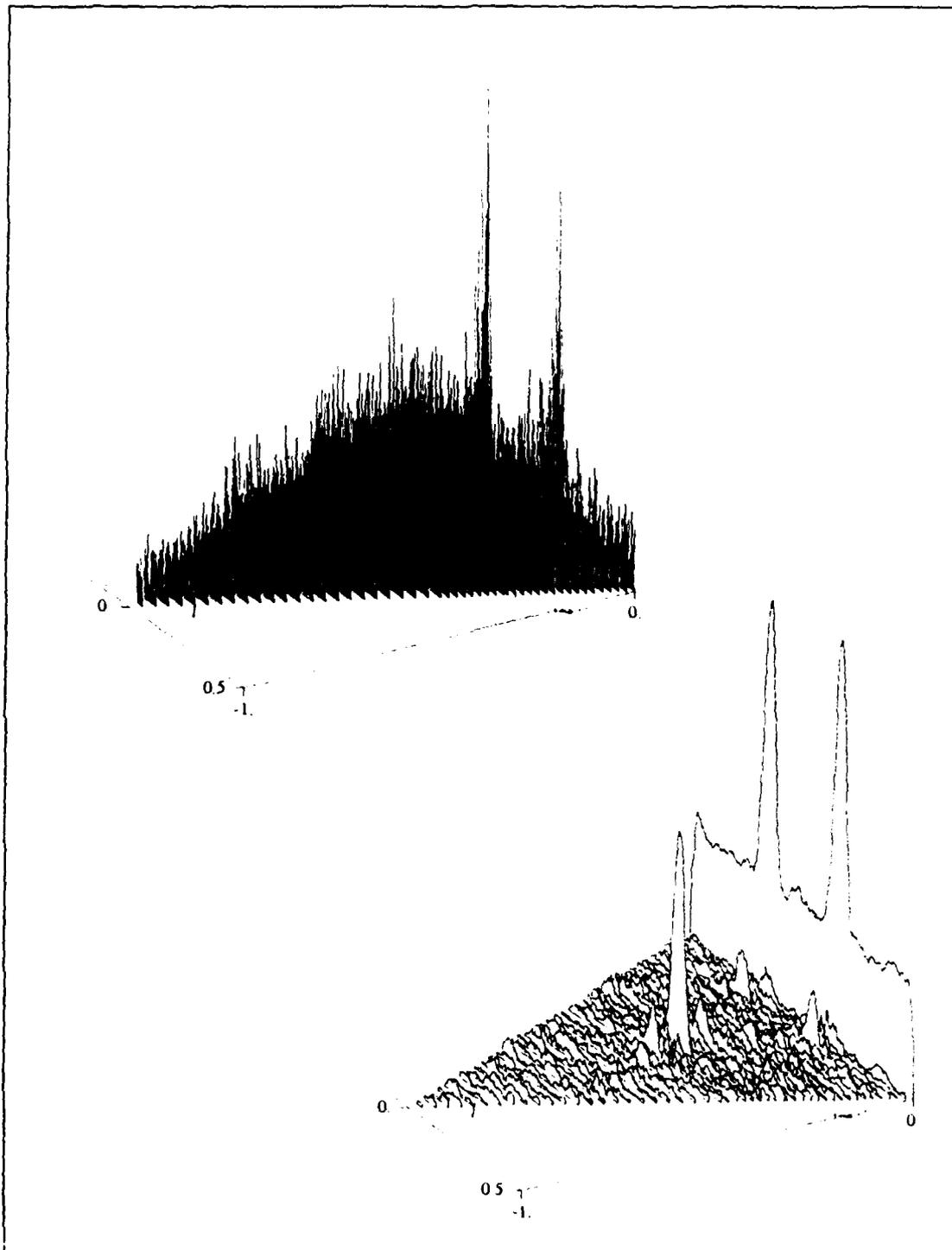


Figure 5-5 Effect of smoothing on bifrequency plane output and detection of exploitable cycle frequency features: a) .01% smoothing, b) 5% smoothing. -5dB SNR input, +7dB in-band SNR.

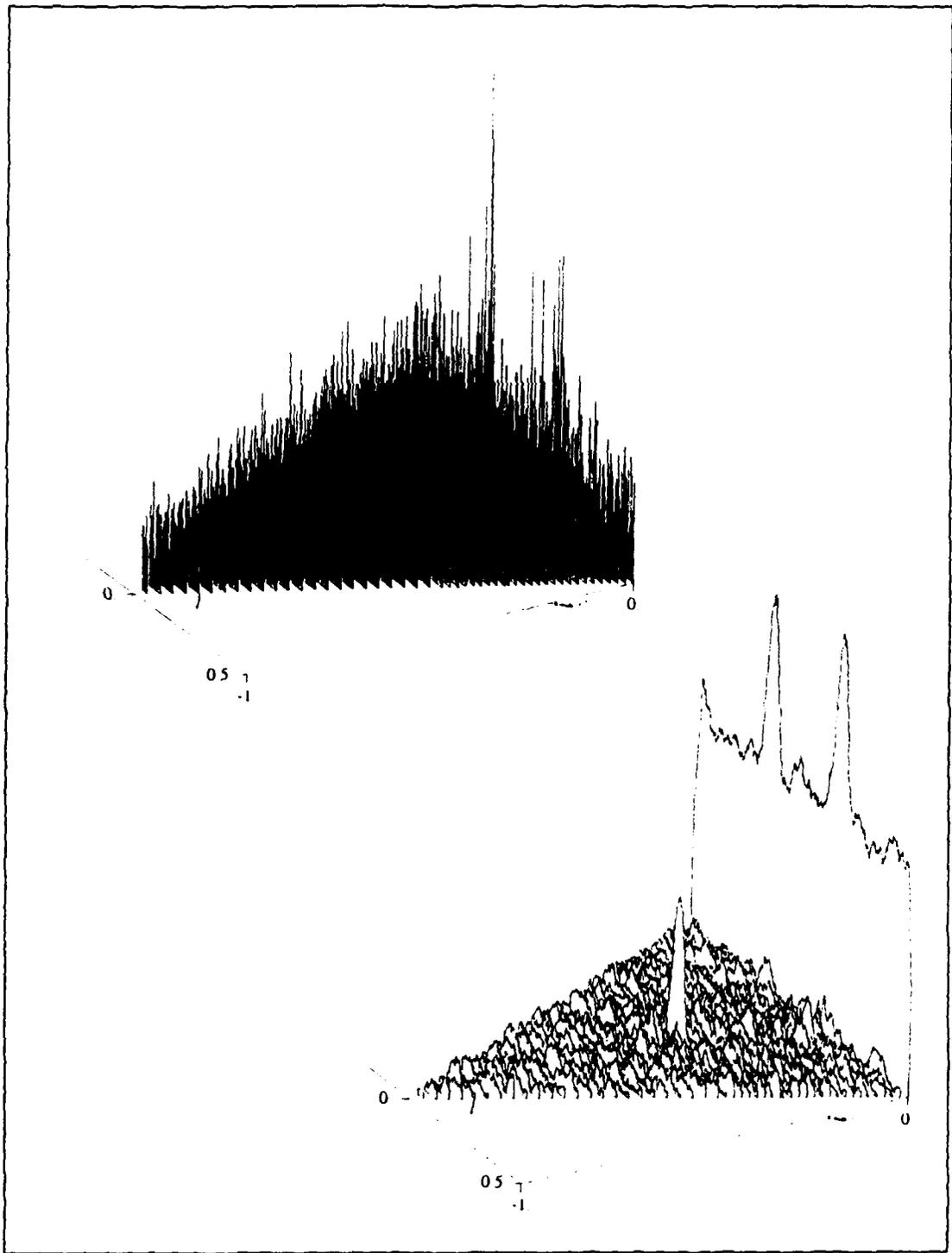


Figure 5-6 Effect of smoothing on bifrequency plane output and detection of exploitable cycle frequency features: a) .01 % smoothing, b) 5 % smoothing. -10dB SNR input, +2dB in-band SNR.

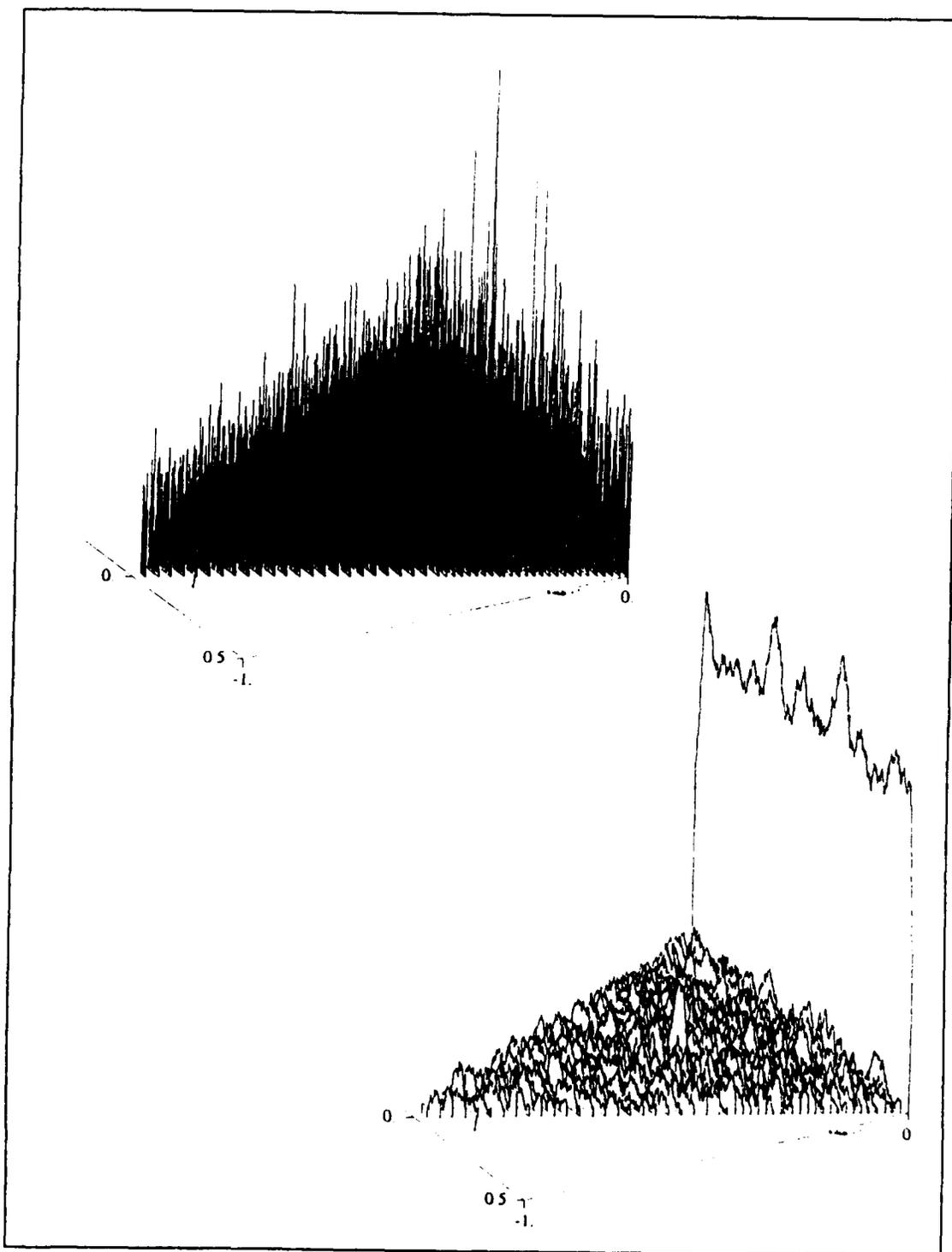


Figure 5-7 Effect of smoothing on bifrequency plane output and detection of exploitable cycle frequency features: a) .01% smoothing, b) 5% smoothing. -15dB SNR input, -3dB in-band SNR.

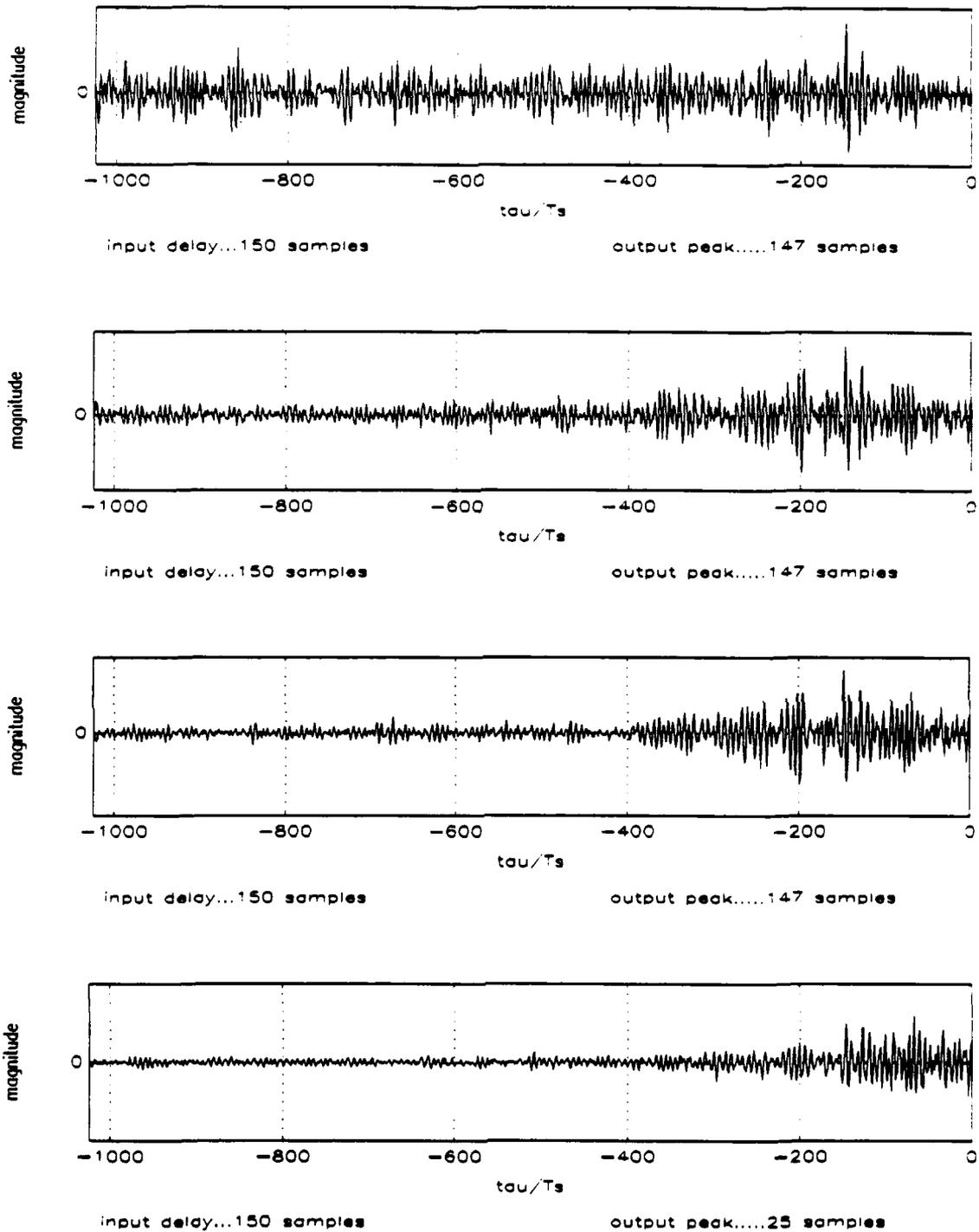


Figure 5-8 Effect of smoothing on SPECCOA output: a) 0.005%, b) 0.09%, c) 0.15%, d) 0.39% with resolution products of 1, 2, 4, and 8, respectively. Input SNR was -5dB with a chip rate of 0.0625 resulting in an in-band SNR of +12dB.

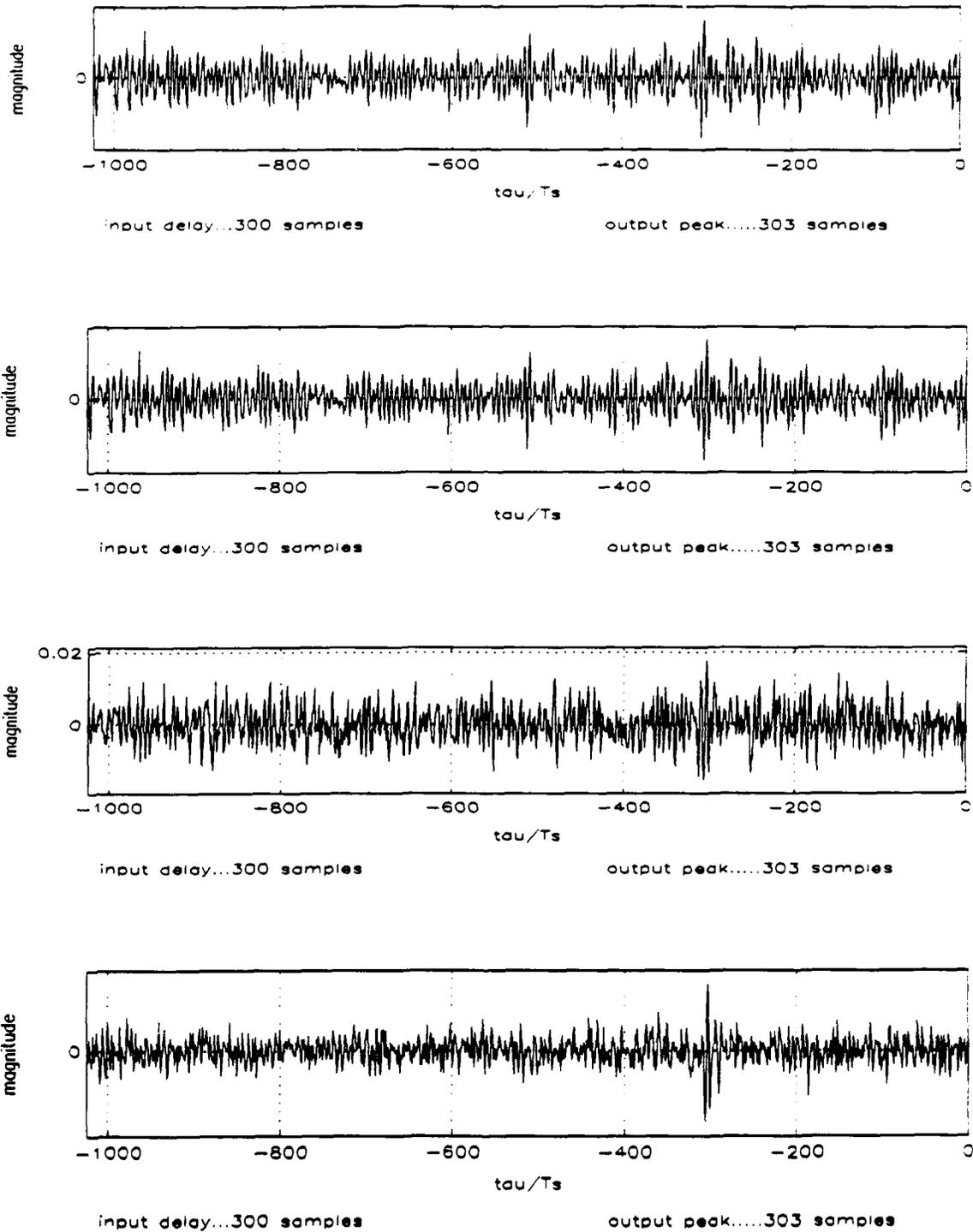
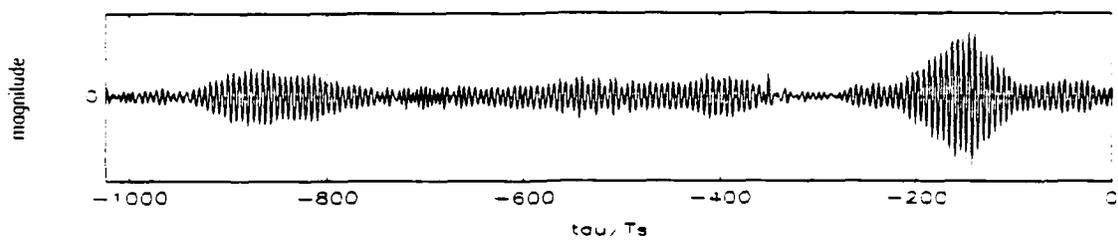
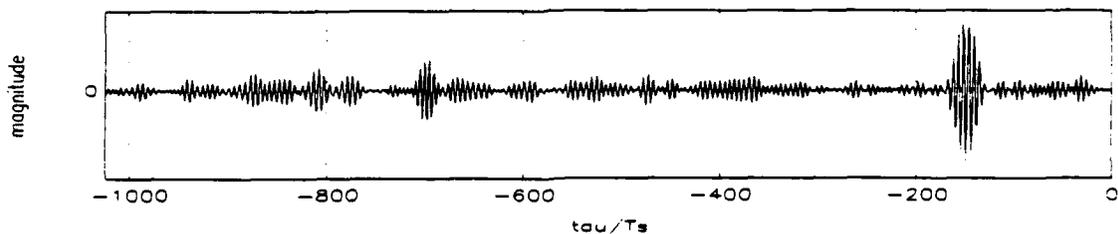


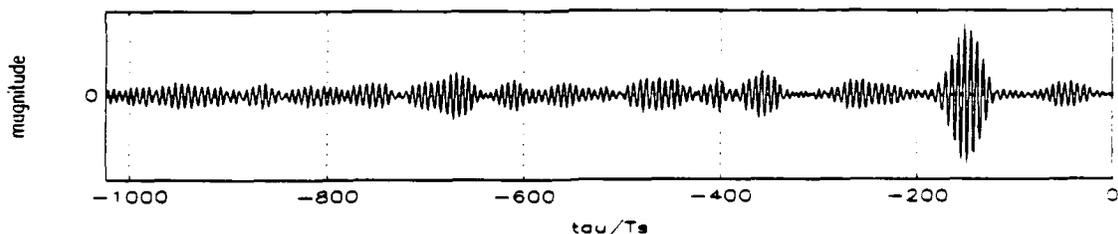
Figure 5-9 Effect of decreasing signal to noise ratio (SNR) on SPECCOA output:
a) 0dB, b) -5dB, c) -10dB, d) -12dB. Cycle frequency, 0.0625, carrier frequency, 0.16,
smoothing, 0.01%, sample lengths, 1920, 1920, 3840, and 15,360, respectively.



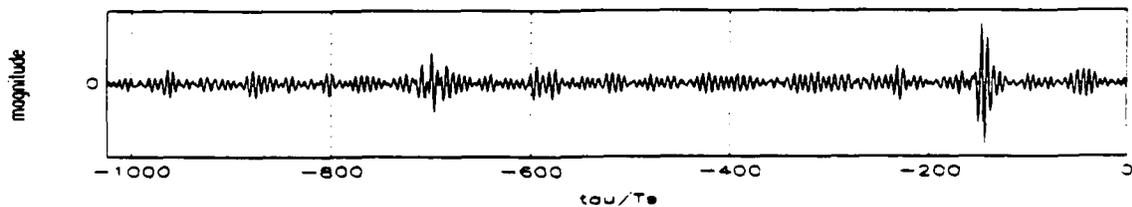
a) Multiple interfering signals.



b) Wide-band interferer.



c) Narrow-band interferer



d) Coband interferer.

Figure 5-10 SPECCOA output results with various interfering signal conditions. $\alpha_0 = 0.0625/T_s$.

VI. CONCLUSIONS

The purpose of this thesis was to implement the SPECCOA algorithm. That objective was met. Tests conducted showed that the algorithm was successful in producing a time-delay output in simulated environments of negative signal to ratio (SNR) and interfering signals. These tests were conducted with relatively short collect lengths of less than 16,384 samples. With larger collect lengths it is anticipated that significantly lower SNR's can be achieved. There is a limit, of course, as to how long a collect length can be gathered. The longer the collect length, the longer the processing time. For systems that operate in near real time, this could become a significant drawback. Thus, a trade-off must be made between the time necessary to process the data and the level of SNR that can be examined.

The most significant result of the testing was that a high degree of smoothing was not necessary in order to produce a time-delay output. In fact, a high degree of smoothing actually degraded the performance of the algorithm. Very small values of smoothing were required, on the order of 0.01% for collect lengths of less than 10,000. This produced a resolution product of near unity.¹ Significant smoothing was still required, however, for developing plots of the bifrequency plane in order to see what features, if any, were exploitable.

¹Recall the resolution product, $\Delta t \Delta f$, equals $(\% \text{ smoothing}/100)(\text{sample length}) \div (1-\alpha_c)$.

Further tests are needed in a real-world environment. Data exists from a special experiment that was conducted in a real-world environment and is available for processing and insertion into the SPECCOA algorithm.

Finally, other methods exist to calculate the spectral correlation densities (SCDs) that are used by the SPECCOA algorithm. Examples are the FFT Accumulation Method (FAM) and the Strip Spectral Correlation Algorithm (SSCA) [Ref 11]. Testing the SPECCOA algorithm using other methods is necessary to determine if any modifications to the algorithm are required. This would produce a general purpose SPECCOA algorithm useful with a variety of SCD calculation methods.

APPENDIX A. HYPERBOLA DEVELOPMENT

Following [Refs. 2 and 9], a proof is given that the geolocation scenario described in Chapter III, part B does indeed produce a hyperbola. Figure A-1 depicts the basic scenario. Receiver platforms one and two are initially located along the x-axis at coordinates $(-c,0)$ and $(c,0)$. The emitter, located at coordinates (x,y) , transmits a spherical wave that travels over distances d_1 and d_2 to arrive at times t_1 and t_2 at receivers one and two, respectively. The

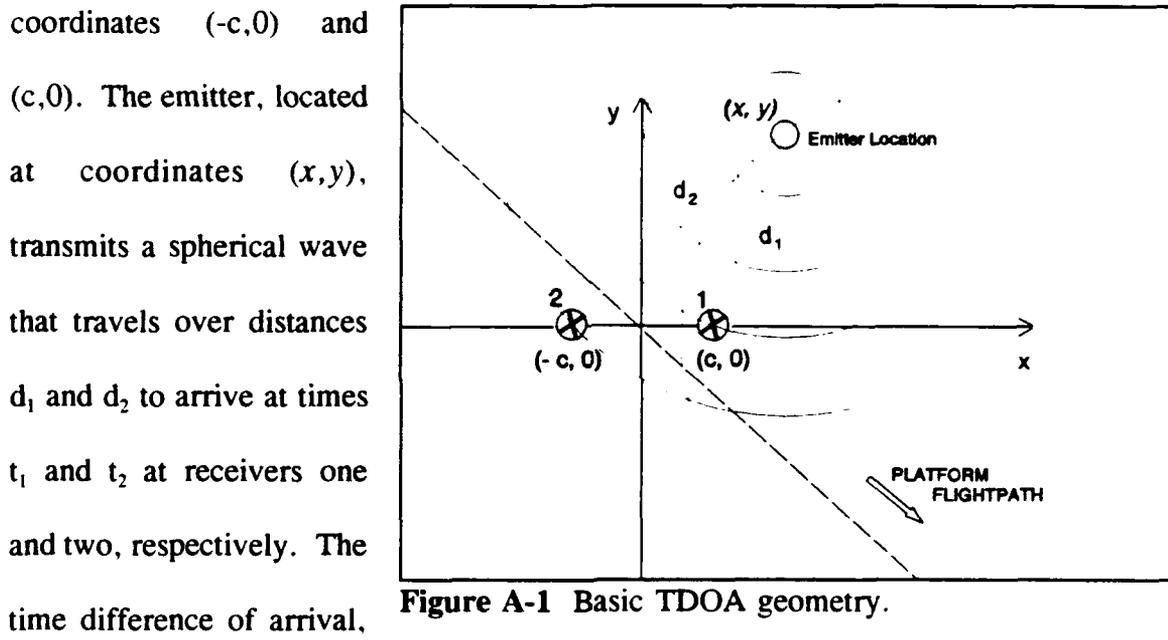


Figure A-1 Basic TDOA geometry.

time difference of arrival, or TDOA, is then

$$TDOA = \Delta t = t_2 - t_1 = \frac{d_2}{v} - \frac{d_1}{v} \tag{A.1a}$$

$$= \frac{1}{v} \sqrt{(x+c)^2 + y^2} - \frac{1}{v} \sqrt{(x-c)^2 + y^2}, \tag{A.1b}$$

where v is the propagation speed of the electromagnetic wave. Multiplying both sides by v to get an equation in the form of a difference of *distances* gives

$$v \Delta t = D = \left| \sqrt{(x+c)^2 + y^2} - \sqrt{(x-c)^2 + y^2} \right|. \quad (\text{A.2})$$

Since a hyperbola is defined as the set of all points whose difference in distance from two *fixed* points is constant, manipulating (A-2) will yield the standard form of a hyperbola. First, moving the second radical to the left side of the equation, squaring both sides, and rearranging gives

$$D^2 + (x-c)^2 + y^2 + 2D\sqrt{(x-c)^2 + y^2} = (x+c)^2 + y^2, \quad (\text{A.3})$$

expanding and simplifying leads to

$$4xc - D^2 = 2D\sqrt{(x-c)^2 + y^2}. \quad (\text{A.4})$$

Again, squaring both sides and rearranging yields

$$4D^2c^2 - D^4 = 16x^2c^2 - 4x^2D^2 - 4D^2y^2. \quad (\text{A.5})$$

One further simplification gives the final result of

$$\frac{x^2}{D^2/4} - \frac{y^2}{(4c^2 - D^2)/4} = 1, \quad (\text{A.6})$$

which is the standard form of the hyperbola:

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1 \quad (\text{A.7})$$

with x-intercept $\pm a$ or $\pm D/2$, and asymptotic to the lines $y = \pm \left(\frac{b}{a}\right)x$ or

$$y = \pm \left[\frac{\sqrt{4c^2 - D^2}}{D} \right] x.$$

APPENDIX B. TEST CASE GRAPHS: α_0 EQUALS CHIP FEATURE

TABLE B1 PARAMETERS FOR GRAPHS OF FIGURE B-1: VARYING SNR

	input delay	% smoothing	resolution product ¹	sample length	α_0 , chip feature ²	carrier frequency ²	SNR dB
a)	300	0.01	1	1920	0.0625	0.16	-0
b)	300	0.01	1	1920	0.0625	0.16	-5
c)	300	0.01	1	3840	0.0625	0.16	-10
d)	300	0.001	1	15360	0.0625	0.16	-12

$${}^1\Delta t\Delta f = (\% \text{ smooth}/100)(\text{sample length})/(1 - \alpha_0)^2 f_i/f_c$$

TABLE B2 PARAMETERS FOR GRAPHS OF FIGURE B-2: VARYING DELAY

	input delay	% smoothing	resolution product ¹	sample length	α_0 , chip feature ²	carrier frequency ²	SNR dB
a)	150	0.01	1	1920	0.0625	0.16	-5
b)	300	0.01	1	1920	0.0625	0.16	-5
c)	700	0.01	1	1920	0.0625	0.16	-5
d)	950	0.01	1	1920	0.0625	0.16	-5

$${}^1\Delta t\Delta f = (\% \text{ smooth}/100)(\text{sample length})/(1 - \alpha_0)^2 f_i/f_c$$

**TABLE B3 PARAMETERS FOR GRAPHS OF FIGURE B-3:
VARYING SMOOTHING**

	input delay	% smoothing	resolution product ¹	sample length	α_0 , chip feature ²	carrier frequency ²	SNR dB
a)	150	0.005	1	1920	0.0625	0.16	-5
b)	150	0.09	2	1920	0.0625	0.16	-5
c)	150	0.15	4	1920	0.0625	0.16	-5
d)	150	0.39	8	1920	0.0625	0.16	-5

$${}^1\Delta t\Delta f = (\% \text{ smooth}/100)(\text{sample length})/(1 - \alpha_0)^2 f_i/f_c$$

TABLE B4 PARAMETERS FOR GRAPHS OF FIGURE B-4: VARYING f_c

	input delay	% smoothing	resolution product ¹	sample length	α_o , chip feature ²	carrier frequency ²	SNR dB
a)	150	0.01	1	1920	0.0625	0.16	-5
b)	150	0.01	1	1945	0.05	0.16	-5
c)	150	0.01	1	1962	0.04167	0.16	-5
d)	150	0.01	1	1979	0.03333	0.16	-5

$${}^1\Delta t\Delta f = (\% \text{ smooth}/100)(\text{sample length})/(1 - \alpha_o)^2 f_c/f_s$$

TABLE B5 PARAMETERS FOR GRAPHS OF FIGURE B-5: VARYING f_c

	input delay	% smoothing	resolution product ¹	sample length	α_o , chip feature ²	carrier frequency ²	SNR dB
a)	150	0.01	1	1920	0.0625	0.10	-5
b)	150	0.01	1	1945	0.0625	0.16	-5
c)	150	0.01	1	1962	0.0625	0.30	-5
d)	150	0.01	1	1979	0.0625	0.42	-5

$${}^1\Delta t\Delta f = (\% \text{ smooth}/100)(\text{sample length})/(1 - \alpha_o)^2 f_c/f_s$$

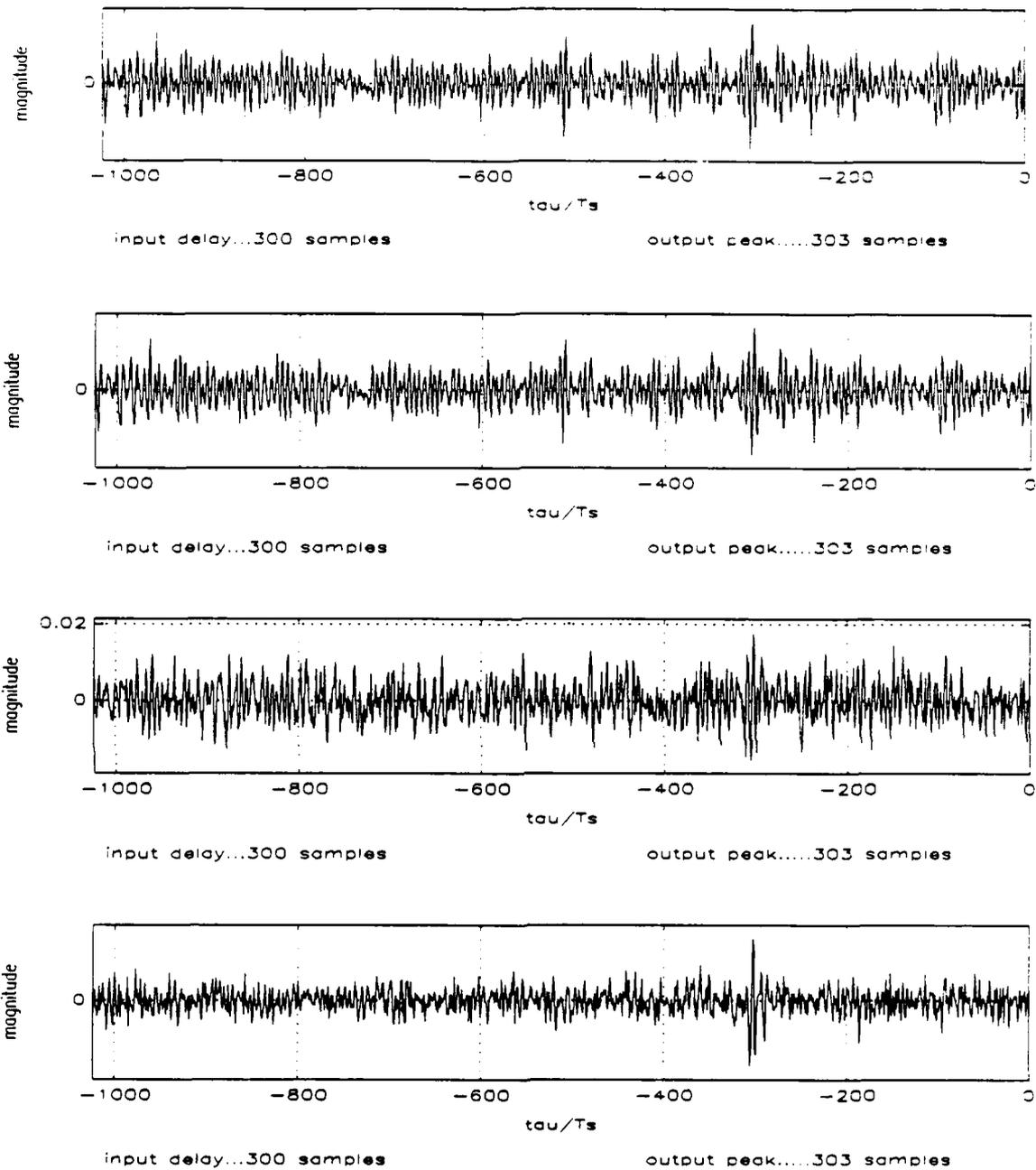


Figure B-1 Effect of decreasing signal to noise ratio (SNR) on SPECCOA output: a) 0dB, b) -5dB, c) -10dB, d) -12dB. Cycle frequency, $.0625/T_s$, carrier frequency, $.16/T_s$, smoothing, 0.01%.

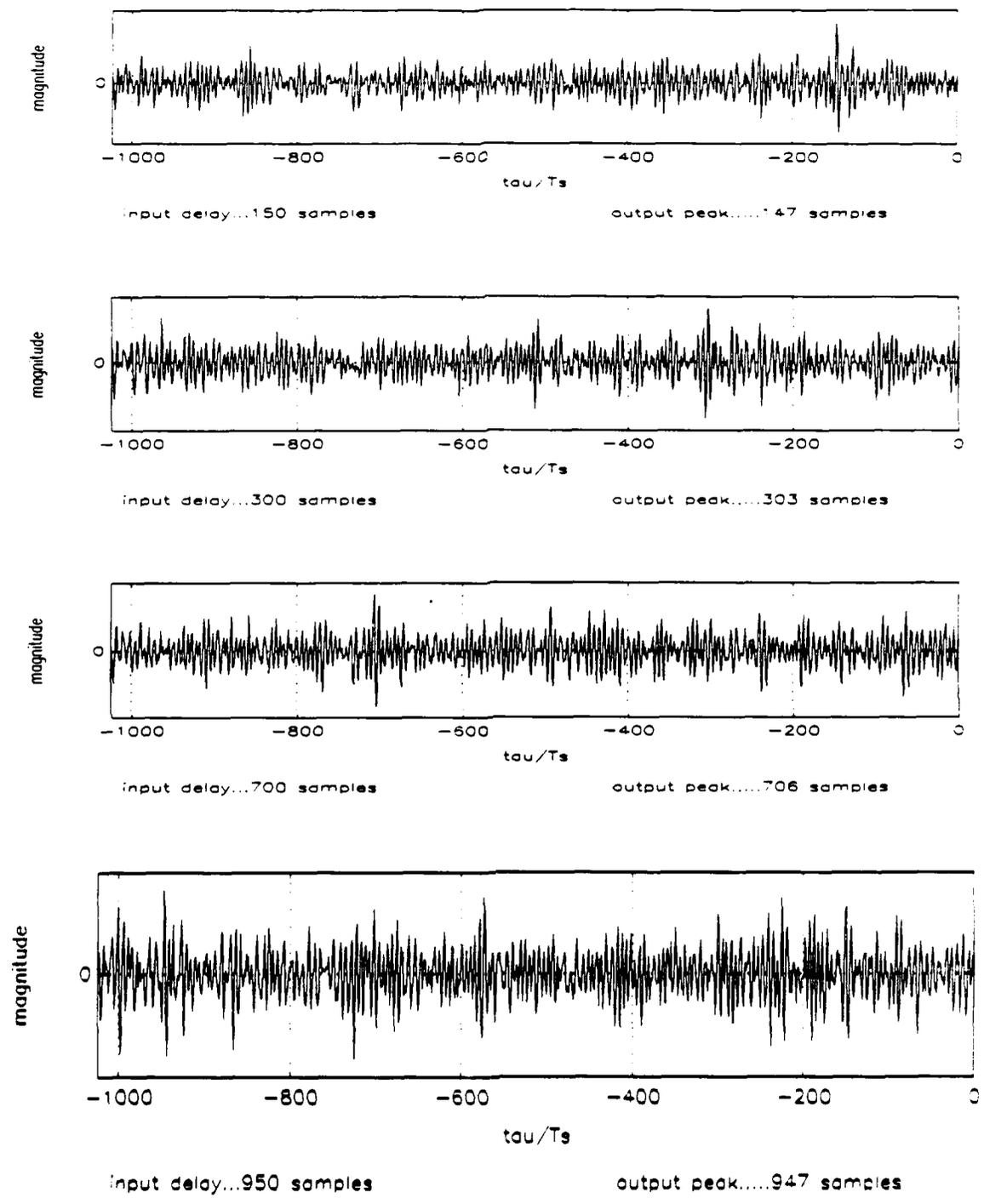


Figure B-2 Output of SPECCOA algorithm for increasing delays.

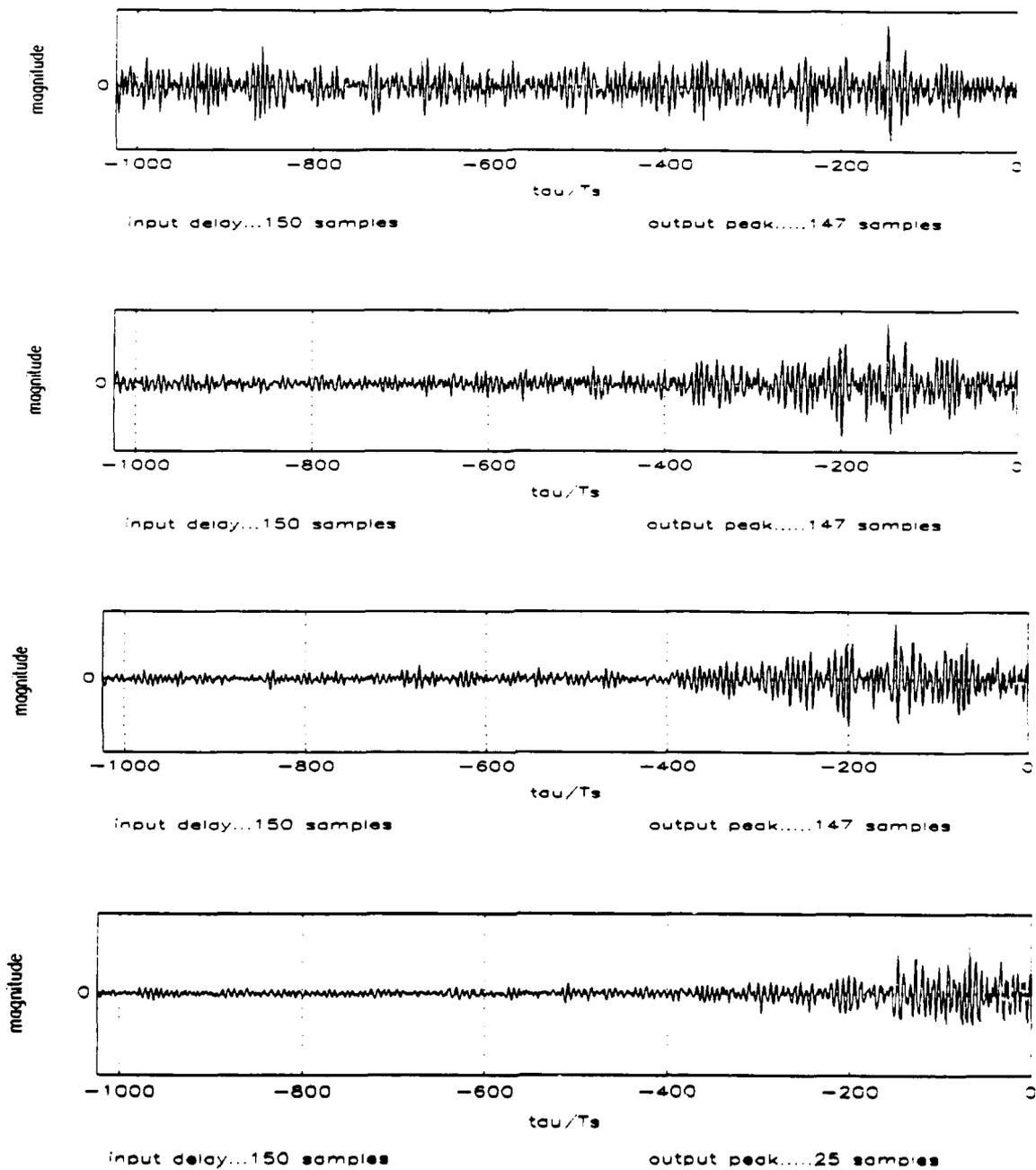
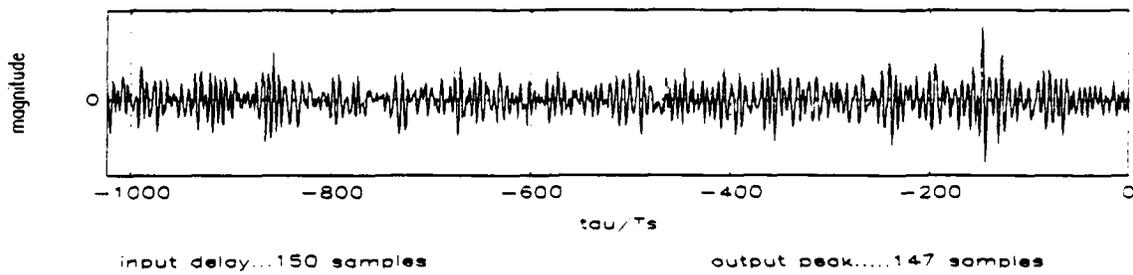
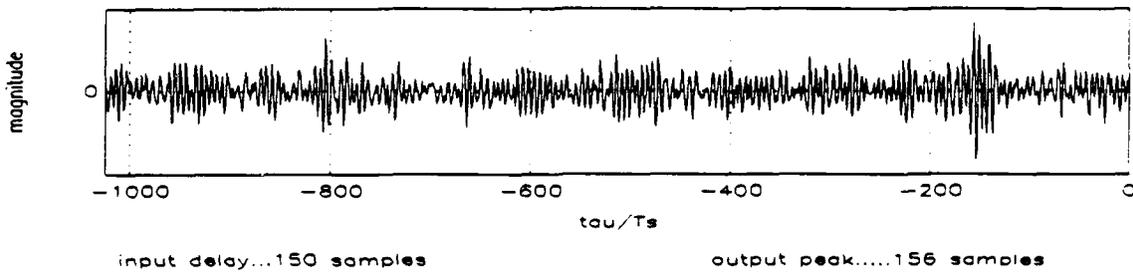


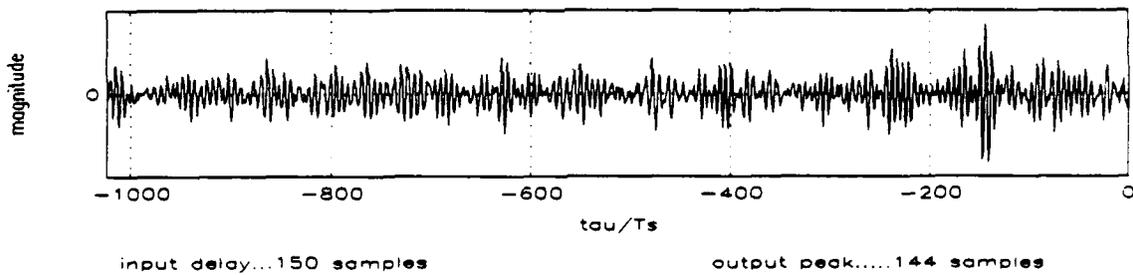
Figure B-3 Effect of smoothing on SPECCOA output: a) 0.005%, b) 0.09%, c) 0.15%, d) 0.39% with resolution products of 1, 2, 4, and 8, respectively. Carrier frequency was $.16/T_s$.



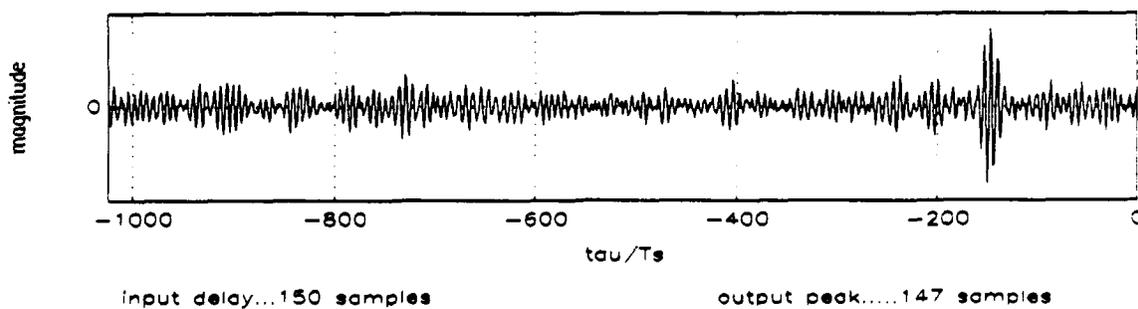
(a) $\alpha = 0.0625/T_s$.



(b) $\alpha = 0.50/T_s$.

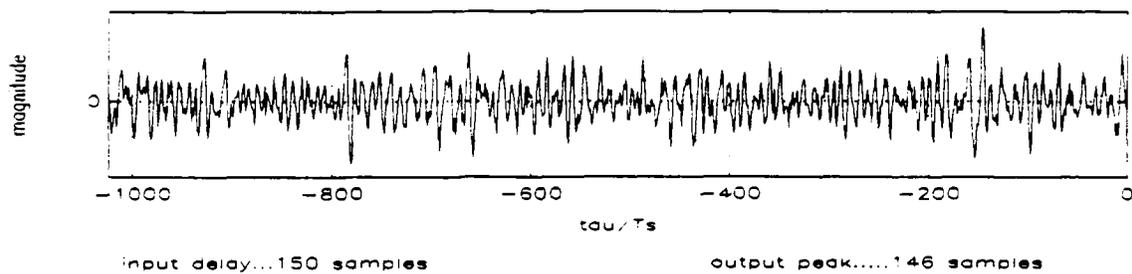


(c) $\alpha = 0.041667/T_s$.

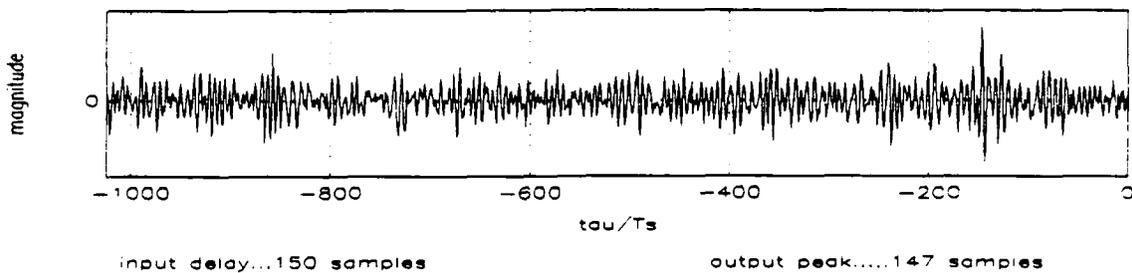


(d) $\alpha = 0.03333/T_s$.

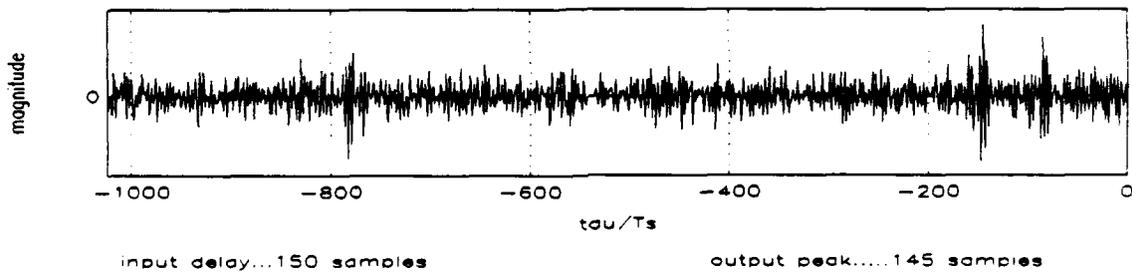
Figure B-4 Effect on SPECCOA output of varying the chip frequency, f_c . Carrier frequency, $0.16/T_s$.



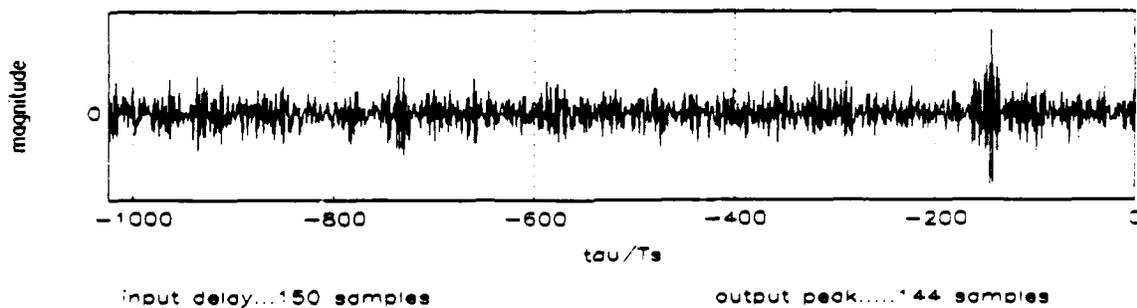
(a) $f_0 = 0.1/T_s$.



(b) $f_0 = 0.16/T_s$.



(c) $f_0 = 0.30/T_s$.



(d) $f_0 = 0.42/T_s$.

Figure B-5 Effect of varying the carrier frequency, f_0/T_s , on SPECCOA output.

APPENDIX C. TEST CASE GRAPHS: α_0 EQUALS TWICE CARRIER FREQ.

TABLE C1 PARAMETERS FOR GRAPHS OF FIGURE C-1: VARYING SNR

	input delay	% smoothing	resolution product ¹	sample length	α_0^2	carrier frequency ²	SNR dB
a)	150	0.20	5	1392	0.32	0.16	0
b)	150	0.20	5	1392	0.32	0.16	-5
c)	150	0.20	5	2392	0.32	0.16	-10
d)	150	0.0305	5	11,141	0.32	0.16	-15

$${}^1\Delta t\Delta f = (\% \text{ smooth}/100)(\text{sample length})/(1 - \alpha_0) \quad {}^2 f_r/f_s, f_c = 0.0625/T_s.$$

TABLE C2 PARAMETERS FOR GRAPHS OF FIGURE C-2: VARYING DELAY

	input delay	% smoothing	resolution product ¹	sample length	α_0^2	carrier frequency ²	SNR dB
a)	150	0.20	5	1392	0.32	0.16	-5
b)	300	0.20	5	2785	0.32	0.16	-5
c)	700	0.05	5	5570	0.32	0.16	-5
d)	900	0.05	5	5570	0.32	0.16	-5

$${}^1\Delta t\Delta f = (\% \text{ smooth}/100)(\text{sample length})/(1 - \alpha_0) \quad {}^2 f_r/f_s, f_c = 0.0625/T_s.$$

**TABLE C3 PARAMETERS FOR GRAPHS OF FIGURE C-3:
VARYING SMOOTHING**

	input delay	% smoothing	resolution product ¹	sample length	α_o^2	carrier frequency ²	SNR dB
a)	150	0.01	1	1392	0.32	0.16	-5
b)	150	0.195	4	1392	0.32	0.16	-5
c)	150	0.35	8	1392	0.32	0.16	-5
d)	150	0.50	11	1392	0.32	0.16	-5

$${}^1\Delta t\Delta f = (\% \text{ smooth}/100)(\text{sample length})/(1 - \alpha_o) \quad {}^2 f_r/f_s, f_c = 0.0625/T_s.$$

TABLE C4 PARAMETERS FOR GRAPHS OF FIGURE C-4: VARYING f_o

	input delay	% smoothing	resolution product ¹	sample length	α_o^2	carrier frequency ²	SNR dB
a)	150	0.20	5	1638	0.20	0.10	-5
b)	150	0.20	5	1515	0.21	0.13	-5
c)	150	0.20	5	1392	0.32	0.16	-5
d)	150	0.20	5	1269	0.38	0.19	-5

$${}^1\Delta t\Delta f = (\% \text{ smooth}/100)(\text{sample length})/(1 - \alpha_o) \quad {}^2 f_r/f_s, f_c = 0.0625/T_s.$$

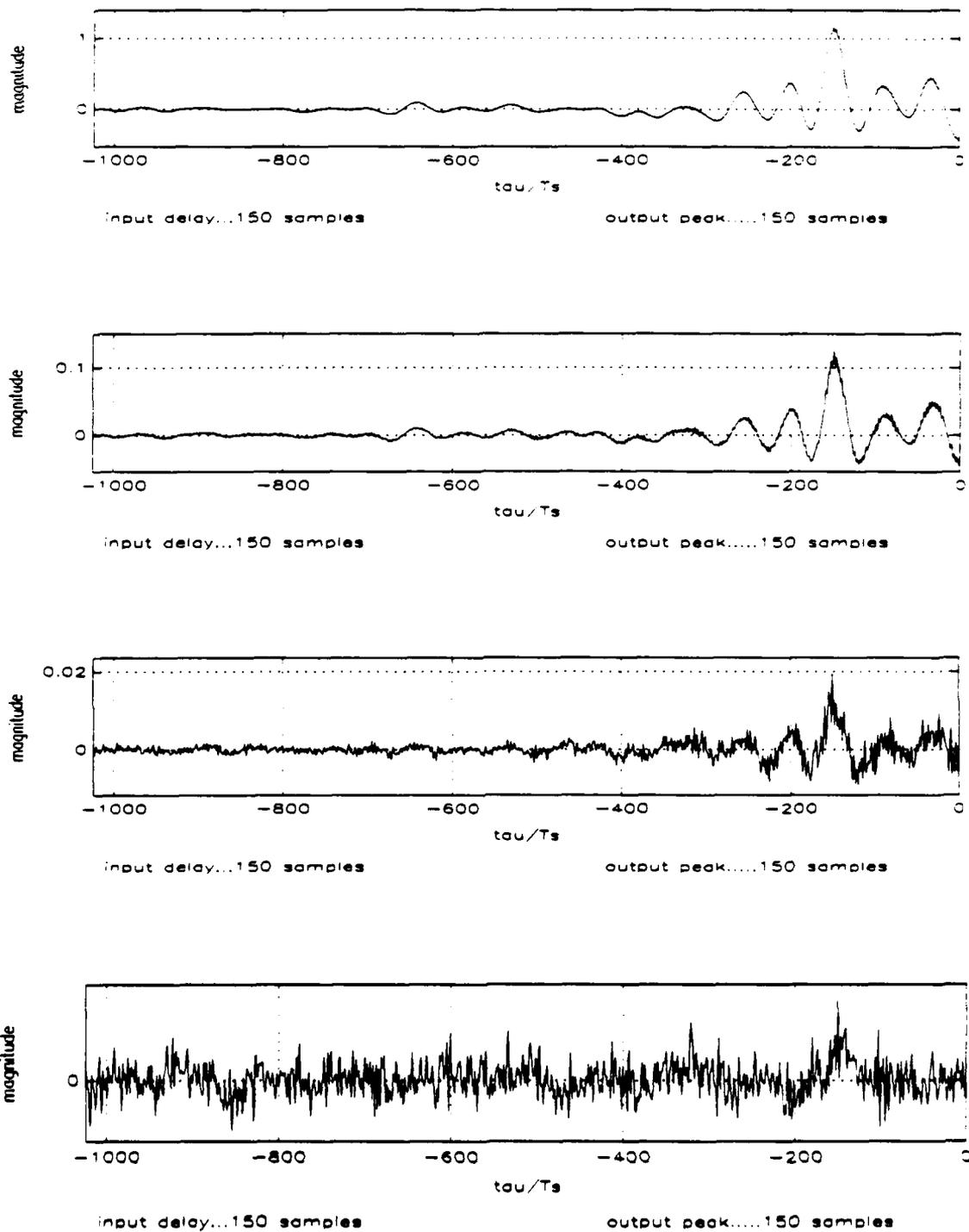


Figure C-1 Effect of decreasing signal to noise ratio (SNR) on SPECCOA output: a) 0dB, b) -5dB, c) -10dB, d) -15dB. Cycle frequency, $\alpha = 2f_0$, carrier frequency, $.16/T_s$, $\Delta t \Delta f = 5$.

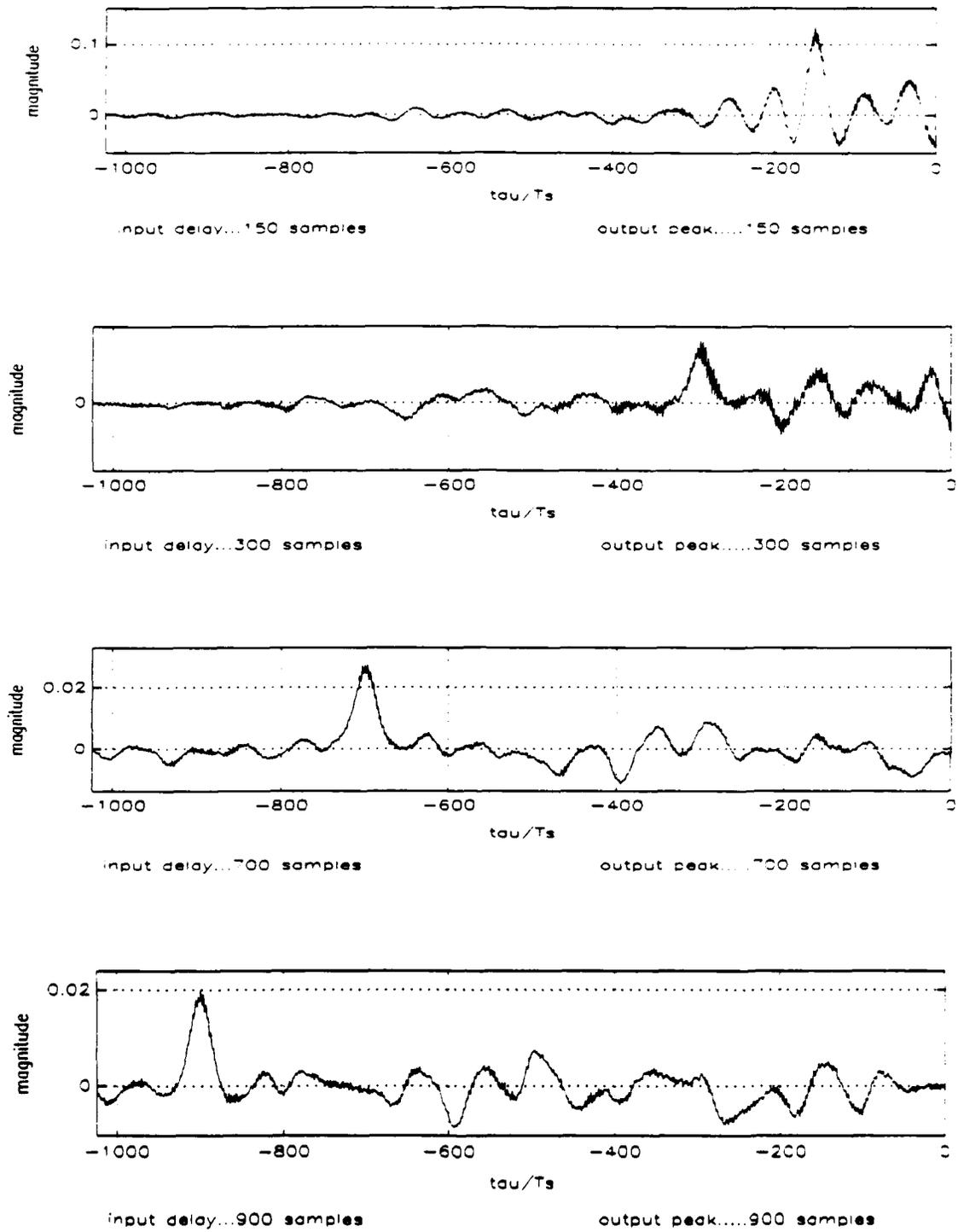


Figure C-2 Output of SPECCOA algorithm for increasing delays.

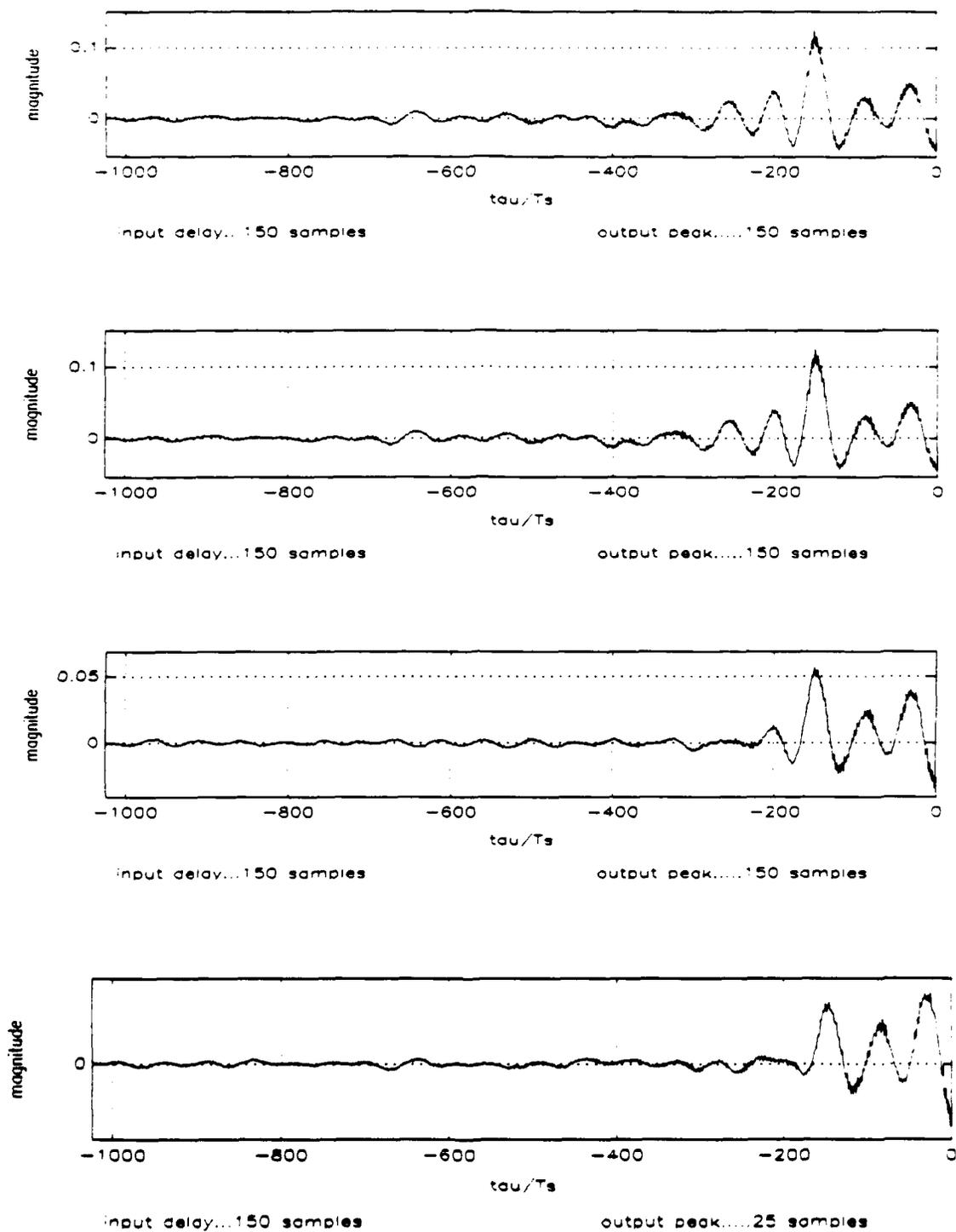


Figure C-3 Effect of smoothing on SPECCOA output: a) 0.01%, b) 0.195%, c) 0.35%, d) 0.5%, with resolution products of 1, 4, 8, 11, respectively. Carrier freq., $0.16/T_s$.

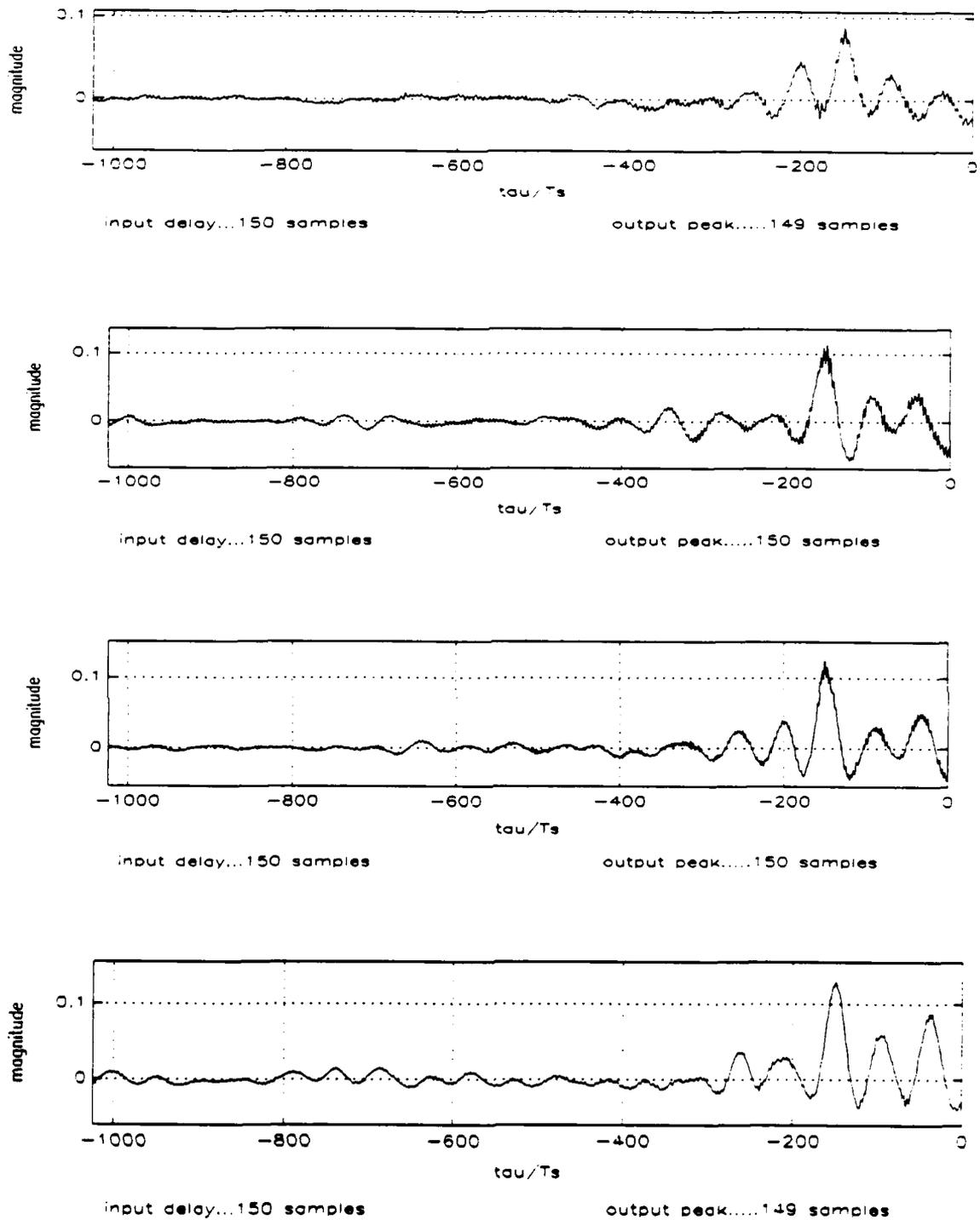


Figure C-4 Effect of varying the carrier frequency, f_c/T_s , on SPECCOA output:
a) $0.1/T_s$, b) $0.13/T_s$, c) $0.16/T_s$, d) $0.19/T_s$. $\alpha_0 = 2f_0$ for all cases, $f_c = 0.0625/T_s$.

**APPENDIX D. TEST CASE GRAPHS: α_0 EQUALS TWICE CARRIER FREQ.
PLUS CHIP FEATURE**

TABLE D1 PARAMETERS FOR GRAPHS OF FIGURE D-1: VARYING SNR

	input delay	% smoothing	resolution product ¹	sample length	α_0^2	carrier frequency ²	SNR dB
a)	150	0.02	1	1264	0.3825	0.16	0
b)	150	0.02	1	1264	0.3825	0.16	-5
c)	150	0.02	1	2529	0.3825	0.16	-10
d)	150	0.005	1	10,117	0.3825	0.16	-15

$${}^1\Delta t\Delta f = (\% \text{ smooth}/100)(\text{sample length})/(1 - \alpha_0)^2 f_c/T_s, f_c = 0.0625/T_s,$$

TABLE D2 PARAMETERS FOR GRAPHS OF FIGURE D-2: VARYING DELAY

	input delay	% smoothing	resolution product ¹	sample length	α_0^2	carrier frequency ²	SNR dB
a)	150	0.02	1	2529	0.3825	0.16	-5
b)	300	0.02	1	2529	0.3825	0.16	-5
c)	700	0.02	1	2529	0.3825	0.16	-5
d)	900	0.02	1	2529	0.3825	0.16	-5

$${}^1\Delta t\Delta f = (\% \text{ smooth}/100)(\text{sample length})/(1 - \alpha_0)^2 f_c/T_s, f_c = 0.0625/T_s,$$

**TABLE D3 PARAMETERS FOR GRAPHS OF FIGURE D-3:
VARYING SMOOTHING**

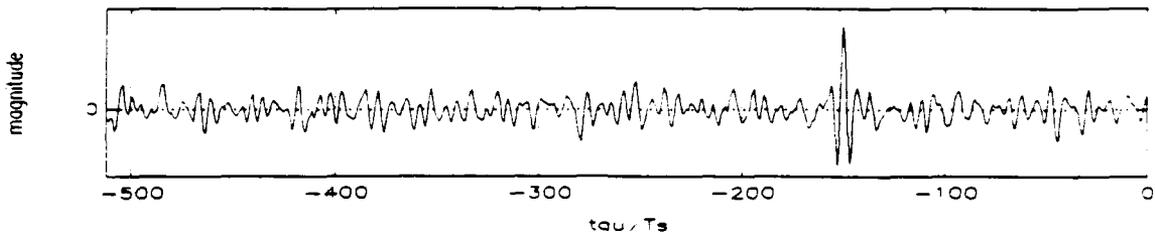
	input delay	% smoothing	resolution product ¹	sample length	α_o^2	carrier frequency ²	SNR dB
a)	150	0.05	2	1264	0.3825	0.16	-5
b)	150	0.10	3	1264	0.3825	0.16	-5
c)	150	0.30	7	1264	0.3825	0.16	-5
d)	150	0.50	11	1264	0.3825	0.16	-5

$${}^1\Delta t\Delta f = (\% \text{ smooth}/100)(\text{sample length})/(1 - \alpha_o)^2 f_r/T_s, f_c = 0.0625/T_s.$$

TABLE D4 PARAMETERS FOR GRAPHS OF FIGURE D-4: VARYING f_o

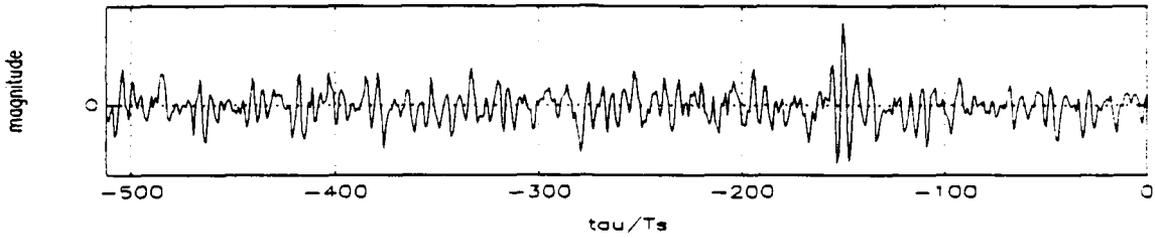
	input delay	% smoothing	resolution product ¹	sample length	α_o^2	carrier frequency ²	SNR dB
a)	150	0.02	1	1510	0.2625	0.10	-5
b)	150	0.02	1	1387	0.3225	0.13	-5
c)	150	0.02	1	1264	0.3825	0.16	-5
d)	150	0.02	1	1141	0.4425	0.19	-5

$${}^1\Delta t\Delta f = (\% \text{ smooth}/100)(\text{sample length})/(1 - \alpha_o)^2 f_r/T_s, f_c = 0.0625/T_s.$$



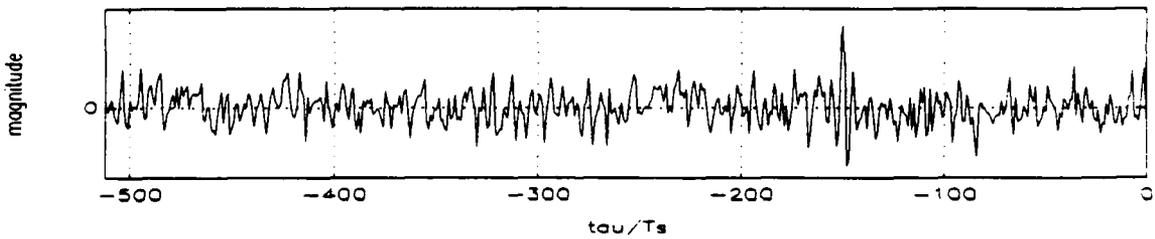
input delay...150 samples output peak.....150 samples

(a) 1264 samples



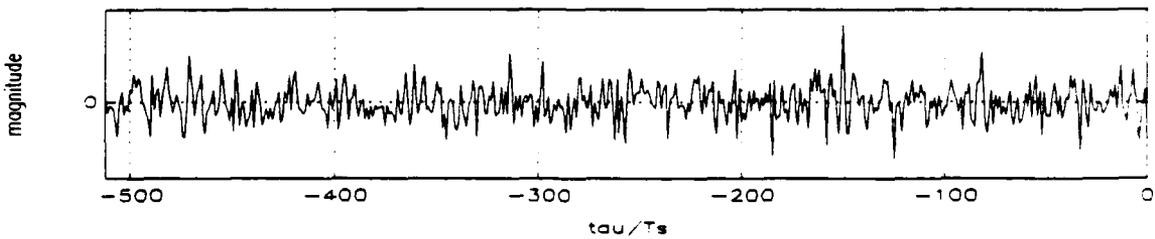
input delay...150 samples output peak.....150 samples

(b) 1264 samples



input delay...150 samples output peak.....150 samples

(c) 2529 samples



input delay...150 samples output peak.....150 samples

d) 10,117 samples

Figure D-1 Effect of decreasing SNR on SPECCOA output: a) 0dB, b) -5dB, c) -10dB, d) -15dB. Carrier frequency, $0.16/T_s$, f_c , $0.0625/T_s$, $\alpha_o = 2 f_o + f_c$.

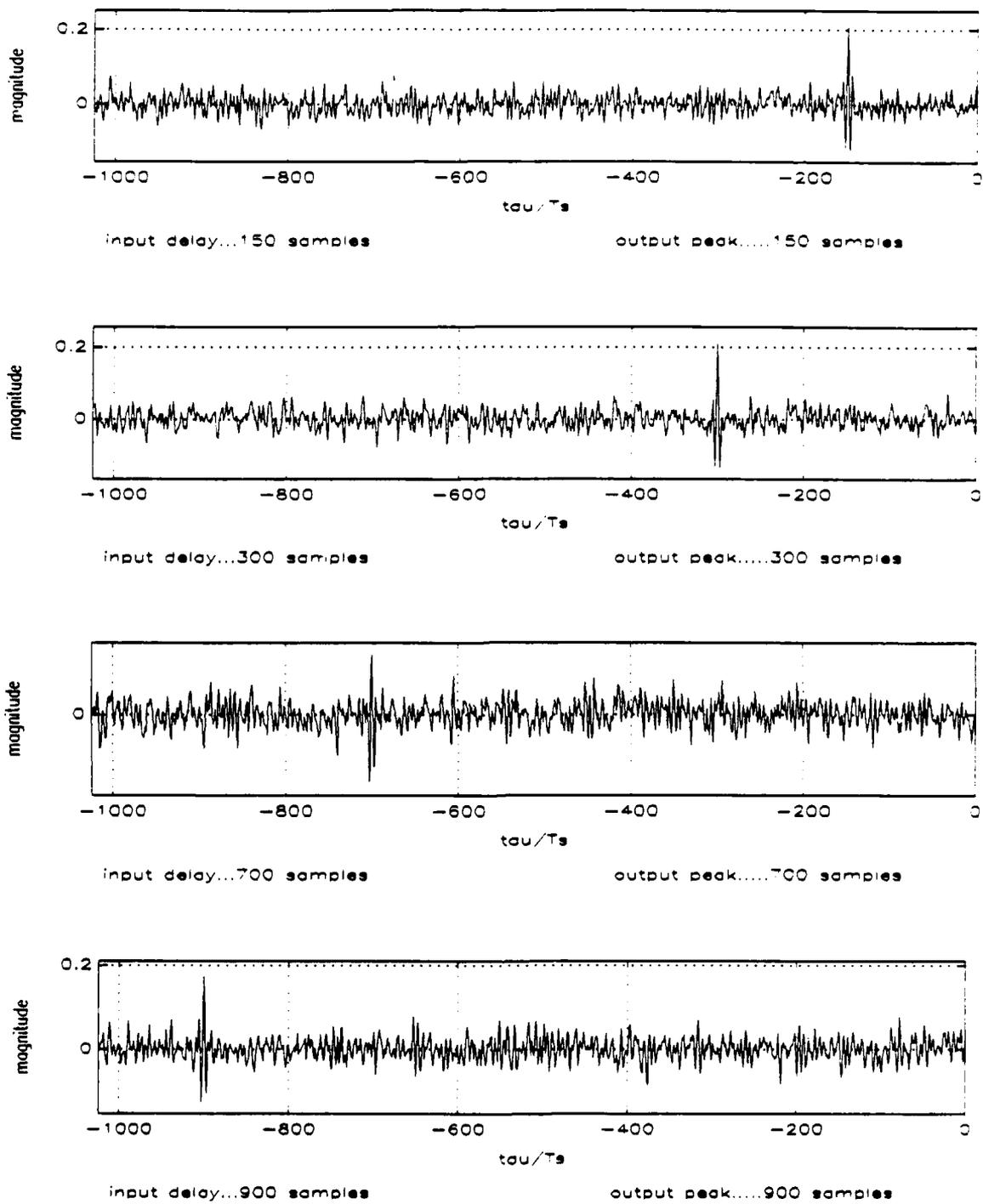


Figure D-2 Output of SPECCOA algorithm for increasing delays.

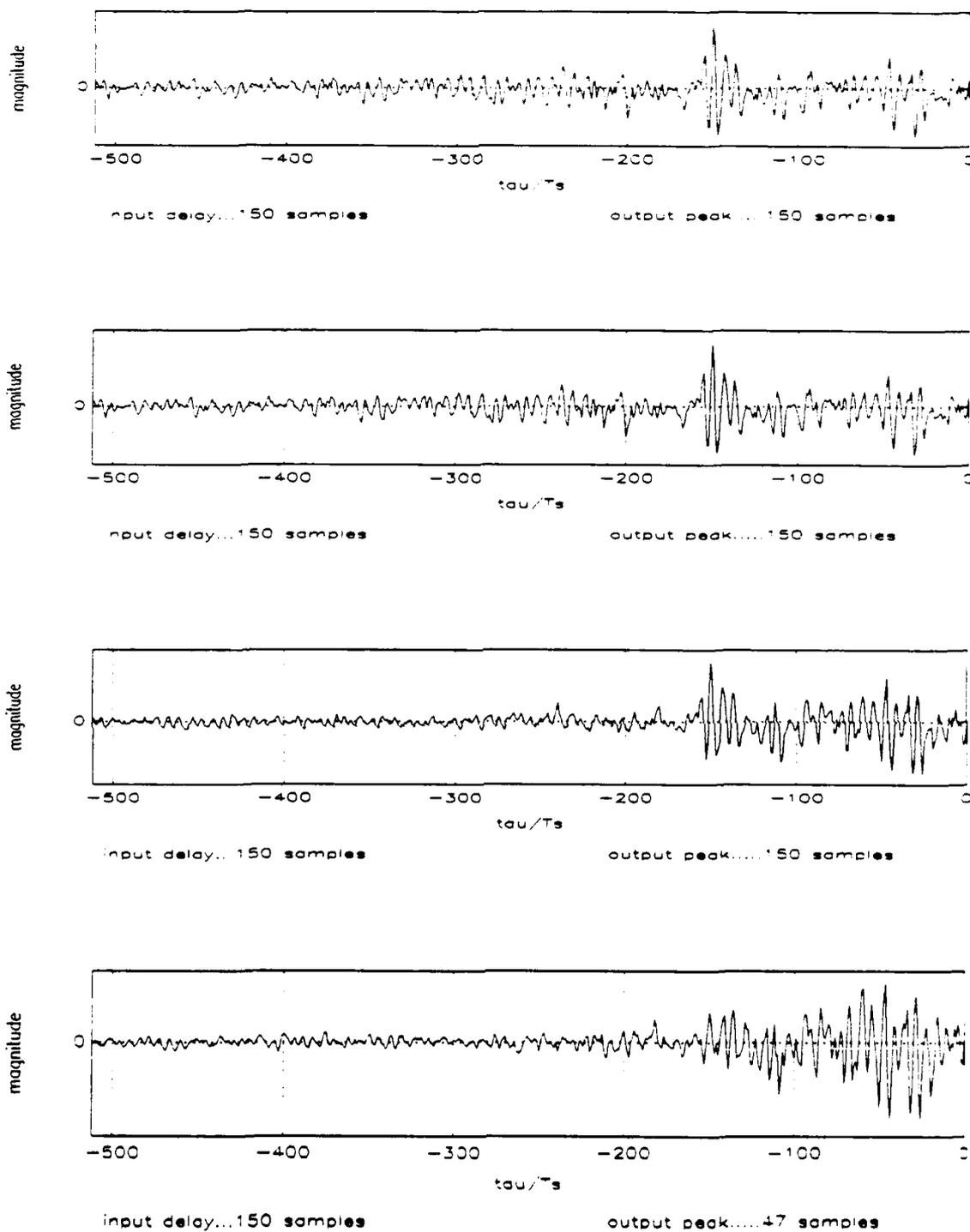
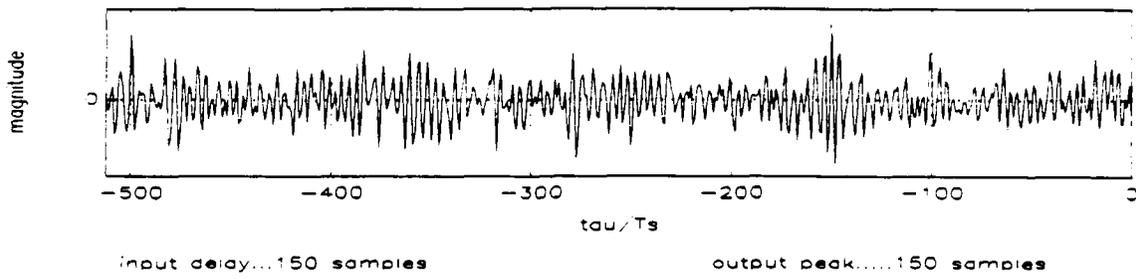
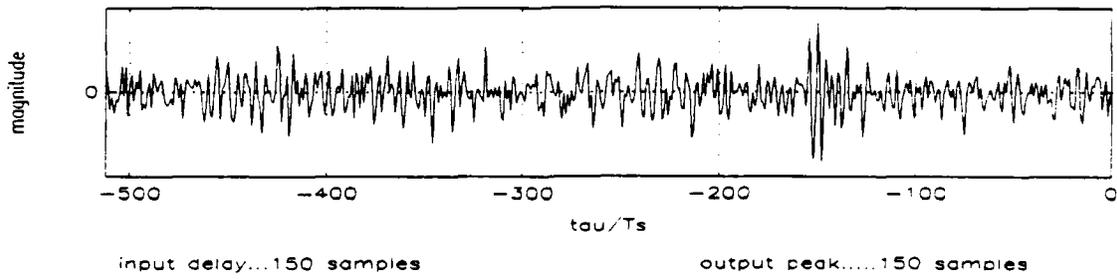


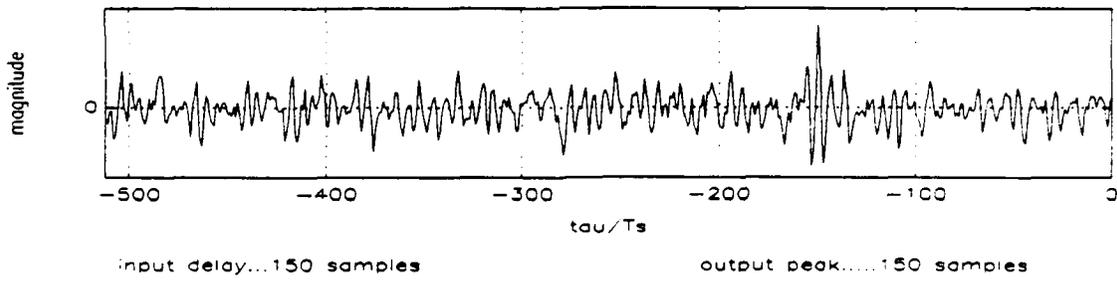
Figure D-3 Effect of smoothing on SPECCOA output: a) 0.05%, b) 0.10%, c) 0.30%, d) 0.50%, with resolution products of 2, 3, 7, 11, respectively. Carrier frequency, $0.16/T_s$.



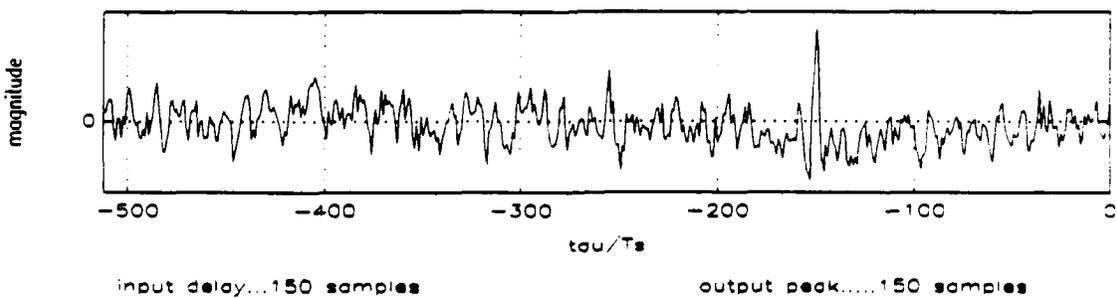
(a) $f_0, 0.1/T_s$. $\alpha_0, 0.2625/T_s$.



(b) $f_0, 0.13/T_s$. $\alpha_0, 0.3225/T_s$.



(c) $f_0, 0.16/T_s$. $\alpha_0, 0.3825/T_s$.



(d) $f_0, 0.1/T_s$. $\alpha_0, 0.4425/T_s$.

Figure D-4 Effect of varying the carrier frequency, f_0 , on SPECCOA output.

APPENDIX E. COMPUTER PROGRAMS

This appendix contains four computer programs and routines that were written during the course of working on this paper: *tdoasmth.bat*, *speca.c*, *fcnvrt.c*, and *plotdata.m*. The main algorithm written is called *speca* and has a companion program, *fcnvrt*, that is used to adjust the data file headers obtained from the SSPI software package so that they can be used by *speca*. If another method is used to calculate the spectral correlation density functions needed in the SPECCOA algorithm, then their output data files need to be formatted properly as noted in the *speca* program. *tdoasmth* is a controlling script file that calls the signal generating program, spectral correlation density calculating program, and the TDOA calculating algorithm. *plotdata* takes the output data produced by the *speca* program and plots it. Each program is thought to be well commented so that an additional in depth discussion will not be given here. Two areas requiring some comment, however, are generating the test signals and the spectral correlation densities, both of which use the SSPI software package.

A. SIGNAL GENERATION SOFTWARE

This section describes the software created by Statistical Signal Processing, Inc. that was used to simulate signals to test the SPECCOA algorithm. Table E-1 lists the parameters that were varied to obtain the desired characteristics of the test signal. One or more of the parameters shown in bold was varied for each test run. The other

parameters could also be varied but were essentially left in their default setting when the input file was first generated. More information can be obtained from [Ref. 10] which should be consulted frequently when using the SSPI software.

B. GENERATING SPECTRAL CORRELATION DENSITIES

The spectral correlation and cross spectral correlation densities were generated using the SSPI software. Script file *tdoasmth.bat* was used to call the SSPI software package to perform the calculations. The steps involved for a particular test run were as follows:

- adjust parameters in the file *pam_filnam.inp* to create the desired test signal.
- set α_o in the *alphas* file to the value of cycle frequency being exploited.
- adjust smoothing and cycle frequency as necessary in the *tdoasmth.bat* file; smoothing is command input for the *sxaf* command (e.g., -w 0.01), and the alpha value is a command line input for the *specca* routine.
- call *tdoasmth.bat* to calculate the spectral and cross-spectral densities; output files are *surfyx_out* and *surfx_out*.
- generated data files can now be processed by the SPECCOA algorithm to determine the time delay; algorithm output is stored in *Syxx.dat* file which was read directly into the MATLAB routine *plotdata.m* to plot the SPECCOA algorithm output.

TABLE E-1 INPUT PARAMETERS IN
SIGNAL GENERATION PROGRAM

nsamples	2048
samples per symbol	16
bits per symbol	1
constellation	psk
pulse type	nyq
pulse BW	1.000000e+00
pulse tail exp	-2
delay samples	300
real freq shift	1
carrier freq	0.16e+00
carrier phase deg	0.000000e+00
over sample	1
under sample	1
signal power dB	0.5000e+01
noise power dB	1.5000e+01
ASCII or binary	ASCII
real output	0
time output	1
freq output	0

TDOASMTH.BAT

#This file is a batch/script file to run the separate programs needed to
 #generate a test signal, calculate the spectral and cross-spectral correlation
 #densities, and determine the time difference of arrival using the SPECCOA
 #algorithm. Refer to the SSPI software manual for details of 'pam', 'combine',
 #and 'sxaf' programs.

#create the test signals x and y. $y = x(t - \text{delay})$
 #the output files generated are pam_xsig.tim, pam_ysig.tim
 #modifying the pam_xsig.inp or pam_ysig.inp file for binary output
 #vice ASCII output will make the program run faster

```
pam -bits -1534 xsigo      #original signal before noise added
pam -bits -1534 ysigo     #original signal before noise added
pam -noise -2066 nnoise   #generate noise with independent seed
pam -noise -4065 mnoise   #generate m-noise with different ind seed
combine pam_xsigo.tim pam_nnoise.tim pam_xsig.tim #add nnoise to xsig
combine pam_ysigo.tim pam_mnoise.tim pam_ysig.tim #add mnoise to ysig
```

#cross correlate xsig, ysig to generate the cross spectral correlation, Syx.
 #Using the (-f a) option makes the output file ASCII, (-f b) for binary.
 #Use 'fcnvrt' program to make the output file header compatible with
 #the 'speca' program. NOTE: cycle frequency, alpha, is read from the default
 #file named 'alphas'.

```
sxaf -f a -w 0.01 -o surfyx_out pam_ysig.tim pam_xsig.tim
fcnvrt surfyx_out syx_out # fcnvrt infile outfile
```

#autocorrelate xsig to generate Sx. 'fcnvrt' makes output file header
 #compatible with the 'speca' program.

```
sxaf -f a -w 0.01 -o surfx_out pam_xsig.tim
fcnvrt surfx_out sx_out
```

#multiply (Syx) & conjugate(Sx). Inverse Fourier transform the product.
 #The specific cycle frequency, alpha, is an input parameter; 0.0625 is shown.
 #The result is stored in the data file 'Syxx.dat', which can be loaded directly
 #into MATLAB for plotting. Format is a col of real nos. and a col of imag. nos.

```
speca 0.0625 syx_out sx_out Syxx.dat
```

SPECA.C

```

#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include "/tools2/SSPI/pam/fft.c"
#include "/tools2/SSPI/pam/radix.c"

main(argc, argv)

int argc;
char *argv[];
{
int i, r_j_flag1, num_data_pts1, n, n2, N, pwr2;
int r_j_flag2, r_j_flag_out, num_data_pts2, num_of_loops, zeroref, tau;
char *file1, *file2, *wfile;
float alpha, rtemp1, rtemp2, itemp1, itemp2;
COMPLEX *Syxx;
static COMPLEX sigtemp;
FILE *fpr1, *fpr2, *fpw;
i=0;
alpha = atoi(argv[1]);      /* input cycle frequency parameter to used in SPECCOA algorithm */
file1 = argv[2];           /* Syx data file */
file2 = argv[3];           /* Sx data file */
wfile = argv[4];           /* Syxx output data file */

/*****
/*
/* This program computes the TIME DIFFERENCE OF ARRIVAL (TDOA) from
/* two input files, one a delayed version of the other
/*
/*
/* written by LT Timothy A. Benson
/* December 1992
/*
/*-----*/
/* files have the following format

input files -
real or complex flag
num data points
data:
real_part    imag_part

output file -
real or complex flag
num data points
data:
real_part    imag_part

----- */
/*
/*****

```

```

/*****
/*
/* VARIABLE DEFINITIONS
/*
    Syxx          - contains the final data of this program
    i             - a counter
    r_j_flag      - real-imaginary flag for files 1, 2, and output file
    num_data_pts  - number of data pts for file 1, 2
    n             - equals number of data pts in smaller file
    N             - power of two next above n
    pwr2          - a number such that 10^pwr2 is greater than or equal to n
    n2            - n2 = n/2
    num_loops     - loop through this many times to read amount of data from both files
                  equal to the amount of data in the smaller file
    zeroref       - finds zero index for data length -N/2 to N/2
    tau           - any value between -N/2 to N/2. Corresponds to the lag value
    alpha         - cycle frequency of spectral correlation being processed
    rtempl,2     - real variable into which data is read from in-file
    itempl,2     - imaginary variable into which data is read from in-file
    sigtemp       - a temporary variable to hold values of Syxx during manipulation
*/

/* check for the correct no. of command line arguments */
if (argc == 1) {
    printf("spec alpha file1 file2 outfile")
    printf(" - takes spectral data from file1 and file2\n")
    printf("and computes TDOA using SPECCOA algorithm:\n")
    printf("delay = real[ifft(SyxSx*)exp(j*pi*alpha*tau)\n");
    exit();
}
else
    if (argc != 5) {
        printf("spec ... improper no. of arguments\n");
        printf("proper format is: spec alpha Syxfile Sxfile outfile\n");
        printf("\n");
        exit();
    }

/* open files and assign pointers to them */
fpr1 = fopen(file1, "r");
fpr2 = fopen(file2, "r");
fpw = fopen(wfile, "w");

/* determine whether data file has complex numbers and get number of data points */

/* first file */
fscanf(fpr1,"%d %d", &r_j_flag1, &num_data_pts1);

/* second file */
fscanf(fpr2,"%d %d", &r_j_flag2, &num_data_pts2);

```

```

/* loop through num of data points for the file with the lesser amount */
if (num_data_pts1 < num_data_pts2)
    n = num_data_pts1;
else
    n = num_data_pts2;
num_of_loops = n;

/* determine if n is a power of two; if not either truncate or zero-pad to a power of 2 */
pwr2=radix(n); /* set equal to smallest power of two >= n */
N=1;
for (i=0; i< pwr2; ++i)
    N=2*N;

/* determine whether to zero-pad or truncate. if n is greater than 35% of the way
to the next power of 2, then zero-pad, otherwise truncate */

if(n > 1.35*N/2) {
    printf("n=%d, N=%d, n will be zero-padded to %d\n", n, N, N);
}

if(n <= 1.35*N/2) {
    N=N/2;
    printf("n=%d, N=%d, truncating n to %d\n", n, N, N);
}

/* test line */
printf("1: n=%d, N=%d\n", n, N);

/* allocate space for Syxx vector */
Syxx=(COMPLEX*)calloc(N,sizeof(COMPLEX));
if (Syxx == NULL) {
    printf("spec...insufficient space to allocate Syxx\n");
    exit();
}

/* if n is not a power of 2, then zero pad symmetrically up to a power of 2, N */
if (N > n) {
    /* determine the number of zeros that have to be padded */
    numzer = N - n;
}

/* test line */
printf("2: n=%d, N=%d\n", n, N);

```

```

/* loop to read in all data and conjugate and multiply as necessary to
form product of (Syx) * conjugate(Sx) */

for(i=0; i<N; ++i) {

/* if zero-padding needed (ie N > n) */
if(N > n) {

/* zero-pad the first (num. of zeros)/2 elements of Syxx */
if (i < (numzer+1)/2) {
    Syxx[i].r = 0;
    Syxx[i].i = 0;
}

/* zero-pad the last (num. of zeros)/2 elements of Syxx */
if (i >= (N - numzer/2)) {
    Syxx[i].r = 0;
    Syxx[i].i = 0;
}

} /* end loop for zero-padding */

if (r_j_flag1 == 1 && r_j_flag2 ==1) { /* both files are REAL only */
    r_j_flag_out = 1;
    fscanf(fpr1,"%e", &rtemp1);
    fscanf(fpr2,"%e", &rtemp2);
    Syxx[i].r = (rtemp1) * (rtemp2);
    Syxx[i].i = 0;
}

if (r_j_flag1 == 1 && r_j_flag2 ==2) { /* file1 real, file2 complx */
    r_j_flag_out = 2;
    fscanf(fpr1,"%e", &rtemp1);
    fscanf(fpr2,"%e%e", &rtemp2, &itemp2);
    Syxx[i].r = rtemp1 * rtemp2;
    Syxx[i].i = rtemp1 * (-1.0) * itemp2;
}

if (r_j_flag1 == 2 && r_j_flag2 ==1) { /* file1 complx, file2 real */
    r_j_flag_out = 2;
    fscanf(fpr1,"%e%e", &rtemp1, &itemp1);
    fscanf(fpr2,"%e", &rtemp2);
    Syxx[i].r = rtemp1 * rtemp2;
    Syxx[i].i = rtemp2 * itemp1;
}
}

```

```

/* add data to Syxx if i is between zero-padded ends */
if (i >= (numzer +1)/2 && i < (N - numzer/2)) {
    if (r_j_flag1 == 2 && r_j_flag2 ==2) { /* both files are complx */
        r_j_flag_out = 2;
        fscanf(fp1, "%e%e", &rtempl, &itemp1);
        fscanf(fp2, "%e%e", &rtemp2, &itemp2);
        Syxx[i].r = (rtempl * rtemp2 - itemp1 * (-1.0) * itemp2)/n;
        Syxx[i].i = (rtempl * (-1.0) * itemp2 + rtemp2 * itemp1)/n;
    }
}
} /* end of for loop. all data is multiplied and stored in Syxx */

n=N;

/* test line */
printf("all data has been read and multiplied\n");

/* test line */
printf("beginning fft operation\n");

/* inverse Fourier transform Syxx */
pwr2=radix(n); /* set equal to smallest power of two >= n */
N=1;
for (i=0; i< pwr2; ++i)
N=2*N;

/* test line */
printf("n=%d, N=%d\n", n, N);

fft(Syxx, n, -1, 2);
/* Syxx - COMPLEX array to be transformed
   n      - number of points in signal - power of 2 >= n
   -1     - inverse FFT
   2      - normalized FFT
*/

/* shift result from (0 to N-1) to (-N/2 to N/2) */
for (i=0; i < n2; ++i) {
    sigtemp = *(Syxx + i);
    *(Syxx + i) = *(Syxx + i + n2);
    *(Syxx + i + n2) = sigtemp;
}

```

```

/* test line */
  printf("ifft routine finished\n")
  printf("multiplying by final factor\n")
/* multiply by exp(j*pi*alpha*tau) where zero ref is n/2 + 1 */
/* this expression is equivalent to cos(pi*alpha*tau) + j*sin(pi*alpha*tau) */
  zeroref = n/2 + 1;
  for (i=0; i<n; ++i) {
    tau = -(zeroref - i);
    Syxx[i].r = Syxx[i].r * cos(PI*alpha*tau) - Syxx[i].i * sin(PI*alpha*tau);
    Syxx[i].i = Syxx[i].r * sin(PI*alpha*tau) + Syxx[i].i * cos(PI*alpha*tau);
  }

/* print data to file */
  printf("writing data to file\n");
  for (i=0; i<n; ++i)
    fprintf(fpw,"%e  %e\n", Syxx[i].r, Syxx[i].i);
  printf("data is written to file\n");

/* close files */
  fclose(fpr1);
  fclose(fpr2);
  fclose(fpw);

} /* end of program */

```

FCNVRT.C

```

#include <stdlib.h>
#include <stdio.h>

main(argc, argv)

int argc;
char *argv[];
{
int i, r_j_flag, num_data_pts, tempint;
char *rfile, *wfile;
float real, imag, tempfloat;
FILE *fpr, *fpw;
rfile = argv[1];
wfile = argv[2];

/*****
/*
/* This program converts a data file's header to run with 'specaf'
/*
/* written by Timothy A. Benson
/* December 1992
/*
*****/

/*-----*/
/*
/* file to be read has the following format - note line spacing.
/* (see page sxaf-8 in SSPI manual)
/* <...> - indicates nos. read by program to control program flow,
/* i.e., other nos. are not needed but correct linespacing is required.
/*
/* <real or complex flag>
/* <num cuts> alpha min alpha max
/* num data points max fmin max fmax max
/*
/* <alpha> <numf> min fmax
/*
/*----- */

```

```

/*****
/*
/*          VARIABLE DEFINITIONS          */
/*
    i          - a counter
    r_j_flag   - real- imaginary flag
    num_dat_pts - number of data points in in-file
    tempint    - a temporary variable
    tempfloat  - a temporary variable
    real       - data variables
    imag       - data variables
*/

/* check for the correct no. of command line arguments */
if (argc == 1) {
    printf("fcnvrt infile outfile - converts header format for use by fmult\n");
    exit();
}
else
    if (argc != 3) {
        printf("fcnvrt ... improper no. of arguments\n");
        printf("proper format is: fcnvrt infile outfile\n");
        printf("\n");
        exit();
    }

/* open files and assign pointers to them */
fpr = fopen(rfile, "r");
fpw = fopen(wfile, "w");

/* determine whether data file has complex numbers and the
total number of data points. */

fscanf(fpr,"%d%d%e%e", &r_j_flag, &tempint, &tempfloat, &tempfloat);
fscanf(fpr,"%d%e%e", &tempint, &tempfloat, &tempfloat);
fscanf(fpr,"%e%d", &tempfloat, &num_data_pts);

/* print header in output file for data information */
fprintf(fpw,"%d\n%d\n", r_j_flag, num_data_pts);

/* adjust file pointer to first line of data */
fscanf(fpr,"%e%e", &tempfloat, &tempfloat);

```

```

/* loop to copy data into outfile */

i=0;
while (i < num_data_pts) {
    ++i;
    if (r_j_flag == 1) {
        fscanf(fpr,"%e", &real);
        fprintf(fpw,"%e\n", real);
    }

    else if (r_j_flag == 2) {
        fscanf(fpr,"%e%e", &real, &imag);
        fprintf(fpw,"%e    %e\n", real, imag);
    }

} /* end while loop to copy all data */

/* close files */
fclose(fpr);
fclose(fpw);

}

```

PLOTDATA.M

%This program plots data from a file named Syxx.dat created by tdoa.bat

clear
clg

delay=101; %value of delay that was inserted into test signal

load Syxx.dat
len=length(Syxx);
Syxxreal=(Syxx(:,1)');

%determine lag value of peak
[maxval indx]=max(Syxxreal);
lag=indx - len/2 - 1;

%plot data
xaxis=-len/2:len/2-1;
V=[-len/2 0 -1.2*max(-Syxxreal) 1.2*max(Syxxreal)];
V=[-512 0 -1.2*max(-Syxxreal) 1.2*max(Syxxreal)];

axis(V);
subplot(211)
plot(xaxis, Syxxreal), grid
title('SPECCOA OUTPUT')
xlabel('tau/Ts')
ylabel('magnitude')

axis;

LIST OF REFERENCES

1. U.S. Navy Hydrographic Office, *Air Navigation*, H. O. Pub. No. 216, pp. 345-348, 1967.
2. Michael G. Contos and William A. Gardner, *Two-Receiver TDOA/FDOA Analysis*, July 1991.
3. William A. Gardner, "Exploitation of Spectral Redundancy in Cyclostationary Signals," *IEEE Signal Processing Magazine*, vol. 8, pp. 14-36, April 1991.
4. William A. Gardner and Chih-Kang Chen, "Signal-Selective Time-Difference-of-Arrival Estimation for Passive Location of Man-Made Signal Sources in Highly Corruptive Environments, Part I: Theory and Method." *IEEE Transactions on Signal Processing*, Vol. 40, No. 5, May 1992.
5. William A. Gardner and Chih-Kang Chen, "Signal-Selective Time-Difference-of-Arrival Estimation for Passive Location of Man-Made Signal Sources in Highly Corruptive Environments, Part II: Algorithms and Performance." *IEEE Transactions on Signal Processing*, Vol. 40, No. 5, May 1992.
6. Leon W. Couch, II, *Digital and Analog Communication Systems*, third edition, pp. 404-412, Macmillan Publishing Company, 1990.
7. William A. Gardner, *Statistical Spectral Analysis: A Nonprobabilistic Theory*, Prentice Hall, 1987.
8. Radian Corporation, *Spread Spectrum Signals and Techniques Handbook*, third edition, Prepared for the National Security Agency, report no. W3-210-81, June 1981.
9. H. H. Loomis, Jr., "Geolocation of Electromagnetic Emitters," Spectral Special Research Group Handout, Naval Postgraduate School, July 1992 (unpublished).
10. Stephan V. Schell and Chad M. Spooner, *Cyclic Spectral Analysis Software Package*, Version 2.0, Statistical Signal Processing, Inc., 1991.

11. Randy S. Roberts, William A. Brown, and Herschel H. Loomis, Jr.,
"Computationally Efficient Algorithms for Cyclic Spectral Analysis", *IEEE Signal Processing Magazine*, vol. 8, pp. 38-49, April 1991.

INITIAL DISTRIBUTION LIST

- | | | |
|----|---|---|
| 1. | Defense Technical Information Center
Cameron Station
Alexandria, Virginia 22304-6145 | 2 |
| 2. | Dudley Knox Library, Code 52
Naval Postgraduate School
Monterey, California 93943-5000 | 2 |
| 3. | Chairman, Code EC
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, California 93943-5000 | 1 |
| 4. | Professor Herschel H. Loomis, Code EC/Lm
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, California 93943-5000 | 3 |
| 5. | Commander
Naval Space Command
Dahlgren, Virginia 22448-5170 | 1 |
| 6. | LT Timothy A. Benson
Naval Submarine School
Box 700 Code 81
Groton, Connecticut 06340-5700 | 2 |
| 7. | LT Vandenberg, Code 9120
Naval Research Laboratory
4555 Overlook Avenue SW
Washington, D. C. 20375-5320 | 1 |
| 8. | LCDR Oxborough, Code 9110
Naval Research Laboratory
4555 Overlook Avenue SW
Washington, D. C. 20375-5320 | 1 |

9. Statistical Signal Processing, Inc. 1
Attn: Dr. William A. Gardner
6950 Yount Street
Yountville, California 94599

10. Signal Science, Inc. 1
Attn: Mrs. Carolyn Koenig
2985 Kifer Road
Santa Clara, California 95051