7

MITIGATION OF MULTIPATH SIGNALS IN GLOBAL POSITIONING SYSTEM RECEIVERS USING INTERVAL AND GENETIC OPTIMIZATION

by

Jason M. Cuevas

B.S., Mechanical Engineering, Northeastern University
Boston, Massachusetts (1987)

Submitted to the Department of Aeronautics and Astronautics
in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE in AERONAUTICS AND ASTRONAUTICS

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 1993

Signature of Author _____
Department of Aeronautics and Astronautics
June 1993

Approved by _____
Dr. Howard Musoff
Principal Member Technical Staff, Charles Stark Draper Laboratory
Thesis Supervisor

Approved by _____
Dr. Richard Greenspan
Principal Member Technical Staff, Charles Stark Draper Laboratory
Thesis Supervisor

Certified by _____
Dr. George T. Schmidt
Division Manager, Charles Stark Draper Laboratory
Lecturer, Department of Aeronautics and Astronautics
Thesis Advisor

Accepted by _____
Professor Harold Y. Wachman
Chairman, Department Graduate Committee

MITIGATION OF MULTIPATH SIGNALS IN GLOBAL POSITIONING SYSTEM RECEIVERS USING INTERVAL AND GENETIC OPTIMIZATION

by

JASON M. CUEVAS

Submitted to the Department of Aeronautics and Astronautics
on May 7, 1993
in partial fulfillment of the requirements for the degree of
Master of Science in Aeronautics and Astronautics

ABSTRACT

An environmental effect that degrades the accuracy of the Global Positioning System (GPS) is signal multipath. Multipath is the result of a signal arriving at a GPS receiver along multiple propagation paths usually understood to be a "direct" line of sight path plus one or more delayed paths created by reflections from surfaces near the receiving antenna. Multipath causes time-varying and non-zero mean ranging errors. Since Multipath is a "local" phenomenon it is not reduced by differential GPS techniques. Therefore it remains as the largest obstacle to achieving the highest level of accuracy potentially available from the GPS.

A model based approach to jointly estimate the direct and multipath signal time delays is developed in this thesis. The procedure reduces the effects of multipath by estimating the time delays rather than suppressing multipath effects experienced by a typical receiver. The technique as disclosed in this thesis is designed to eliminate the effects of one specular multipath signal. Interval and Genetic optimization algorithms were developed to take the outputs from a bank of correlators and to match them with the proper choice of the signal parameters. Simulations with different multipath scenarios and signal to noise ratios were made. The technique was able to estimate the time delays to better than 0.03 chips corresponding to 8.5 meters for the C/A code in the presence of one specular multipath signal and at a 30 dB SNR.

Thesis Supervisor:	Dr. Howard Musoff Principal Member Technical Staff, C. S. Draper Laboratory
Thesis Supervisor:	Dr. Richard Greenspan Principal Member Technical Staff, C. S. Draper Laboratory
Thesis Advisor:	Dr. George T. Schmidt Division Manager, C. S. Draper Laboratory Department of Aeronautics and Astronautics

Acknowledgments

I would like to express my gratitude to all those who have had a part in the creation of this thesis, and who have supported me throughout the course of my graduate work at MIT and the Draper Laboratory.

Special thanks to Professor Wallace Vander Velde and Dr. Howard Musoff for providing invaluable assistance with the technical base of this thesis, and for giving me their precious time.

To Dr. Richard Greenspan who proposed the multipath problem out of which this thesis evolved and for his guidance during the writing of this thesis.

To Dr. George Schmidt my thesis supervisor and Division Leader at the Draper Laboratory for his comments and ideas that helped shape my thesis.

To Dr. Mathew Bottkol and Ronald Harris for their interest and help in the project.

To Paul G. van Deventer for the use of portions of his Genetic optimization software.

Finally special thanks to Concetta for her patience and support.

This thesis was researched and written at the Charles Stark Draper Laboratory.

Publication of this thesis does not constitute approval by the Draper Laboratory of the findings or conclusions contained herein. It is published for the exchange and stimulation of ideas.

I hereby assign my copyright of this thesis to the Charles Stark Draper Laboratory, Inc., of Cambridge, Massachusetts.

Jason M. Cuevas

Permission is hereby granted by the Charles Stark Draper Laboratory, Inc. to the Massachusetts Institute of Technology to reproduce and to distribute copies of this thesis document in whole or in part.

Table of Contents

List of Illustrations

List of Tables

Chapter 1

Introduction

1.1 Problem Background

The emergence of the Global Positioning System (GPS) as an accurate navigation system is opening the door to an exciting variety of GPS applications. Precision navigation of aircraft, ships, trains, and automobiles in real time are examples of emerging applications. Geodesists use the GPS for surveying. Military systems such as smart munitions and high precision weapons are being designed to make use of the GPS. All of these applications require the highest level of accuracy available from the GPS.

Techniques to provide such accuracy exist and are being constantly improved. Standard GPS P-code tracking provides 5-10 meter level horizontal (CEP) accuracy under ideal conditions. Differential or Relative GPS navigation takes advantage of common environmental error sources in the GPS code measurements that can be eliminated by the use of two closely spaced GPS receivers that track the same satellites. One to three meter level horizontal accuracy (CEP) is common in these modes of operation. Geodesists use interferometric techniques involving the signal carrier to refine differential processing to obtain decimeter level accuracy. All of these techniques provide the levels of accuracy mentioned above in a clean signal environment.

Unfortunately a clean signal environment is not always available. One environmental effect that cannot be reduced by simple means such as

averaging is signal multipath. Multipath is the result of a signal arriving at a receiver along multiple propagation paths usually understood to be a "direct" line of sight path plus one or more delayed paths created by reflections from surfaces near the receiving antenna. Multipath is not peculiar to the Global Positioning System; its effects are seen in everyday life. For example, "Ghosts" that are seen on television sets are the result of multipath. Since multipath is a "local" phenomenon it is not reduced by differential techniques. Therefore it remains as the largest obstacle to achieving the highest level of accuracy potentially available from the Global Positioning System.

Typical GPS receivers operate by tracking the transit time of the pseudorandom noise (PRN) modulated ranging signal broadcast by the GPS satellites along their paths to the receiver. This measurement is accomplished by tracking the phase or time delay of the PRN code relative to a locally generated version of the code in a process called code tracking. GPS positioning is based on one way transit time measurements to four or more satellites. The fourth measurement is needed to solve for the user clock offset from the GPS system time. Generally GPS receivers do not need ultra-stable clocks because four or more satellites are always visible.

Given at least four pseudorange measurements the user can determine position in three coordinates and solve for the time bias from a set of simultaneous equations given by:

$$\varphi_i = \sqrt{(x_i - x)^2 + (y_i - y)^2 + (z_i - z)^2} + c\tau_c + \varphi_e \qquad (1.1\text{-}1)$$

where: φ_i = the pseudorange to the i^{th} satellite

x_i, y_i, z_i = the coordinates of the i^{th} satellite

x, y, z, = the unknown coordinates of the receiver

c = the speed of light

τ_c = the unknown receiver clock bias

φ_e = other errors

The accuracy of the navigation solution is dependent on the pseudorange measurements. Multipath effects cause time varying non-zero mean pseudorange errors [2], [11], [12]. In principal, multipath can be so severe as to cause the receiver to stop tracking the broadcast signals all together. Van Nee demonstrated this analytically in [11] and Tranquilla and Carr [8] observed loss of lock during their field tests. This thesis develops a model based technique to jointly estimate the direct and multipath signal time delays. The procedure reduces the effects of multipath by estimating the time delays rather than by attempting to suppress multipath effects experienced by a typical receiver.

The technique disclosed in this thesis is a model based optimization procedure designed to eliminate the effects of one specular multipath signal. The amplitudes of the direct and multipath signals as well as the their respective time delays are estimated, by means of Interval and Genetic optimization algorithms. These were developed to efficiently associate the outputs from a bank of correlators to the proper choice of the signal delay parameters.

A simulation of the procedure was done that included over fifty different optimization runs with different multipath scenarios and signal to noise ratios. The technique was able to estimate the time delays of the direct and multipath signals to better than 0.03 chips, corresponding to 8.5 meters for the C/A code. Conventional receivers operating in a multipath environment similar to the simulated cases without noise would have errors as large as 60 meters for the C/A code. A factor of seven improvement was achieved. The accuracy of the procedure developed in this thesis is not sensitive to the strength or delay of the multipath signal. Thus the factor of improvement would increase for worse multipath environments and decrease for less severe multipath environments.

1.2 Prior Research

A number of articles exist that model multipath and describe its effects on typical receiver architecture. Only a handful of papers describe

techniques to mitigate the effects of multipath. The papers tend to fall into two broad categories, model based characterization and mitigation techniques based on antenna siting. Five papers proved to be the most useful.

Van Nee has published two papers detailing the effects of multipath on two mechanizations of code tracking loops. He developed signal models to include multipath effects on the coherent Delay Lock Loop (DLL) and the non coherent DLL [11] and [12]. Delay lock loops are the standard control systems that perform the code tracking. Simulation results from his work detail the pseudorange errors caused by multipath. Mean errors on the order of 30 meters for the C/A code and 3 meters for the P code are possible when the multipath approaches the direct path amplitude. More importantly, his work showed that pseudorange errors caused by multipath are not zero mean processes.

A doctoral dissertation by Braasch [2] characterized multipath errors for aircraft in a study of GPS applied to precision approach and landing. The GPS navigation technique he studied was differential GPS (DGPS). In this case a master station with a GPS receiver determines GPS measurement corrections that it transmits to the airborne GPS receiver. The master station obtains the corrections by taking advantage of the fact that it is positioned at a known (surveyed) location. One DGPS technique uses position estimates that are obtained from the master station receiver and compares them to the known surveyed position. Any differences in these positions are attributed to propagation delays and satellite clock errors that are common to both the master station and airborne receivers.

However, if either, or both the master station and the airborne receiver experience multipath effects these corrections become erroneous. The work by Braasch detailed these effects at the master station and airborne receiver. Optimal placement of the master station was studied as well. Braasch concludes that the problem of multipath needs to be resolved by good station siting, excellent antenna design, and receiver mitigation techniques before GPS precision approach can be used to its utmost potential.

Tranquilla and Carr [8] present measured results of the effects of multipath. Their work consisted of taking GPS measurements in the vicinity of a variety of known multipath inducing obstacles. These included dams, large bodies of water, high tension power lines, and a railway crossing. In each of these cases the multipath effects were clearly visible causing pseudorange errors in C/A code tracking as large as 20 meters.

These papers demonstrate the potentially crippling effects of multipath but say little about receiver techniques to mitigate multipath. Van Dierendonck [10] describes a technique for multipath mitigation in which the code tracking loops are tuned to be much tighter than normal. In this case the errors induced by multipath are reduced for weak multipath signals. The work by van Nee [12] argues that larger multipath signal amplitudes induce errors to values comparable to errors that a typical receiver would have under similar circumstances. Thus, to date detailed characterizations of multipath were done but very little to mitigate the effects of multipath.

1.3 Thesis Outline

This thesis develops a technique to eliminate the effects of multipath. Chapter 2 develops the necessary signal model and a typical receiver code tracking loop model. Examples are given in Chapter 2 that demonstrate the effects of multipath in a standard receiver. Chapter 3 develops a strategy for eliminating the effect of multipath when there is only a single interfering path. The problem is reduced to an optimization problem where the signal model parameters, including the direct signal delay, are estimated.

Chapters 4 and 5 explore the Genetic and Interval optimization techniques used to eliminate the effects of multipath. Examples from each procedure provided insight into the optimization problem which ultimately lead to the combination of the Interval and Genetic optimization scheme. The operation of the mitigation procedure in the presence of noise is detailed in Chapter 6. The results presented in Chapter 6 provide the basis

for a new receiver architecture. Finally, Chapter 7 discusses the results and conclusions of the thesis and presents a plan for future research.

The major result of this thesis is a GPS receiver architecture that can eliminate the effects of a single multipath signal on GPS satellite navigation. This thesis lays the ground work for future research into multipath mitigation. The technique developed is fully capable of being expanded to cases where more than one multipath signal is present. Research should continue in this area to achieve the highest levels of accuracy available from the GPS which will remove the major obstacle to accomplish a variety of GPS applications.

Chapter 2

Signal Model and DLL Architecture

2.1 GPS Signal Structure

The signal broadcast by a GPS satellite is an L-band carrier that is phase modulated with a PRN ranging code whose epoch is determined by an ultra-stable atomic clock. The broadcast signal is also phase modulated with a low rate data message that contains information related to the health of the satellites and their ephemerides. The data message can be ignored in the analysis because its effects in a typical receiver are removed by a combination of data demodulation techniques that are relatively insensitive to multipath, and by decision feedback. Complex signal notation will be used to represent the received signal. The analysis begins with a noise free assumption leading to an estimation algorithm whose performance in the presence of noise is evaluated in Chapter 6.

Each GPS satellite broadcasts carrier signals at two different frequencies. The carriers, L1 and L2, are broadcast at 1575 MHz and 1227 MHz respectively. Two orthogonal PRN codes are broadcast in phase quadrature on L1. The two PRN codes are the C/A code and the P code. The C/A code is a 1023 bit "Gold" code that is unique to each satellite. Phase transitions of the code, the "chipping rate", occur at 1.023 MHz. The P code is a biphase modulated pseudorandom sequence with a chipping rate of 10.23 MHz. The L2 signal can transmit either the C/A or P code. In the following signal model the PRN code can represent either the C/A or P code.

2.2 Multipath Signal Model

The GPS signal received in the presence of specular multipath can be modeled as a direct signal plus attenuated and delayed versions of the direct signal. Specular multipath occurs when the broadcast signal is reflected by a surface near the receiving antenna that causes an extra delayed replica of the signal to reach the receiving antenna. When only one multipath signal is present, the received signal is:

$$r(t) = a_0\sqrt{2}p(t-\tau_0)e^{j(\omega_c+\omega_0)t+\theta_0} + a_m\sqrt{2}p(t-\tau_0-\tau_m)e^{j(\omega_c+\omega_m)t+\theta_m} \qquad (2.2\text{-}1)$$

where: $r(t)$ = the received signal

a_0, a_m = the direct and multipath signal amplitudes

$p(t)$ = the GPS PRN code

τ_0 = the direct signal delay

τ_m = the multipath signal delay relative to the direct path

ω_c = the carrier frequency

ω_0, ω_m = the Doppler shifts of the direct and multipath signals

θ_0, θ_m = the phase angles associated with the Doppler shifts

The Doppler shifts for the direct and multipath signals are different. This happens because the line of sight distance to the satellites change at different rates along the direct and multipath paths. An expression for the difference between the Dopplers is developed assuming that the signal broadcast by a single GPS satellite has a plane wave front. This assumption is valid since the GPS satellites are in such high orbits that their signals appear to be broadcast from a source an infinite distance away

from a receiver on the surface of the earth. Figure 2.2-1 is a diagram of a receiver and reflector, the direct signal path, and the reflected signal path.

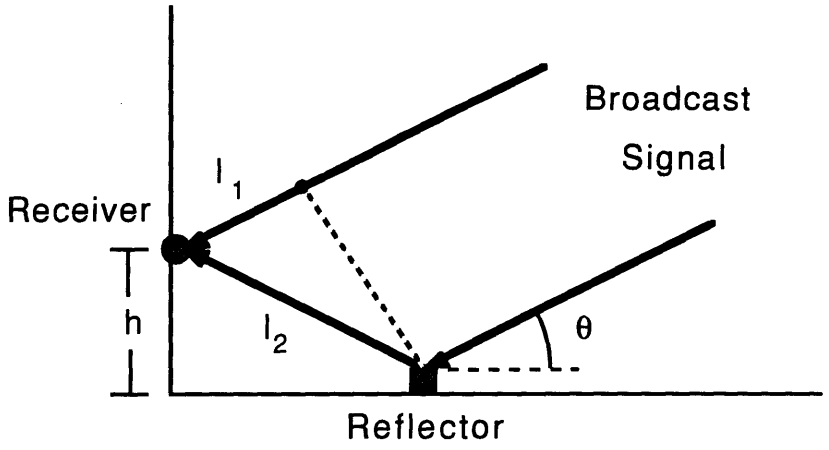

Figure 2.2-1 Receiver Reflector Diagram

The differential frequency is:

$$\Delta f = \frac{2\pi}{\lambda}\frac{d}{dt}(l_2 - l_1) \qquad (2.2\text{-}2)$$

where: λ = the wave length of the carrier

l_1 = the incremental distance along the direct signal path

l_2 = the incremental distance along the reflected path

The length difference term in equation 2.2-2 is a function of the angle of incidence between the signal and reflector and is:

$$l_2 - l_1 = 2h\sin(\theta) \qquad (2.2\text{-}3)$$

where: h = the height of the receiver

θ = the angle of incidence

The frequency difference or spread is a function of the angle of incidence which changes as the satellite moves in its orbit. The frequency difference in terms of θ is:

$$\Delta f = -\frac{4\pi h}{\lambda}(\cos(\theta))\frac{d\theta}{dt} = \omega_0 - \omega_m \qquad (2.2\text{-}4)$$

This frequency spread is called the fading bandwidth. Typical values of the fading bandwidth are discussed in section 2.4.

2.3 Non-coherent DLL Architecture

The non-coherent DLL which is one typical realization of a code tracking loop is examined. Other receiver realizations exist; but these other receiver designs behave in response to multipath in a fashion similar to the non-coherent DLL. Van Nee [11] published work that showed during low dynamics the maximum errors induced by multipath were the same for both the coherent and non-coherent DLL. A block diagram of a portion of a non-coherent DLL is shown in Figure 2.3-1 [11].

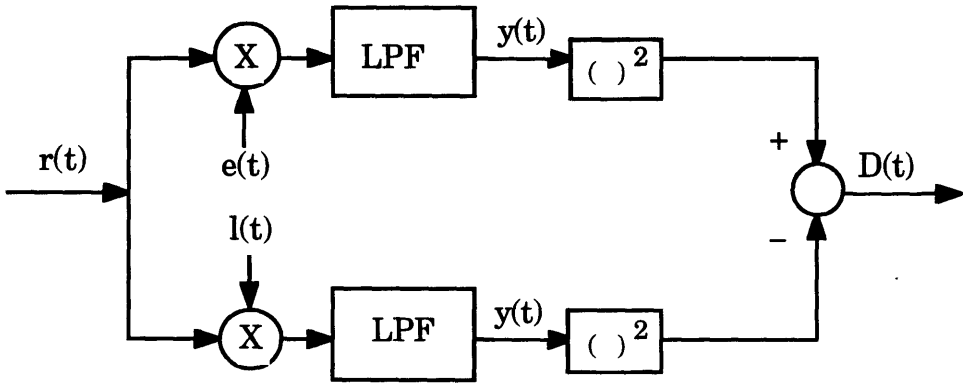

Figure 2.3-1 Non-Coherent DLL

The input signal r(t) is coherently correlated with locally generated PRN codes. The local codes used for correlation are typically offset from the direct signal phase by plus and minus a half of a code chip, often referred to as the "early" and "late" codes. These are:

$$e(t) = \frac{1}{\sqrt{2}} p(t - \hat{\tau}_0 + \frac{T_c}{2}) e^{-j\omega_c t} \qquad (2.3\text{-}2)$$

$$l(t) = \frac{1}{\sqrt{2}} p(t - \hat{\tau}_0 - \frac{T_c}{2}) e^{-j\omega_c t} \qquad (2.3\text{-}3)$$

where: $\hat{\tau}_0$ = the receiver's estimate of the direct signal delay

T_c = the PRN code chip length

The product signals are low pass filtered to obtain the early and late correlation functions of the PRN code. The output of each filter is the sum of the delayed correlation functions for the direct and multipath signal components. These are:

$$y_e(\tau, t) = a_0 \Phi(\tau_0 - \hat{\tau}_0 + \frac{T_c}{2}) e^{j\omega_0 t + \theta_0} + a_m \Phi(\tau_0 - \hat{\tau}_0 + \tau_m + \frac{T_c}{2}) e^{j\omega_m t + \theta_m} \qquad (2.3\text{-}4)$$

$$y_l(\tau, t) = a_0 \Phi(\tau_0 - \hat{\tau}_0 - \frac{T_c}{2}) e^{j\omega_0 t + \theta_0} + a_m \Phi(\tau_0 - \hat{\tau}_0 + \tau_m - \frac{T_c}{2}) e^{j\omega_m t + \theta_m} \qquad (2.3\text{-}5)$$

where: $\Phi(\alpha)$ = the autocorrelation function of the PRN code

The autocorrelation function of an ideal PRN code is:

$$\Phi(\alpha) = 1 - |\alpha| \qquad |\alpha| \le T_c \qquad (2.3\text{-}6)$$

$$\Phi(\alpha) = 0 \qquad |\alpha| \ge T_c$$

The autocorrelation function of a PRN code that a receiver realizes is slightly different from the above model. The function becomes rounded at the peak due to the finite integration time of the low pass filter. Also, the function may take on non-zero values outside of the plus and minus one chip band. These effects are small relative to the autocorrelation function amplitude and for this reason equation 2.3-6 will suffice.

The non-coherent DLL forms the error discriminator as the difference between the magnitude squared of the early and late correlator outputs. In the absence of multipath the discriminator output as a function of the tracking loop error e is:

$$D(e) = a_0^2(\Phi(e+\frac{T_c}{2})^2 - \Phi(e-\frac{T_c}{2})^2) \qquad (2.3\text{-}7)$$

where: $e = \tau_0 - \hat{\tau}_0$ = error in the delay estimate

The code tracking loop uses the discriminator output as an error signal to control its tracking of the direct signal delay. Figure 2.3-2 is a plot of the discriminator as a function of the delay error parameter. The receiver tracks the null of the discriminator as opposed to tracking a correlation peak which is a more difficult task in the presence of noise and when filtering effects in the receiver smooth the peak of the signal correlation function. Essentially the loop balances the outputs of the early and late correlators. In the absence of multipath the discriminator takes a value of zero when the loop is tracking the direct signal.

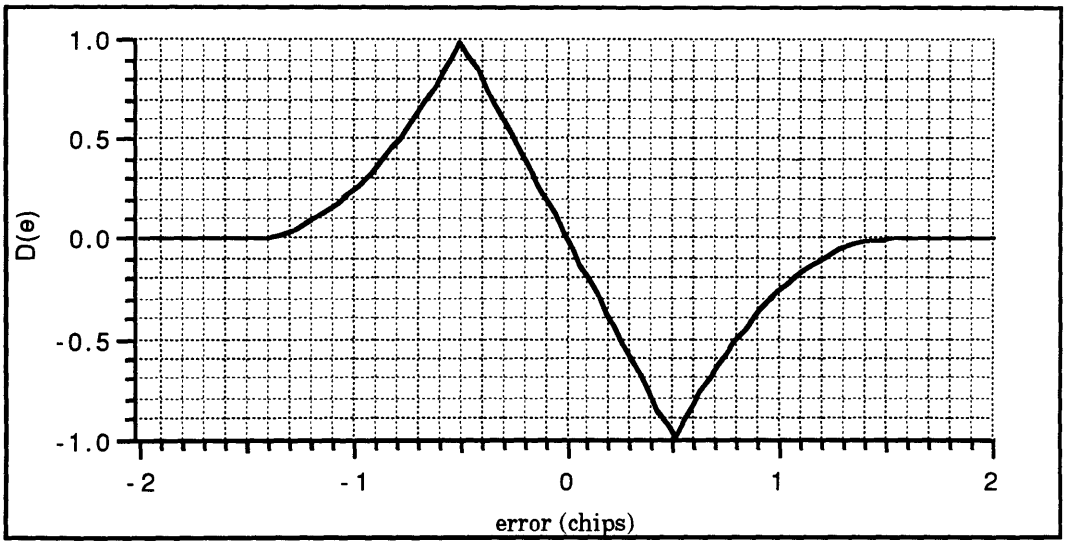

Figure 2.3-2 Normalized Non-Coherent DLL Discriminator

The discriminator output becomes dependent on time as well as the signal delays when multipath is present. In this case the discriminator characteristic is:

$$D(t,e) = a_0^2(\Phi(e+\frac{T_c}{2})^2 - \Phi(e-\frac{T_c}{2})^2) + a_m^2(\Phi(e+\tau_m+\frac{T_c}{2})^2 - \Phi(e+\tau_m-\frac{T_c}{2})^2) - 2a_0a_m(\Phi_e - \Phi_l)\cos(\Delta\omega t + \Delta\theta) \quad (2.3\text{-}8)$$

where: $\Phi_e = \Phi(e+\frac{T_c}{2})\Phi(e+\tau_m+\frac{T_c}{2})$

$\Phi_l = \Phi(e-\frac{T_c}{2})\Phi(e+\tau_m-\frac{T_c}{2})$

$\Delta\omega = \omega_0 - \omega_m$

$\Delta\theta = \theta_0 - \theta_m$

The cosine term causes the discriminator to beat or fade as a function of time. The fading bandwidth is the difference between the Doppler frequencies, $\Delta\omega$. The fading bandwidth typically has 10-15 minute long periods [9] that cause the output of each low pass filter to fade slowly in and out. More importantly, the discriminator output does not equal zero when the direct signal is being accurately tracked. The multipath signal introduces a time varying shift in the zero crossing of the discriminator. The resulting pseudorange error is a non-zero mean process. Examples of this behavior are given in section 2.5 and in [2], [11], [12].

2.4 Phasor Interpretation

The signal fading associated with the multipath signal is easily visualized by phasor diagrams. Figure 2.4-1 diagrams the phasors associated with the direct and multipath signals after the correlation process. The magnitude squared of the early and late correlator outputs are taken and differenced forming the discriminator. Since the discriminator function is periodic in time, the root of it will be also. The root is considered for convenience.

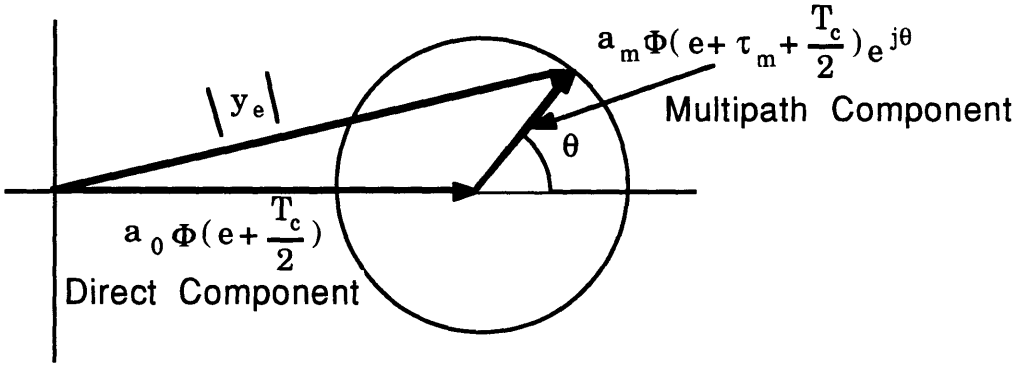

Figure 2.4-1 Phasor Diagram

The point where the discriminator equals zero is a function of the angle θ:

$$Z_\theta(e) = \left| a_0\Phi(e+\frac{T_c}{2}) + a_m\Phi(e+\tau_m+\frac{T_c}{2})e^{j\theta} \right| - \left| a_0\Phi(e-\frac{T_c}{2}) + a_m\Phi(e+\tau_m-\frac{T_c}{2})e^{j\theta} \right| \quad (2.4\text{-}1)$$

where: $\theta = \Delta\omega t + \Delta\theta$

In the nominal case without multipath, the zero crossing is at zero as expected. A limiting case can be examined to bound the maximum error caused by one multipath signal. Assume that the signal amplitudes are positive and the angle θ equals zero. In this case the absolute values on equation 2.4-1 can be dropped because $\Phi(\alpha)$ is an even function. The autocorrelation terms in 2.4-1 are expanded in series with Φ' defined as the derivative of Φ:

$$\Phi(e+\frac{T_c}{2}) = \Phi(\frac{T_c}{2}) + \Phi'(\frac{T_c}{2})e \quad (2.4\text{-}2)$$

$$\Phi(e-\frac{T_c}{2}) = \Phi(\frac{T_c}{2}) - \Phi'(\frac{T_c}{2})e \quad (2.4\text{-}3)$$

$$\Phi(e+\tau_m+\frac{T_c}{2}) = \Phi(\frac{T_c}{2}) + \Phi'(\frac{T_c}{2})(e+\tau_m) \quad (2.4\text{-}4)$$

$$\Phi(e+\tau_m-\frac{T_c}{2}) = \Phi(\frac{T_c}{2}) - \Phi'(\frac{T_c}{2})(e+\tau_m) \quad (2.4\text{-}5)$$

Considering the case where θ equals zero and substituting 2.4-2 - 2.4-5 into equation 2.4-1 yields:

$$Z_0(e) = 0 = 2\Phi'(\frac{T_c}{2})(a_0 e + a_m(e+\tau_m)) \qquad (2.4\text{-}6)$$

Equation 2.4-6 is valid within the linear range of a detector having a one chip spacing between the correlators. Since $\Phi'(\frac{T_c}{2}) \neq 0$, the zero value of equation 2.4-6 (i.e. the tracking error), occurs for:

$$Te = -\frac{a_m}{a_0 + a_m}\tau_m \qquad (2.4\text{-}7)$$

Equation 2.4-7 is valid within the linear range of the discriminator. A more general result presented by van Nee [12] is described in Appendix A. Van Nee's equations are general for arbitrary correlator spacing. Thus far the case where the difference between the early and late correlators equals one code chip was developed.

2.5 Example Pseudorange Errors

Multipath induces time varying pseudorange errors because the satellites move relative to the user. As mentioned earlier the discriminator is distorted by multipath. The instantaneous tracking errors associated with the distortion are calculated as a function of the signal to multipath ratio (SMR = a_0 / a_m), the fading bandwidth, and the multipath delay relative to the direct signal delay from equation 2.3-8. The tracking error is the value $e = \tau_0 - \hat{\tau}_0$ that sets the discriminator equal to zero. Three cases are presented with different multipath delays.

The three cases analyzed all have signal to multipath ratios of 6 dB and a fading bandwidth of 0.62 Hz. Relative signal delays between the direct and multipath signal of 0.05, 0.3, and 0.5 chips are presented. Figure 2.5-1 displays the resulting pseudorange errors plotted as a function of

time. The signal fading is apparent in each curve. Errors as large as 0.23 chips (67 meters for the C/A code) are visible. The mean errors for each case are 0.0002, 0.0041, and 0.0057 chips (0.05 - 1.6 meters for the C/A code). The instantaneous errors are large and non-zero mean.

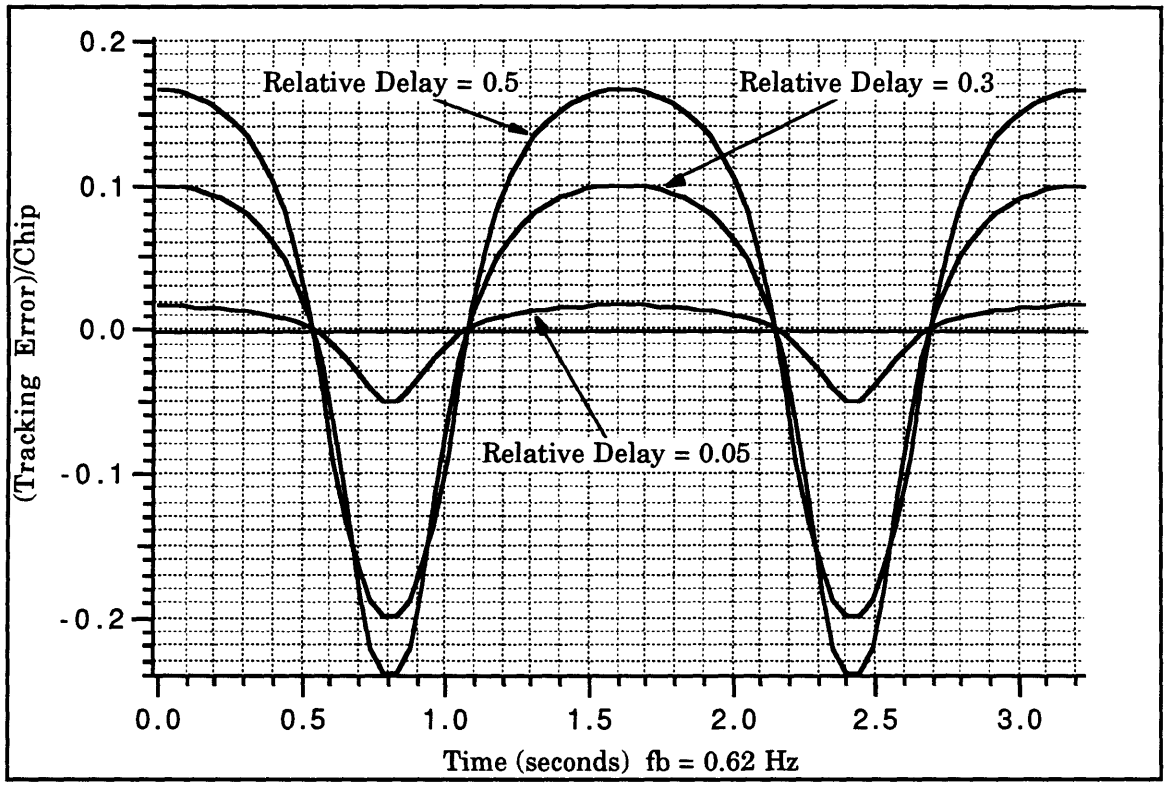

Figure 2.5-1 Example Tracking Errors

In order to get a better feel for the nature of multipath an ensemble of instantaneous tracking errors from different cases were made. In each case the mean and standard deviation of the tracking errors were calculated. Figure 2.5-2 is a plot of the mean error per chip versus the relative multipath delay. Figure 2.5-3 contains the standard deviation. Each plot contains four curves that give results for different SMRs. Similar plots are in [2], [11], and [12]. The largest mean error in the four cases was 37 meters for the C/A code and occurred when the SMR equaled 3 dB.

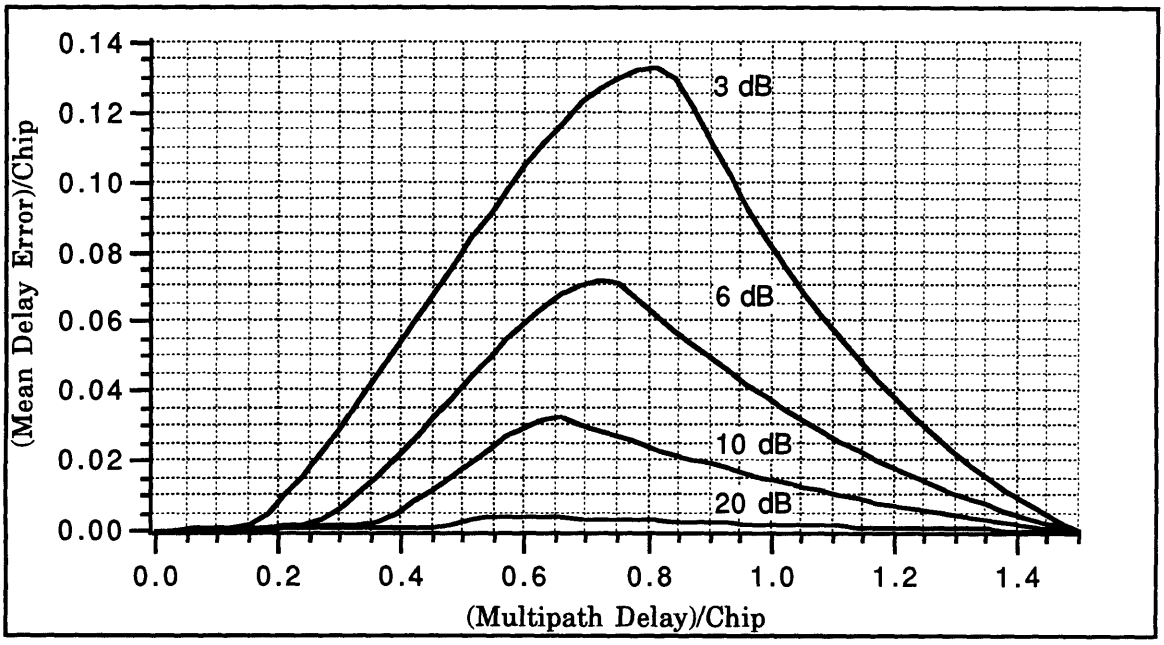

Figure 2.5-2 Mean Tracking Error

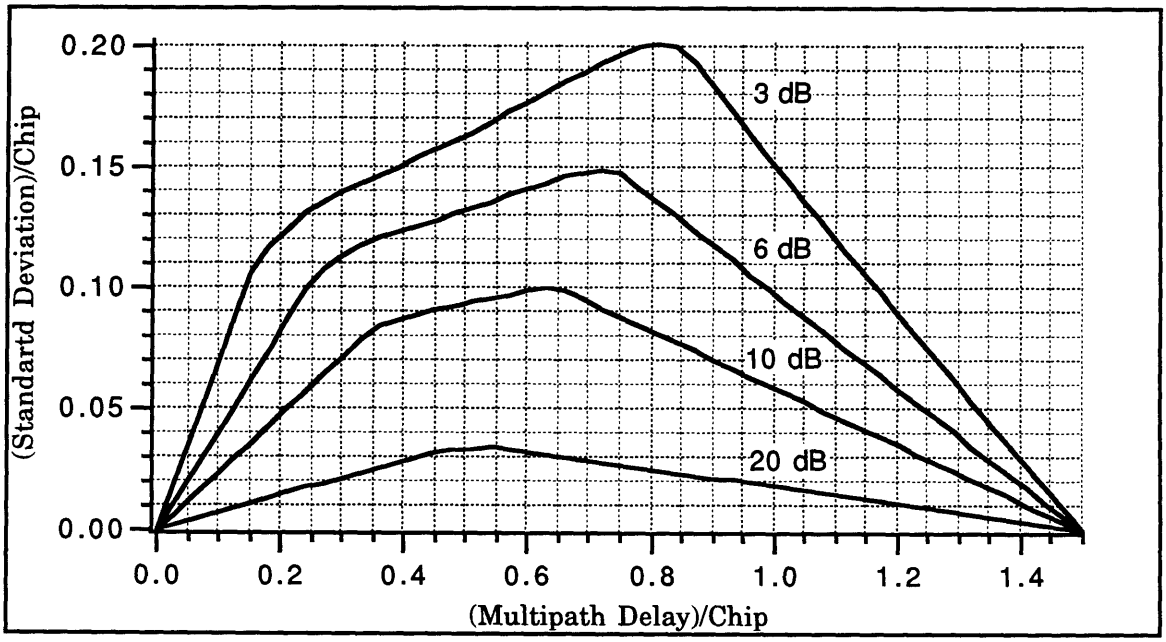

Figure 2.5-3 Standard Deviation of Tracking Errors

In all of the above simulations the fading bandwidth was set to 0.62 Hz. Van Nee [11] provides a procedure for calculating the fading bandwidth for a stationary receiver and one reflector. Under these conditions the maximum fading bandwidth is 0.62 Hz. A note about mobile receivers is in order. The fading bandwidth under these conditions can be much larger than the stationary case with typical values between 0.62 Hz and 50 Hz [11]. DLL operation during dynamics produces pseudorange errors similar in magnitude to those of a stationary non-coherent DLL receiver [12]. Hence, only stationary receiver cases were presented.

2.5 Summary

This chapter presented an overview and model of a typical code tracking loop. The effects of one specular multipath on a non coherent DLL were shown. The multipath effects demonstrated in this Chapter were large. Averaging the receiver outputs resulted in non zero-mean errors as large as 30 meters for the C/A code. The motivation for developing a multipath mitigation scheme is provided from these results.

Chapter 3

Multipath Mitigation

3.1 Introduction

The navigation errors due to multipath were characterized in Chapter 2. A strategy to reduce these errors is needed. This chapter presents a mitigation technique based on estimating the true direct signal delay in the presence of multipath. The multipath delay is estimated as a by product of the procedure. This approach is illustrated in the context of interference from a single specular multipath component. However, the optimization scheme that is developed is extendible to cases where more than one specular multipath signal is present.

3.2 PRN Code Autocorrelation Function

The output of each correlators is disturbed by the multipath signal. The outputs for a pair of correlators are given by equations 2.3-4 and 2.3-5. An optimization procedure is developed that takes advantage of the fact that these correlator outputs are the weighted sum of the delayed autocorrelation values of the direct signal. In order to show this, a new slightly different signal and correlation model is developed. The significance of the changes becomes obvious in later sections. The received

signal is given by:

$$r(t) = \alpha_0 \sqrt{2} p(t-\tau_0) e^{j(\omega_c+\omega_0)t+\theta_0} + \alpha_m \sqrt{2} p(t-\tau_m) e^{j(\omega_c+\omega_m)t+\theta_m} \qquad (3.2\text{-}1)$$

where: α_0, α_m = the direct and multipath signal amplitudes

τ_0, τ_m = the direct and multipath signal delays

ω_0, ω_m = the Doppler shifts of the direct and multipath signals

θ_0, θ_m = the phase angles associated with the Doppler shifts

The signal model represented by equation 3.2-1 is slightly different than that presented in Chapter 2. The changes are made for convenience and to ease the model formulation. The signal amplitude symbols and the time argument in the multipath PRN code are different. The weights of the correlation function that is developed are functions of the signal amplitudes, α_0 , α_m, and the Doppler terms. The weights are denoted by a_0 and a_m. The other change is the multipath delay, and in this case the multipath delay is constrained to be greater than the direct signal delay. This is true as long as the direct signal path is not obstructed in which case the multipath signal would reach the antenna first.

If it were possible to continuously correlate the received signal r(t) against a locally generated reference signal, the complete PRN code cross correlation function could be obtained. The resulting function is given by:

$$y(\tau, t) = \alpha_0 \Phi(\tau+\tau_0) e^{j\omega_0 t+\theta_0} + \alpha_m \Phi(\tau+\tau_m) e^{j\omega_m t+\theta_m} \qquad (3.2\text{-}2)$$

where: $\Phi(\bullet)$ = delayed version of the PRN code autocorrelation function

This correlation function can be broken up into real and imaginary parts:

$$y_R(\tau,t) = \alpha_0\Phi(\tau+\tau_0)\cos(\omega_0 t+\theta_0)+\alpha_m\Phi(\tau+\tau_m)\cos(\omega_m t+\theta_m) \quad (3.2\text{-}3)$$

$$y_I(\tau,t) = j\alpha_0\Phi(\tau+\tau_0)\sin(\omega_0 t+\theta_0)+j\alpha_m\Phi(\tau+\tau_m)\sin(\omega_m t+\theta_m) \quad (3.2\text{-}4)$$

The sinusoidal terms can be treated as constants assuming the correlation processes is done fast compared to the Doppler periods. The resulting signals are:

$$\Phi_{TR}(\tau) = a_{0R}\Phi(\tau+\tau_0)+a_{mR}\Phi(\tau+\tau_m) \quad (3.2\text{-}5)$$

$$\Phi_{TI}(\tau) = a_{0I}\Phi(\tau+\tau_0)+a_{mI}\Phi(\tau+\tau_m) \quad (3.2\text{-}6)$$

where: $a_{0R} = \alpha_0\cos(\omega_0 t+\theta_0)$

$a_{mR} = \alpha_m\cos(\omega_m t+\theta_m)$

$a_{0I} = j\alpha_0\sin(\omega_0 t+\theta_0)$

$a_{mI} = j\alpha_m\sin(\omega_m t+\theta_m)$

The correlation functions given by equations 3.2-5 and 3.2-6 are each the sum of weighted delayed versions of the PRN code autocorrelation function. A sample function is shown in Figure 3.2-6. The function in this case is made up of a direct signal having a peak amplitude a_0 and a multipath signal with a peak amplitude of a_m. The multipath signal is delayed two chips relative to the direct signal.

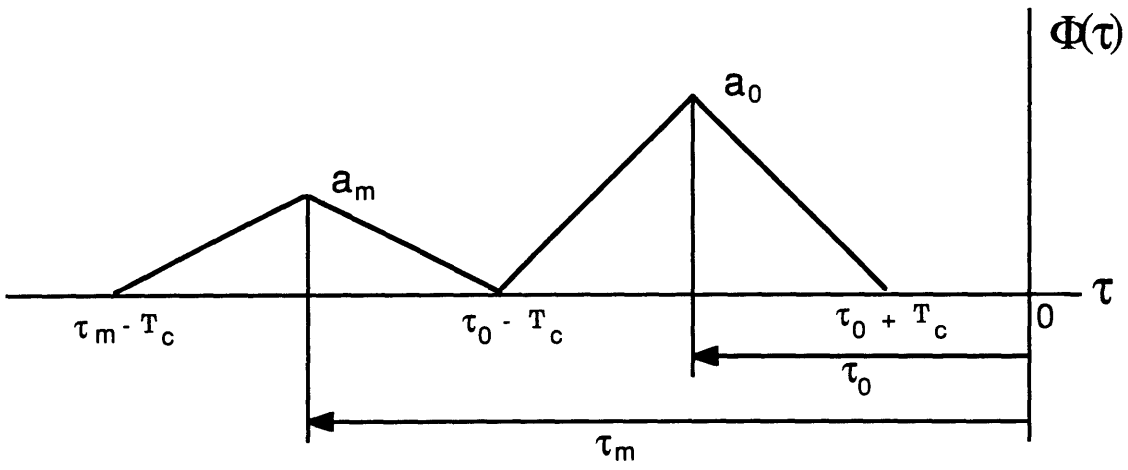

Figure 3.2-6 Sample correlation function

3.3 Signal Parameter Estimation Using the Correlation Function

The real and imaginary parts of the disturbed correlation function were described. For convenience only the real signal is considered in the balance of this study. In any implementation of the technique the real and imaginary parts would be used for parameter estimation because the estimation process developed for the real part of the signal is directly applicable to processing the imaginary part of the signal. Therefore the subscripts denoting the real and imaginary signals will be dropped in the following analysis. The correlation signal to be considered is then:

$$\Phi_T(\tau) = a_0\Phi(\tau+\tau_0)+a_m\Phi(\tau+\tau_m) \qquad (3.3\text{-}1)$$

The signal parameters, a_0, a_m, τ_0, τ_m , can be estimated provided that the correlation function can be measured.

3.3.1 The Closed Form Solution

The correlation function given by equation 3.3-1 is dependent on four unknowns, a_0, a_m, τ_0, τ_m. It seems reasonable to expect that if four measurements of the function are made then one need only analytically solve the resulting four simultaneous equations for the parameters. Unfortunately this does not work out well in real application. The localized PRN code autocorrelation function is piece wise linear and is expressed as:

$$\Phi(\tau) = 1 - \tau_d - \tau \qquad \text{when} \quad -\tau_d < \tau < 1 - \tau_d$$

$$\Phi(\tau) = 1 + \tau_d + \tau \qquad \text{when} \quad -\tau_d - 1 < \tau < -\tau_d$$

$$\Phi(\tau) = 0 \qquad \text{elsewhere} \qquad (3.3.1\text{-}1)$$

where: τ_d = the signal delay parameter (either τ_0 or τ_m)

The resulting four equations, that are made up from $\Phi(\tau)$, are non-linear functions of the signal parameters. Closed form solutions of these equations do not exist in most cases. Thus, an optimization procedure to estimate the signal parameters is utilized.

3.4 Optimization

An optimization procedure is used to estimate the signal parameters in the presence of multipath. Measurements of the disturbed correlation function are made and the set of signal parameters that produces the best match to the measured function are estimated. A cost function is defined and the optimization is carried out by minimizing this function.

The error between the estimated correlation and true correlation at

any measurement point is:

$$E(\tau_i) = \hat{\Phi}(\tau_i) - \Phi(\tau_i) \tag{3.4-1}$$

where:

$$\hat{\Phi}(\tau_i) = \hat{a}_0 \Phi(\tau_i + \hat{\tau}_0) + \hat{a}_m \Phi(\tau_i + \hat{\tau}_m)$$

$$\Phi(\tau_i) = a_0 \Phi(\tau_i + \tau_0) + a_m \Phi(\tau_i + \tau_m)$$

τ_i = a measurement time

$\hat{a}_0, \hat{a}_m, \hat{\tau}_0, \hat{\tau}_m$ = estimates of the signal parameters

The cost function used in the optimization is the sum of the squared measurement residuals and is:

$$J(\hat{a}_0, \hat{a}_m, \hat{\tau}_0, \hat{\tau}_m, \Phi(\tau_i)) = \sum_{i=1}^{n} E(\tau_i)^2 \tag{3.4-2}$$

where: n = the number of measurements

At this stage the only remaining steps in the optimization are forming the measurements and selecting a suitable optimization procedure to minimize the cost function.

3.5 Measurement Formulation

Traditional receivers typically compute two points of the correlation function and form the error discriminator for tracking. However many samples of the correlation function are needed for signal parameter estimation by optimization. These samples are obtained from the output of a bank of correlators. Figure 3.5-1 is a diagram of n correlators.

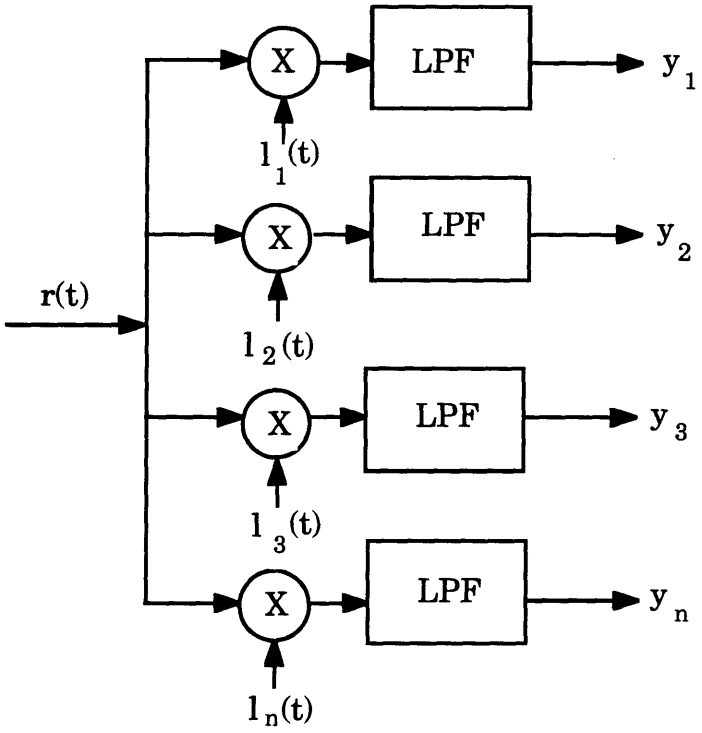

Figure 3.5-1 Bank of correlators

In this case the input signal is coherently correlated with locally generated codes all offset from each other in order to measure the disturbed correlation function.

The local codes are:

$$l_i(t) = \frac{1}{\sqrt{2}} p(t+\tau_i) e^{-j\omega_c t} \qquad (3.5\text{-}1)$$

The outputs of the correlators are:

$$y_i(t,\tau_i) = \alpha_0 \Phi(\tau_i + \tau_0) e^{j\omega_0 t+\theta_0} + \alpha_m \Phi(\tau_i + \tau_m) e^{j\omega_m t+\theta_m} \qquad (3.5\text{-}2)$$

The real part of the measured correlation signal is:

$$\Phi(\tau_i) = a_0\Phi(\tau_i + \tau_0) + a_m\Phi(\tau_i + \tau_m) \qquad (3.5\text{-}3)$$

where: $a_0 = \alpha_0 \cos(\omega_0 t + \theta_0)$

$a_m = \alpha_m \cos(\omega_m t + \theta_m)$

Equation 3.5-3 represents computed samples of equation 3.3-1. The sampled correlation function given by equation 3.5-3 can take on zero values if the measurements are not spaced properly. So far a fine measurement process has been considered. The autocorrelation function $\Phi(\bullet)$ can be zero as shown by equation 3.3.1-1. In order to ensure that the non-zero parts of the function are sampled an acquisition process is first performed.

The samples are taken at times spaced around the direct signal delay τ_0 resulting in a sampling time of the form:

$$\tau_i = c_i + \tilde{\tau}_0 \qquad (3.5\text{-}4)$$

where: c_i = the local time offsets

$\tilde{\tau}_0$ = the initial direct signal delay estimate

The acquisition process provides the initial coarse estimate of the direct signal delay. A number of discrete measurement points are then taken relative to the initial estimate as laid out by the fixed local offsets. The initial estimate is obtained by external aiding or by a search process combined with threshold detection. During acquisition correlations are taken over a large region until the output of any one correlator is above a threshold. The sample time for that measurement is used as the initial estimate.

3.6 Proof of Concept

The problem of signal parameter estimation was reduced to a minimization problem. A suitable optimization strategy to minimize the cost function J is needed. The cost function limits the choice of optimization procedures because components of it are piece wise linear and do not have continuous derivatives. Also it is not clear without some visualization of the cost function over the parameter space if there are local minima and saddle points. Because displaying the cost function over the four dimensional parameter space is not practical a simplified correlation model and simple optimization procedure that was easy to analyze was developed to explore the viability of optimization used as a technique for multipath mitigation.

Instead of working with the correlation function and its measurements directly, the Fourier series representations of them were used. Figure 3.6-1 shows a plot of the unit correlation function in terms of chips. The time period of the signal is defined by the parameter T. The series representation for the function is:

$$\Phi_f(\tau) = \frac{1}{T} + \sum_{k=0}^{\infty} \frac{T}{\pi^2 k^2} \left(1 - \cos\left(\frac{2\pi k}{T}\right)\right) \cos\left(\frac{2\pi k \tau}{T}\right) \qquad (3.6\text{-}1)$$

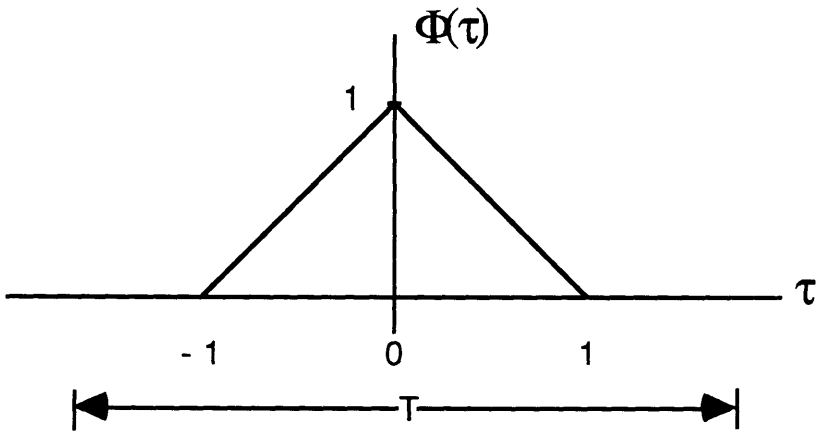

Figure 3.6-1 Unit correlation function

Only the fundamental series term is used for the signal and measurements. The cost function with n measurements is defined by:

$$J = \sum_{i=1}^{n} \{\hat{a}_0 A + \hat{a}_0 B \cos(C(\tau_i + \hat{\tau}_0)) + \hat{a}_m A + \hat{a}_m B \cos(C(\tau_i + \hat{\tau}_m)) - \Phi_f(\tau_i)\}^2 \quad (3.6\text{-}2)$$

where: $A = \frac{1}{T}$

$$B = \frac{T}{\pi^2}(1 - \cos(\frac{2\pi}{T}))$$

$$C = \frac{2\pi}{T}$$

The cost function now has continuous derivatives. A simple optimization procedure based on Newton's method is used to minimize J. The partial derivatives of J with respect to each of the signal parameters are set to zero. The partial derivatives are:

$$\frac{\partial J}{\partial \hat{a}_0} = g_1 = 2\sum_{i=1}^{n} \{\alpha(\hat{a}_0 A + \hat{a}_0 B \cos(C(\tau_i + \hat{\tau}_0)) + \hat{a}_m A + \hat{a}_m B \cos(C(\tau_i + \hat{\tau}_m)) - \Phi_f(\tau_i))\} \quad (3.6\text{-}3)$$

$$\frac{\partial J}{\partial \hat{a}_m} = g_2 = 2\sum_{i=1}^{n} \{\beta(\hat{a}_0 A + \hat{a}_0 B \cos(C(\tau_i + \hat{\tau}_0)) + \hat{a}_m A + \hat{a}_m B \cos(C(\tau_i + \hat{\tau}_m)) - \Phi_f(\tau_i))\} \quad (3.6\text{-}4)$$

$$\frac{\partial J}{\partial \hat{\tau}_0} = g_3 = 2\sum_{i=1}^{n} \{\phi(\hat{a}_0 A + \hat{a}_0 B \cos(C(\tau_i + \hat{\tau}_0)) + \hat{a}_m A + \hat{a}_m B \cos(C(\tau_i + \hat{\tau}_m)) - \Phi_f(\tau_i))\} \quad (3.6\text{-}5)$$

$$\frac{\partial J}{\partial \hat{\tau}_m} = g_4 = 2\sum_{i=1}^{n} \{\gamma(\hat{a}_0 A + \hat{a}_0 B \cos(C(\tau_i + \hat{\tau}_0)) + \hat{a}_m A + \hat{a}_m B \cos(C(\tau_i + \hat{\tau}_m)) - \Phi_f(\tau_i))\} \quad (3.6\text{-}6)$$

where: $\alpha = A + B\cos(C(\tau_i + \hat{\tau}_0))$

$$\beta = A + B\cos(C(\tau_i + \hat{\tau}_m))$$

$$\phi = -2\hat{a}_0 BC\sin(C(\tau_i + \hat{\tau}_0))$$

$$\gamma = -2\hat{a}_m BC\sin(C(\tau_i + \hat{\tau}_m))$$

These four equations are used to solve for the signal parameters. Define the vector $\underline{x} = [\hat{a}_0, \hat{a}_m, \hat{\tau}_0, \hat{\tau}_m]^T$. Newton's method solves the four equations for $\underline{x}$ by iteration:

$$\underline{x}^{new} = \underline{x}^{curr} - L^{curr^{-1}}\,\underline{g}^{curr} \tag{3.6-7}$$

The vector $\underline{g}$ is the vector of partial derivatives. The matrix L is a Jacobian of the functions given by equations 3.6-3 -3.6-6 and is:

$$L = \begin{bmatrix} \frac{\partial g_1}{\partial \hat{a}_0} & \frac{\partial g_1}{\partial \hat{a}_m} & \frac{\partial g_1}{\partial \hat{\tau}_0} & \frac{\partial g_1}{\partial \hat{\tau}_m} \\ \frac{\partial g_2}{\partial \hat{a}_0} & \frac{\partial g_2}{\partial \hat{a}_m} & \frac{\partial g_2}{\partial \hat{\tau}_0} & \frac{\partial g_2}{\partial \hat{\tau}_m} \\ \frac{\partial g_3}{\partial \hat{a}_0} & \frac{\partial g_3}{\partial \hat{a}_m} & \frac{\partial g_3}{\partial \hat{\tau}_0} & \frac{\partial g_3}{\partial \hat{\tau}_m} \\ \frac{\partial g_4}{\partial \hat{a}_0} & \frac{\partial g_4}{\partial \hat{a}_m} & \frac{\partial g_4}{\partial \hat{\tau}_0} & \frac{\partial g_4}{\partial \hat{\tau}_m} \end{bmatrix} \tag{3.6-8}$$

The signal parameters are estimated by iteration using equation 3.6-7 starting with some initial guess of the signal values.

3.6.1 Optimization Results

A simulation was run using the outlined procedure to estimate signal parameters from two test cases. Inputs to the simulation were measurements of the true disturbed correlation signal fundamental and

initial estimates of the signal parameters. Table 3.6.1-1 contains a summary of the inputs for the two test cases.

Case	Truth Model a_0, a_m, τ_0, τ_m	Initial Condition $\hat{a}_0, \hat{a}_m, \hat{\tau}_0, \hat{\tau}_m$
1	1.0, 0.5, 0.2, 0.5	0.9, 0.4, 0.2, 0.5
1A	1.0, 0.5, 0.2, 0.5	0.9, 0.4, 0.1, 0.4
2	1.0, -0.5, 0.2, 0.5	0.9, -0.4, 0.2, 0.5

Table 3.6.1-1 Simulations

Five iterations of the process were conducted during each test case. Output from the test cases is displayed in Figures 3.6.1-1 - 3.6.1-3. Each figure is made up of four subplots that show the sampled correlation function, error in the signal estimates, the partial derivatives, and the cost function. The first subplot displays the sampled correlation function that is used in the minimization. Plots of the attenuated and delayed PRN code autocorrelation functions that make up the sampled correlation function are superimposed on this subplot as well. The next subplot contains the percent error of the signal estimates. The percent difference between the true value and the estimated value for each signal parameter is displayed in this plot. The third subplot displays the results of evaluating equations 3.6-3 - 3.6-6 (g_1, g_2, g_3, g_4) with the parameter estimates and the samples from the total correlation function. The last plot shows the resultant cost function at each iteration step.

Results from the first case, shown in Figure 3.6.1-1, are poor. The first iteration improved the amplitude estimates, but drove the delay estimates off the correct values. Subsequent iterations corrupted all of the parameter estimates. The first case was run a second time with new initial conditions. The parameter estimates, shown in Figure 3.6.1-2, were improved in this case. The divergence problem displayed in the previous case was not present. One final case was run with the sign reversed on the

multipath signal amplitude. Results from this case, Figure 3.6.1-3, look similar to the first case. The amplitude estimates were improved while the delay estimates were made worse.

Some conclusions can be drawn from the results of the three cases. The cost function shows little variation for large changes in the parameters. In all three simulation runs, the cost function decreases to zero very quickly. The problem formulation seems to be very sensitive to initial conditions. The results between the first and second run bear this out. Small changes in the initial conditions resulted in large changes in the parameter estimates.

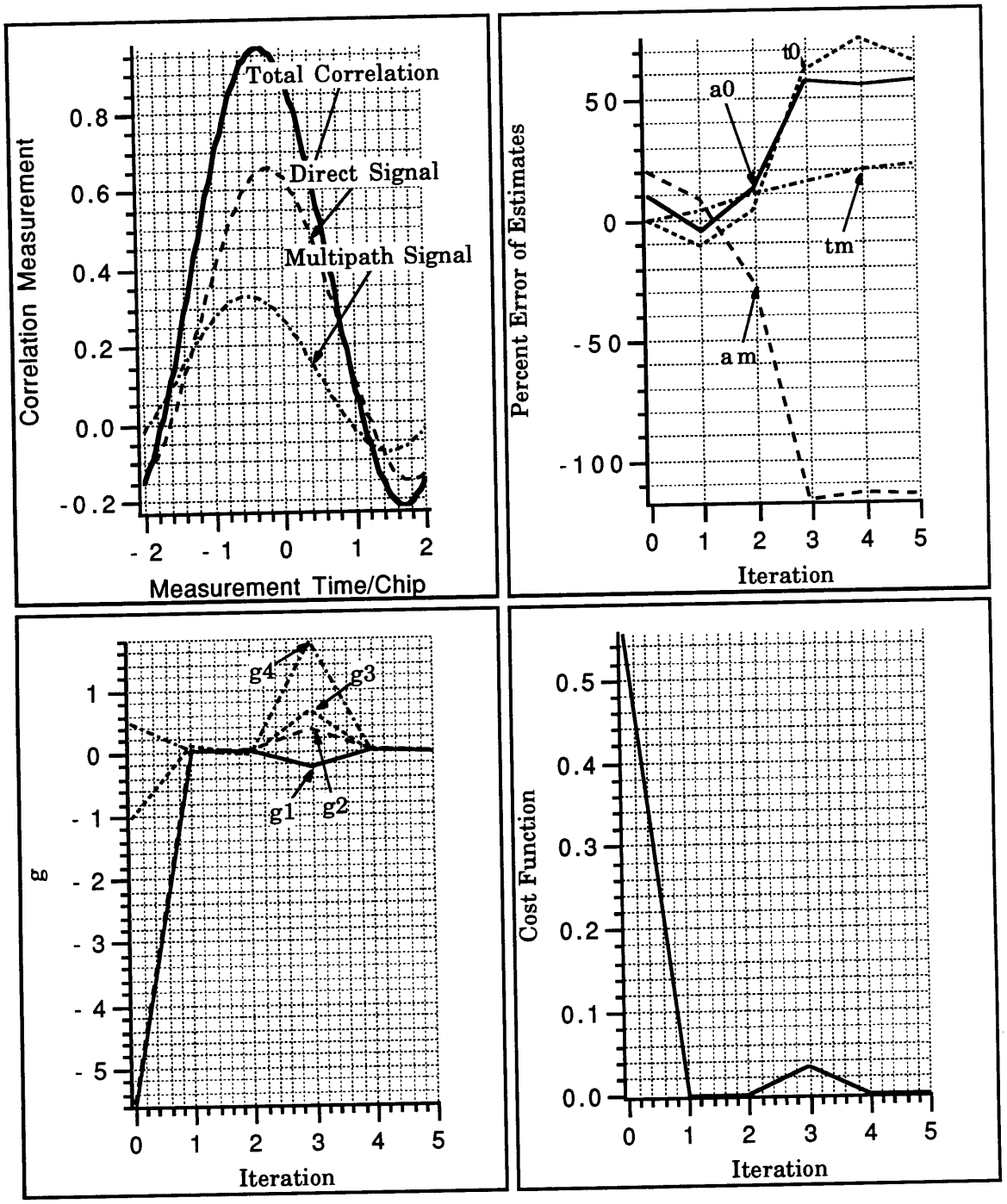

Figure 3.6.1-1 Results from simulation 1

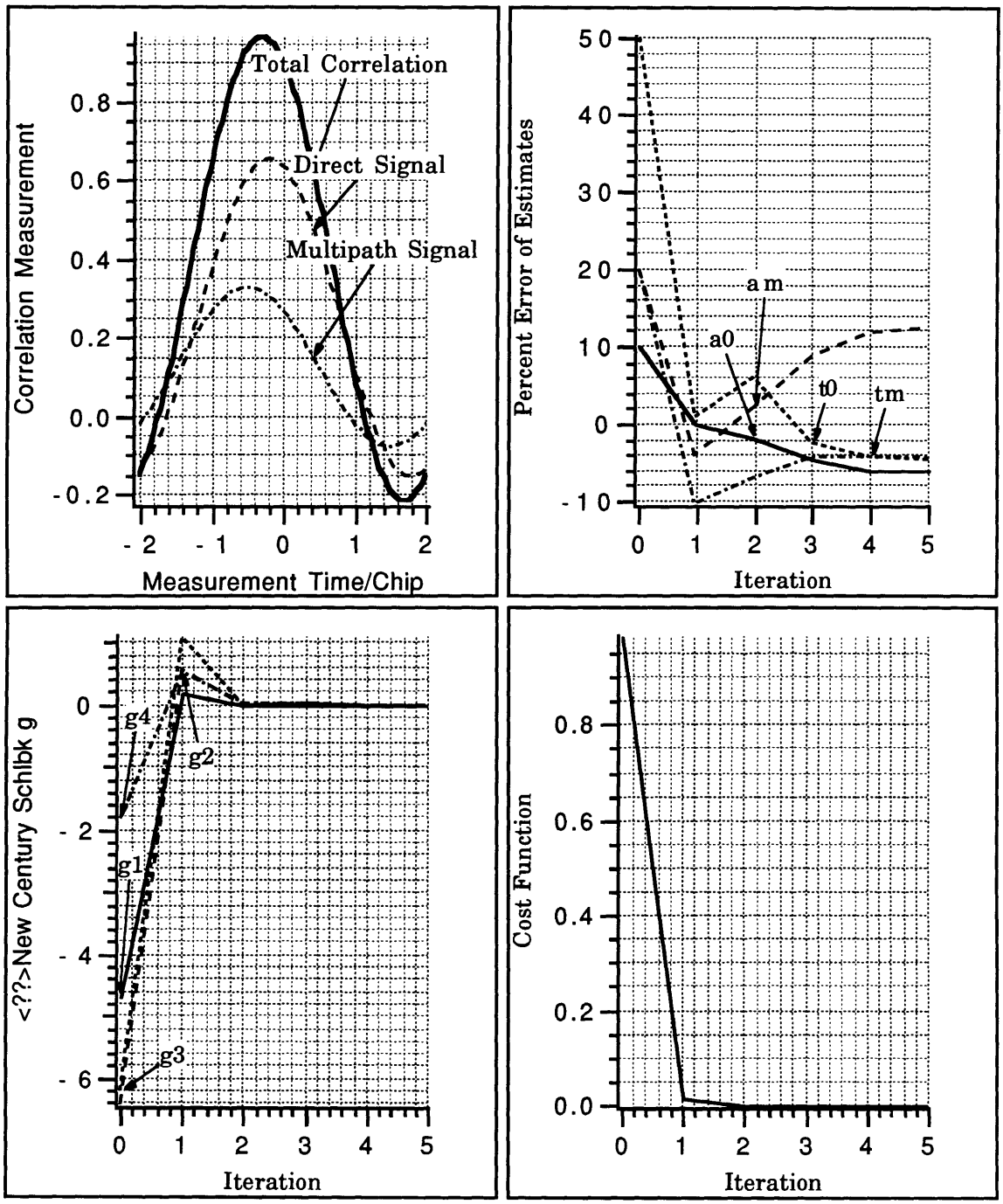

Figure 3.6.1-2 Results from simulation 2

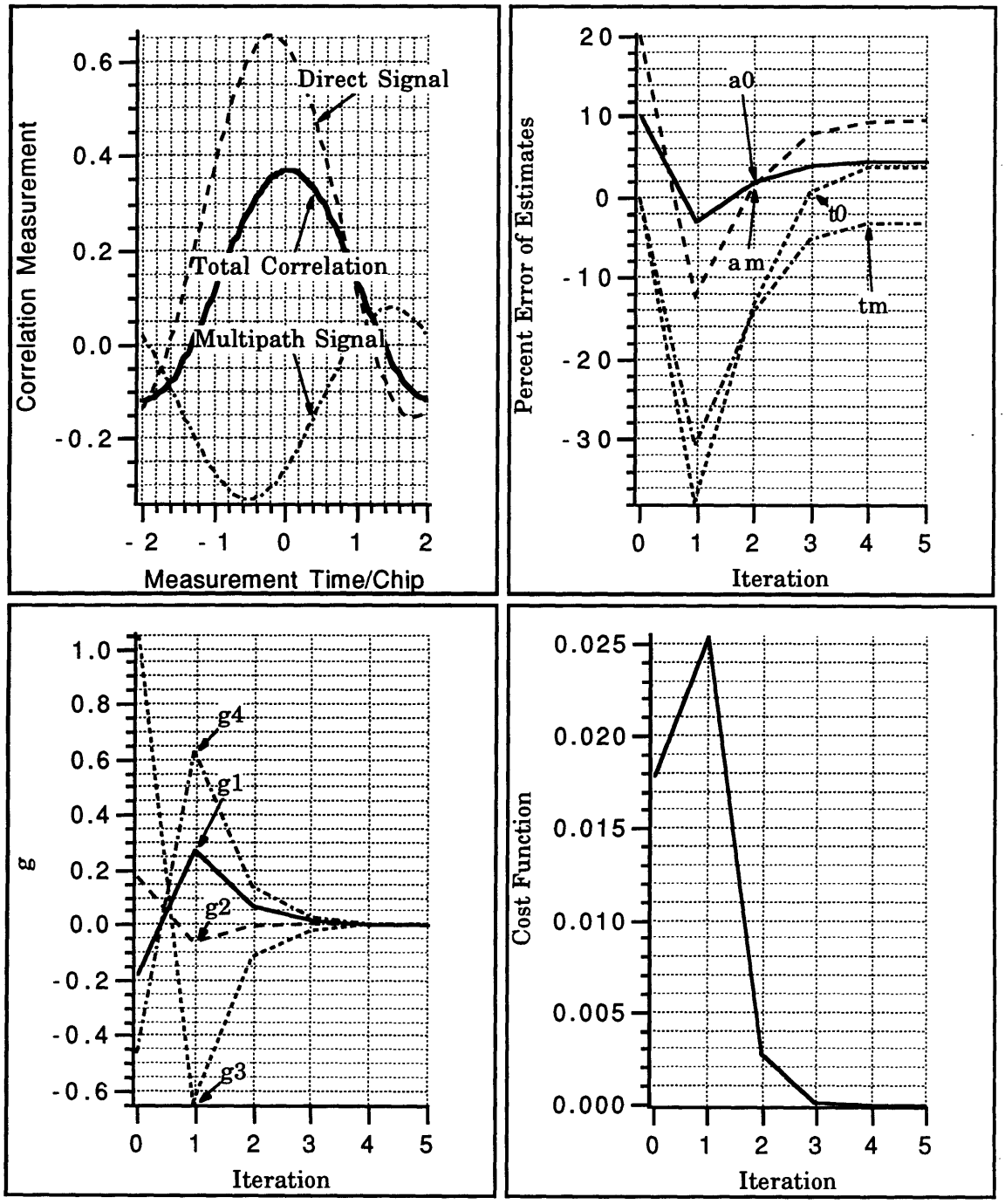

Figure 3.6.1-3 Results from simulation 3

In order to understand the behavior of the results better, a mesh plot of the cost function was made for case 1. The parameter values displayed in the plot were taken from a subset of the actual iteration. In order to display the plot the time delays were fixed at 0.22 and 0.5 chips and the signal amplitudes varied from 0.84-1.2 and 0.44-0.64 respectively. The resulting cost function is shown in Figure 3.6.1-4. The cost function plot shows a steep transition zone from 0.003 to 1.8. The optimization seems to have run into difficulty because of the flat to steep transition. The resulting iteration steps after this point were led into a region away from the minimum.

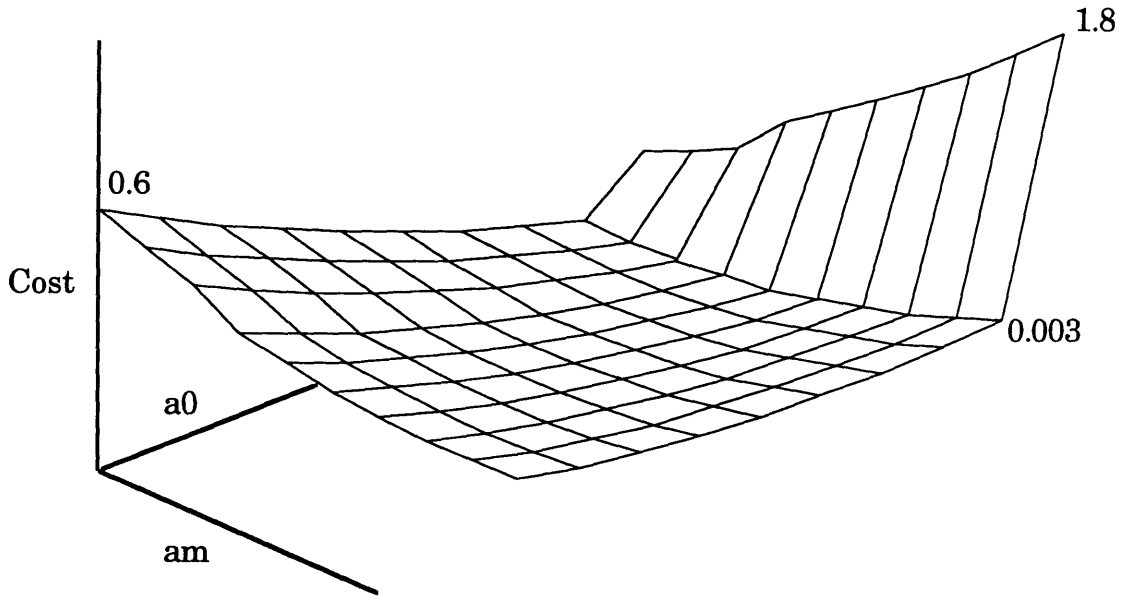

Figure 3.6.1-4 Cost function mesh plot

3.7 Summary

The results of the trial optimization were encouraging. The procedure improved the estimates of the signal parameters in one of the cases. The fact that the procedure worked at all considering the problem

was formulated using only the relatively flat correlation signal fundamental offered hope that a formulation using the actual correlation signals would work. The test case results did show that the cost function was flat in large regions and not very sensitive to parameter variations.

Multipath mitigation by optimization is possible with the appropriate procedure. The optimization must be able to handle piece wise linear functions that are not continuously differentiable and be relatively insensitive to initial conditions. This leads to the use of the Genetic and Interval optimization procedures. In these cases the optimizations are done with the actual correlation and cost functions and is shown in Chapters 4 and 5.

Chapter 4

The Genetic Algorithm

4.1 Introduction

The Genetic algorithm is a variant of traditional random search algorithms that emulate certain features of natural selection and genetics [4], [9]. The Genetic algorithm is not a random search based on chance. The difference between a random search and the Genetic algorithm is best described by van Deventer [9]: "Genetic algorithms use random choice to guide a directed search". These properties allow the genetic algorithm to search through complex parameter spaces thoroughly without relying on auxiliary information such as continuity and derivatives.

Genetic algorithms are rooted in genetics and computer science. The terminology associated with the Genetic algorithm is a mix of these fields of study. The algorithm works with string structures to form an organized stochastic search procedure that promotes survival of the fittest among the strings. The algorithm is unique in that it works with a coding of the parameter set and not the parameters themselves. Genetic algorithms use only cost function or objective function information; derivatives are not used. The rules that make up the procedure are based on stochastic operators. The genetic algorithm searches through a population of possible solutions and not single point solutions.

A brief description of the Genetic algorithm is presented in this chapter along with parameter estimation results. Under ideal starting conditions the algorithm is able to jointly estimate the direct and multipath

signal delays to better than 0.02 chips under conditions where a conventional receiver error would exceed 0.2 chips. A more complete description of the algorithm along with the mathematical foundations that make it up are given in Appendix B. More detailed information on the procedure that includes several examples is presented in both [4] and [9].

4.2 Reproduction, Crossover, and Mutation

Conceptually the basic Genetic algorithm is very simple and is really only based on three operations:

1) reproduction

2) crossover

3) mutation

These three operators define how the string structures are copied and swapped to form a search. The string structures are a coding of the parameters and together are called a genotype. The decoded values associated with a particular genotype is a phenotype. The individual coded parameters that make up a genotype are called chromosomes. Each chromosome is made up of characters called genes. The algorithm manipulates or evolves an initial population of strings into new strings that improve over successive generations. Reproduction, crossover, and mutation are the operators that make up this process.

4.2.1 Reproduction

Individual genotypes are selected and copied into a mating pool according the their fitness. This process is reproduction. The fitness of a particular genotype is a measure of the parameters that make up the genotype. Fitness is the numerical value of a non-negative objective function given a genotype. In the case of multipath mitigation the fitness

function is closely related to the cost function that was defined by equation 3.4-2. Genotypes with larger fitness values are more likely to be selected to the mating pool. The high fitness genotypes are the parents of successive generations that evolve with higher and higher fitness.

Genotypes are selected to the mating pool in two steps. Each genotype is assigned a relative fitness value to measure its strength. Based on this fitness a string is chosen or sampled to enter the mating pool. The mating pool is formed based on the expected value of a string which is the number of times a string is expected to reproduce in the next generation. It is defined by:

$$Ex_i = \frac{F_i}{F_{avg}} \qquad (4.2.1\text{-}1)$$

where: F_i = the fitness of an individual string

F_{avg} = the average fitness of the current population

Strings are copied into the mating pool an average of Ex_i times. For example if a particular string had an expected value of 3.75 then it is copied into the mating pool an average of 3.75 times.

Sampling is the process in which the expected values are used to create the mating pool. Ideally each string is copied exactly Ex_i times. Unfortunately fractional string copies cannot be realized, so a sampling algorithm is used. Many sampling algorithms [4], [10] exist, one algorithm is the remainder stochastic independent sampling algorithm (RSIS). RSIS is efficient and has minimum spread [4]. The spread is a measure of the consistency of the sampling algorithm and is the range of copies an individual can achieve in the mating pool. Another advantage of RSIS is that it has less bias than other methods. A string has zero bias when its expected value equals the actual sampling of the individual.

In this algorithm each member of the population contributes into the pool exactly the integer value of its expected value. Each member's

fractional expected value is then used as a probability for further selection. The population is evaluated again and members are selected stochastically based on their fractional expected value. Usually the mating pool is filled on this second pass. Additional passes are done as necessary.

4.2.2 Crossover and Mutation

A new generation of strings are produced by crossover and mutation after a mating pool is formed. Two strings are selected at random and crossed to form a new string. Figure 4.2.2-1 diagrams the mating pool, crossover, and the beginning of a new generation. Each string is made up of a coding of parameters. In the figure a simple binary coding is diagrammed. After two strings are selected from the pool, parts of them corresponding to the encoded parameters are swapped to form two new strings. The two new strings then represent different combinations of the encoded values of the parameters. The parts of the strings that get swapped are chosen at random with probability p_c. Each new string can be changed further by mutation. Individual sites in strings are changed with probability p_m. Usually the probability of mutation, p_m, is low.

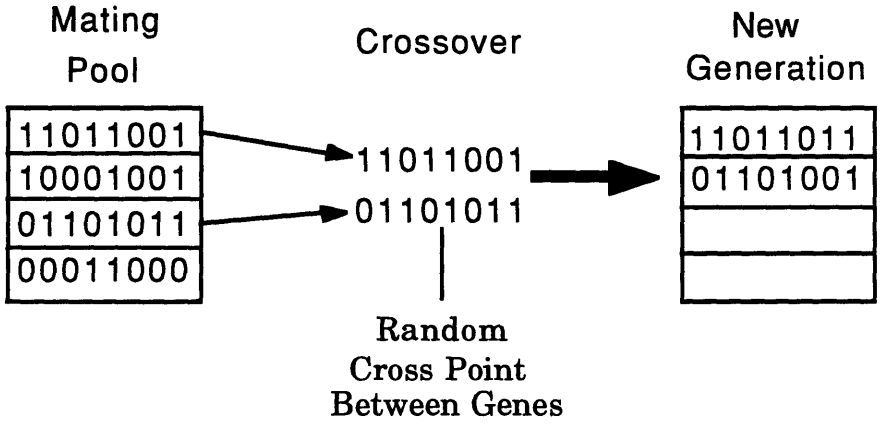

Figure 4.2.2-1 Crossover

Crossover is the process that exchanges information between strings and facilitates an improved parameter evolution. Figure 4.2.2-1 detailed a simple one point cross over scheme. Other schemes exist [4], [9] that result in greater information exchange between strings. However care must be exercised in choosing a scheme. Too many crossover sites will lead to a random shuffle of strings.

The mutation operator changes string characters a small percentage of the time. The role of mutation is to guard against a loss of useful information in the reproduction and crossover processes. Reproduction and crossover have stochastic operators as part of their make up. Occasionally valuable genetic information is lost due to the probabilistic nature of the processes. Mutation places a damper on these effects by changing individual string characters after mating.

4.3 Parameter Coding

One of the fundamental underpinnings of the Genetic algorithm is the parameter coding. Coding is the process of taking the search parameters and mapping them into a structure. For example, in the case of signal parameter estimation the terms a_0, a_m, τ_0, τ_m are coded to form a binary string. Figure 4.3-1 depicts the basic string structure. Each parameter is given a specified number of characters that form a chromosome. The chromosomes are then concatenated to form the string structure (genotype). The parameters in Figure 4.3-1 are made up of four 4 bit chromosomes that together form a 16 bit genotype.

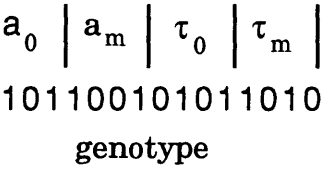

Figure 4.3-1 String Structure

The basic underlying principal when choosing a coding as stated by Goldberg [4] is : " The user should select the smallest possible alphabet that permits a natural expression of the problem". The maximum amount of information per bit of the coding is desirable.

The phenotype or the real values of the parameters are obtained by a mapping. A simple way to map the variables from real values to the binary code is a linear transformation. Each parameter is given a maximum, U_i^{max}, and minimum, U_i^{min}, bound prior to optimization. In the case above, 4 bit chromosomes were chosen. The direct signal amplitude a_0 would equal U_i^{min} when the four bit portion of the genotype associated with a_0 equaled 0000. The amplitude would equal U_i^{max} when the bits are set to 1111. Intermediate values are assigned linearly in between the minimum and maximum values.

4.4 The Fitness Function and Fitness Scaling

The Genetic algorithm works with the fitness of the genotypes. The algorithm maximizes a non-negative fitness function. Therefore, the cost function defined by equation 3.4-2 needs to be rewritten in fitness function form. The translation from cost to fitness that is used is:

$$F_i = \begin{cases} J_{ceil} - J_i & \text{when } J_i < J_{ceil} \\ 0 & \text{otherwise} \end{cases} \qquad (4.4\text{-}1)$$

where: J_{ceil} = a constant specified prior to optimization

J_i = the cost function defined by equation 3.4-2

i = the i^{th} member of the population

Therefore the lower the cost, that is the goal, the higher the fitness.

The Genetic algorithm starts from an initial population of strings generated at random. The fitness, F_i, of each member of the initial population is calculated, and based on this, reproduction and crossover are done to form a new population. This procedure is then repeated with the new generation. It is common for the initial population to have a few highly fit individuals that dominate the population. Subsequent reproduction and crossover to form a new generation will also be dominated by these fit members. Early domination by a few fit members will lead to premature convergence. An opposite affect can happen as well. In a population where most of the members have a high fitness and are about the same for all members of the population, a survival of the average can occur. In this case a random walk among the mediocre is likely. These two situations are handled by fitness scaling.

Fitness scaling is performed to ensure that no individuals dominate the process early on and to enhance competition among individuals in later generations. The scaled fitness is a linear function of the raw fitness and is:

$$F_i^s = cF_i + d \tag{4.4-2}$$

The constants c and d are selected during each generation so that the average of the scaled fitness values equals the average of the raw fitness values. The expected value of an average population member remains the same after scaling. One last scaling operation is done. The maximum expected offspring that the fittest member will contribute to new generations is set prior to optimization. The new maximum fitness is re-scaled to be:

$$F_{max}^s = C_m F_{avg} \tag{4.4-3}$$

where: F_{avg} = the average raw fitness of the population

C_m = the expected number of offspring of the fitness member

In some instances the scaling may force the fitness of a member to be negative. To satisfy the constraint that fitness functions be non-negative, the value of C_m is adjusted such that the minimum scaled fitness equals zero.

An example of the scaling procedure, taken from [4], is shown in Figure 4.4-1. In this example the multiplication factor C_m is set to a value of 2. The figure shows the scaled fitness as a function of the raw fitness. The scaled and raw average fitness are the same and the maximum scaled fitness equals twice the average fitness.

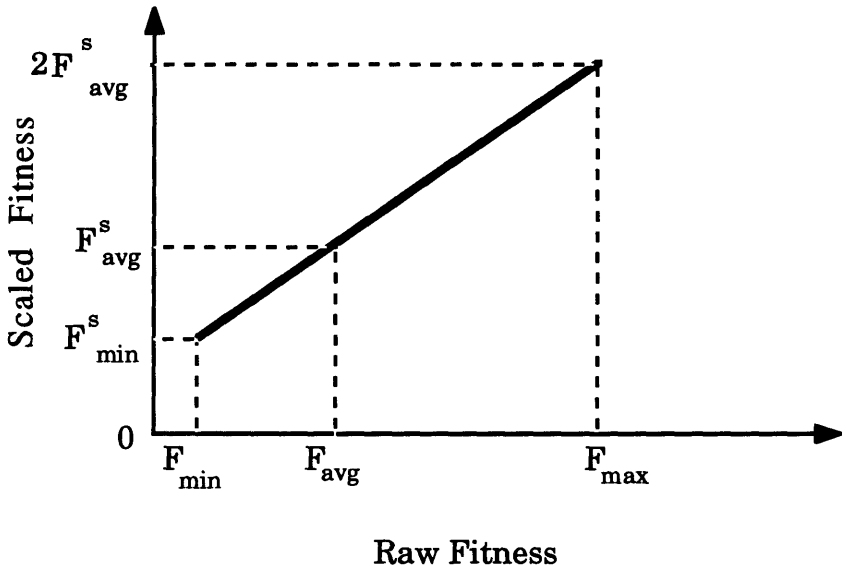

Figure 4.4-1 Fitness Scaling

4.5 Signal Parameter Estimation

Joint estimation of the signal parameters a_0, a_m, τ_0, τ_m was reduced to an optimization problem. The Genetic algorithm was used for optimization and signal parameter estimates were obtained. Results from two cases are

presented in this section. Although a number of simulations were run, these two cases provide a representative insight into the behavior of the optimization as used for parameter estimation. The two cases simulated were both done with the same signal truth model. Forty measurements of the correlation function were used. The direct and multipath signal amplitudes were equal to 1.0 and 0.5 respectively and the direct and multipath delays were equal to 0.2 and 0.5 chips. Different initial starting conditions were assumed for the second simulation.

4.5.1 Problem Formulation

The optimization procedure uses measurements of the disturbed correlation function to provide estimates of the four signal parameters, a_0, a_m, τ_0, τ_m. These four parameters were coded into a forty bit genotype made up of four chromosomes. Each chromosome represents a signal parameter. A simple binary coding scheme was used. Figure 4.5.1-1 contains an example genotype break down.

Chromosomes

1101101101 1101100011 0001011100 0001100101

a_0 a_m τ_0 τ_m

1101101101110110001100010111000001100101

Genotype

Figure 4.5.1-1 Example Genotype

A population of eighty genotypes was used. The initial population was chosen at random with each bit having an equally likely chance of being a 1 or 0. The range of the parameters was specified prior to optimization and is shown in Table 4.5.1-1. The selection of the parameter ranges plays a significant role in the performance of the algorithm. The parameter

maximum and minimum values were used to map the genotype in to real values as discussed in section 4.3.

Parameter	a_0	a_m	τ_0	τ_m
Case 1 Max	1	0.5	0.5	1.0
Case 1 Min	0.5	0.0	0.0	0.5
Case 2 Max	1.0	1.0	1.0	1.0
Case 2 Min	-1.0	-1.0	0.0	0.0

Table 4.5.1-1 Parameter Ranges

After the initialization process, the algorithm was run in an iterative process consisting of reproduction, crossover, and mutation to form new generations of solutions. Fitness scaling was used with a multiplication factor of 2. A crossover probability of 0.7 and a mutation probability of 0.02 were used. These probabilities are typical values [4], [9] used in Genetic optimization problems.

Choice of these control parameters might have been important in terms of algorithm performance. Therefore a number of simulations were carried out to investigate the influence of population sizes, genotype lengths, scaling factors, crossover probabilities, and mutation probabilities. The optimization results in terms of accuracy and rate of convergence were not sensitive to these parameters.

4.5.2 Example Results

The two cases presented were both run with the same signal model parameter values. The only difference between simulations was the initial parameter ranges, shown in Table 4.5.1-1. During the simulation a population of 80 members was used and evolved over forty generations. The signal estimates at any generation are the ones with the highest fitness in that particular generation.

The first simulation was run with tight initial parameter ranges. The maximum and minimum amplitudes for both the direct and multipath signal did not overlap. The delay parameters were given non-overlapping conditions as well. These conditions placed constraints on the regions that the optimization algorithm needed to search. Figure 4.5.2-1 contains the results from the first simulation. The figure is made up of four subplots. The first subplot displays the error in the signal amplitude estimates at each generation. The second subplot contains the error in the signal delay estimates. The error is the difference between the true parameter and the estimate. The third subplot displays the maximum fitness as a function of the generations. The maximum fitness is the fitness of the best member of a generation. The last subplot shows the average fitness. The average fitness is the average of each members fitness in a particular generation.

The results show that after the fifth generation the signal estimates track the true signal parameters. The errors are small compared to the signal parameters. Table 4.5.2-1 contains summary of the mean and standard deviation of the error in the parameter estimates. The table also contains the maximum error caused by multipath for a conventional receiver. A conventional receiver experiences a maximum pseudorange error of 0.2 chips (C/A) for the simulated level of multipath. The mean error in the direct signal delay estimate from optimization is 0.004 chips (C/A) with a standard deviation of 0.02 chips.

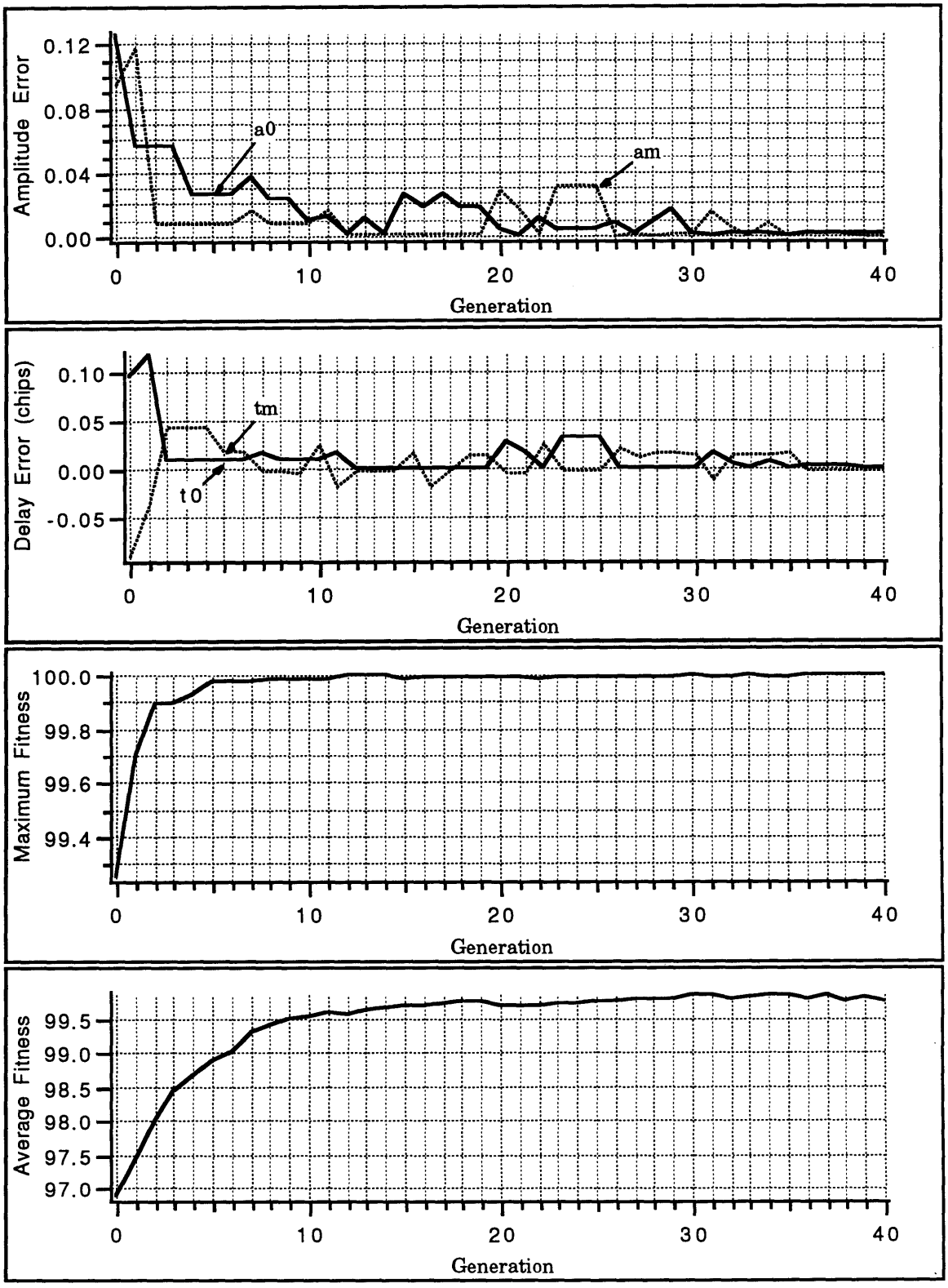

Figure 4.5.2-1 Non-overlapping Initial Parameters

The plots of the fitness behave as expected. The maximum fitness increases rapidly and remains stable in later generations. The initial population of solutions was very good. The average fitness increases with each new generation. The solutions became better as the generations evolved.

Parameter	Mean Error 1	Mean Error 2	Standard Deviation 1	Standard Deviation 2	Maximum DLL Error
a_0	0.016	0.21	0.023	0.16	-
a_m	0.012	-0.18	0.023	0.18	-
τ_0	0.0046	-0.022	0.022	0.045	0.2
τ_m	-0.027	0.096	0.024	0.097	-

Table 4.5.2-1 Summary of Results

The next simulation was run to mimic more realistic starting conditions. The ranges on the parameters were opened up and overlapped. The results shown in Figure 4.5.2-2 were not nearly as good in this case. The amplitude estimates vary and do not converge to reasonable values. The amplitude estimates display a survival of the mediocre. The signal delay estimates are slightly worse than the first case. A summary of these results is in Table 4.5.2-1. The maximum and average fitness do increase and remain flat.

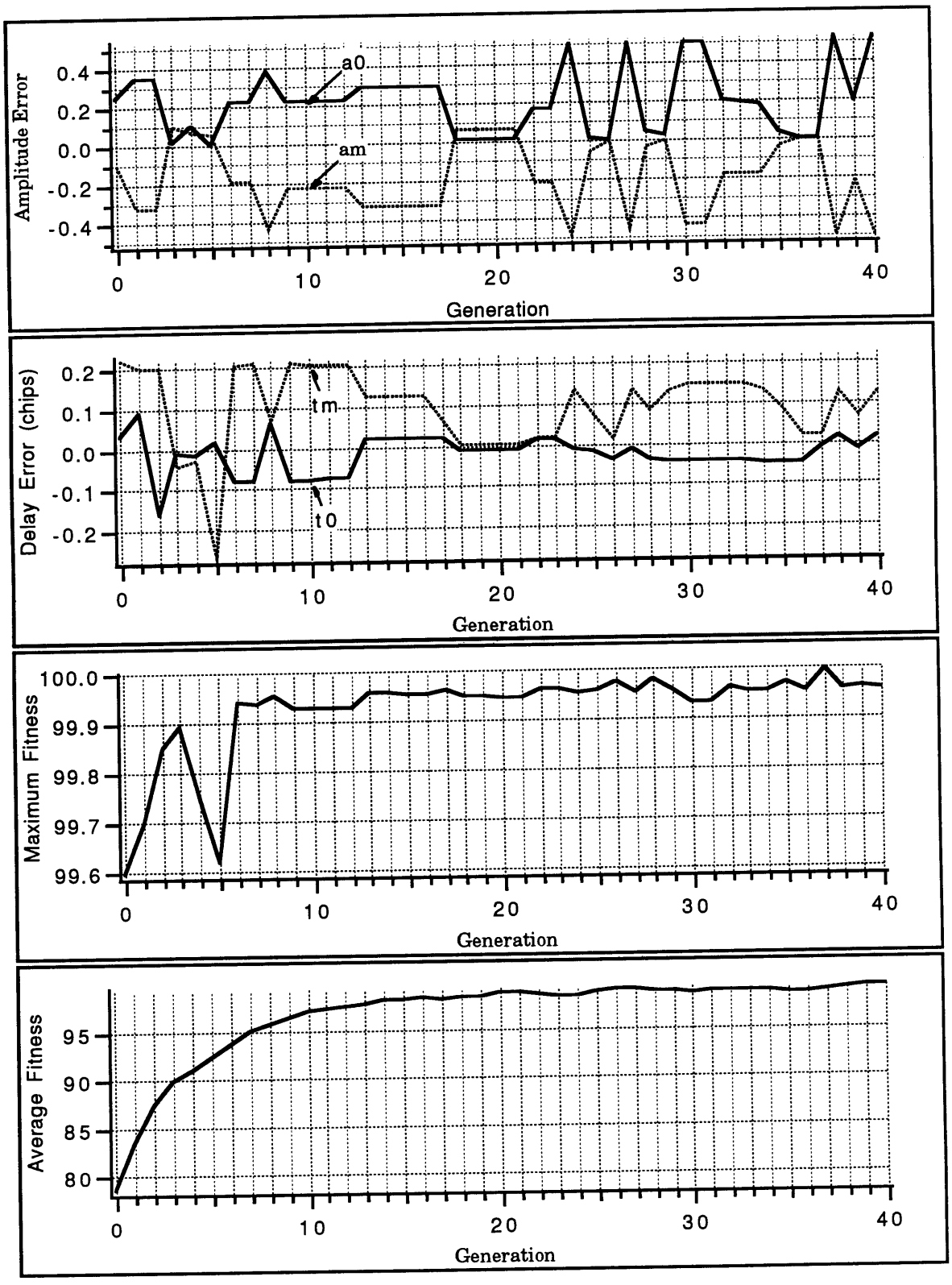

Figure 4.5.2-3 Overlapping Initial Parameters

4.6 Summary

The results summarized above are not unique. Several other cases were simulated with overlapping parameter ranges. In most cases at least one of the signal parameter estimates did not converge. In some instances the direct signal parameters more closely tracked the multipath parameters. Quadratic penalty weights were introduced into the cost function to eliminate this effect. However, the penalty weighted simulations still behaved in an erratic manner.

The non-convergence problem is most likely caused by the relative flatness of the underlying cost function. Each member of the population looked as good as the next. Large changes in the parameter estimates only cause small changes in the cost function. This facet of the problem is demonstrated by the behavior of the fitness plots. The average and maximum fitness plots look similar in each simulation.

The non-convergence problem only manifested itself when the parameter ranges overlapped. Simulations with non-overlapping ranges worked well as shown in the first simulation. The tighter and non-overlapping ranges limit the search space and allow the algorithm to perform a more directed search. In this case, survival of the fittest occurs and not survival of the mediocre. Given reasonable initial parameter ranges the Genetic algorithm works well. Development of an algorithm to provide these initial ranges is discussed in the next chapter.

Chapter 5

Interval Methods

5.1 Introduction

The Genetic algorithm was able to estimate the signal parameters when the initial parameter ranges were small and did not overlap. This chapter presents an algorithm that is capable of supplying tight non-overlapping initial parameter ranges to be used as initial conditions for the Genetic algorithm. An optimization procedure based on Interval mathematics is developed and used to provide the initial parameter ranges or intervals. Interval optimization in the absence of noise can jointly estimate the direct and multipath signal delays within 0.01 chips. In fact, the procedure was able to estimate the time delays to the same accuracy even when the relative delay between the direct and multipath signals was 0.05 chips. The presence of noise which is discussed in the next chapter reduces the effectiveness of the Interval procedure. However, Interval methods do reduce the initial parameter ranges to levels that Genetic optimization can work with effectively in the presence of signal noise.

Interval mathematics originates from the study of computational errors due to rounding and uncertain data. The book "Interval Analysis" by Moore [6] developed a theory to analyze errors in computed data. Interval mathematics replaces real or complex numbers and arithmetic, with interval numbers and interval arithmetic. A powerful optimization procedure capable of solving the global optimization problem was developed by Hansen [5] using Interval mathematics. The procedure is attractive

because it is not limited to functions that have continuous derivatives. Joint estimation of the signal parameters in the presence of multipath is possible with Interval optimization

5.2 Interval Mathematics

An interval number is the set of numbers between and including two end points. The terms interval number and interval are used interchangeably for convenience. A standard notation for an interval is:

$$X = [a,b] \tag{5.2-1}$$

where: a = the lower bound

b = the upper bound

A number x is an element of the set X, $x \in X$, for any x such that $a \leq x \leq b$.

Any real non-interval number can be represented by an interval. A real number x is equivalent to $X = [x,x]$. An interval X is positive when $a \geq 0$ and negative when $b \leq 0$. The interval X = [a,b] is less than the interval Y = [c,d] only when b < c. The important notion of the interval is that it contains the continuum of points between and including the end points. This definition is exploited during optimization.

A theory governing Interval mathematics exists and for example contains the operators of addition, subtraction, multiplication, and division. Functions of intervals are defined and are evaluated using the rules of Interval mathematics. Consider an example with two real numbers x and y and two intervals X and Y. If standard mathematical operations of x and y are denoted as x op y then the interval operation is:

$$X \text{ op } Y = \{ x \text{ op } y : x \in X, \ y \in Y \} \tag{5.2-2}$$

The interval X op Y resulting from the operation is every possible number that can be formed as x op y for each $x \in X$ and $y \in Y$. This definition is used to form rules for generating the endpoints of operations between two intervals X = [a,b] and Y = [c,d]. For example, the addition rule is:

$$X + Y = [a + c, b + d] \tag{5.2.-3}$$

A complete set of rules is given in Appendix C. The rules are more complicated for different operators. Some care is needed when evaluating operands to ensure that the correct interval bounds are obtained. A good example of this is, the interval $Y = X^2$. If the interval X = [-1,2] then Y = [0,4] and not [1,4]. The evaluation of the interval $[-1,2]^2$, does not equal $[-1,2][-1,2] = [1,4]$. Special rules are used to obtain the correct endpoints that are [0,4]. The rules contained in Appendix C are developed to ensure the proper endpoints are obtained.

5.3 Interval Optimization

The optimization procedure works with interval values of the search parameters. The interval signal parameters $\hat{a}_0, \hat{a}_m, \hat{\tau}_0, \hat{\tau}_m$ are used to evaluate the interval value of the cost function defined by equation 3.4.2. The algorithm minimizes the cost function by iterating on the parameter intervals. The parameter intervals represent a search space or box. As the optimization is carried out the parameter box is reduced in a region that always contains the minimum of the cost function, subsequent iterations keep on reducing this box until a pre-selected limit is reached [5]. The resulting parameter intervals are as large as the preset limit and are guaranteed to contain the cost function minimum.

5.3.1 The Interval Cost Function

In order to use the algorithm the interval cost function needs to defined. The cost function from Chapter 3 is:

$$J = \sum_{i=1}^{n} \left\{ \hat{a}_0 \Phi(\tau_i + \hat{\tau}_0) + \hat{a}_m \Phi(\tau_i + \hat{\tau}_m) - a_0 \Phi(\tau_i + \tau_0) - a_m \Phi(\tau_i + \tau_m) \right\}^2 \qquad (5.3.1\text{-}1)$$

The signal parameter estimates $\hat{a}_0, \hat{a}_m, \hat{\tau}_0, \hat{\tau}_m$ are interval parameters. Therefore the operations defined by the cost function are done with interval mathematics as defined in Appendix C. The sampled correlation function portion of the cost function is:

$$\Phi(\tau_i) = a_0 \Phi(\tau_i + \tau_0) + a_m \Phi(\tau_i + \tau_m) \qquad (5.3.1\text{-}2)$$

The sampled correlation function is not an interval function and is obtained as outputs from the bank of correlators described in Chapter 3.

The total cost function is an interval function. Evaluation of the function is straight forward using interval mathematics with the exception of the PRN code autocorrelation function components. The correlation function is:

$$\begin{aligned} \Phi(\tau_i) &= 1 - \hat{\tau}_d - \tau_i \qquad \text{when } -\hat{\tau}_d < \tau_i < 1 - \hat{\tau}_d \\ \Phi(\tau_i) &= 1 + \hat{\tau}_d + \tau_i \qquad \text{when } -\hat{\tau}_d - 1 < \tau_i < -\hat{\tau}_d \\ \Phi(\tau_i) &= 0 \qquad \text{elsewhere} \end{aligned} \qquad (5.3.1\text{-}3)$$

where: τ_i = the correlation sampling time

$\hat{\tau}_d$ = the interval estimate of either $\hat{\tau}_0$ or $\hat{\tau}_m$

The interval parameter $\hat{\tau}_d$ represents the range of possible delay estimates including the end points corresponding to the given interval parameter $\hat{\tau}_0$ or $\hat{\tau}_m$. Thus the autocorrelation function at any sampling time can take values in the range of 0 to 1. Figure 5.3.1-1 contains an example function with unit amplitude, measurement time of -0.5 chips and a interval signal delay, $\hat{\tau}_0$ = [0, 0.75] chips. The figure contains plots of three correlation functions. The rightmost function represents the function when $\hat{\tau}_0 = 0$. The left most function is the case where $\hat{\tau}_0 = 0.75$. The figure also contains a plot for an intermediate value of $\hat{\tau}_0$. Over the range of delays 0 - 0.75 chips, the peak of the autocorrelation function shifts left 0.75 chips. The peak of the function passes the measurement point of -0.5 chips. The function takes values ranging from 0.5 to 1 at the measurement time -0.5 chips. Therefore the correct range of the interval function is [0.5, 1] and not [0.5, 0.75] which is the result of evaluating equation 5.3.1-3 at the endpoints of the interval $\hat{\tau}_0$ = [0,0.75].

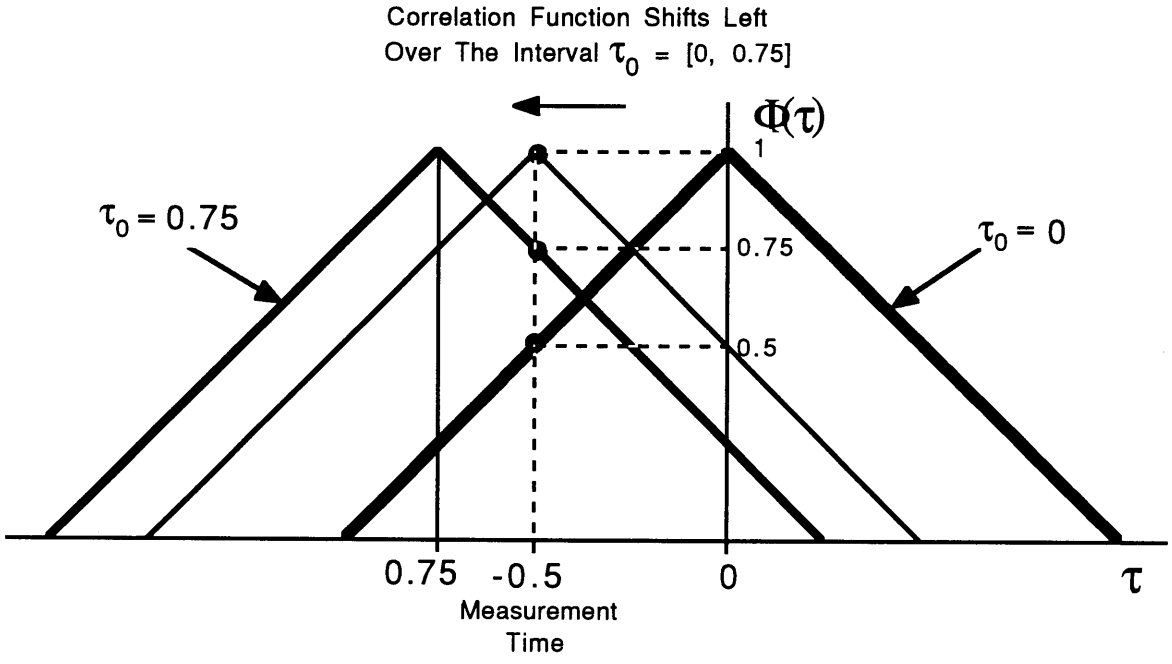

Figure 5.3.1-1 Interval Evaluation of the PRN Autocorrelation Function

The lower and upper bounds of the cost function are calculated for any given set of interval signal parameters using interval rules that ensure proper end point evaluation.

5.3.2 The Optimization Procedure

The procedure starts with a number of initializations. Initial parameter ranges are specified that bound search region. Table 5.3.2-1 contains starting intervals for the joint estimation problem, the interval width desired at the end of optimization, and the interval width of the cost function. The parameter intervals are wider than those used for the genetic algorithm and are realistic in terms of what a receiver of this type would use. During execution, the interval estimates of the signal parameters are reduced to the specified limit and converge around the true values. The cost function interval is reduced and converges to zero.

Interval	$\hat{a}_0$	$\hat{a}_m$	$\hat{\tau}_0$	$\hat{\tau}_m$
Maximum	1	1	1	1
Minimum	-1	-1	-1	-1

Table 5.3.2-1 Initial Parameter Intervals

Conceptually the algorithm is simple. The initial search space or box is divided up into regions. A simple approach is to cut the parameter ranges in half. In the signal parameter case this results in sixteen sub boxes. In actuality, the number of boxes is reduced by the constraint that the direct signal delay is less than the multipath delay. Therefore, some of the sub boxes can be discarded. The cost function is evaluated over each sub box. This results in interval values of the cost function for each sub box. The intervals of the cost function for any sub box give the lower and upper limit that the cost function takes for any combination of parameters within

and on the boundaries of the sub box. The lower limit of each sub box cost interval are compared and the sub box with the lowest limit is chosen for subdivision and further processing.

The basic procedure eliminates vast regions of the search space by looking at the boundaries of the sub regions. The interval evaluation in the absence of noise produces **the minimum and maximum** bounds on the cost function in the sub box **without evaluating every possible combination of parameters** in the sub box [5]. The sub box with the lowest limit is chosen for subdivision because the goal of the procedure is to find the minimum of the cost function. The process of subsequent subdivisions reduces and keeps reducing the parameter intervals to the preset limit that contain the function minimum.

In order to appreciate the Interval procedure consider a simple two dimensional example used to minimize the function:

$$f(x,y) = -e^{-(x^2+y^2)} \qquad (5.3.2\text{-}1)$$

The function has a minimum of -1.0 located at x = 0 and y = 0. Define an initial box as X = [-1, 1] and Y = [-1, 1]. Figure 5.3.2-1 diagrams the initial box and box subdivision. For example purposes, divide the box into four sub boxes as shown in the figure. Evaluate the interval value of equation 5.3.2-1 for each sub box. Table 5.3.2-2 contains the sub boxes and lower limit of the function intervals for each sub box. The sub box with the smallest lower limit contains the region where the minimum of the function is. Three regions are immediately discarded and the procedure is started again with the chosen sub box. Subsequent divisions and selections reduce the region defined by the X and Y intervals and these converge to the location of the minimum of the function.

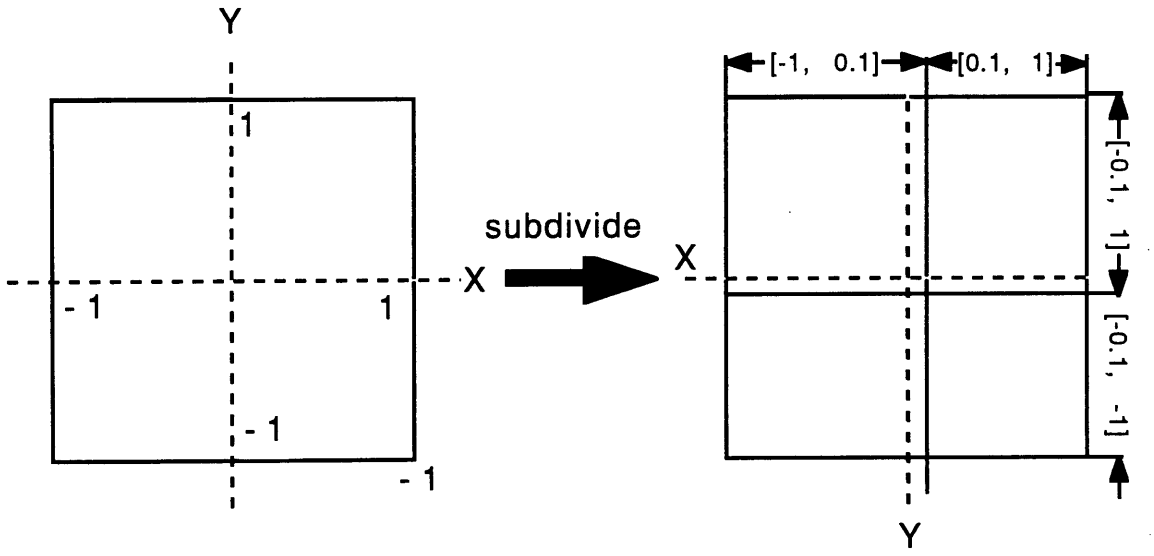

Figure 5.3.2-1 Example Search Box and Subdivision

Box	x low	x up	y low	y up	f low
1	0.1	1	-0.1	1	-0.99
2	0.1	1	-1	-0.1	-0.98
3	-1	0.1	-0.1	1	-1.00
4	-1	0.1	-1	-0.1	-0.99

Table 5.3.2-2 Sub Boxes and Function Lower Limit

5.4 Interval Optimization Results

A number of simulations were run using the Interval algorithm. Results from two cases are presented in this section. Two different truth models were simulated. The amplitudes of the direct and multipath signals were 1.0 and 0.5 for each simulation. The signal delays were different. The first simulation used a direct signal delay of 0.2 chips along with a multipath delay of 0.5 chips. The second simulation had a direct

signal delay of 0.1 chips and a multipath delay of 0.15 chips. Forty measurements from the disturbed correlation signal were used in each simulation. The algorithm was run eight iterations starting from an initial box of [-1, 1] for each of the signal parameters. The stopping criteria which is the difference between the lower and upper interval bounds was 0.01 for the parameter intervals and 0.03 for the cost function.

The first simulation was a repeat of the case described in Chapter 4. During the Genetic optimization the signal amplitude estimates displayed a random walk type of behavior. The simulation done using the Interval procedure was initialized with even wider initial conditions than the Genetic algorithm case. However, the results were dramatically better with the Interval method. Figure 5.4-1 contains results from the simulation.

The figure contains four subplots. The subplots display the intervals for each of the signal parameters, $\hat{a}_0, \hat{a}_m, \hat{\tau}_0, \hat{\tau}_m$ starting from the initial point and for each iteration. Each interval is reduced to the 0.01 level of accuracy by the end of the optimization.

The results shown in Figure 5.4-1 are dramatic. Every parameter was estimated to within 0.01. The delay estimates converged to the true simulated values of 0.2 and 0.5 chips respectively. The amplitudes converged to 1.0 and 0.5. A standard receiver exhibits a maximum pseudorange error of 0.2 chips for the C/A code under the simulated conditions. The Interval procedure was able to estimate the direct signal delay to better than 0.01 chips; a factor of 20 improvement.

The next simulation was run with the direct and multipath signal delays spaced apart by 0.05 chips. This case represents the situation of a receiver with a reflector that is very close to the receiver's antenna. Figure 5.4-2 contains the results. The figure contains the interval plots of each parameter. The results from this case are no less impressive. The algorithm was able to estimate all the signal parameters to within 0.01. The signal delay estimates converged to the simulated values of 0.1 and 0.15 chips. A typical receiver under these conditions would have a 0.05 chip error. The Interval procedure was able to jointly estimate the signal delays to better than 0.01 chips.

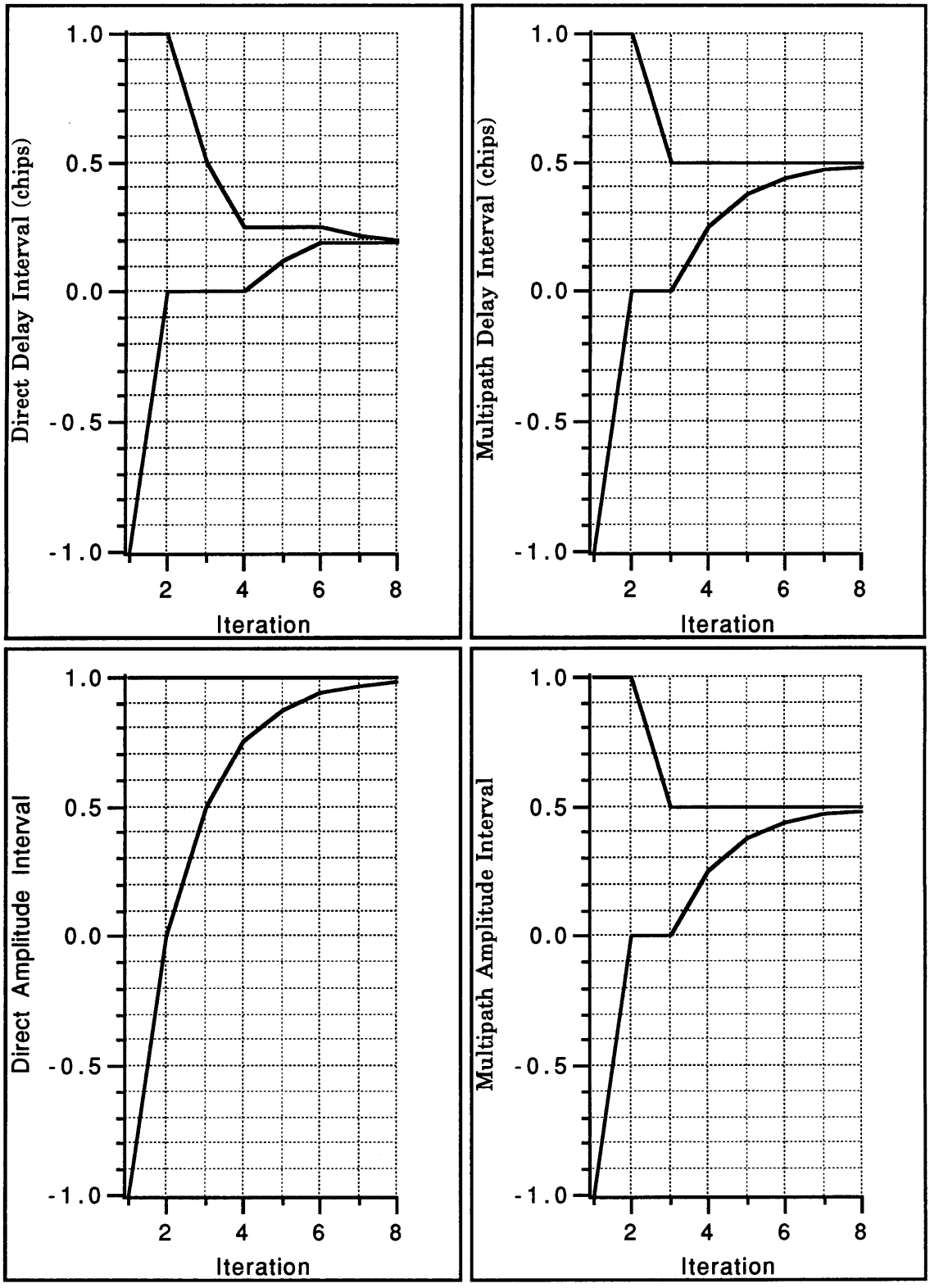

Figure 5.4-1 Interval Results, $\tau_0 = 0.2$, $\tau_m = 0.5$

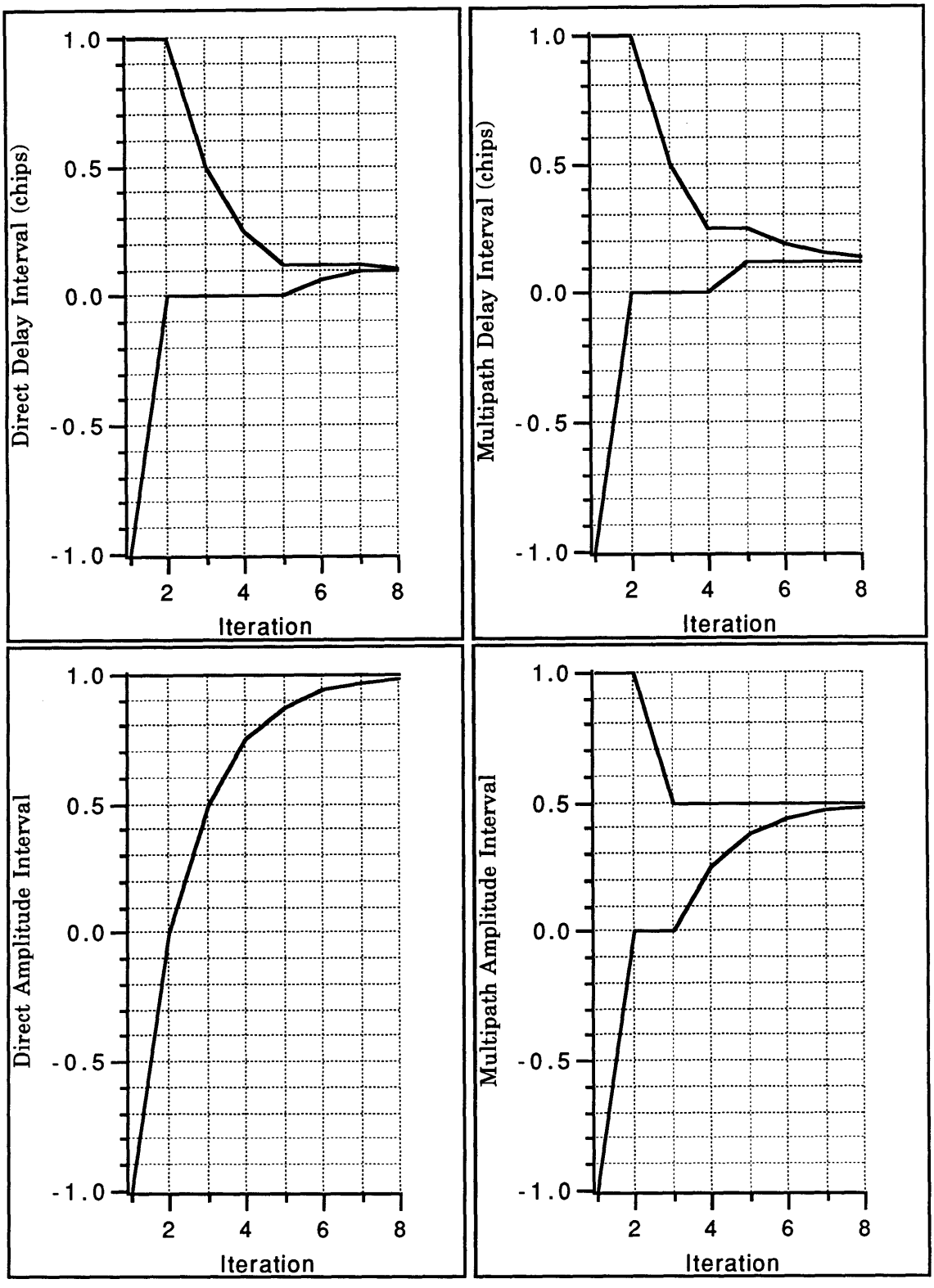

Figure 5.4-2 Interval Results, $\tau_0 = 0.1$, $\tau_m = 0.15$

5.5 Summary

The interval method was used successfully to jointly estimate the signal parameters. A number of different simulations with different combinations of direct and multipath signal parameters was run. In every case the algorithm was able to successfully estimate each parameter to the prescribed 0.01 interval width accuracy. However, these simulations were done without noise on the signal. Unfortunately the procedure broke down when noise was introduced.

The algorithm is based on examining regions of a search space, comparing lower interval bounds, and then discarding large portions of the search space. The presence of noise affects the cost function interval evaluation such that the procedure discards the proper solution region. Fortunately this behavior does not occur early on in the process.

The occurrence of this behavior was discovered only after many Monte Carlo simulations were run with signal noise. Chapter 6 develops the noise model used in the simulations and presents simulation results that highlight this behavior. However, because this behavior occurs, the Interval method was combined with the Genetic algorithm for joint estimation of the signal parameters in the presence of noise. The Interval method was used to supply tight non-overlapping parameter ranges as the starting point for the Genetic algorithm. Results from simulations run using this procedure are presented in Chapter 6.

Chapter 6

Joint Signal Parameter Estimation in the Presence of Signal Noise

6.1 Introduction

The previous chapters addressed the joint estimation problem in the absence of signal noise. The Interval optimization method was able to estimate all signal parameters to within 1% of their true values. The introduction of noise into the simulations changed these results. The Interval method can break down in the presence of noise. Therefore a combination of Interval and Genetic optimization was used to estimate the signal parameters. A number of Monte Carlo simulations was run to examine the estimation problem. The direct and multipath signal time delays were estimated to within 0.03 chips in the presence of post correlation noise with a 30 dB signal to noise ratio.

This chapter presents a noise model for the post correlation noise at the output of the bank of correlators. The post correlation noise model is taken from Bowles [1]. That work presented an algorithm that used the output of a bank of correlators to extend code loop tracking. Typical post correlation noise values for traditional receivers are given. Results from simulations using the Interval method with noise are presented. Finally the results from the combined Interval and Genetic optimization are shown.

6.2 Noise Model

The signal at the receiver is a combination of the direct signal, multipath signal, and noise. The received signal is:

$$s(t) = r(t) + n_T(t) \tag{6.2-1}$$

where: $r(t)$ = the direct and multipath signal components (eqn. 3.2-1)

$n_T(t)$ = the signal noise at the antenna

Typically the noise is treated as band limited additive white Gaussian noise [1], [7]. The noise represents thermal noise and diffuse multipath signals. The noise at the antenna is:

$$n_T(t) = N_0 n_0(t) \tag{6.2-2}$$

where: N_0 = the intensity of the noise

$n_0(t)$ = is white noise with unit intensity

The post correlation properties of the noise are determined by analyzing the noise as it passes through the front end of a receiver. The signal received at the antenna is passed through an RF filter to remove the carrier. The signal is then split into real and imaginary components and sampled. At this point the resulting sampled signal components are correlated with a bank of locally generated codes and integrated to form the sampled correlation signal.

The noise after demodulating the signal components into real and imaginary parts is treated as band limited and white [1]. The power spectrum of the noise is shown in Figure 6.2-1. The frequency range of the spectrum is the cut off frequency of the RF filter.

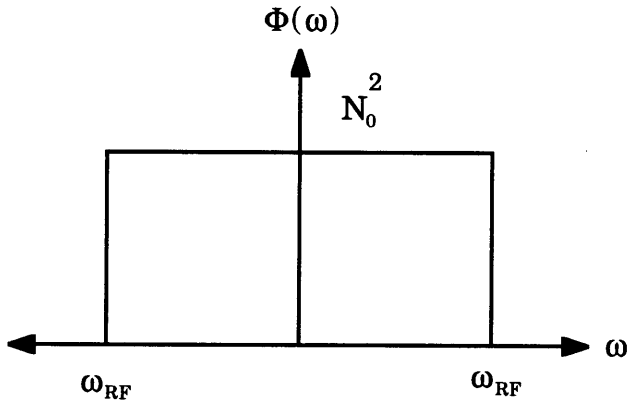

Figure 6.2-1 Signal Noise After Demodulation

The real and imaginary components are correlated with locally generated codes and integrated as described in Chapter 3. The integration time is large compared to the time period of the RF filter cut off frequency. Thus the noise is still treated as band limited and white. This assumption also results in the real and imaginary components being uncorrelated and zero mean. Figure 6.2-2 shows a bank of correlators and filters with signal noise as the input. The noise is correlated with locally generated codes and filtered.

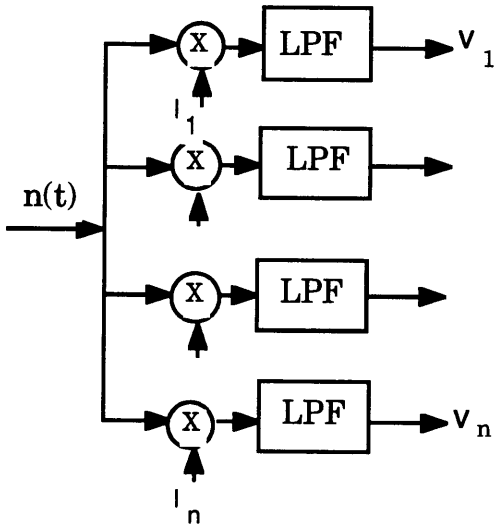

Figure 6.2-2 Bank of Correlators

The local codes are:

$$l_i(t) = p(t+\tau_i) \tag{6.2-3}$$

where: p(t) = the locally generated PRN code

The output of the filters is the convolution of the noise, the locally generated code, and the impulse response of the low pass filter:

$$v_i(t) = \int_0^T n(\tau)p(\tau+\tau_i)h(t-\tau)d\tau \tag{6.2-4}$$

where: h(t) = the impulse response of the filter

T = the integration time

The bandwidth of the filter is high enough so that the filter lets the noise and signal through. The output of the filters for n correlators is:

$$\overline{v} = \begin{bmatrix} n(t)p(t+\tau_i) \\ \cdot \\ \cdot \\ \cdot \\ n(t)p(t+\tau_n) \end{bmatrix} \tag{6.2-5}$$

The covariance matrix for the post correlation noise is:

$$P = E\left\{\overline{v}\overline{v}^T\right\} \tag{6.2-6}$$

The elements that make up the covariance matrix are obtained by substituting equation 6.2-5 into equation 6.2-6 and taking the expectation.

The resulting covariance matrix is a function of the PRN code autocorrelation function and noise intensity. The covariance matrix is:

$$P(i,j) = N_{pc}\Phi([i-j]\Delta) \tag{6.2-7}$$

where: i and j = the i^{th} and j^{th} correlators

N_{pc} = the noise intensity

$\Phi(\bullet)$ = the PRN code autocorrelation function

Δ = the spacing between correlators

The estimation procedure uses the output from a bank of correlators to sample the disturbed correlation function. The noise on each of these samples can be obtained from equation 6.2-7. The post correlation noise on the samples is:

$$\bar{n}_{pc} = C\bar{n} \tag{6.2-8}$$

where: C = the square root of the covariance matrix P

$\bar{n}$ = a vector of white unity variance Gaussian noise

Noise was simulated as a process described by equation 6.2-8. Specification of the noise level on the post correlation signal is given by the Signal to Noise Ratio (SNR) defined as:

$$SNR = \frac{\{\max(\Phi_T(\tau))\}^2}{N_{pc}} \tag{6.2-9}$$

where: $\max(\Phi_T(\tau))$ = the maximum correlation signal amplitude

6.3 Interval Optimization Results

Several simulations were run in order to evaluate the effects of noise on the performance of the Interval method. The simulations consisted of running different combinations of direct signal, multipath signal, and noise intensity through the Interval optimization with 10 different noise vectors chosen at random. Table 6.3-1 summarizes the inputs for the simulations presented in this section. The simulations that are summarized were run with amplitudes of 1.0 and 0.5 and delays of 0.2 and 0.5 chips for the direct and multipath signals respectively. The difference between the simulations was the noise intensity. The SNRs for the simulations were 30 and 50 dB. The 30 dB SNR represents a typical post correlation noise level in a standard receiver with a 0.02 second integration time [1], [7]. A total of twenty optimizations were run with two different SNRs and the same combination of signal parameters. The noise was generated by using equation 6.2-7 and a random number generator. Each simulation was run with the same end constraints. The interval stopping width was set to 0.01 as in the previous Interval optimizations.

Results from the first sequence of ten simulations with a SNR of 50 dB are shown in Figure 6.3-1. The figure contains two plots that contain the direct and multipath signal interval estimates. Each plot contains results from the ten individual runs with the same signal parameters. The interval estimates start from the initial conditions of -1.0 and 1.0 and quickly converge to the proper signal estimates of 0.2 and 0.5 chips. The intervals open back up again at the start of a new run with different noise and quickly converge to the correct levels. The plot contains results from ten simulations run with different noise.

The results shown in the figure are impressive. The Interval procedure was able to estimate both the direct and multipath signal delays to within 0.01 chips in the presence of noise. In each case the estimates converged to 0.2 and 0.5 chips within nine iterations. The estimates of the signal amplitudes although not shown were just as good.

Simulation	a_0	a_m	τ_0	τ_m	# of Runs	SNRs
1	1.0	0.5	0.2	0.5	10	50 dB
2	1.0	0.5	0.2	0.5	10	30 dB

Table 6.3-1 Simulation Inputs

The second simulation consisted of the same delay parameters but a lower more realistic SNR. The results are shown in Figure 6.3-2. These plots show the interval delay estimates for the ten different noise cases as was done in the first simulation. The plots also contain a line that shows the true signal delay parameters.

As in the first simulation the intervals start at -1.0 and 1.0 chips and converge to the specified 0.01 interval width accuracy, and open up, then converge again in later simulations. However in this case, half of the time the estimates do not converge to the correct signal delay values. The direct delay estimate plot shows that in some cases the estimates converge to a value 0.06 chips below the proper level. The multipath delay estimates behave the same. The presence of noise was enough to send the Interval search into the wrong region.

The Interval method works by examining large regions of the search space and discards regions based on interval evaluations. The presence of noise causes the algorithm to pick the wrong regions to discard. The parameter space where the solution lies is thrown out. The results in Figure 6.3-2 show this. When this happens there is no way to return to that region. The results from the second simulation are a good example of this. Three more simulations were run with different combinations of direct and multipath signal parameters and 30 dB SNRs to examine this behavior further. The results from these simulations displayed the same behavior as shown in Figure 6.3-2. The simulation results converged to incorrect estimates about half of the time late in the optimization procedure. However because the procedure broke down, it was viewed as prudent to develop a procedure where this cannot occur.

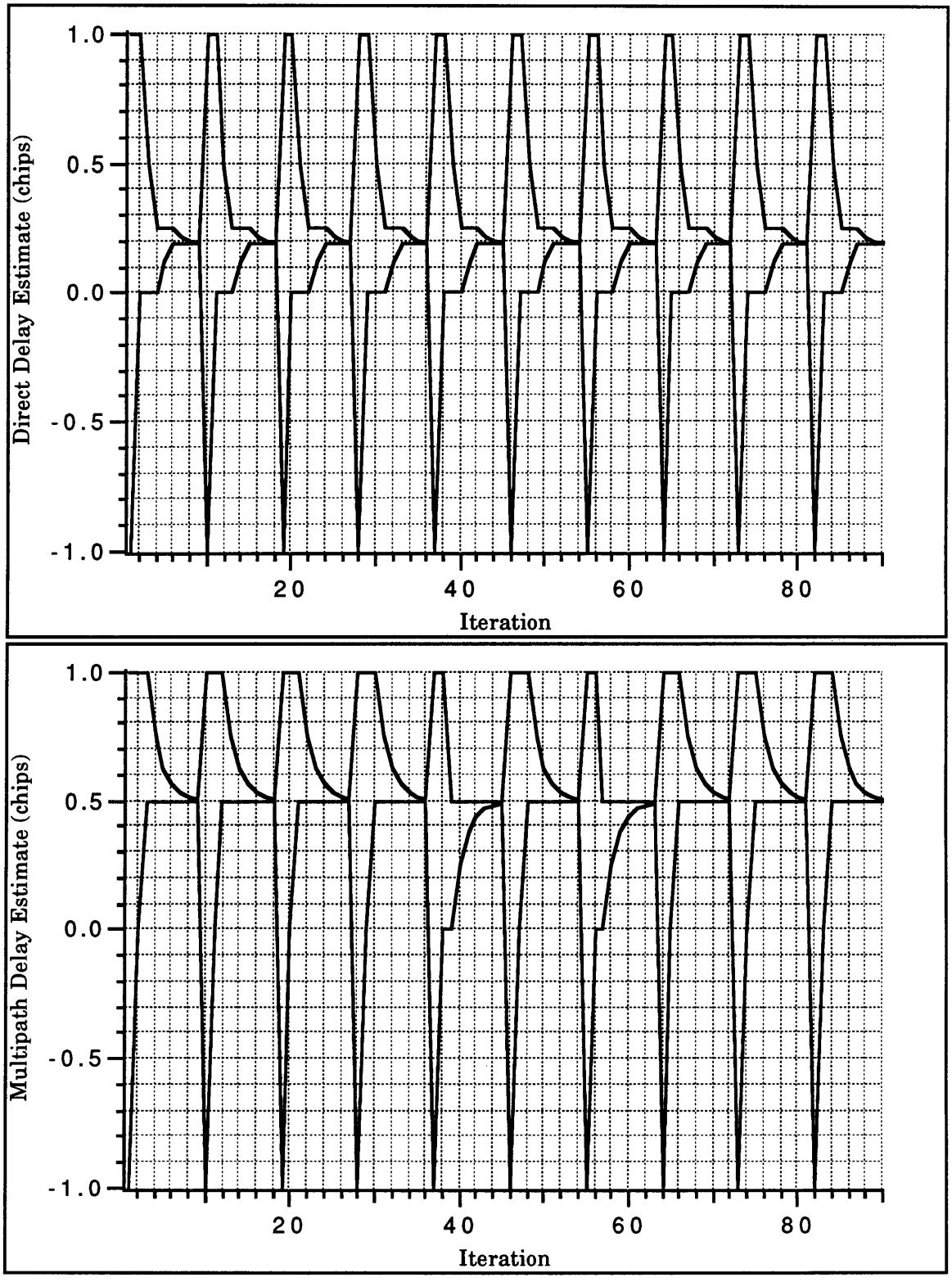

Figure 6.3-1 Monte Carlo Interval Results for Simulation 1, SNR = 50 dB

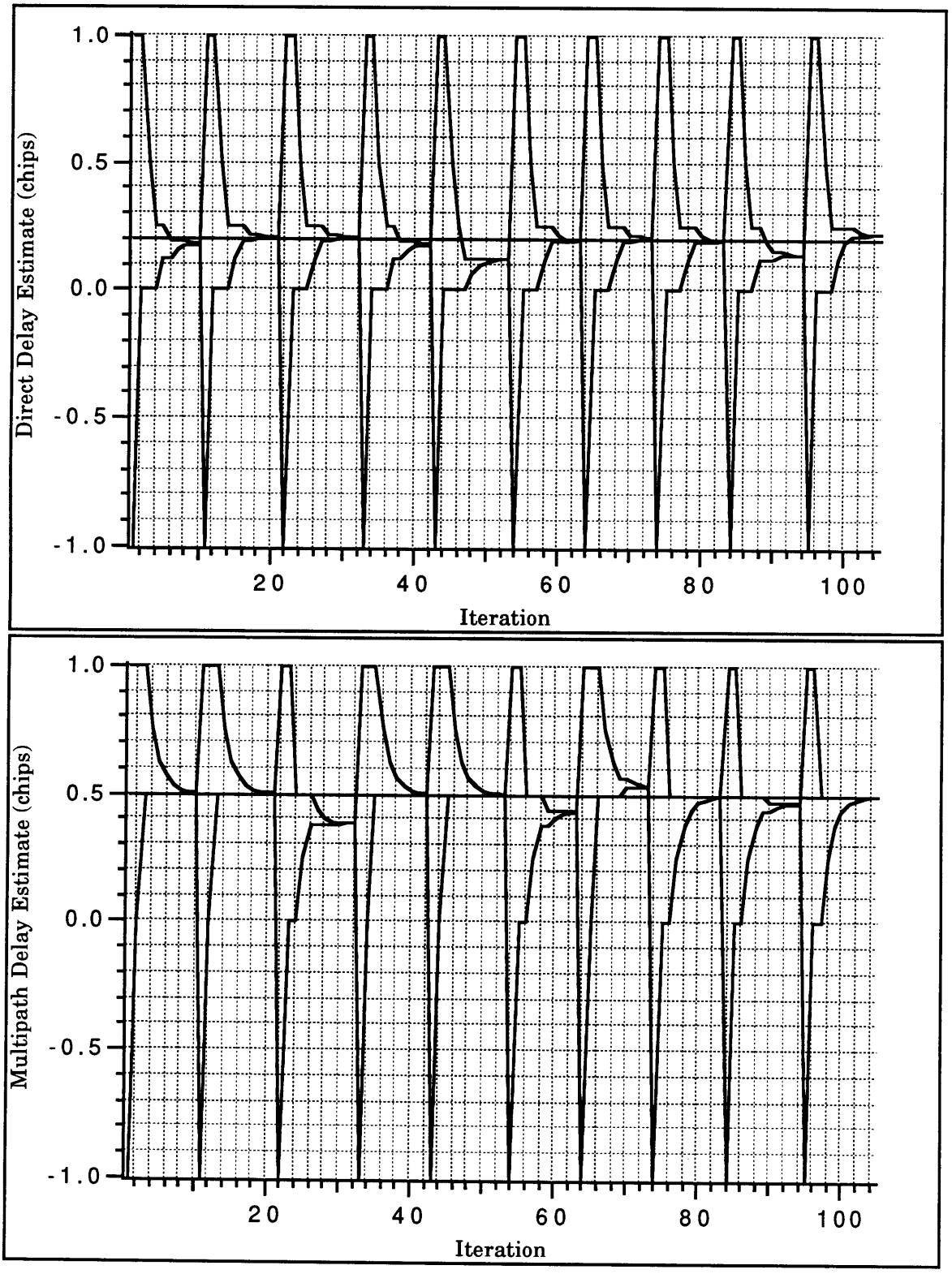

Figure 6.3-2 Monte Carlo Interval Results for Simulation 2, SNR = 30 dB

6.4 Genetic Optimization Results

The Interval method was combined with the Genetic optimization and used to estimate the signal parameters in the presence of noise. Genetic optimization does not explore a parameter space and discard regions of it. The algorithm conducts a directed search of many regions of the space at once and focuses on regions that are more likely to contain the correct solution parameters. The operations of reproduction, crossover, and mutation build a robustness to noise into the algorithm. This algorithm given the right initial conditions is capable of estimating the signal parameters in the presence of noise.

Fortunately the non-convergence problem described in section 6.3 did not occur early on in the Interval optimization. The initial parameter estimate intervals were reduced in every simulation enough such that the Genetic algorithm was able to finish the estimation process accurately. The same evaluation procedure used for the Interval procedure was used for the Genetic algorithm procedure. Monte Carlo analyses were run using the Genetic algorithm. Two different simulations were run with SNRs of 30 dB. The signal amplitudes were 1.0 and 0.5 and the delays were 0.2 and 0.5 chips for the first simulation. The multipath amplitude was changed to -0.5 for the second simulation. The starting conditions are shown in Table 6.4-1.

Parameter	a_0	a_m	τ_0	τ_m
Case 1 Max	1	0.5	0.5	0.5
Case 1 Min	0.5	0.0	0.0	0.0
Case 2 Max	1.0	-0.5	0.5	0.5
Case 2 Min	0.5	-1.0	0.0	0.0

Table 6.4-1 Initial Parameter Ranges

A total of twenty optimizations were run with the two different combinations of signal parameters. The 30 dB post correlation noise was

the same as the noise used in the Interval evaluation. The algorithm was run with forty measurements of the correlation function and evolved forty generations. A population size of 80 and a crossover probability of 0.7 were used.

The results from both simulations are presented in Table 6.4-2. The table contains the mean and standard deviation of the error in the signal estimates. The first ten Monte Carlo runs correspond to the simulation with the positive signal amplitudes. The last ten runs correspond to the second simulation with the negative multipath signal amplitude. The error is defined as the difference between the true and the estimated signal parameter. The table summarizes the twenty different optimization simulations. In every case the procedure was able to estimate the delay parameters to within 0.03 chips. This is nearly a factor of 7 improvement over a conventional receiver operating under these multipath conditions but without noise.

Figure 6.4-1 contains a plot of the results from one of the simulations. The direct and multipath signal amplitudes were 1.0 and -0.5. The direct and multipath delays were 0.2 and 0.5 chips. The plot contains the four signal parameter estimates at each generation. The estimates converged to the correct values by the tenth generation Each of the other nineteen simulations behaved in a very similar manner.

Monte Carlo Run	a_0		a_m		τ_0 (chips)		τ_m (chips)	
	mean	σ	mean	σ	mean	σ	mean	σ
1	0.013	0.009	0.005	0.007	-.016	0.009	0.015	0.018
2	0.016	0.009	0.003	0.004	-.022	0.031	0.016	0.031
3	0.003	0.006	0.003	0.005	-.013	0.012	0.018	0.016
4	0.024	0.015	0.028	0.019	-.044	0.031	0.040	0.029
5	0.007	0.014	0.028	0.015	-.042	0.025	0.009	0.022
6	0.001	0.001	0.005	0.005	0.023	0.009	0.012	0.009
7	0.000	0.000	0.003	0.006	-.017	0.028	0.008	0.013
8	0.007	0.009	0.005	0.007	-.016	0.02	0.04	0.034
9	0.010	0.009	0.005	0.007	-.012	0.011	0.006	0.006
10	0.021	0.006	0.003	0.005	-.038	0.007	0.001	0.04
11	0.007	0.011	0.008	0.015	-.007	0.021	0.031	0.030
12	0.018	0.005	0.005	0.004	0.003	0.012	0.007	0.013
13	0.007	0.029	0.007	0.012	0.005	0.022	0.005	0.028
14	0.004	0.007	0.009	0.013	-.029	0.018	0.012	0.024
15	0.030	0.011	0.019	0.012	0.004	0.024	0.058	0.019
16	0.006	0.005	0.003	0.005	0.014	0.034	0.008	0.009
17	0.007	0.009	0.003	0.004	0.024	0.010	0.0127	0.020
18	0.009	0.009	0.059	0.022	-.010	0.007	0.003	0.006
19	0.009	0.009	0.029	0.012	0.003	0.020	0.003	0.004
20	0.008	0.012	0.008	0.010	-.039	0.019	0.029	0.028

Table 6.4-2 Summary of Genetic Optimization Results

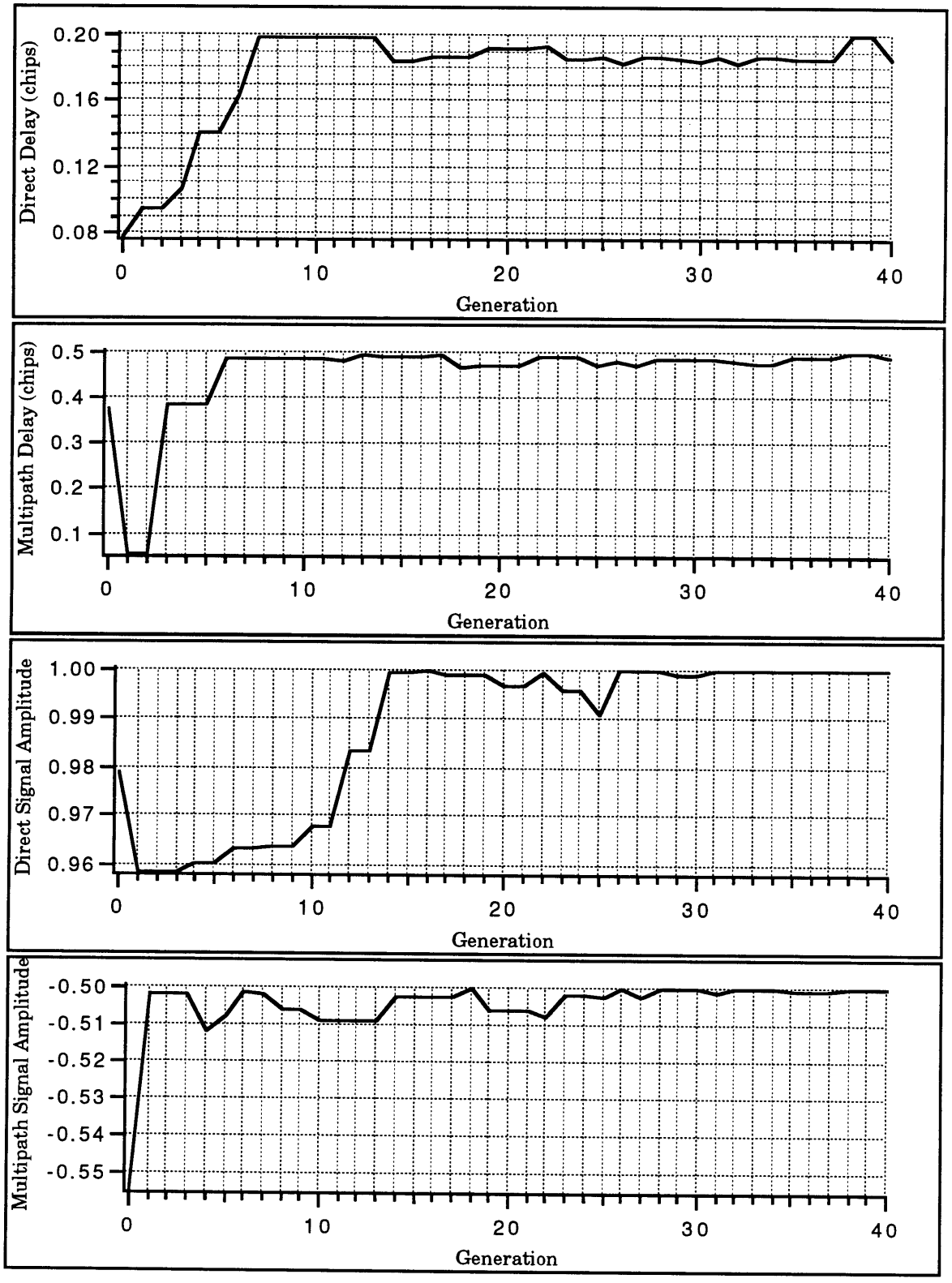

Figure 6.4-1 Genetic Optimization Results in the Presence of Noise

6.5 Summary

The results from the combined Interval and Genetic optimization were impressive. The procedure was used to estimate the time delay of the direct signal in the presence of one multipath signal and noise. The algorithm works with the same degree of accuracy even when no multipath is present. The Interval procedure was used to supply initial parameter ranges the Genetic algorithm for parameter estimation. A SNR of 30 dB was used. The combined procedure demonstrates the capability to jointly estimate the signal parameters a factor of seven times better than a conventional receiver under identical but noiseless multipath conditions could.

Chapter 7

Conclusion

7.1 Conclusions

A technique to estimate the transit time of the timing signal broadcast by the GPS satellites was developed. The procedure works in the presence of one specular multipath signal and signal noise. The procedure estimates the time delays of the direct and multipath signal rather than suppressing the effects of multipath that a traditional receiver experiences. The algorithm was able to estimate the time delays to better than 0.03 chips (8.5 meters for the C/A code) in the presence of multipath and noise with a 30 dB SNR. The technique works just as well when no multipath signal is present. Results from the simulated cases as shown in Chapter 6 demonstrated a factor of seven improvement over a non coherent DLL receiver operating in the same multipath environment without noise. The accuracy of the procedure developed in this thesis was not sensitive to the strength or delay of the multipath signal. Thus the factor of seven improvement would increase for more severe multipath environments and decrease for less severe environments.

Implementing the algorithm in a receiver is practical. Most new receivers use digital processing and adapting them to contain multiple correlation channels is feasible. A digital signal processor (DSP) would execute the algorithm in real time using outputs from the correlation channels. Estimates of the signal parameters would be available at a rate proportional to the throughput of the DSP.

The combined Interval and Genetic optimization scheme executes approximately 2 million operations in order to obtain the signal parameter estimates. The 2 million operations are the total of the additions, subtractions, etc. of the optimization routines as coded in MATLAB. The digital signal processors that are available are capable of performing operations at 0.1 micro second speed. Thus, estimates would be available every 0.2 seconds. Section 7.2 offers suggestions to decrease the processing time.

The receiver would also need a random number generator and processor capable of performing the Interval mathematics. For example, a processor built by Intel has the necessary directed rounding capability used as part of the interval endpoint formulation. The processor need only be programmed with the necessary transcendental function routines peculiar to Interval optimization such as the PRN code autocorrelation function. Coding of the Genetic algorithm would be much simpler to program because the procedure works with string structures coded as binary numbers and manipulating these structures is simple.

The technique developed in the thesis is constrained by the ability to perform the correlations and estimation fast enough such that the Doppler terms can be treated as constants. The use of modern digital signal processing technology should allow for fast correlations to be done whether in parallel or a staggered configuration with shared correlators. The entire algorithm needs to be executed fast compared to the Doppler terms. A trade off between receiver dynamics and algorithm accuracy needs to be done.

7.2 Recommendations For Further Research

Future research can follow two different paths; continued effort with the model based approach or alternate non model based techniques. Because a non-model based technique is unlikely the recommendations summarized in this section will focus on further characterization of the technique invented in this thesis and extensions to it. The procedure eliminates the effects of one specular multipath signal. The algorithm

relies on prior information about how many multipath signals are present. The effects of mismodeling the multipath environment should be examined. The receiver operates under the assumption that no multipath or one multipath signal is present. The performance of the algorithm needs to be analyzed when more than one multipath signal is present.

Another recommendation is to modify the procedure to include the capability to handle more specular multipath signals. The optimization procedures are easily modified to add additional signal parameters. The cost function defined by equation 3.4-2 would be modified to include more multipath signals. Another amplitude and delay would need to be estimated for every multipath signal added. The initial conditions for both the Genetic and Interval methods would be increased to match the number of extra multipath signals added. One last modification is necessary. The box subdivision portion of the Interval method would need to be modified to handle the increased parameter space.

Implementation issues with the actual execution of the algorithm in real time need to be addressed. As mentioned previously the algorithms execute approximately 2 million instructions in order to get parameter estimates. Investigation of possible ways to reduce the operation count is needed. The operation count is dominated by the Genetic algorithm. The Genetic algorithm as coded propagates a population of 80 members over 40 generations. The fitness evaluation for each member is operation intensive and is the main source of the high operation count.

Reducing the fitness evaluation sequence is practical. The fitness is the sum of the difference squared between the estimated correlation function and the sampled function. An easy way to reduce the operation count is to reduce the input measurements. Another approach would be to use a smaller population size. It may be possible to run the Genetic algorithm with interval strings. Ranges of different populations could be evaluated at once which would reduce the computation count. Trade studies of these alternatives need to be done in order to evaluate their effect on performance.

Finally, the optimization code needs to be fine tuned and coded for efficiency. The crossover and mutation probabilities used in the Genetic

algorithm should be studied and altered. Quicker convergence to the proper estimates may be possible. The optimization routines were coded for functionality and not real time execution. The binary nature of the Genetic algorithm may lend itself to parallel architectures. Execution of the Genetic algorithm in parallel would offer significant throughput savings that would greatly facilitate the realization of the algorithms into a real time system.

Appendix A

Maximum Pseudorange Errors

As stated in Chapter 2, van Nee developed closed form equations for the maximum pseudorange error in the presence of one multipath signal. The largest errors occur when the multipath signal is out of phase by 180 degrees or exactly in phase with respect to the direct signal. In this case the discriminator is [12]:

$$D(t,e) = a_0^2(\Phi(e+d)^2 - \Phi(e-d)^2) + a_m^2(\Phi(e+\tau_m+d)^2 - \Phi(e+\tau_m-d)^2) \qquad \text{(A-1)}$$

where: a_0 and a_m = the direct and multipath signal amplitudes

$\Phi(\bullet)$ = the PRN code autocorrelation function

$e = \tau_0 - \hat{\tau}_0$ = the difference between the true direct signal delay and its estimate

τ_m = the multipath delay

d = the offset of the locally generated codes

The offset of the locally generated codes was set to plus and minus a half of a code chip for the work done in the thesis. The equations developed by van Nee were done with arbitrary offsets. The maximum pseudorange error due to the distortion is obtained

from the roots of equation A-1. The roots are:

$$\varphi_e = \frac{a_m}{a_0 + a_m}\tau_m \quad \text{when} \quad 0 \le \tau_m < \frac{(a_0 + a_m)d}{2a_0} \tag{A-2}$$

$$\varphi_e = \frac{a_m d}{2a_0} \quad \text{when} \quad \frac{(a_0 + a_m)d}{2a_0} \le \tau_m \le T_c - d\left(1 - \frac{a_0 + a_m}{2a_0}\right) \tag{A-3}$$

$$\varphi_e = \frac{a_m}{2a_0 - a_m}\left(T_c + \frac{d}{2} - \tau_m\right) \quad \text{when} \quad T_c - d\left(1 - \frac{a_0 + a_m}{2a_0}\right) < \tau_m \le T_c + \frac{d}{2} \tag{A-4}$$

$$\varphi_e = 0 \quad \text{when} \quad \tau_m > T_c + \frac{d}{2} \tag{A-5}$$

Appendix B

Foundations of the Genetic Algorithm

The underlying mathematical foundations of the Genetic algorithm are described in this Appendix. The theory behind the algorithm relies on schemata which are similarity templates that describe subsets of strings. The description was taken as an excerpt from [9] with the author's permission.

The effect of reproduction, crossover, and mutation on the information contained in the population is investigated in a more rigorous way using schemata which exploit the underlying similarities between the strings of the population. A schema is a similarity template describing subsets of strings with similarities at specific loci. It can also be though of as a hyper-plane in the search space. Using the binary alphabet {0,1}, define a new element * which functions as a wild card. Schemata are then created over the extended alphabet {0,1,*} as a pattern matching mechanism where * can be used to match either a 1 or 0. A schema matches a particular string if at every location 1 in the schema fits 1 in the string, 0 matches 0, or * matches either. An example of schemata and sets of matching strings is shown below.

schema	100*	1*0*	10**
sets of	1000	1000	1000
matching	1001	1001	1001
strings		1100	1010
		1101	1011

For alphabets of cardinality k and string length l there are $(k+1)^l$ schemata. Because each string position can take its own value or the wildcard symbol, each string is a match to k^l schemata. A population of n strings from the binary alphabet will therefore display between 2^l and $2^l n$ schemata, depending on the diversity of the members of the population. For a population of n members an estimated $O(n^3)$ schemata are processed. Thus, for each generation requiring n fitness evaluations, n^3 are processed in parallel without memory other that the population itself. This important feature of Genetic algorithms is known as implicit parallelism.

Two definitive properties which distinguish between schemata are the schema order and schema defining length. The order $o(H)$ of a schema H is the number of specific bits (1's and 0's) in the template. For example, if H is given by *1**10*, then $o(H) = 3$. The defining length $\delta(H)$ of a schema H is the distance between the first and last specific bit positions. For the same H as above $\delta(H) = 4$. In the case where $o(H) = 1$ the defining length by definition $\delta(H) = 0$.

Schemata provide the means to analyze the effects of the genetic operators on the information contained in the population. The effect of reproduction on the expected number of schemata is determined by treating each schema as a random variable with mean estimated by the average fitness of its occurrences in the population. During reproduction a string A_i is copied according to its fitness F_i, by being selected to the mating pool with probability $p_i = F_i / \sum F_j$. The expected number of copies of the string A_i in the mating pool is then given by np_i. The growth or decay of a particular schema can be described in a similar manner. Define $m(H,t)$ to be the number of instances of a particular schema H contained in the population A(t) at time step t. The expected occurrences of H in A(t+1) is then given by the equation:

$$m(H,t+1) = n\frac{m(H,t)f(H)}{\sum F_j} \qquad \text{(B-1)}$$

where $f(H)$ is the average fitness of the strings representing schema H at time t. Recognizing that the average fitness of the entire population $F_{avg} = \sum F_j / n$, equation B-1 can be reduced to:

$$m(H,t+1) = m(H,t)\frac{f(H)}{F_{avg}} \tag{B-2}$$

Equation B-2 is known as the reproductive schema growth equation. From this equation, the number of above average schemata in the population will grow and below average schemata will become less. If $f(H)$ remains consistently above the average fitness of the population, such that $f(H) = (1+b)F_{avg}$, where b is a constant, then equation B-2 is:

$$m(H,t+1) = (1+b)m(H,t) \tag{B-3}$$

Equation B-3 is the discrete equivalent of the exponential form. Note that this does not account for the effect of crossover and mutation. Reproduction thus yields and exponentially increasing (decreasing) number of above (below) average schemata in progressive populations.

Reproduction is a highly exploitive search of the parameter space. If reproduction was the only operator employed, successive populations would consist only of increasing numbers of above average strings. Crossover allows for a structured information exchange between strings, which creates new strings and thus promotes exploration of the search space. Reproduction and crossover are opposing forces in that reproduction tends to increase the number of above average schemata while crossover destroys schemata. If crossover between two strings occurs at random with probability p_c, then the lower bound on the probability that a schema survives crossover is:

$$p_s \geq 1 - p_c \frac{\delta(H)}{l-1} \tag{B-4}$$

Combining reproduction and crossover, the expected occurrences of schema H in the population A(t+1) is:

$$m(H,t+1) \geq m(H,t) \frac{f(H)}{F_{avg}} \left[1 - p_c \frac{\delta(H)}{l-1} \right] \tag{B-5}$$

The significant terms in equation B-5 are $f(H)$ and $\delta(H)$. Effectively, schemata that have above average performance and short defining lengths will increase exponentially in successive populations.

Mutation has the same tendency to destroy schemata as crossover. For schema to survive mutation all the specified bits (there are $o(H)$ of them) have to survive mutation. The probability that a schema survives mutation is then $(1-p_m)^{o(H)}$ where p_m is the probability that a bit will mutate. For small values of p_m this expression can be approximated by $1-o(H)\,p_m$. Combining all of the operators the expected number of copies that a particular schema H will have in the next generation is:

$$m(H,t+1) \geq m(H,t) \frac{f(H)}{F_{avg}} \left[1 - p_c \frac{d(H)}{l-1} - o(H) p_m \right] \tag{B-6}$$

The cross products in equation B-6 have been ignored. Equation B-6 proves that above average schemata of low order with short defining lengths increase exponentially in successive populations. This result is known as the fundamental theorem of Genetic algorithms.

Appendix C

Interval Mathematics

The mathematical rules governing the generation of the Interval endpoints is summarized in this Appendix. The operators are defined in terms of two intervals X = [a,b] and Y = [c,d].

$$X + Y = [a+c,\ b+d] \tag{C-1}$$

$$X - Y = [a-d,b-c] \tag{C-2}$$

$$X * Y = \begin{cases} [ac, bd] & \text{if } a \geq 0 \text{ and } c \geq 0 \\ [bc, bd] & \text{if } a \geq 0 \text{ and } c < 0 < d \\ [bc, ad] & \text{if } a \geq 0 \text{ and } d \leq 0 \\ [ad, bc] & \text{if } a < 0 < b \text{ and } c \geq 0 \\ [bd, ad] & \text{if } a < 0 < b \text{ and } d \leq 0 \\ [ad, bc] & \text{if } b \leq 0 \text{ and } c \leq 0 \\ [ad, ac] & \text{if } b \leq 0 \text{ and } c < 0 < d \\ [bd, ac] & \text{if } b \leq 0 \text{ and } d \leq 0 \\ [\min(bc, ad), \max(ac, bd)] & \text{if } a < 0 < b \text{ and } c < 0 < d \end{cases} \tag{C-3}$$

$$\frac{1}{Y} = \left[\frac{1}{d}, \frac{1}{c}\right] \qquad 0 \notin Y \tag{C-4}$$

$$\frac{X}{Y} = X * \frac{1}{Y} \qquad 0 \notin Y \tag{C-5}$$

$$X^n = \begin{cases} [1,1] & \text{if } n = 0 \\ [a^n, b^n] & \text{if } a \geq 0 \text{ or if } a \leq 0 \leq b \text{ and } n \text{ is odd} \\ [b^n, a^n] & \text{if } b \leq 0 \\ [0, \max(a^n, b^n)] & \text{if } a \leq 0 \leq b \text{ and } n \text{ is even for } n = 0, 1, 2, \ldots \end{cases} \tag{C-6}$$

Rules for division of intervals containing zeros is called extended interval arithmetic. These rules can be found in [5].

References

1. Bowles, Michael W., "Correlation Tracking", Doctoral Dissertation, Massachusetts Institute of Technology, June 1980.

2. Braasch, Michael S., "On The Characterization Of Multipath Errors in Satellite-Based Precision Approach And Landing Systems", Doctoral Dissertation, Ohio University, June 1992.

3. Georgiadou, Y. and A. Kleusberg, "On Carrier Signal Multipath Effects in Relative GPS Positioning", Manuscripta Geodaetica, 1988, Vol. 13.

4. Goldberg, David E., Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley Publishing Company, INC., Massachusetts, 1989.

5. Hansen, Eldon, Global Optimization Using Interval Analysis, Marcel Dekker, Inc., New York, 1992.

6. Moore, R. E., Interval Analysis, Prentice-Hall, New Jersey, 1966.

7. Spilker, James J., Digital Communications by Satellite, Prentice-Hall, New Jersey, 1977.

8. Tranquilla, J. M., and Carr, J. P., "GPS Multipath Field Observations at Land and Water Sites", Navigation, Journal of the Institute of Navigation, vol. 37, no. 4, Winter 1990-91, pp. 393-414.

9. van Deventer, Paul G., "Flight Control Command Generation In A Real-Time Mission Planning System Using Constrained Genetic Optimization", Massachusetts Institute of Technology, June 1992.

10. Van Dierendonck, A. J., "Theory And Performance Of Narrow Correlator Spacing In A GPS Receiver", Institute of Navigation National Technical Meeting, January 1992.

11. van Nee, D. J., "Multipath Effects On GPS Code Phase Measurements", Proceedings of ION GPS-91, New Mexico, September 1991, pp. 915-924.

12. van Nee, D. J., "GPS Multipath and Satellite Interference", Proceedings of ION GPS-92, Washington DC, July 1992.