

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

ROBUST GPS AUTONOMOUS SIGNAL QUALITY MONITORING

A DISSERTATION

SUBMITTED TO THE DEPARTMENT OF MECHANICAL ENGINEERING

AND THE COMMITTEE ON GRADUATE STUDIES

OF STANFORD UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

Awele N. Ndili

August 1998

UMI Number: 9908831

**Copyright 1998 by
Ndili, Awele Nnaemeka**

All rights reserved.

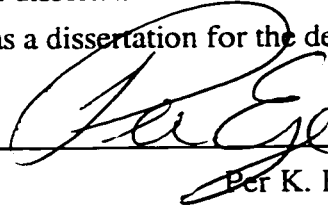
**UMI Microform 9908831
Copyright 1998, by UMI Company. All rights reserved.**

**This microform edition is protected against unauthorized
copying under Title 17, United States Code.**

UMI
300 North Zeeb Road
Ann Arbor, MI 48103

Copyright © 1998 by Awele N. Ndili
All Rights Reserved.

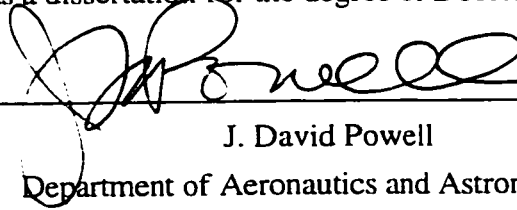
I certify that I have read this dissertation and that in my opinion it is fully adequate,
in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.



Per K. Enge

Department of Aeronautics and Astronautics
Principal Adviser

I certify that I have read this dissertation and that in my opinion it is fully adequate,
in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.



J. David Powell

Department of Aeronautics and Astronautics

I certify that I have read this dissertation and that in my opinion it is fully adequate,
in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.



Jonathan P. How

Department of Aeronautics and Astronautics

Approved for the University Committee on Graduate Studies:



*To Ada,
with all my love*

Abstract

The Global Positioning System (GPS), introduced by the U.S. Department of Defense in 1973, provides world-wide navigation capabilities to both military and commercial users. It achieves this purpose through a constellation of 24 satellites in global orbit, each emitting a low-power radio-frequency signal for ranging. GPS receivers receive these transmitted signals and compute position from range measurements made to four or more visible satellites. An end user with a GPS receiver can determine their location anywhere on the globe to within 30 meters.

GPS has become very popular in recent years, finding a wide range of applications, including aircraft navigation, marine oceanic, coastal and inland-waterway navigation, search and rescue, agriculture, surveying, space borne attitude and position determination, and vehicle navigation. Each application places demands on GPS for various levels of accuracy, integrity, system availability, and continuity of service. For example, aircraft Category I precision approach requires a 5m vertical accuracy, achieved 99.9% of the time.

Radio frequency interference (RFI), which results from many sources such as TV/FM harmonics, radar or Mobile Satellite Systems (MSS), presents a challenge in the use of GPS, by posing a threat to the accuracy, integrity and availability of the GPS navigation solution. This threat is a result of the low power levels of received GPS satellite signals relative to RFI. With increasing interference, GPS accuracy degrades, with a resulting loss in integrity. At high enough RFI power levels, the GPS receiver loses lock, resulting in system unavailability and a loss of continuity of service. This presents a severe problem to GPS users, especially for integrity-critical applications such as aircraft high-precision approach and auto-land, where an integrity breach could result in loss of life.

In order to use GPS for integrity-sensitive applications, it is therefore necessary to monitor the quality of the received signal, with the objective of promptly detecting the presence of RFI, and thus provide a timely warning of the degradation of system accuracy, thereby boosting the integrity of GPS. This presents a challenge, since the myriad kinds of RFI effect the GPS receiver in different ways. In addition, there are other forms of interference, such as physical blockage and attenuation, which though not being of the radio frequency type, nonetheless contribute to the integrity threat. What is required then, is a *robust* method of detecting GPS accuracy degradation which is effective regardless of the origin of the threat.

This dissertation presents a new method of robust signal quality monitoring for GPS. Algorithms for receiver autonomous interference detection and integrity monitoring are presented. Candidate test statistics are derived from fundamental receiver measurements of in-phase and quadrature correlation outputs, and the gain of the Active Gain Controller (AGC). Performance of selected test statistics is evaluated in the presence of RFI: broadband interference, pulsed and non-pulsed interference, coherent CW at different frequencies; and non-RFI: GPS signal fading due to physical blockage and multipath. Results are presented which verify the effectiveness of the methods proposed.

The benefits of pseudolites in reducing service outages due to interference are demonstrated. Pseudolites also enhance the geometry of the GPS constellation, improving overall system accuracy. Designs for pseudolites signals, to reduce the near-far problem associated with pseudolites use, are also presented.

Acknowledgements

I wish to thank my adviser, Professor Per Enge, for providing an excellent research environment, for his encouragement, technical guidance and stimulation, and exposure to excellent resources. I will always be grateful not only for an excellent student-faculty relationship, but also for his keen interest in my welfare.

I owe a great deal of gratitude to the other members of my reading committee: Professor David Powell, started me out in the doctorate path on a strong academic and financial footing, and has continued to play an important role in my career. Professor Jonathan How willingly accepted the role of examiner and committee member under short notice, going the extra mile to review this work.

I would like to thank the Department of Aeronautics and Astronautics here at Stanford, for a productive and enjoyable three year support, during which in my role as Network Administrator, I was privileged to interact with a large number of students, staff and faculty. In particular I wish to thank the department administrator, Lisa Akselrad, and building manager, Vadim Matte, for the great support I continue to receive.

I would also like to thank my colleagues, members of the Stanford GPS group, for their support, and willingness to listen and review my research over the years. In particular but by no means conclusive I would like to thank Jock Christie, Todd Walter, Boris Pervan, Y.J. Tsai, Andrew Hansen, Sam Pullen and Hiro Uematsu.

This work was supported in part by a Research Grant from the FAA. I am grateful for the FAA support.

I would like to thanks my friends. for encouragement and support. and for providing a well-balanced environment. making this endeavor all the more worthwhile. I owe a lot of gratitude to my family. for believing in me. A more solid springboard one could not ask for.

Finally I would like to thank my wife, Ada, who has 'lived' through the writing of this thesis, and whose presence, encouragement and support made all the difference. It is to her I dedicate this thesis.

Contents

ABSTRACT.....	VII
ACKNOWLEDGEMENTS.....	IX
LIST OF TABLES	XIII
LIST OF FIGURES.....	XIV
1 INTRODUCTION.....	1
1.1 GLOBAL POSITIONING SYSTEM OVERVIEW	1
1.2 THE INTERFERENCE PROBLEM.....	7
1.3 INTEGRITY MONITORING.....	9
1.4 THESIS APPROACH AND GOALS	10
1.5 RESEARCH CONTRIBUTIONS.....	13
1.6 READERS GUIDE	14
2 CANDIDATE INTEGRITY MONITOR STATISTICS.....	15
2.1 INTRODUCTION.....	15
2.2 GPS RECEIVER OVERVIEW.....	16
2.2.1 <i>Antenna / Low Noise Amplifier</i>	16
2.2.2 <i>Down Conversion</i>	17
2.2.3 <i>Analog to Digital Conversion</i>	18
2.2.4 <i>Correlation</i>	19
2.2.5 <i>Acquisition</i>	21
2.2.6 <i>Code Tracking</i>	22
2.2.7 <i>Carrier Tracking</i>	24
2.3 CANDIDATE TEST STATISTICS	26
2.3.1 <i>Correlator Output Power</i>	26
2.3.2 <i>Correlator Output Power Variance</i>	28
2.3.3 <i>Carrier Phase Jitter</i>	29
2.3.4 <i>AGC Gain</i>	31
3 INTERFERENCE ANALYSES.....	35
3.1 INTRODUCTION.....	35
3.2 RF INTERFERENCE TO GPS.....	35
3.3 GPS RECEIVER SUSCEPTIBILITY TO RF INTERFERENCE	38
3.4 EXPECTED INTERFERENCE SIGNAL POWER FOR LOSS OF LOCK.....	44
3.5 COHERENT CW INTERFERENCE.....	46
3.6 IMPACT OF INTERFERENCE ON GPS	51
3.6.1 <i>Impact on Accuracy</i>	51
3.6.2 <i>Impact on Integrity and Continuity</i>	53
3.6.3 <i>Impact on Availability</i>	53
4 SIMULATION AND BENCH TEST SETUP.....	55
4.1 INTRODUCTION.....	55
4.2 GPS SIMULATION.....	56
4.2.1 <i>Signal Generation and Downconversion</i>	58
4.2.2 <i>Analog to Digital Converter</i>	62
4.2.3 <i>Correlation</i>	63
4.2.4 <i>Tracking</i>	64

4.3	BENCH TEST SETUP.....	65
4.4	TEST PROCEDURES.....	67
5	INTEGRITY MONITORING RESULTS.....	71
5.1	INTRODUCTION.....	71
5.2	IMPACT OF INTERFERENCE ON CODE AND CARRIER TRACKING LOOPS.....	72
5.2.1	<i>Impact of Continuous RF Interference on Code and Carrier Tracking Loops</i>	72
5.2.2	<i>Impact of Pulsed RF Interference on Code and Carrier Tracking Loops</i>	74
5.2.3	<i>Impact of Signal Attenuation on Code and Carrier Tracking Loops</i>	76
5.3	IMPACT OF INTERFERENCE ON TEST STATISTICS.....	78
5.3.1	<i>Impact of Continuous RF Interference on Test Statistics</i>	78
5.3.2	<i>Impact of Pulsed RF Interference on Test Statistics</i>	80
5.3.3	<i>Impact of Signal Attenuation on Test Statistics</i>	84
5.4	COHERENT INTERFERENCE AND LOOP CAPTURE.....	84
5.5	TEST STATISTICS PERFORMANCE.....	87
5.5.1	<i>Performance of Correlator Output Power</i>	88
5.5.2	<i>Performance of Correlator Output Power Variance</i>	90
5.5.3	<i>Performance of Carrier Phase Jitter</i>	91
5.5.4	<i>Performance of AGC Gain</i>	92
5.6	FREQUENCY ERROR MONITORING USING TEST STATISTICS.....	93
5.7	BENCH TEST RESULTS.....	107
5.8	CONCLUSIONS.....	110
6	PSEUDOLITES TO MITIGATION SERVICE OUTAGES DUE TO INTERFERENCE.....	113
6.1	INTRODUCTION.....	113
6.2	INTERFERENCE MITIGATION THROUGH USE OF AIRPORT PSEUDOLITES.....	114
6.2.1	<i>APL Covariance Analyses</i>	114
6.2.2	<i>Results of APL Covariance Analyses</i>	118
6.3	PSEUDOLITE SIGNAL DESIGN.....	129
6.3.1	<i>Introduction</i>	129
6.3.2	<i>Random Sequence Analyses</i>	133
6.3.3	<i>Code Realizations</i>	137
6.3.4	<i>Pseudolite Signal Design Conclusions</i>	142
7	CONCLUSIONS AND FUTURE WORK.....	145
7.1	SUMMARY.....	145
7.1.1	<i>Autonomous Signal Quality Monitoring</i>	145
7.1.2	<i>Interference Mitigation via Use of Airport Pseudolites</i>	146
7.1.3	<i>Pseudolite Signal Design</i>	147
7.2	RECOMMENDATIONS FOR FUTURE WORK.....	147
	APPENDIX A: SIMULATION RESULTS.....	149
	APPENDIX B: BENCH TEST RESULTS.....	199
	APPENDIX C: SIMULATION SOFTWARE CODE.....	237
	APPENDIX D: HARDWARE CONFIGURATION INPUT FILES.....	311
	APPENDIX E: RUN CONFIGURATION INPUT FILES.....	327
	BIBLIOGRAPHY.....	337

List of Tables

TABLE 1: ACCURACY REQUIREMENTS FOR AIRCRAFT PRECISION APPROACHES.....	4
TABLE 2: INTEGRITY REQUIREMENTS FOR AIRCRAFT PRECISION APPROACHES.....	6
TABLE 3: POTENTIAL SOURCES OF INTERFERENCE TO GPS RECEIVERS.....	37
TABLE 4: SAMPLE RECEIVER HARDWARE CONFIGURATION INPUT FILE.....	57
TABLE 5: SAMPLE RUN CONFIGURATION FILE FOR CONTINUOUS CW INTERFERENCE, WITH POWER LEVEL RANGING FROM 0 TO 40 DB.....	68
TABLE 6: OVERVIEW OF PRESENTED RESULTS	71
TABLE 7: LEAST SQUARES ERROR LINEAR FIT SLOPES AND RMS DATA SPREADS FROM LINEAR FIT, FOR CORRELATOR OUTPUT POWER WHEN USED AS AN INTEGRITY MONITORING TEST STATISTIC.....	89
TABLE 8: LEAST SQUARES ERROR LINEAR FIT SLOPES AND RMS DATA SPREADS FROM LINEAR FIT, FOR CORRELATOR OUTPUT POWER VARIANCE WHEN USED AS AN INTEGRITY MONITORING TEST STATISTIC.....	90
TABLE 9: LEAST SQUARES ERROR LINEAR FIT SLOPES AND RMS DATA SPREADS FROM LINEAR FIT, FOR CARRIER PHASE JITTER WHEN USED AS AN INTEGRITY MONITORING TEST STATISTIC.....	91
TABLE 10: LEAST SQUARES ERROR LINEAR FIT SLOPES AND RMS DATA SPREADS FROM LINEAR FIT, FOR AGC GAIN WHEN USED AS AN INTEGRITY MONITORING TEST STATISTIC.....	92
TABLE 11: LEAST SQUARES ERROR LINEAR FIT SLOPES AND RMS DATA SPREADS FROM LINEAR FIT, FOR AGC GAIN USING A FAST ACTING AGC, WHEN USED AS AN INTEGRITY MONITORING TEST STATISTIC.....	93
TABLE 12: COMPARISON OF TEST STATISTIC PERFORMANCE.....	111
TABLE 13: PROCESSING GAIN USING LENGTH-4X31 CODES (IN DB)	141
TABLE 14: PROCESSING GAIN USING LENGTH-4X1023 CODES (IN DB).....	141

List of Figures

FIGURE 1: GPS CONSTELLATION OF 24 SATELLITES.....	2
FIGURE 2: POSITION DETERMINATION VIA FOUR PSEUDORANGE MEASUREMENTS.....	3
FIGURE 3: EFFECTIVE JAMMING DISTANCE AS A FUNCTION OF JAMMER POWER.....	8
FIGURE 4: INTEGRITY MONITORING APPROACHES.....	10
FIGURE 5: TEST STATISTICS DECISION MATRIX.....	11
FIGURE 6: PSEUDOLITE POWER AND PULSING SCHEME.....	13
FIGURE 7: SCHEMATIC OF A GENERIC GPS RECEIVER.....	16
FIGURE 8: 5-STAGE DOWN CONVERSION.....	17
FIGURE 9: GPS RECEIVER DIGITIZATION.....	18
FIGURE 10: CORRELATION PEAK.....	19
FIGURE 11: INPHASE AND QUADRATURE COMPONENTS OF THE EARLY LOCAL SIGNAL.....	20
FIGURE 12: SCHEMATIC OF A SINGLE CORRELATOR CHANNEL.....	21
FIGURE 13: CODE AND CARRIER TRACKING LOOPS.....	22
FIGURE 14: SECOND ORDER DELAY LOCKED LOOP.....	23
FIGURE 15: LOW PASS FILTERING OF INSTANTANEOUS CORRELATOR OUTPUT POWER.....	27
FIGURE 16: CORRELATOR OUTPUT POWER FOR A GPS RECEIVER TRACKING SATELLITE PRN #17. SATELLITE TRACKING STARTS AT TIME = 17 SECONDS.....	28
FIGURE 17: CARRIER PHASE JITTER FOR A GPS RECEIVER TRACKING SATELLITE PRN #17.....	30
FIGURE 18: TWO-BIT ADAPTIVE QUANTIZER (AGC).....	31
FIGURE 19: AGC GAIN VS. AWGN.....	32
FIGURE 20: BANDPASS FILTERING AN IF ₄ SIGNAL (CENTER FREQUENCY = 1.41 MHz).....	39
FIGURE 21: SPREADING OF 50 Hz GPS DATA.....	40
FIGURE 22: NOISE AND INTERFERENCE IN THE GPS SIGNAL.....	41
FIGURE 23: DESPREADING AND DATA RECOVERY.....	42
FIGURE 24: CW INTERFERENCE ON L1 BEFORE AND AFTER SPREADING BY RECEIVER.....	46
FIGURE 25: CW INTERFERENCE ON WORST CASE C/A SPECTRAL LINE BEFORE AND AFTER SPREADING BY RECEIVER.....	47
FIGURE 26: CORRELATOR OUTPUT POWER AND PSEUDORANGE ERROR AS A FUNCTION OF CW INTERFERENCE FREQUENCY OFFSET.....	48
FIGURE 27: DOPPLER OF GPS CONSTELLATION OVER 12 HOURS.....	49
FIGURE 28: DOPPLER RATE OF SATELLITE PRN 14 VS. ELEVATION.....	50
FIGURE 29: VERTICAL POSITION ERROR OF A RECEIVER AS A FUNCTION OF NOISE PLUS INTERFERENCE POWER.....	52
FIGURE 30: INTERFERENCE POWER LEVEL AS A FUNCTION OF AIRCRAFT DISTANCE FROM RUNWAY THRESHOLD.....	54
FIGURE 31: RECEIVED SATELLITE SIGNAL STRENGTH AS A FUNCTION OF SATELLITE ELEVATION – A CURVE FIT BASED ON EMPIRICAL DATA.....	59
FIGURE 32: GPS SIMULATION SIGNAL GENERATION AND DOWNCONVERSION.....	60
FIGURE 33: RF SIGNAL FROM SINGLE GPS SATELLITE AT IF ₁ = 4.31 MHz.....	61
FIGURE 34: COMPOSITE SIGNAL FROM ALL (11) IN-VIEW SATELLITES AT IF ₁ = 4.31 MHz.....	61
FIGURE 35: 8 dB THERMAL NOISE AT IF ₁	61
FIGURE 36: THERMAL NOISE PLUS ALL-IN-VIEW RF SIGNAL PRIOR TO FILTERING.....	61
FIGURE 37: FILTERED IF ₁ SIGNAL SAMPLED AT 5.714 MHz, NOMINAL FREQ. = 1.41 MHz.....	63
FIGURE 38: SAMPLED AND QUANTIZED COMPOSITE IF ₁ SIGNAL.....	63
FIGURE 39: CORRELATION ON SINGLE CHANNEL.....	64
FIGURE 40: BENCH TEST LAYOUT.....	66
FIGURE 41: BENCH TEST HARDWARE SHOWING THE RECEIVER-UNDER-TEST (LEFTMOST), CW BROADBAND AND PULSE GENERATOR (STACKED IN MIDDLE), AND GPS SIGNAL GENERATOR (RIGHT).....	66
FIGURE 42: SAMPLE SIMULATION RUNS: 26 dB AWGN, 18 dB CW INTERFERENCE.....	70

FIGURE 43: CODE TRACKING LOOP ERROR AS FUNCTION OF CONTINUOUS RF INTERFERENCE POWER: PSEUDORANGE ERROR VS. $C/(N_0+I_0)$	73
FIGURE 44: CARRIER TRACKING LOOP ERROR AS FUNCTION OF CONTINUOUS RF INTERFERENCE POWER: MEAN CARRIER FREQUENCY ERROR VS. $C/(N_0+I_0)$	74
FIGURE 45: CODE TRACKING LOOP ERROR AS FUNCTION OF PULSED RF INTERFERENCE POWER: PSEUDORANGE ERROR VS. PULSE DUTY CYCLE	75
FIGURE 46: CARRIER TRACKING LOOP ERROR AS FUNCTION OF PULSED RF INTERFERENCE POWER: MEAN CARRIER FREQUENCY ERROR VS. PULSE DUTY CYCLE	76
FIGURE 47: CODE TRACKING LOOP ERROR AS FUNCTION OF SIGNAL ATTENUATION: PSEUDORANGE ERROR VS. SIGNAL ATTENUATION	77
FIGURE 48: CARRIER TRACKING LOOP ERROR AS FUNCTION OF SIGNAL ATTENUATION: MEAN CARRIER FREQUENCY ERROR VS. SIGNAL ATTENUATION	78
FIGURE 49: IMPACT OF CONTINUOUS RF INTERFERENCE ON TEST STATISTICS.....	79
FIGURE 50: IMPACT OF PULSED RF INTERFERENCE ON TEST STATISTICS	82
FIGURE 51: IMPACT OF SIGNAL ATTENUATION ON TEST STATISTICS	83
FIGURE 52: LOOP CAPTURE WITH STRONG COHERENT CW INTERFERENCE: CORRELATOR OUTPUT POWER VS. CW INTERFERENCE FOR CW AT $L1 + 7$ KHZ. A STRONG SPECTRAL LINE. POINTS IN BOX INDICATE RECEIVER HAS LOST LOCK.....	85
FIGURE 53: LOOP CAPTURE WITH STRONG COHERENT CW AT $L1 + 7$ KHZ. CW INTERFERENCE POWER TO SIGNAL RATIO IS 40 DB FOR A $C/(N_0+I_0)$ OF 15 DB-HZ. (A) PSEUDORANGE ERROR OVER TIME. (B) CORRELATOR OUTPUT POWER OVER TIME.	86
FIGURE 54: TEST STATISTICS DECISION MATRIX SHOWING (A) NEGATIVE SLOPE MATRIX AND (B) POSITIVE SLOPE MATRIX.....	87
FIGURE 55: PSEUDORANGE ERROR AS A FUNCTION OF CORRELATOR OUTPUT POWER FOR AWGN, CW, PULSED AWGN, PULSED CW, SIGNAL ATTENUATION, CW AT $L1 + 1$ KHZ, AND CW AT $L1 + 7$ KHZ.	97
FIGURE 56: PSEUDORANGE ERROR AS A FUNCTION OF CORRELATOR OUTPUT POWER VARIANCE FOR AWGN, CW, PULSED AWGN, PULSED CW, SIGNAL ATTENUATION, CW AT $L1 + 1$ KHZ, AND CW AT $L1 + 7$ KHZ.....	98
FIGURE 57: PSEUDORANGE ERROR AS A FUNCTION OF CARRIER PHASE JITTER FOR AWGN, CW, PULSED AWGN, PULSED CW, SIGNAL ATTENUATION, CW AT $L1 + 1$ KHZ, AND CW AT $L1 + 7$ KHZ	99
FIGURE 58: PSEUDORANGE ERROR AS A FUNCTION OF AGC GAIN FOR AWGN, CW, PULSED AWGN, PULSED CW, SIGNAL ATTENUATION, CW AT $L1 + 1$ KHZ, AND CW AT $L1 + 7$ KHZ	100
FIGURE 59: PSEUDORANGE ERROR AS A FUNCTION OF AGC GAIN FOR A FAST ACTING AGC, FOR AWGN, CW, PULSED AWGN, PULSED CW, SIGNAL ATTENUATION, CW AT $L1 + 1$ KHZ, AND CW AT $L1 + 7$ KHZ.....	101
FIGURE 60: FREQUENCY ERROR AS A FUNCTION OF CORRELATOR OUTPUT POWER FOR AWGN, CW, PULSED AWGN, PULSED CW, SIGNAL ATTENUATION, CW AT $L1 + 1$ KHZ, AND CW AT $L1 + 7$ KHZ	102
FIGURE 61: FREQUENCY ERROR AS A FUNCTION OF CORRELATOR OUTPUT POWER VARIANCE FOR AWGN, CW, PULSED AWGN, PULSED CW, SIGNAL ATTENUATION, CW AT $L1 + 1$ KHZ, AND CW AT $L1 + 7$ KHZ.....	103
FIGURE 62: FREQUENCY ERROR AS A FUNCTION OF CARRIER PHASE JITTER FOR AWGN, CW, PULSED AWGN, PULSED CW, SIGNAL ATTENUATION, CW AT $L1 + 1$ KHZ, AND CW AT $L1 + 7$ KHZ ...	104
FIGURE 63: FREQUENCY ERROR AS A FUNCTION OF AGC GAIN FOR AWGN, CW, PULSED AWGN, PULSED CW, SIGNAL ATTENUATION, CW AT $L1 + 1$ KHZ, AND CW AT $L1 + 7$ KHZ.....	105
FIGURE 64: BENCH TEST RESULTS FOR CORRELATOR OUTPUT POWER VS. $C/(N_0+I_0)$	108
FIGURE 65: BENCH TEST RESULTS FOR CORRELATOR OUTPUT POWER VARIANCE VS. $C/(N_0+I_0)$	109
FIGURE 66: BENCH TEST RESULTS FOR CARRIER PHASE JITTER VS. $C/(N_0+I_0)$	109
FIGURE 67: INTERFERENCE MITIGATION SCENARIO: AN AIRCRAFT ON A 3-DEGREE APPROACH PATH	115
FIGURE 68: VERTICAL ACCURACY OVER 24 HOURS AND OVER 30 DB INTERFERENCE POWER RANGE, FOR A DGPS-ONLY USER AT SAN FRANCISCO INTERNATIONAL AIRPORT. FULL GPS CONSTELLATION OF 24 SATELLITES ARE AVAILABLE	121

FIGURE 69: VERTICAL ACCURACY OVER 24 HOURS AND OVER 30 dB INTERFERENCE POWER RANGE, FOR A 1 APL + GPS USER AT SAN FRANCISCO INTERNATIONAL AIRPORT. FULL GPS CONSTELLATION OF 24 SATELLITES ARE AVAILABLE	122
FIGURE 70: VERTICAL ACCURACY OVER 24 HOURS AND OVER 30 dB INTERFERENCE POWER RANGE, FOR A 2 APLS + GPS USER AT SAN FRANCISCO INTERNATIONAL AIRPORT. FULL GPS CONSTELLATION OF 24 SATELLITES ARE AVAILABLE	123
FIGURE 71: VERTICAL ACCURACY OVER 24 HOURS AND OVER 30 dB INTERFERENCE POWER RANGE, FOR A DGPS-ONLY USER AT SAN FRANCISCO INTERNATIONAL AIRPORT. ANALYSIS ASSUMES A 3 GPS SATELLITE FAILURE, LEAVING ONLY 21 HEALTHY SATELLITES	124
FIGURE 72: VERTICAL ACCURACY OVER 24 HOURS AND OVER 30 dB INTERFERENCE POWER RANGE, FOR A 1 APL + GPS USER AT SAN FRANCISCO INTERNATIONAL AIRPORT. ANALYSIS ASSUMES A 3 GPS SATELLITE FAILURE, LEAVING ONLY 21 HEALTHY SATELLITES	125
FIGURE 73: VERTICAL ACCURACY OVER 24 HOURS AND OVER 30 dB INTERFERENCE POWER RANGE, FOR A 2 APLS + GPS USER AT SAN FRANCISCO INTERNATIONAL AIRPORT. ANALYSIS ASSUMES A 3 GPS SATELLITE FAILURE, LEAVING ONLY 21 HEALTHY SATELLITES	126
FIGURE 74: GPS AVAILABILITY OVER 24 HOURS FOR A USER LOCATED AT SAN FRANCISCO, 24 HEALTHY SATELLITES	127
FIGURE 75: GPS AVAILABILITY OVER 24 HOURS FOR A USER LOCATED AT SAN FRANCISCO, 3 SATELLITE FAILURE (21 HEALTHY SATELLITES)	128
FIGURE 76: PSEUDOLITE/SATELLITE MAXIMUM POWER RATIO VS. PSEUDOLITE ACQUISITION DISTANCE FOR AN AIRCRAFT ON A 3-DEGREE GLIDE SLOPE ASSUMING CLOSEST DISTANCE IS 50 METERS	130
FIGURE 77: NEAR-FAR PROBLEM WITH PSEUDOLITE LOCATED 1KM FROM RUNWAY THRESHOLD.....	131
FIGURE 78: LENGTH 2N SEQUENCES, WITH ONE HAVE ITS SECOND HALF SWITCHED IN SIGN	137
FIGURE 79: AMBIGUITY FUNCTION FOR 2 GOLD CODES OF ORDER 5 (LENGTH 31x2, SWITCH = +/-).....	139
FIGURE 80: FIVE STAGE LINEAR FEEDBACK SHIFT REGISTER.....	139
FIGURE 81: MODIFIED NEAR-FAR PROBLEM WITH PSEUDOLITE 1 KM FROM RUNWAY THRESHOLD.....	143
FIGURE 82: BAC-1-11 RADIATION PATTERN SPHERICAL COVERAGE (COURTESY OF J.I.R. OWEN)	143

Chapter 1

INTRODUCTION

1.1 The Global Positioning System

The Global Positioning System (GPS) is a satellite-based navigation system introduced by the US Department of Defense which enables a user to achieve three dimensional positioning anywhere on the globe by range measurements from orbiting satellites. The position accuracy is between 30 to 100 m for civilian users, and below 10 m for military use. Introduced in 1973, GPS reached full operational status in 1994, and a wide range of military, commercial and civilian uses.

GPS consists of three segments: the space segment, ground control segment and user segment. The space segment is comprised of 24 satellites in 12-hour orbits. Satellites orbit in 6 inclined planes, with 4 satellites per orbit (Figure 1). Each satellite continuously transmits an RF signal at L1 center frequency of 1575.42 MHz containing the C/A (Clear Acquisition) signal - available to civilians, and the P/Y code - currently available only for the military. A second signal containing P/Y code only is also transmitted at L2: 1227.6 MHz.

The L1 C/A signal is a low power spread spectrum signal synthesized by bi-phase shift modulating the carrier frequency with pseudorandom 'noise-like' (PRN) sequences known as Gold codes. Each satellite transmits a unique Gold code of length 1023 chips at a chipping rate of 1023 MHz. Gold codes are nearly orthogonal. The autocorrelation function features a distinct peak of 1023 chips (0 dB) for perfect match or zero chip offset, with a maximum match of 65 chips (-24 dB) for all other offsets. The cross-correlation between any two different codes also produces very low output (-24 dB) compared to the

autocorrelation peak. This property permits *multiple channel access*, the transmission of numerous GPS satellite signals within the same frequency band.

A 50 Hz data stream is bi-phase modulated onto the carrier and Gold code signals. This data stream contains satellite specific messages such as time of transmission, satellite ephemeris and almanac information. It enables a GPS receiver to decode signal measurements into satellite range measurements.

The signal power received from the satellites, approximately -160 dBW, is so low it is not discernible above the noise floor with a spectrum analyzer. This low power nature of the GPS signal makes interference from other RF sources a major challenge. This topic will be discussed further later in this chapter.

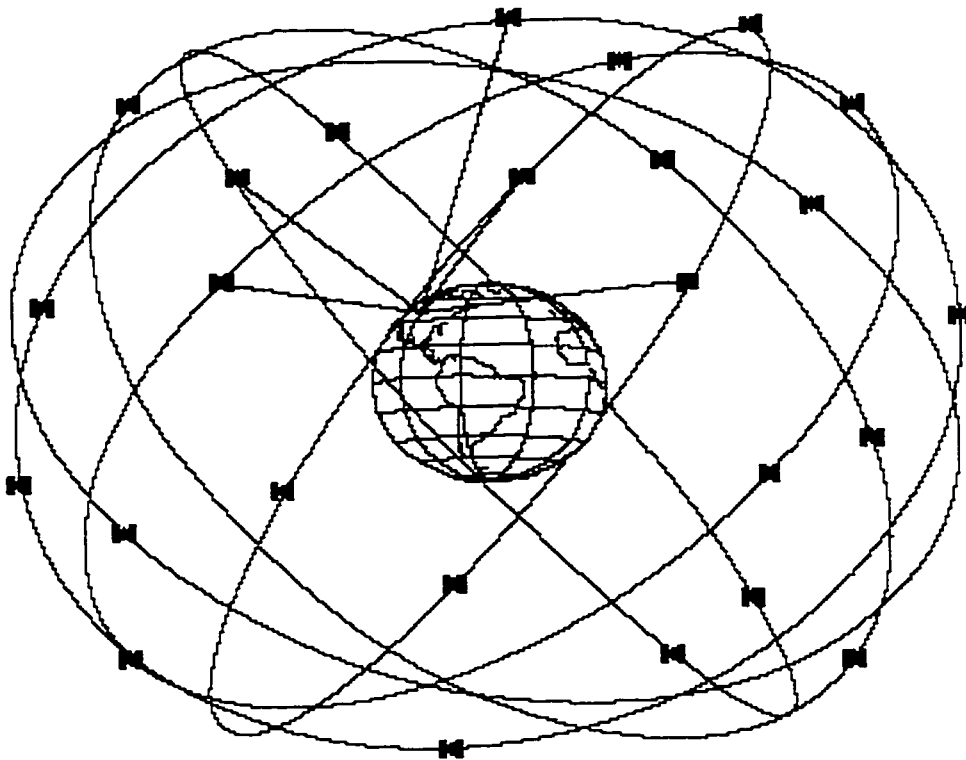


Figure 1: GPS Constellation of 24 satellites

The ground control segment consists of 5 monitor stations located around the globe: Hawaii, Colorado Springs, Ascension Island, Diego Garcia (Indian Ocean) and Kwajalein Island (West Pacific Ocean). The station in Colorado Springs is the Master Control Site, while the other 4 stations serve only as monitor and upload stations. All five stations each contain a number of GPS receivers which monitor the signals transmitted by each satellite in view. These received signals are processed to estimate satellite ephemeris, clock error, health and other satellite parameters. This information is then uploaded to the satellites, usually once a day, and applied by the satellite processor as fine ephemeris corrections in the broadcast 50 Hz data, to maintain GPS system accuracy.

The GPS user segment consists of a wide variety of applications employing GPS to determine position, velocity and timing information to varying degrees of precision. At the core of any GPS application is the GPS receiver, which receives the signals transmitted by the GPS satellites. A GPS receiver determines position by measuring its distance from 4 or more tracked satellites, as shown in Figure 2.

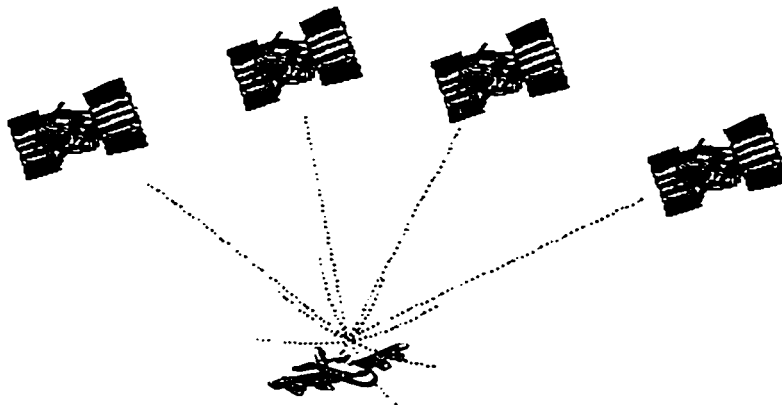


Figure 2: Position Determination Via Four Pseudorange Measurements

Distance from a satellite is computed by multiplying the time of travel of the signal from satellite to receiver by the speed of light. This range measurement is known as

pseudorange.. At least four pseudorange measurements are needed to determine the x-y-z components of a 3 dimensional solution, and to determine the unknown receiver clock bias.

Applications of GPS include aircraft navigation, including enroute, approach and landing phases, marine navigation: inland waterways, coastal and oceanic, land navigation, surveying and geodesy, spacecraft attitude and position, time transfer, and search and rescue systems. Each application places various demands and requirements on the GPS system. These requirements fall into four categories: accuracy, integrity, system availability and continuity of service.

Accuracy is the difference between the estimated and true position [1]. This is also known as the navigation sensor error (NSE). It can be defined in terms of percentages, as the error which is not exceeded over a specified percentage of the time, usually 95% or 99%. End user applications place different requirements on system accuracy, for instance, aircraft en route navigation requires less accuracy, on the order of 2 nautical miles, than aircraft in a terminal area, which requires accuracies on the order of 0.3 nautical miles. Precision approach places even more stringent requirements. Table 1 below shows the 95% accuracy requirements for precision aircraft approaches, with corresponding decision heights at which the pilot must decide whether to complete or abort the landing [2].

Table 1: Accuracy Requirements for Aircraft Precision Approaches

Precision Approach Type	Decision Height	Accuracy Requirement (m)
Category I	200+ ft	
Horizontal		16.5
Vertical		3.4
Category II	100+ ft	
Horizontal		6.5
Vertical		1.6
Category III	0 – 100 ft	
Horizontal		4.1
Vertical		0.5

The least stringent requirements are for Category I approach, with a decision height of 200 feet. Category III precision approach involves autoland with zero visibility, and therefore has the most stringent requirements for accuracy, 0.5 m vertical, 95% of the time. Non-augmented GPS has accuracies on the order of 30 m, which can be improved to better than 2 m using local and wide area augmentation [3]. Using carrier phase measurements, accuracies on the order of 2 centimeters are achievable [19].

Integrity may be defined as a measure of confidence that the system is indeed functioning as predicted. Integrity is the ability of a system to provide timely warning to users when the system should not be used for navigation [4]. A system 'with integrity' is one that indeed performs up to its stated specifications, and issues a warning when specifications are not being met. For example for an aircraft on a Category III precision approach the navigation system is required to report, within 2 seconds, any vertical errors exceeding 2.5m. The probability of failing to detect can be no greater than 10^{-9} .

Integrity is usually characterized by four parameters:

- i. Protection level: the maximum tolerable position error, beyond which the system must issue a timely warning. Vertical or horizontal protection level (VPL or HPL) is the maximum tolerable error in the vertical or horizontal direction.
- ii. Time to alarm: the maximum time permitted from the time the protection level is exceeded to the sounding of the corresponding alarm.
- iii. Missed detection (MD) probability: the probability of an undetected failure, or:

$$\Pr[\text{MD}] = \Pr[\text{no alarm} \mid \text{error} > \text{protection level}] \quad (1.1.1)$$

- iv. False alarm (FA) probability: the probability that an alarm was sounded when there really was no system failure, or:

$$\Pr[\text{FA}] = \Pr[\text{alarm} \mid \text{error} < \text{protection level}] \quad (1.1.2)$$

Integrity requirements for GPS end users vary depending on the application and on what is at stake. The consequences of an unreported 5 mile error for a ocean-going vessel during a transoceanic voyage are likely less severe than for a 5 meter unreported error for an aircraft on a Cat III precision approach and landing phase. Table 2 shows tentative integrity requirements for various aircraft precision approaches [5].

Table 2: Integrity Requirements for Aircraft Precision Approaches

Approach Category	Protection Level Direction	Protection Level (m)	Allowable duration out of protection level (s)	Probability of undetected error per landing
CAT I	Lateral	13.5	10	1×10^{-7}
	Vertical	4.8	6	1×10^{-7}
	Total			2×10^{-7}
Cat II	Lateral	8.2	5	1×10^{-7}
	Vertical	2.3	2	1×10^{-7}
	Total			2×10^{-7}
Cat III	Lateral	6.1	2	0.5×10^{-9}
	Vertical	2.3	2	0.5×10^{-9}
	Total			1×10^{-9}

Consequences of missed detections are generally more severe than false alarms, and therefore have more stringent specifications. This thesis focuses on methods to ensure integrity of GPS via monitoring of received signal quality.

Continuity of service is the probability that the system functions throughout an operation, given that it was available at the beginning of the operation. Continuity is an important

requirement during critical phases of aircraft landing, when signal interruptions could result in an aborted approach.

The *availability* of a system is the percentage of time that the services of the system are usable with the specified accuracy, integrity and continuity [4]. For aircraft terminal area navigation the FAA requires an availability of 99.999% [5]. GPS is able to meet availability requirements through the use of ground and space based augmentation.

1.2 The Interference Problem

Radio frequency interference (RFI) are man-made signals that interfere with the ability of a receiver to track the GPS signal. Noise can be similarly defined as RF interference, however noise is caused by natural causes. For example thermal noise is the result of the natural effect of heat energy on electrons. Common sources of interference include radar, FM and TV harmonics, mobile cellular, and ham radio.

Interference impacts all aspects of GPS. Accuracy degrades as interference levels increase, until a certain level where the interferer jams the GPS receiver, causing a loss in continuity and availability. More severe than loss of continuity or availability is the degradation in integrity. When interference degrades GPS accuracies beyond specified protection levels unbeknownst to the user, a breach in integrity results, with potential severe consequences. Cruise vessels have run aground when navigation systems failed in integrity¹. The consequences for aircraft approach and landing are even more disastrous.

Given the low power nature of the GPS signal it is easy to see why interference would be a significant issue: any RF signal of even moderate power within the GPS frequency band

¹On June 10, 1995, the Panamanian cruise vessel ROYAL MAJESTY ran aground 10 miles off Nantucket Island, MA, after its primary navigation sensor GPS suffered an unnoticed fault line (antenna) failure [26].

would constitute interference. Figure 3 shows the effective range of a CW jammer as a function of jamming power. It is assumed that the jamming frequency lies within GPS L1 band, and that the receiver can no longer track GPS when the interference to GPS signal power ratio (J/S) exceeds 25dB, an average based on results discussed in chapter 5.

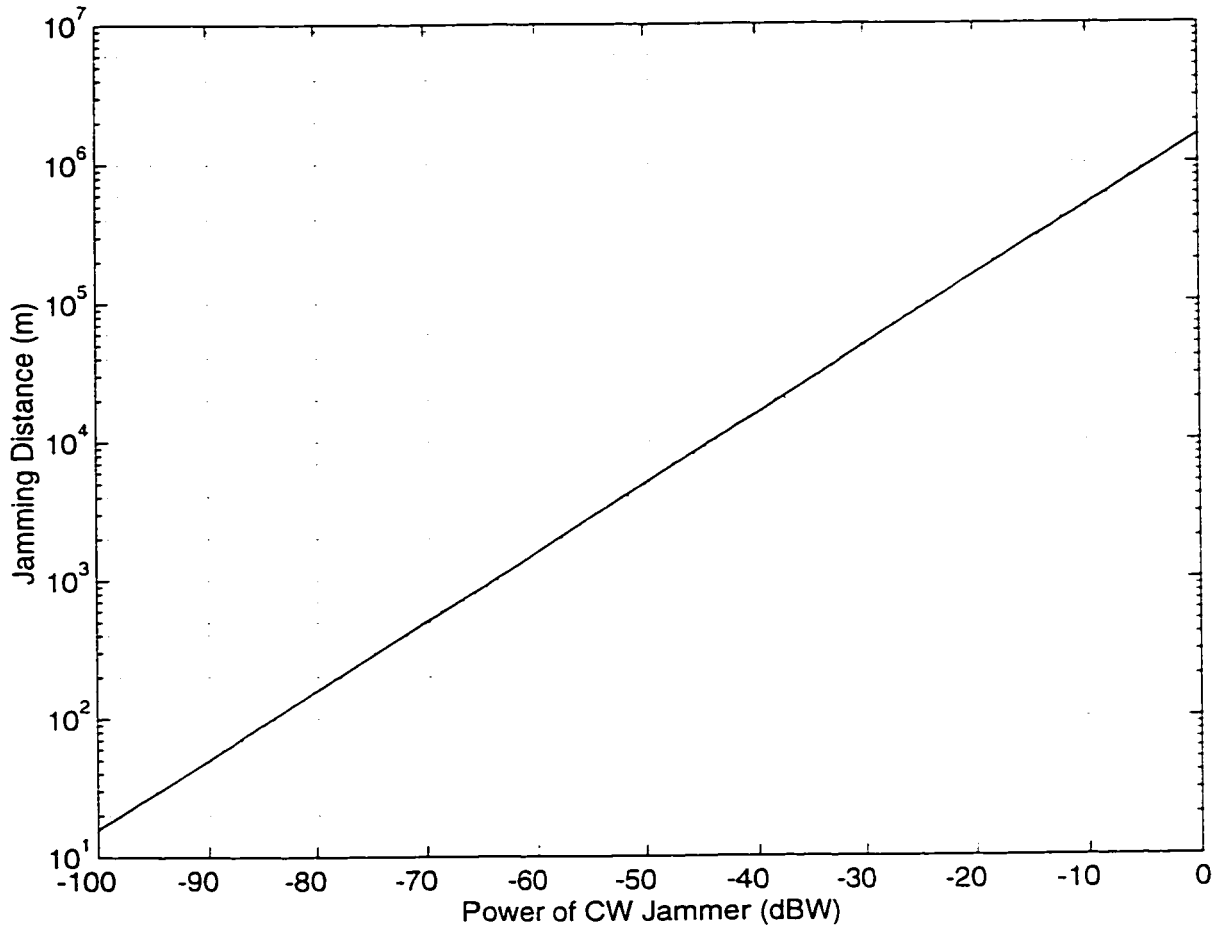


Figure 3: Effective Jamming Distance as a function of Jammer Power

This figure shows that a suitably located 1 watt CW jammer has the potential to disable GPS operations in an area of radius 1.000 km. Of course Figure 3 assumes $1/R^2$ propagation and so it overbounds the effective range of a terrestrial jammer.

1.3 Integrity Monitoring

Autonomous integrity monitoring allows the receiver to detect the presence of interference, before hazardous misleading information (HMI) results. Three approaches to integrity monitoring have been considered:

i. Parity checks

Stand alone GPS provides some level of integrity information in its transmitted data message in the form of parity bits, which enable a receiver to check for errors in the data stream. Parity checks are necessary, however not sufficient, as parity information alone may not be timely for certain applications.

ii. Ground based integrity monitoring:

This consists of ground based monitor stations that measure signal quality based on known surveyed locations. System errors are detected, isolated and reported via broadcast media to users in the vicinity of the monitor station. Local area augmentation systems (LAAS) employ a form of ground monitors to meet accuracy and integrity requirements. While ground based monitoring is important, it is not sufficient to detect all modes of failures, especially since certain interference scenarios present onboard an aircraft may be invisible to a ground based monitor. Examples of such localized interference include RF harmonics from onboard electronics and personal communication devices.

iii. Receiver Autonomous Interference Monitoring (RAIM):

RAIM attempts to detect and isolate failed satellites based on monitoring solution residuals [7]. Position solutions are computed using different sets of satellites, and compared, to detect satellites with degraded signals. A number of different schemes have evolved, including range comparison and least squares-residuals method [8]. While RAIM does indeed colocate the integrity scheme with the receiver, it requires an over-specified

navigation solution, and does not leverage more sensitive measures of signal quality.

Neither parity checking, RAIM nor ground monitoring can always detect a small but perhaps dangerous error.

1.4 Thesis Approach and Goals

This research focuses on autonomous integrity monitoring using a different approach than either of the previous discussed methods. We propose a novel method of integrity monitoring based on observation of fundamental GPS receiver measurements of AGC gain, I/Q correlator output and derived quantities. Figure 4 shows a simple schematic of a GPS receiver, including ground based and residual based integrity monitoring (the monitor station and navigation filter in this figure). Our approach focuses on signal quality monitoring via correlator-level measurements.

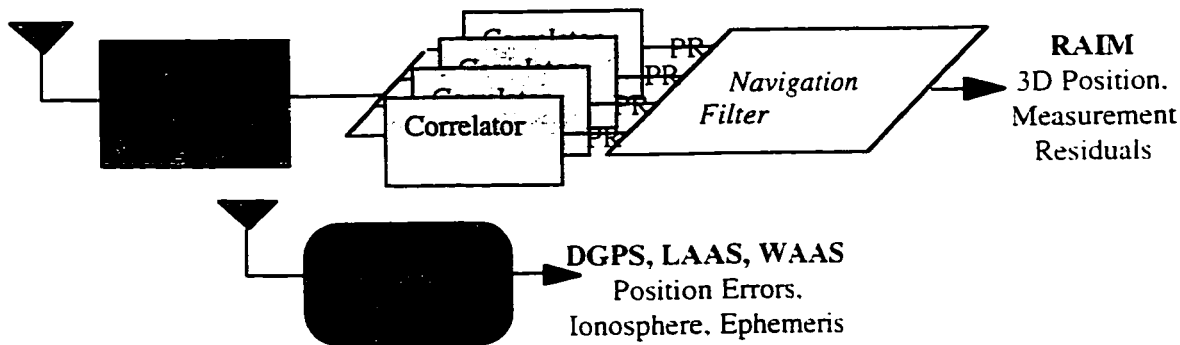


Figure 4: Integrity Monitoring Approaches

We have studied the effects of interference on pseudorange error and on the raw receiver measurements: adaptive A-D quantizer thresholds (AGC gain), correlator output power, variance of correlator output power, and carrier phase jitter. Based on this study we have observed the relationship between pseudorange accuracy degradation as caused by interference and changes in the observed receiver measurements.

We demonstrate the effectiveness of these proposed parameters as integrity monitor test statistics. Integrity monitoring is accomplished by using a simple decision table:

If Test Statistic(i) is greater than some set threshold,
 then sound alert,
 Else indicate all-is-OK.

This is shown in Figure 5.

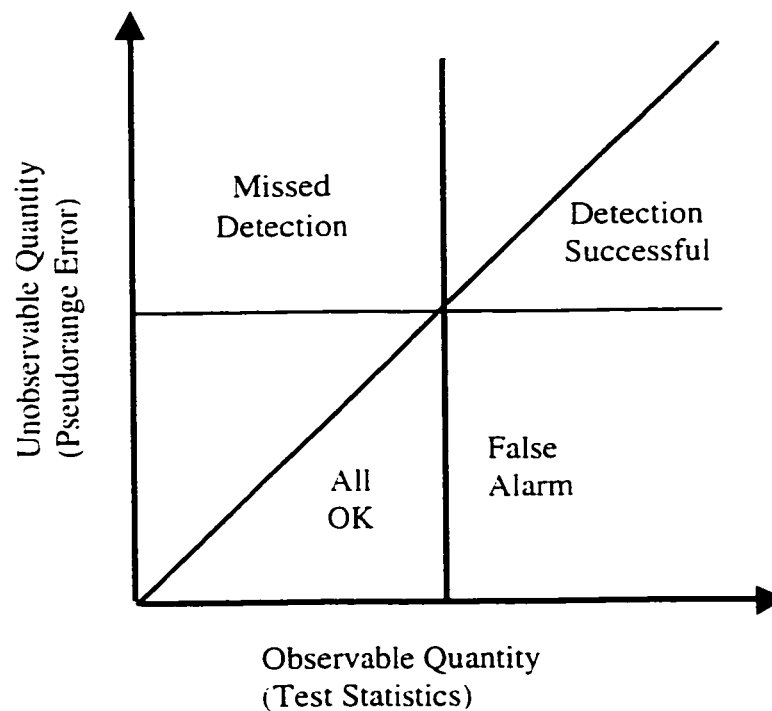


Figure 5: Test Statistics Decision Matrix

The upper half of Figure 5 corresponds to the incidents when the unobservable (or protected) parameter, pseudorange error, is degraded beyond the specified alarm limit. The right half corresponds to when the selected test statistic degrades beyond its specified threshold. Successful detection occurs when test statistic degrades beyond a set threshold

at the same time as when pseudorange error exceeds the protection level, as indicated by the top right corner of Figure 5. Missed detection occurs when pseudorange error exceeds the protection level, while the test statistics remain below the set threshold, corresponding to the top left corner of Figure 5. On the other hand, false alarms occur when pseudorange error remains below the protection level while our test statistics exceed the set threshold, as shown by the bottom right corner of Figure 5.

Our selected test statistics aim to minimize incidents of missed detection, while also keeping low the incidents of false alarm. To meet this specification, the test statistics must be sensitive to interference, and respond to changing interference power levels. However at the same time it is also important that sensitivity to variations in *kinds* of interference is minimized. This is necessary in order to ensure robustness of our test statistics, ensuring reliable detection of accuracy degradation independent of the type of interference causing it.

We demonstrate the good performance and robustness of our selected test statistics in the face of different types of interference, including CW, AWGN, pulsed interference, and signal attenuation.

We also propose solutions to counter degradation in GPS availability, continuity and accuracy through the use of pseudolites, ground based satellites, to augment the GPS constellation. Pseudolites broadcast a GPS-like signal at the same frequency as, and synchronized to the GPS satellites. As a result pseudolite signals can easily be acquired and tracked by GPS receivers with little modification to receiver firmware. The additional pseudorange measurements augment the accuracy of the navigation solution.

Pseudolites transmit a strong signal which is less susceptible to interference due to its superior signal power, as shown in Figure 6.

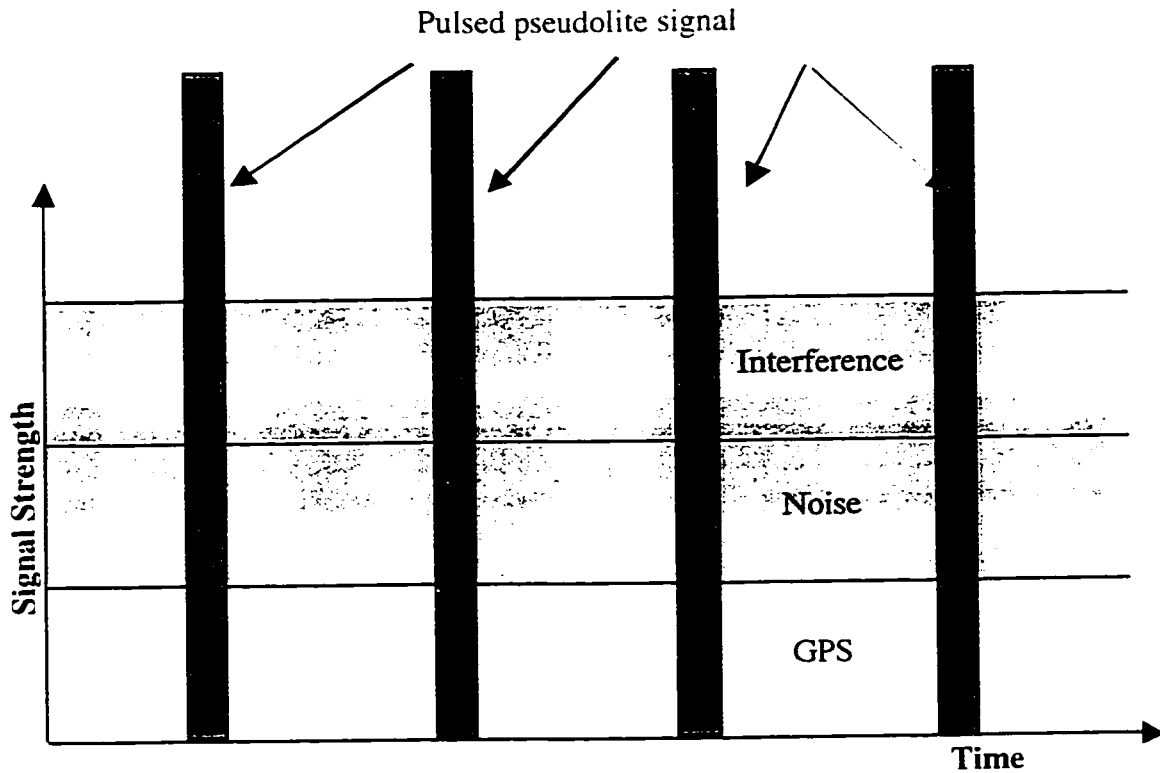


Figure 6: Pseudolite Power and Pulsing Scheme

We demonstrate the effectiveness of ground based satellites in mitigating the effects of interference to GPS availability.

The added immunity to interference comes at a cost: the pseudolites themselves may present interference to GPS. However this interference can be minimized by careful design of the pseudolite signal. To further reduce interference to GPS caused by pseudolites, we propose a signal design for pseudolites based on pulsed signals and faster codes.

1.5 Research Contributions

- We have developed algorithms to autonomously monitor the quality of received GPS signals. Correlator output power is chosen as our primary test statistic, combined with

GPS data parity checks to detect loop capture. Variance of correlator output power, carrier phase jitter and AGC gain provide redundancy and added robustness, with AGC gain providing the added ability to auto-detect types of interference:

- We have demonstrated the robustness of these algorithms to multipath and signal fading;
- The performance of proposed algorithms were verified via software simulation and bench testing;
- We have established the role of airport-located pseudolites (APLs) in mitigating interference: we show that APLs, due to their greater signal strength, are robust to interference and improve GPS system availability even in the presence of interference;
- We also show that APLs improve GPS accuracy and integrity;
- We also demonstrate that APLs provide robustness against satellite failures;
- We propose signal designs for pseudolites to reduce pseudolite-to-GPS and pseudolite-to-pseudolite interference. One such proposed signal for longer, faster pulsed codes, is currently adopted by RTCA as a standard for pseudolite signals.

1.6 Readers Guide

Chapter 2 presents an overview of GPS receiver operation, introducing our candidate integrity monitor statistics. Chapter 3 discusses noise and interference modeling, and also presents an analysis of coherent CW interference, the most severe form of GPS RF interference. The software simulation used for this study is discussed in Chapter 4. Also presented in this chapter are the bench test setup, and the test strategy for both software and bench testing. Chapter 5 presents results showing effect of interference on pseudorange and on the selected test statistics. The performance of the selected parameters as integrity monitor test statistics is also reported in chapter 5. Results in this chapter include both simulation and bench tests. Chapter 6 presents analyses which demonstrate the performance of APLs in mitigating interference. Pseudolite signal designs are also presented. Conclusions and future work are discussed in chapter 7.

Chapter 2

CANDIDATE INTEGRITY MONITOR STATISTICS

2.1 Introduction

The goal of integrity monitoring is to provide a timely early warning to the user whenever the navigation system accuracy degrades beyond a predefined threshold. In practice the accuracy of any navigation system cannot be observed directly, except for situations where true position is known ahead of time or via some other means, for example a surveyed ground monitor station or an aircraft on an approach being tracked simultaneously by laser. It is therefore the objective of integrity monitoring to identify observable parameters that are similarly effected by interference, and that provide a good indication of accuracy degradation. These parameters constitute our integrity monitor statistics. For this study, the unobservable GPS output to be protected is the pseudorange measurements from receiver to satellite, which is used to compute position. Our candidate integrity monitor statistics are fundamental GPS receiver measurement quantities: adaptive A-D quantizer thresholds (AGC gain), correlator output power, variance of correlator output power, and carrier phase jitter. These parameters are defined further in this chapter. By observing the effect of interference on our selected test statistics and the correlation with true pseudorange error, we protect specified levels of GPS accuracy, minimizing rates of false alarm and missed detections.

This chapter begins with a description of the GEC Plessey GPS Card, a GPS receiver, with a goal of clearly defining the meaning of selected candidate test statistics. The GEC Plessey Receiver was used in this work for bench tests and as a template for the software simulation, due to its open architecture, and ease of access to receiver measurements. Each statistic is discussed in the second section of this chapter.

2.2 GPS Receiver Overview

The GPS receiver tracks signals transmitted by the GPS constellation of satellites, extracting range measurements and other data to compute position, velocity and time fixes. Figure 7, a schematic of a generic GPS receiver, identifies five main parts of any GPS receiver: signal reception by antenna and amplification, down conversion, analog to digital conversion, correlation, and code/carrier tracking. These parts are discussed in the following sections.

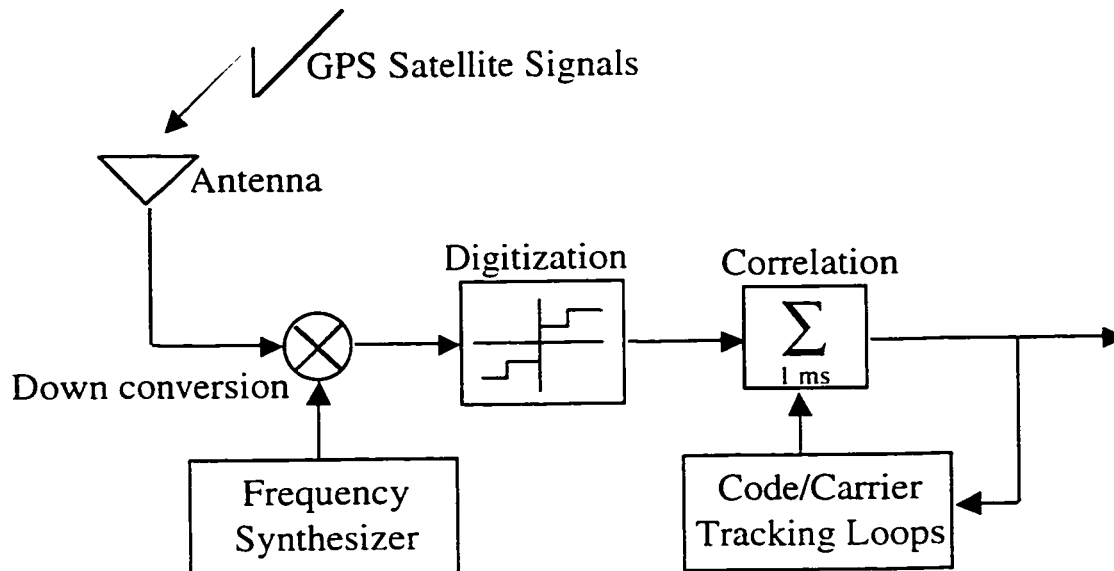


Figure 7: Schematic of a Generic GPS Receiver

2.2.1 Antenna / Low Noise Amplifier

The RF front end of the receiver includes the antenna which receives the GPS satellite signals and passes it through filtering and amplification. The GPS satellite radio frequency signals are converted by the antenna to an analog electrical signal. This electrical signal contains wideband noise and interference, from natural and manmade RFI sources. A low noise amplifier (LNA) is usually used to boost receiver signal strength. The LNA reduces noise by amplifying only the L1 GPS frequency band.

2.2.2 Down Conversion

The RF signal is then downconverted to an intermediate frequency (IF) by mixing with a local oscillator (LO). The GEC Plessey card employs a 5-stage downconversion process. 3 stages of which are analog, occurring in the receiver RF front end (Figure 8). The 1575.42 MHz GPS signal is first mixed with a carrier at a frequency of 1400 MHz, which downconverts it to an intermediate frequency (IF₁) of 175.42 MHz, according to the equation:

$$2 \cos(2\pi f_1 t) \times x(t)D(t) \cos(2\pi f_2 t) = x(t)D(t) [\cos(2\pi(f_1 + f_2)t) + \cos(2\pi(f_1 - f_2)t)]$$

which after bandpass filtering $= x(t)D(t) \cos(2\pi(f_1 - f_2)t)$

(2.2.2.1)

where f_1 = LO frequency 1400 MHz:

f_2 = GPS L1 frequency 1575.42 MHz:

$IF_1 = f_1 - f_2 = 175.42$ MHz:

$D(t)$ is the C/A code; $D(t) = \pm 1$:

and $x(t)$ is the 50 Hz data stream: $x(t) = \pm 1$.

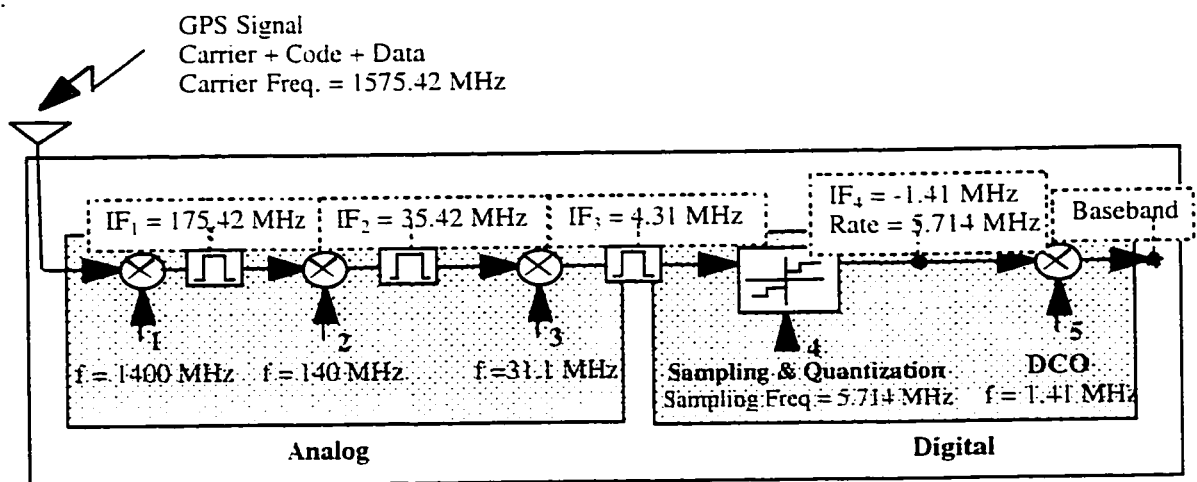


Figure 8: 5-Stage Down Conversion

IF_1 is again mixed with a carrier at 140 MHz to produce IF_2 at 35.42 MHz. Finally IF_2 is mixed down to IF_3 at 4.31 MHz via a carrier at 31.1 MHz. Further downconversion occurs digitally, and is discussed below. Outputs of each analog downconversion are filtered as shown in Figure 8 to eliminate the higher frequency signal which results from mixing

2.2.3 Analog to Digital Conversion

An analog to digital converter is used by the GEC Plessey receiver to sample the IF_3 signal (4.31 MHz) with a sampling rate of 5.714 MHz, which results in a downconverted IF_4 signal at -1.41 MHz (phase reversal), having 5.714×10^6 samples per second (Figure 9). It is important to be aware of this phase reversal as tracking loop corrections for Doppler frequency shifts will have to take it into account.

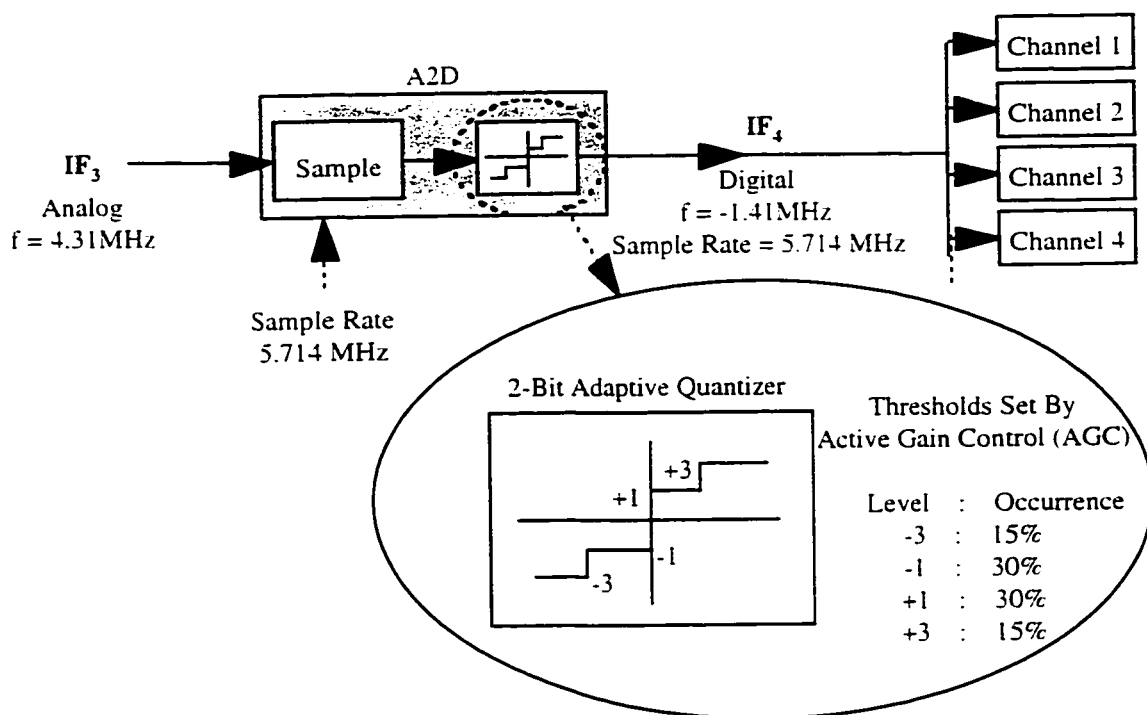


Figure 9: GPS Receiver Digitization

Quantization in the GEC Plessey receiver is performed by a 2-bit adaptive A-D quantizer (Figure 9), which produces 4 output levels, represented by +1, -1, +3 and -3. Quantizer thresholds are continuously adjusted by an active gain controller (AGC) every 0.0005 second to maintain the digital signal output levels -3:-1:+1:+3 at the ratios 15%, 30%, 30% and 15% respectively.

2.2.4 Correlation

The output signal from the digitization stage, IF_4 , contains signals from all GPS satellites in view, each signal having a unique pseudorandom sequence or Gold code. The almost orthogonal properties of Gold codes permit multiple signals to share the same frequency band. To decode each satellite signal the receiver employs a bank of correlators (shown in Figure 9), each correlator dedicated to decoding a single satellite². The correlator matches the incoming IF_4 signal with a locally generated copy in order to determine signal travel time, and thus pseudorange.

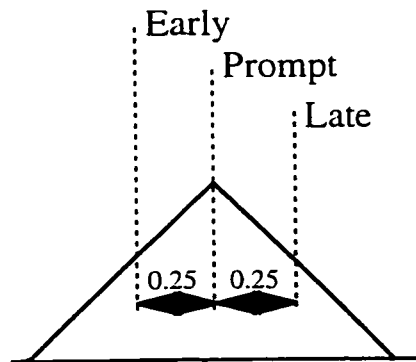


Figure 10: Correlation Peak

For each correlator, early and late replicas of the GPS satellite signal being tracked are generated with a carrier frequency of 1.41 MHz, at a sample rate of 5.714 MHz (same as IF_4). In general, the spacing for early / late could be anywhere from half a Gold code chip ($0.5 \times 1\mu s$) to a tenth of a chip ($0.1 \times 1\mu s$) away from the prompt signal. The GEC

²In certain receivers a single channel may be used to track multiple satellites via multiplexing.

Plessey receiver uses a 0.5 chip correlator spacing (Figure 10). The amplitude of the locally generated signal is evenly distributed between -1, +1, -2 and +2, with a full cycle represented by the sequence: +2 +2 +1 -1 -2 -2 -1 +1.

An inphase signal (I) and a quadrature signal (Q), which is shifted in carrier phase by -90 degrees relative to the inphase, as shown in Figure 11, are generated for both the early and late local signals.

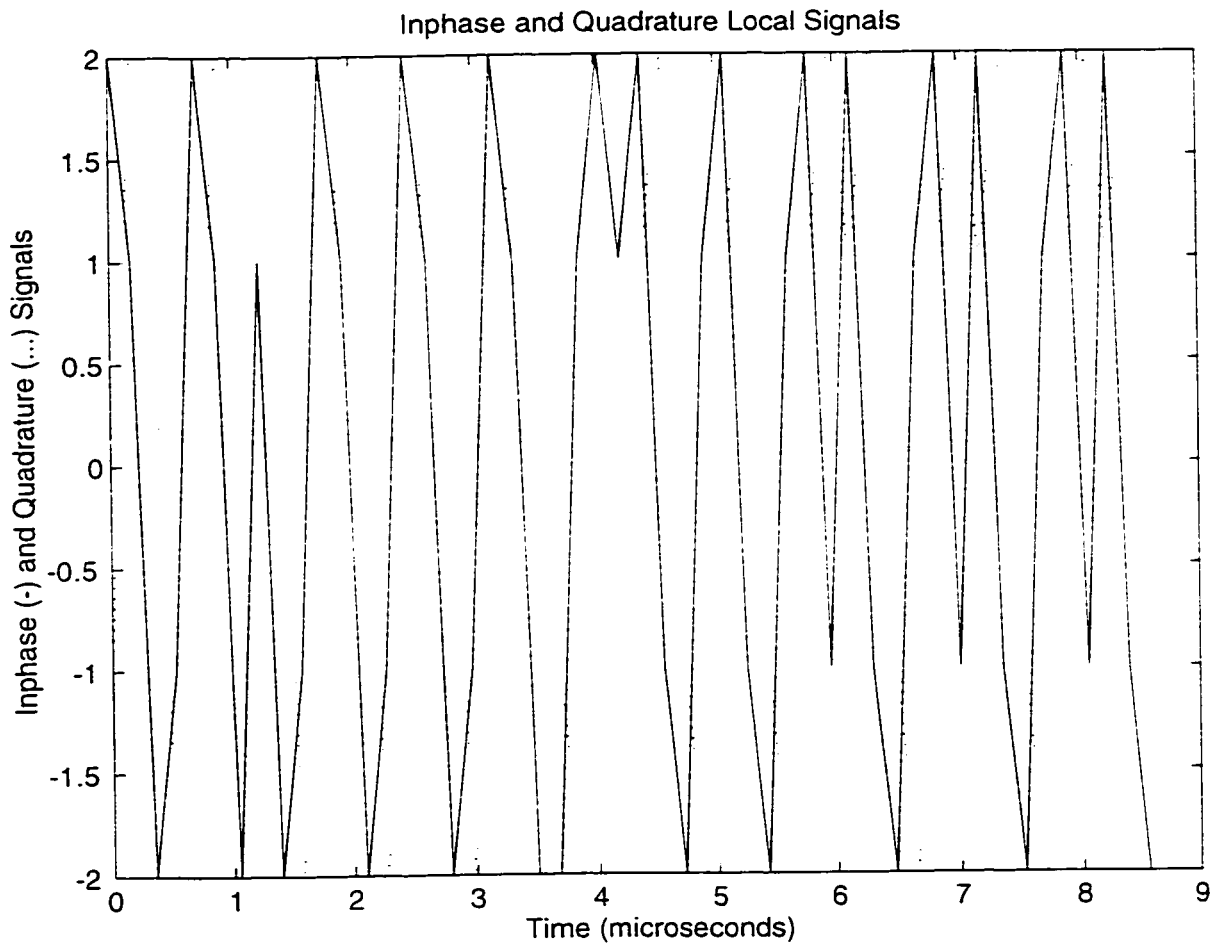


Figure 11: Inphase and Quadrature Components of the Early Local Signal

All four local signals are mixed with the incoming GPS signal, and the output is integrated and dumped by 1 ms accumulators, as shown in the schematic of a single correlator

channel in Figure 12.

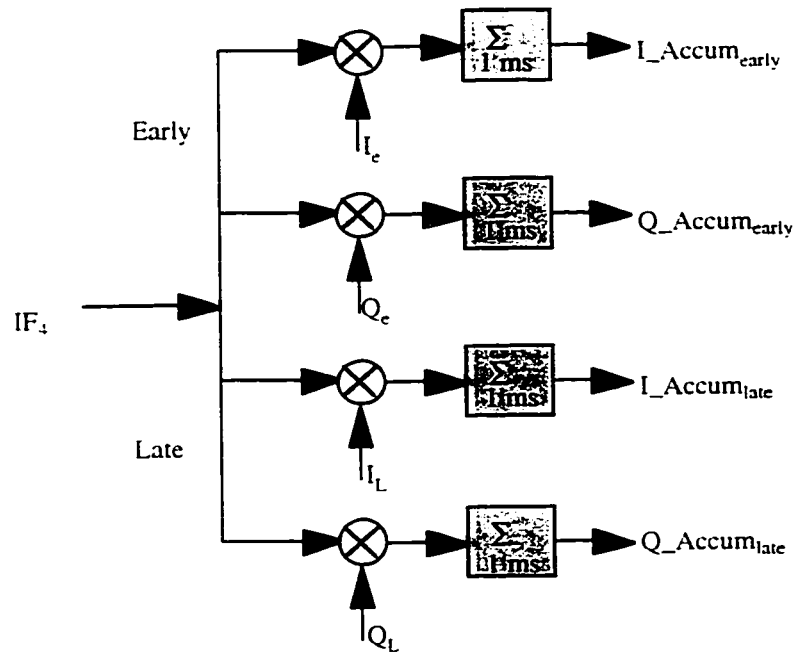


Figure 12: Schematic of a Single Correlator Channel

The resulting output signals shall be denoted as I_Accum_{early} , Q_Accum_{early} , I_Accum_{late} , and Q_Accum_{late} . A prompt signal is also generated in certain receivers, however, the GEC Plessey receiver uses a virtual prompt channel, computed from the average of the early and late:

$$\begin{aligned}
 I_Accum_{prompt} &= (I_Accum_{early} + I_Accum_{late}) / 2 \\
 Q_Accum_{prompt} &= (Q_Accum_{early} + Q_Accum_{late}) / 2
 \end{aligned}
 \tag{2.2.4.1}$$

2.2.5 Acquisition

To track the signal from a satellite the receiver must first search for and acquire it. It does this by generating a local C/A code slightly faster than expected from the incoming signal. This result in a sliding autocorrelation peak search. The carrier frequency of the local

signal is shifted in 500 Hz steps to search across all possible satellite Doppler frequency offsets. Signal acquisition is achieved when the receiver aligns its local replica signal in time with the incoming signal. This results in large accumulator power outputs, beyond a specified *detection* threshold - 6 dB in the GEC Plessey receiver. Once the receiver has locked onto an incoming satellite signal, the local C/A code is adjusted to the correct C/A code speed as determined by the Doppler frequency of the signal, and the receiver's code and carrier tracking loops attempt to maintain lock (Figure 13).

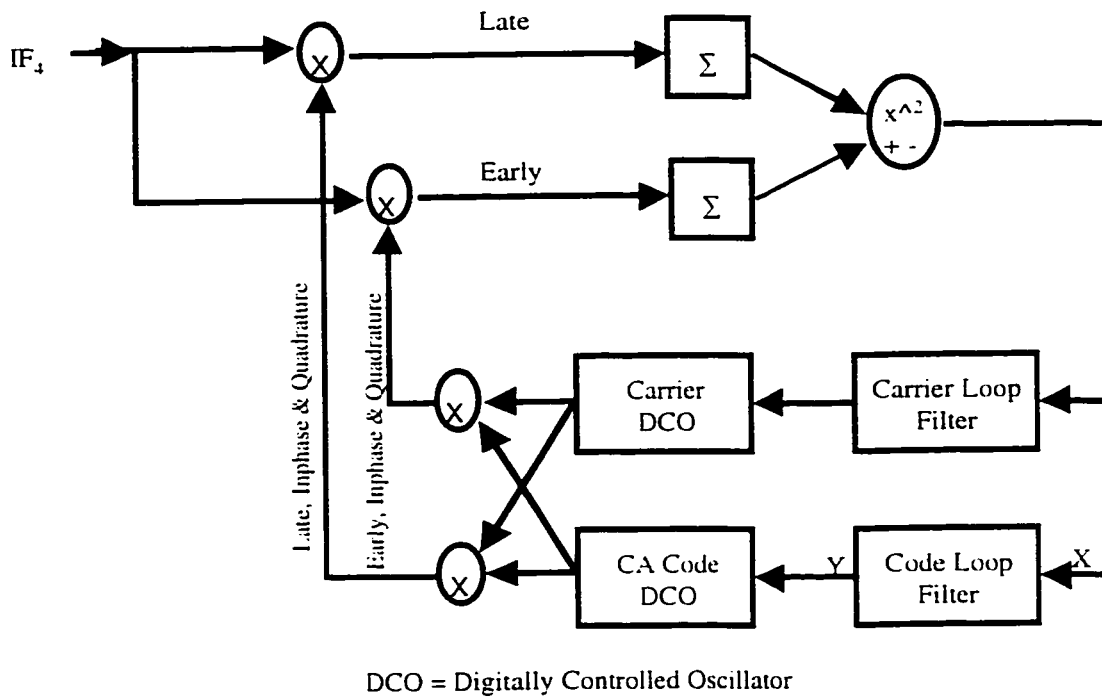


Figure 13: Code and Carrier Tracking Loops

2.2.6 Code Tracking

The code tracking loop maintains lock on the GPS C/A code, from which pseudorange measurements are made. Generally the code tracking loop is a second order delay lock loop (DLL) with the following open loop transfer function [9]:

$$\frac{Y(s)}{X(s)} = G(s) = \frac{T_2s + 1}{T_1s} \quad (2.2.6.1)$$

where $X(s)$ is the input to the code loop filter (Figure 14).

$Y(s)$ is the code loop filter output, and

T_1 and T_2 are time constants which determine loop response.

Stability requires $T_1 > T_2 > 0$, and the loop natural frequency is given by:

$$\omega_n = \sqrt{K_\phi + K_o / T_1} \quad (2.2.6.2)$$

where K_ϕ and K_o are the phase error detector gains and digitally controlled oscillator (DCO) gain respectively, as shown in Figure 14.

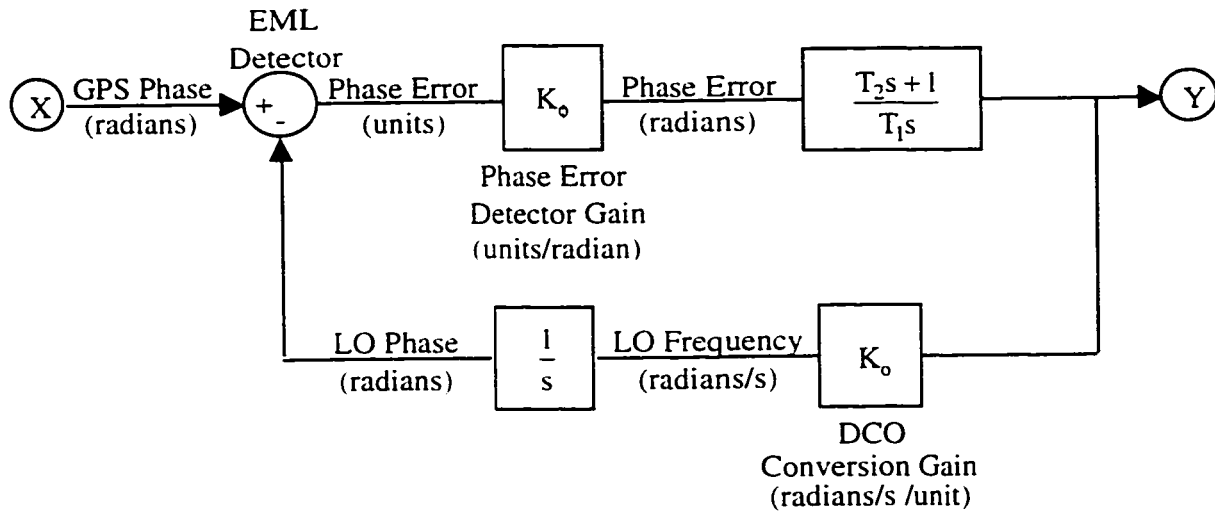


Figure 14: Second Order Delay Locked Loop

K_ϕ converts numerical outputs from the phase error detector into units of radians. For a 1/4 chip correlator, it can be shown that $K_\phi = 928379$ units per radian [9]. K_o is determined by the manufacturer of the DCO. For our studies $K_o = 0.2675$ radians/s per control unit [9].

The damping factor is given by:

$$\zeta = T_2 / 2 \times \omega_n \quad (2.2.6.3)$$

The phase detector used for the code tracking loop error signal is the output of the early-minus-late detector:

$$EML = (I_E^2 + Q_E^2) - (I_L^2 + Q_L^2) \quad (2.2.6.4)$$

To implement the code tracking loop (2.2.6.1) in software the following transformations are made:

$$Y(s) \times T_1 s = X(s)(T_2 s + 1) \quad (2.2.6.5)$$

which, in time domain becomes:

$$T_1 \frac{dy}{dt} = T_2 \frac{dx}{dt} + x \quad (2.2.6.6)$$

Using backwards difference to compute the derivatives, equation (2.2.6.6) can now be calculated recursively as:

$$y_i = y_{i-1} + \frac{T_2}{T_1} (x_i - x_{i-1}) + x_i (\Delta T) \quad (2.2.6.7)$$

2.2.7 Carrier Tracking

The carrier tracking loop is used to keep the locally generated carrier synchronized to the incoming carrier. Carrier loop closure can occur only after code loop closure, otherwise the signal will be too weak. Carrier tracking commonly employs a second order phase

locked loop (PLL). PLLs are suitable for low dynamics applications, but are relatively sensitive to interference. For higher dynamics and high interference environments, frequency locked loops offer better performance [10]. The GEC receiver and the receiver software simulation used in this study employ a second order discrete Jaffe-Rechtin frequency locked loop (FLL) [9]. The frequency error is computed from comparing inphase and quadrature measurements on the prompt channel from successive accumulations:

$$\text{Frequency Error} = Q_k I_{k-1} - I_k Q_{k-1} \quad (2.2.7.1)$$

Initial coarse carrier tracking is achieved using a 4-quadrant frequency discriminator, which rapidly reduces frequency errors from up to 300 Hz down to 10 Hz. This discriminator is used subsequently whenever large phase rate changes are detected, to rapidly reduce the frequency errors. The following corrections, computed from successive prompt channel accumulator outputs, are applied by the 4-quadrant discriminator:

$$\begin{aligned} \Delta I &= I_k - I_{k-1} \\ \Delta Q &= Q_k - Q_{k-1} \end{aligned} \quad (2.2.7.2)$$

These corrections are then applied based on which direction the carrier phasor is drifting, as:

$$\begin{aligned} &\text{IF } |I_k| > |Q_k| \\ &\quad \text{IF } I_k > 0 \\ &\quad \quad \text{correction} = \Delta Q \\ &\quad \text{ELSE} \\ &\quad \quad \text{correction} = -\Delta Q \\ &\text{ELSE} \\ &\quad \text{IF } Q_k > 0 \\ &\quad \quad \text{correction} = -\Delta I \\ &\quad \text{ELSE} \\ &\quad \quad \text{correction} = \Delta I \end{aligned} \quad (2.2.7.3)$$

2.3 Candidate Test Statistics

The following sections define our selected candidate signal quality monitoring test statistics.

2.3.1 Correlator Output Power

Correlator Output Power (COP) is the measured power output from the (virtual) prompt correlator channel, divided by the expected thermal noise floor. The thermal noise floor can be derived by computing the expected COP when there is no signal present:

From section 2.2.2:

Input signal = ± 1 70% of the time
 ± 3 30% of the time

The action of the adaptive A-D quantizer ensures that the magnitude of input signal is always distributed according to this ratio.

From section 2.2.4:

Local signal (I or Q) over one cycle: +2 +2 +1 -1 -2 -2 -1 +1

Computing the product of local and incoming signals, for a random input signal (when no GPS signals are present):

+3 multiplied by the local signal would produce the sequence: +6 +6 +3 -6 -6 -3 +3

mean square value = 22.5

+1 would produce the sequence: +2 +2 +1 -2 -2 -1 +1

mean square value = 2.5

Therefore over a 1 ms accumulation period, at a sample rate of 5.714 MHz, the expected power out on either I or Q channels is given by:

$$\begin{aligned} I^2 &= [0.3 \cdot 22.5 + 0.7 \cdot 2.5] \cdot 5.714 \times 10^3 \\ &= 48571 \end{aligned}$$

Therefore, the expected noise floor when no signal is present is given by:

$$I^2 + Q^2 = 97142 \quad (2.3.1.1)$$

Instantaneous COP ($COP_{\text{instantaneous}}$) is therefore given as

$$\text{Correlator Power Output}_{\text{instantaneous}} = \frac{I_p^2 + Q_p^2}{97142} \quad (2.3.1.2)$$

where I_p and Q_p are the prompt inphase and quadrature accumulator outputs respectively. The quantity, COP, used as a test statistic in this thesis, is a low-pass filtered version of $COP_{\text{instantaneous}}$ obtained as a running average maintained over the most recent 256 samples, equivalent to 0.25 seconds. This process is shown in Figure 15.

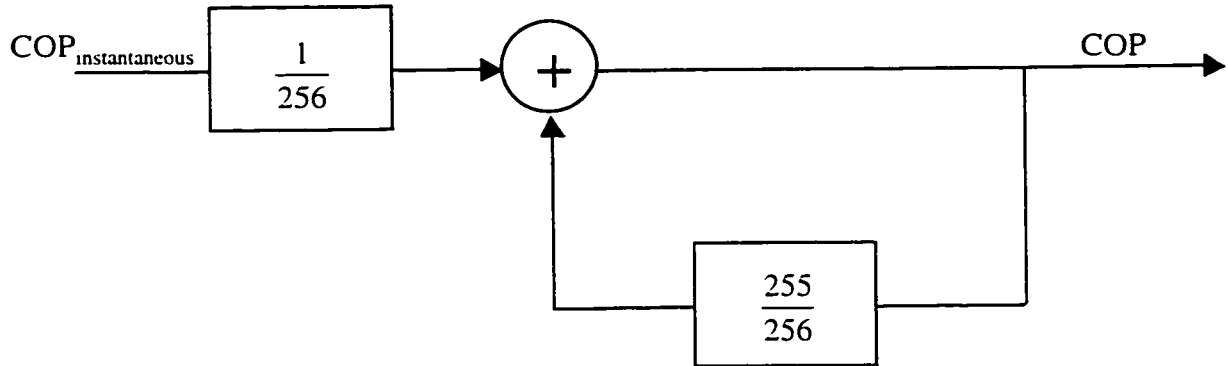


Figure 15: Low Pass Filtering of Instantaneous Correlator Output Power

The instantaneous COP of a real GPS receiver before and after acquisition of satellite PRN 17 is displayed in Figure 16. It can be clearly observed that COP jumps from an average value below zero to about 14 dB relative to the noise floor (equation 2.3.1.1), at the point of signal acquisition. We observe that the value of COP during tracking is an indicator of noise and interference in the received signal, and therefore a good candidate for integrity monitor statistics.

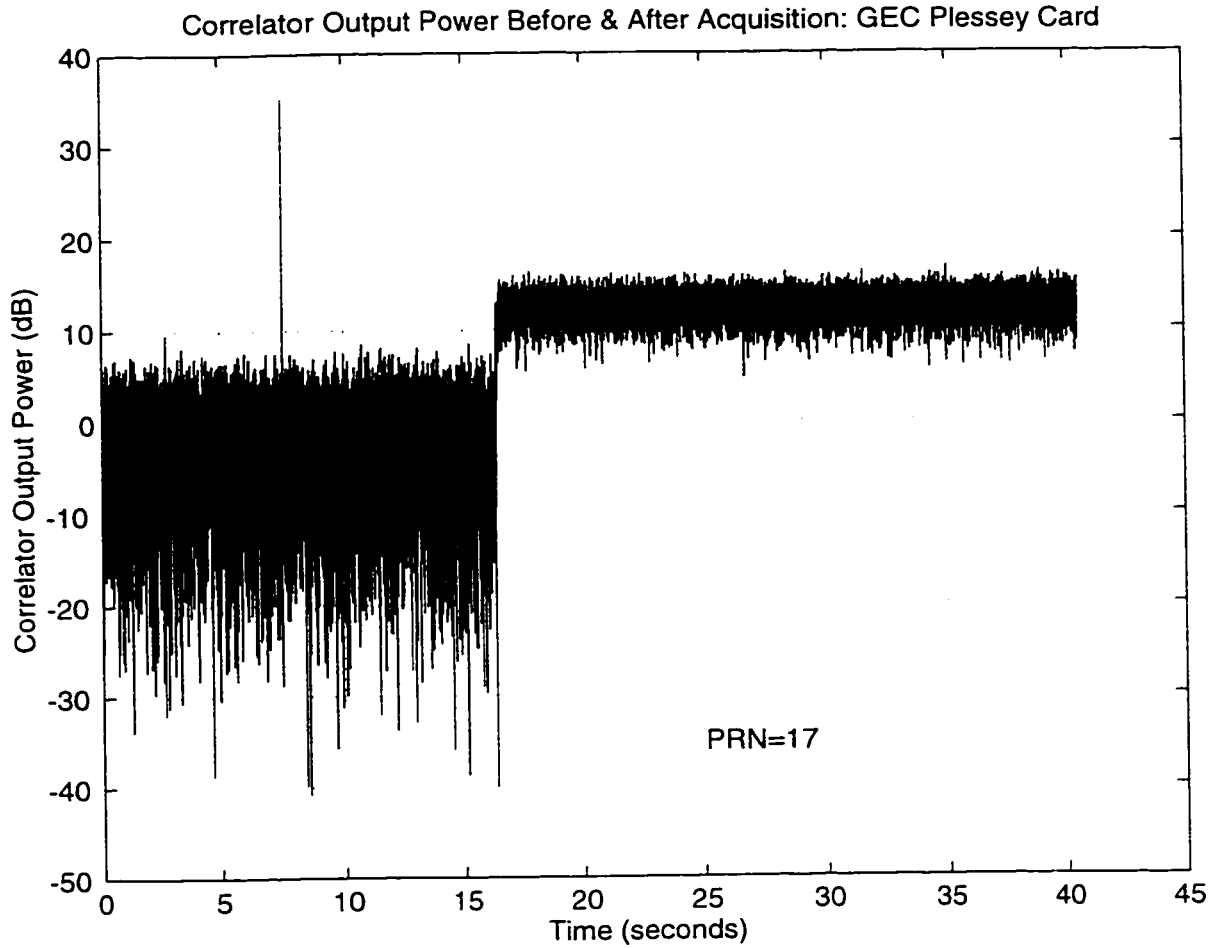


Figure 16: Correlator Output Power for a GPS Receiver tracking Satellite PRN #17. Satellite tracking starts at time = 17 seconds.

2.3.2 Correlator Output Power Variance

Correlator Output Power variance ($COP\sigma$), the variance of COP, can also be observed from Figure 16 as the spread around the mean of COP. It is given by the equation:

$$COP\sigma = \sqrt{E[COP - E(COP)]^2}$$

which, after expansion, becomes:

$$COP\sigma = \sqrt{E(COP^2) - [E(COP)]^2} \quad (2.3.2.1)$$

where $E()$ stands for “expected value of ()”.

There is a significant reduction in $\text{COP}\sigma$ following signal acquisition. We observed that $\text{COP}\sigma$ also is a function of noise and interference in the received signal, and therefore a good candidate for integrity monitoring. A 256 millisecond running average is maintained of $\text{COP}\sigma$ in the same manner as for COP, shown in Figure 15.

2.3.3 Carrier Phase Jitter

From equation 2.2.2.1, the product of the local prompt carrier phase and incoming intermediate frequency after low pass filtering is given by:

$$\text{Local x incoming carrier} = D(t)\cos(2\pi(f_1 - f_2)t) \quad (2.3.3.1)$$

Neglecting code and data modulation, and taking into account initial phase offsets from zero, equation 2.3.3.1 can be rewritten as:

$$\text{Local x incoming carrier} = \cos(\omega_\delta t + \phi) \quad (2.3.3.2)$$

where $\omega_\delta = 2\pi(f_1 - f_2)$, and ϕ is the difference between initial phases of incoming and local carrier signals.

Using Euler's theorem [39], equation 2.3.3.2 can be represented as the real part of a complex sinusoid:

$$\begin{aligned} \cos(\omega_\delta t + \phi) &= \text{Re}\left[e^{j(\omega_\delta t + \phi)}\right] \\ &= \text{Re}\left[e^{j\phi}e^{j\omega_\delta t}\right] \end{aligned} \quad (2.3.3.3)$$

where j is defined as the square root of -1 .

Equation 2.3.3.3 is the *phasor* representation of 2.3.3.2. The terms in the bracket can be viewed as a rotating vector in a complex plane, with a phasor angle $= (\omega_{\delta}t + \phi)$.

The phasor angle of the prompt channel carrier signal can be computed from the arc tangent of the prompt quadrature and inphase signals:

$$\text{Phasor angle} = \arctan (Q_Accum_{\text{prompt}} / I_Accum_{\text{prompt}}) \quad (2.3.3.1)$$

In an ideal phase locked loop, this angle would equal zero. However this is usually not the case. Figure 17 shows a plot of the phasor angle for a real GPS receiver with a FLL tracking satellite PRN 17.

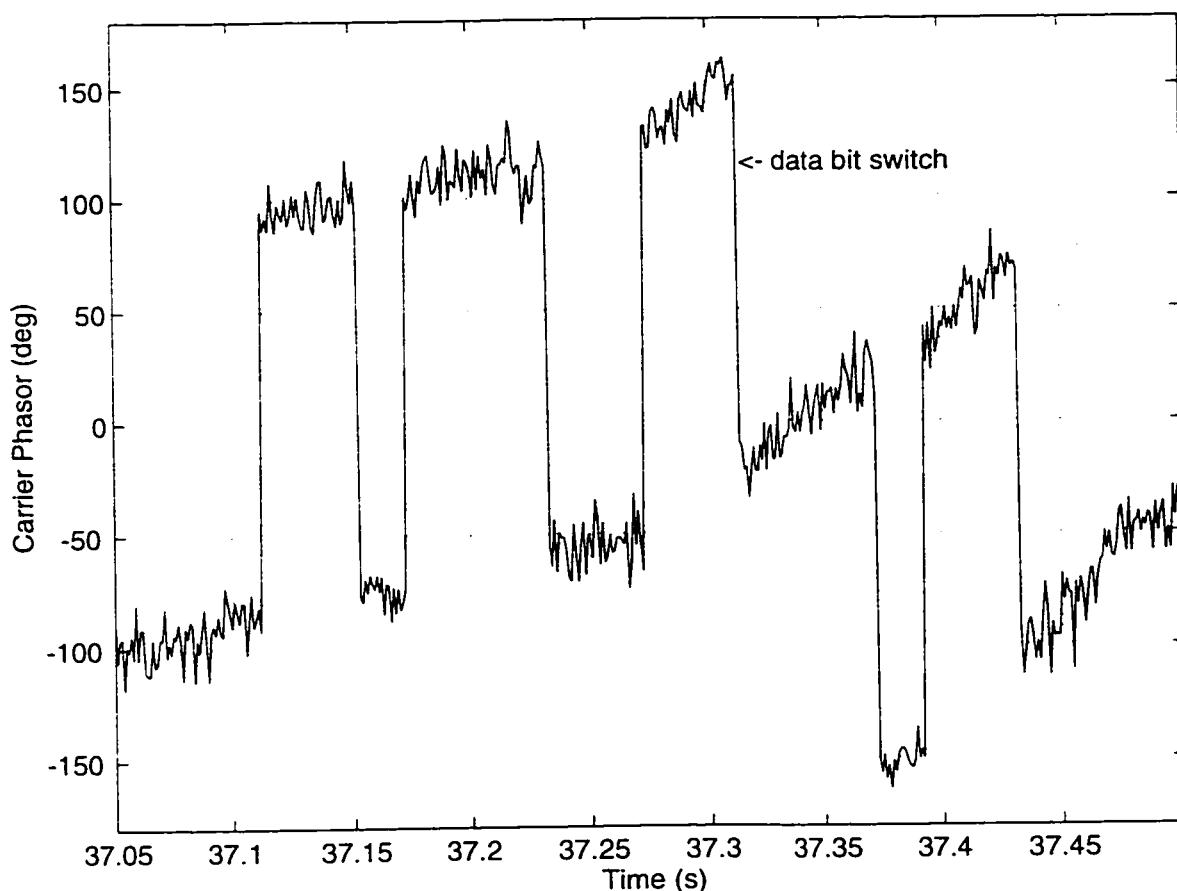


Figure 17: Carrier Phase Jitter for a GPS Receiver tracking Satellite PRN #17

Notice the slow drift in phase, to be expected for a frequency locked loop (a PLL would attempt to drive this trend to zero). The 180 degree swings occur due to transitions in data bits. Observe also the rapid fluctuations about the slow drifting mean. Carrier phase jitter (CPJ) is a measure of these fluctuations, and is given by

$$\text{Carrier Phase Jitter} = \text{running average}[\text{abs}\{\text{Phasor angle}_i - \text{Phasor angle}_{i-1}\}]$$

where i is the 1 ms epoch time index, and Phasor angle is as defined in equation (2.3.3.1) above. A running average is maintained over 250 samples (0.25 seconds). Phase changes greater than 180 degrees are reduced by 180 degrees, to handle data bit switches.

Carrier phase jitter occurs as a result of noise on the incoming signal, satellite clock and receiver clock noise.

2.3.4 AGC Gain

As discussed in section 2.2.2, the active gain control adjusts threshold levels of the A-D quantizer to maintain the output levels +3 : +1 : -1 : -3 in the ratio 15% : 35% : 35% : 15% respectively. For a two bit adaptive quantizer operating on an RF signal (Figure 18) the center threshold, r_2 , is usually zero, and the lower and upper thresholds, r_1 and r_3 , are usually equal.

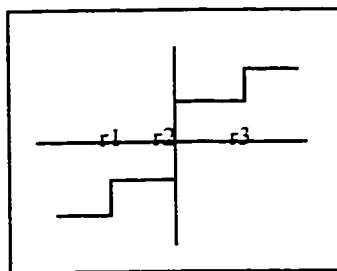


Figure 18: Two-bit Adaptive Quantizer (AGC)

Therefore the magnitude of r_1 (or r_3) is an indication of the gain of AGC controller. The term *AGC gain* is used synonymously with magnitude of the adaptive quantizer thresholds.

Figure 19 shows AGC Gain as a function of noise power level introduced into the GPS receiver. A range of AWGN noise power levels are applied to the receiver, starting from 0 dB (relative to the tracked GPS signal), and up to 37 dB (point at which receiver loses lock) in 1 dB steps. AGC gain increases monotonically with increasing noise power.

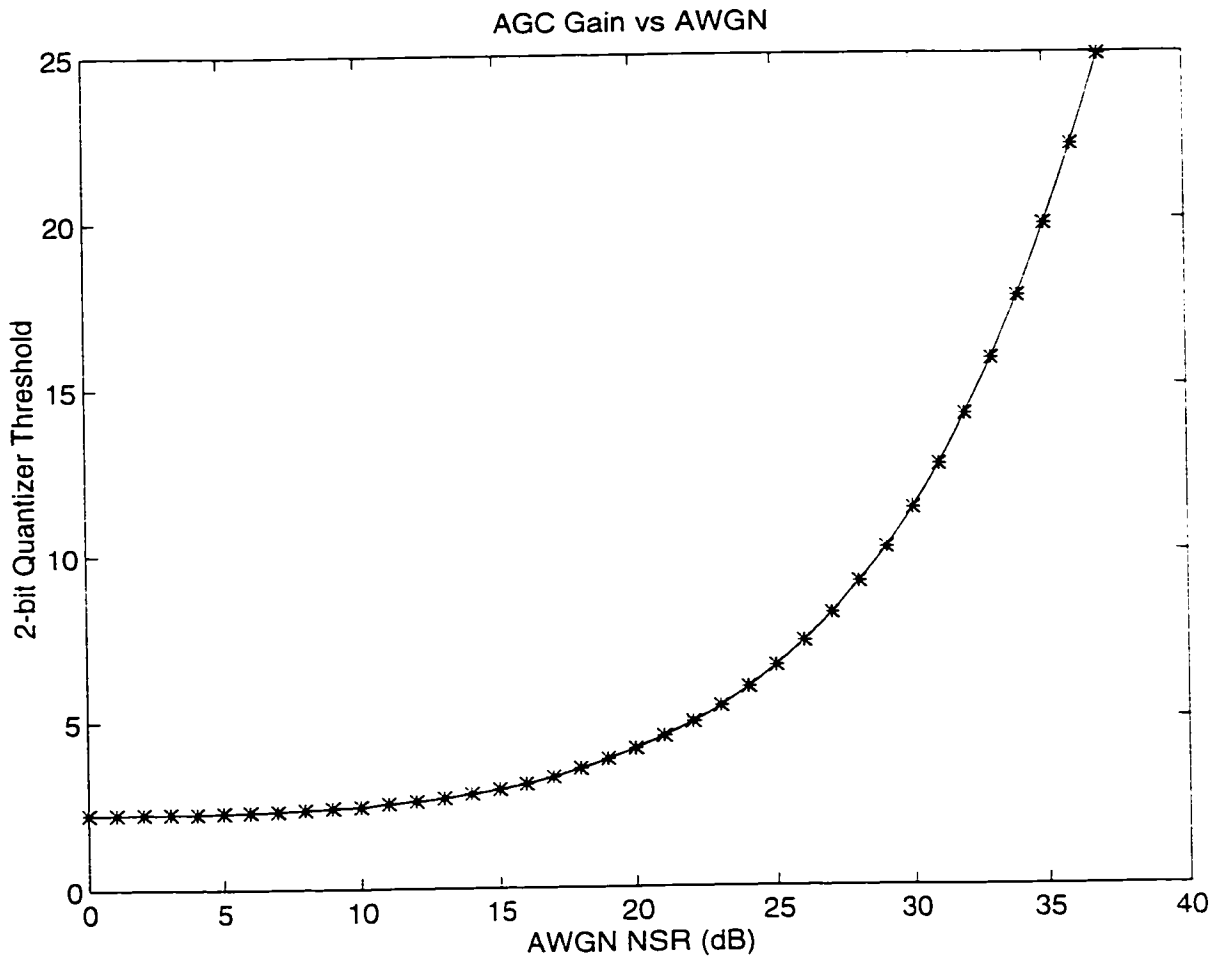


Figure 19: AGC Gain vs. AWGN

AGC Gain is a measure of the total power in the incoming signal, and therefore a good candidate for detecting the presence of interference.

When used as a test statistic, a 256 ms running average is maintained of AGC Gain, in the same manner as for COP (Figure 15).

Chapter 3

INTERFERENCE ANALYSES

3.1 Introduction

There are numerous sources of RF interference (RFI) to a GPS receiver. These sources may be on the same vehicle as the receiver, other vehicles or ground-based. Interference may result from harmonics, intermodulation products of on-board electronic devices, direct CW and broad/narrow band noise RFI emissions. It may fall within the GPS L1 band which is from 1563 to 1587 MHz. This chapter presents an overview of possible interference, and includes an analysis of GPS receiver susceptibilities. Interference types and levels used for later analysis are also presented.

3.2 RF Interference to GPS

One source for interference to GPS receivers onboard aircraft are radios using the civil aviation band, 118 MHz to 136 MHz. This band, within the very high frequency (VHF) range which spans 30 MHz to 300 MHz, is set aside for air to ground voice and data communication. Civil aviation VHF voice bands consist of 25 KHz channels. The 12th and 13th harmonics of several 25 KHz bands fall directly within the GPS spectrum [11]. For example, the 12th harmonic of the VHF channel 131.25 MHz = $12 \times 131.25 \text{ MHz} = 1575 \text{ MHz}$, the center of GPS L1 spectrum. The 13th harmonics of the channels 121.150, 121.175 and 121.200 MHz also fall within GPS band [41].

Another potential source of GPS interference comes from the introduction of High Definition Television (HDTV). HDTV is a new TV standard currently being adopted by the FCC, which will transmit crystal clear, wide screen video and audio at resolutions up to five times greater than current television. Since the signals are digital, HDTV transmitters

will not have to keep up the stringent out-of-band signal suppression currently used by analog TV stations to minimize ‘ghosting’, a phenomenon that results in degraded analog TV pictures due to strong out-of-band transmissions. The current FCC minimum requirements for TV out-of-band signal suppression will be inadequate to prevent significant HDTV interference to GPS.

Other sources include pulsed radar signals, improperly filtered TV signals, and accidental transmissions by RF experimenters. Table 3 below lists a few sources of RFI, classified by source, and whether it falls in-band or out of GPS band [11], [42].

Depending on its bandwidth, RFI Interference can be classified as broadband, narrow band or CW. Interference may also be pulsed (gated on / off) or continuous. Another type of interference to GPS, which is not RFI, occurs due to physical objects that obscure the GPS signal or cause multipath fading to occur. To capture all these various types of interference, the following categories are defined and used in this research:

i. Narrowband Interference:

Narrowband interference is modeled as a pure tone RFI, consisting of a continuous wave (CW), at a specified frequency. When the frequency of CW interference coincides with that of a GPS signal, it is described as being *coherent*. *Non-coherent* CW interference does not coincide with any spectral line of the GPS signal.

ii. Broadband Interference:

Broadband interference and thermal noise have a flat power spectral density over a wide range of frequencies, specifically including the GPS L1 band, 1563 through 1587 MHz. This type of interference is also referred to as Additive White Gaussian Noise (AWGN).

Table 3: Potential Sources of Interference to GPS Receivers

RFI Source	Type	Location
Mobile Satellite Services (MSS) (1610 – 1620 MHz)	In-band Out-of-band	Non-A/C
VHF Comm Harmonics & Passive Intermod. Products	In-Band	Same A/C Nearby A/C
Satellite Communications (SATCOM) Aeronautical Mobile Satellite Service (AMSS)	In-band Out-of-band	Same A/C Nearby A/C
Aircraft Addressing and Reporting System (ACARS) Harmonics	In-Band	Same A/C Nearby A/C Non-A/C
Flight Telephone Services (1626.5 – 1660.5 MHz)	Out-of-band	Same A/C Nearby A/C
Distance Measuring Equipment (DME)	Out-of-band	Same A/C Nearby A/C Non-A/C
HF Harmonics	In-Band	Same A/C
Mode S	In-band Out-of-band	Same A/C Nearby A/C Non-A/C
Amateur Radio	In-Band	Non-A/C
FM Harmonics and Passive Intermodulation	In-Band	Non-A/C
UHF TV Harmonics (787.21 – 788.24 MHz) (524.80 – 5254.48 MHz)	In-Band (2 nd and 3 rd harmonics)	Non-A/C
VHF/UHF Land Mobile Harmonics 394, 315, 262.5 MHz 197, 175, 143.2, 131.3, 121.1 MHz	In-Band	Non-A/C
VHF OmniRange (VOR) Harmonics	In-Band	Non-A/C
Personal Electronic Devices (PED)	In-Band	Same A/C

iii. Pulsed Interference:

Pulsed interference is present only a fraction of the time. It is therefore characterized by a pulse duty cycle – the fraction of time which the pulse is on, pulse repetition rate or pulse width, and peak power. The effects of both pulsed CW and AWGN are considered. The pulsing scheme is derived from the Wide Area Augmentation System (WAAS) Minimum Operational Performance Standards (MOPS) for pulsed interference, which specifies minimum interference rejection capabilities for WAAS GPS receivers. Specifically, peak pulse power used for subsequent analysis is 30dBm, or 150 dB interference-to-signal ratio (ISR).

iv. Multipath

Physical objects close to the antenna cause signal reflections that interfere with this direct signal. This phenomenon is known as *multipath*. The effect of multipath is studied by attenuating the signal being tracked.

v. Signal Blockage or Attenuation

The incoming GPS signal could be blocked or attenuated by physical objects such as buildings, foliage, and terrain. The signal from a setting or rising GPS satellite is also attenuated as a result of the longer travel distance through the troposphere. In addition the gain pattern of an antenna may attenuate the received signal. Signal attenuation and blockage is not interference of any sort, however its effects must be considered when designing a robust monitor.

3.3 GPS Receiver Susceptibility to RF Interference

A GPS receiver suppresses interference by two primary methods: RF filtering, and spread spectrum. A fast acting AGC or adaptive analog to digital converter will also suppress pulsed interference.

Firstly, the RF front end passes the incoming signal through a series of bandpass filters, which greatly reduce out-of-band interference power. Figure 20 demonstrates this reduction using a 4th order Butterworth bandpass filter acting on intermediate frequency IF4, frequency = 1.41 MHz (see down conversion scheme in Figure 8). The filter bandwidth is 1 MHz.

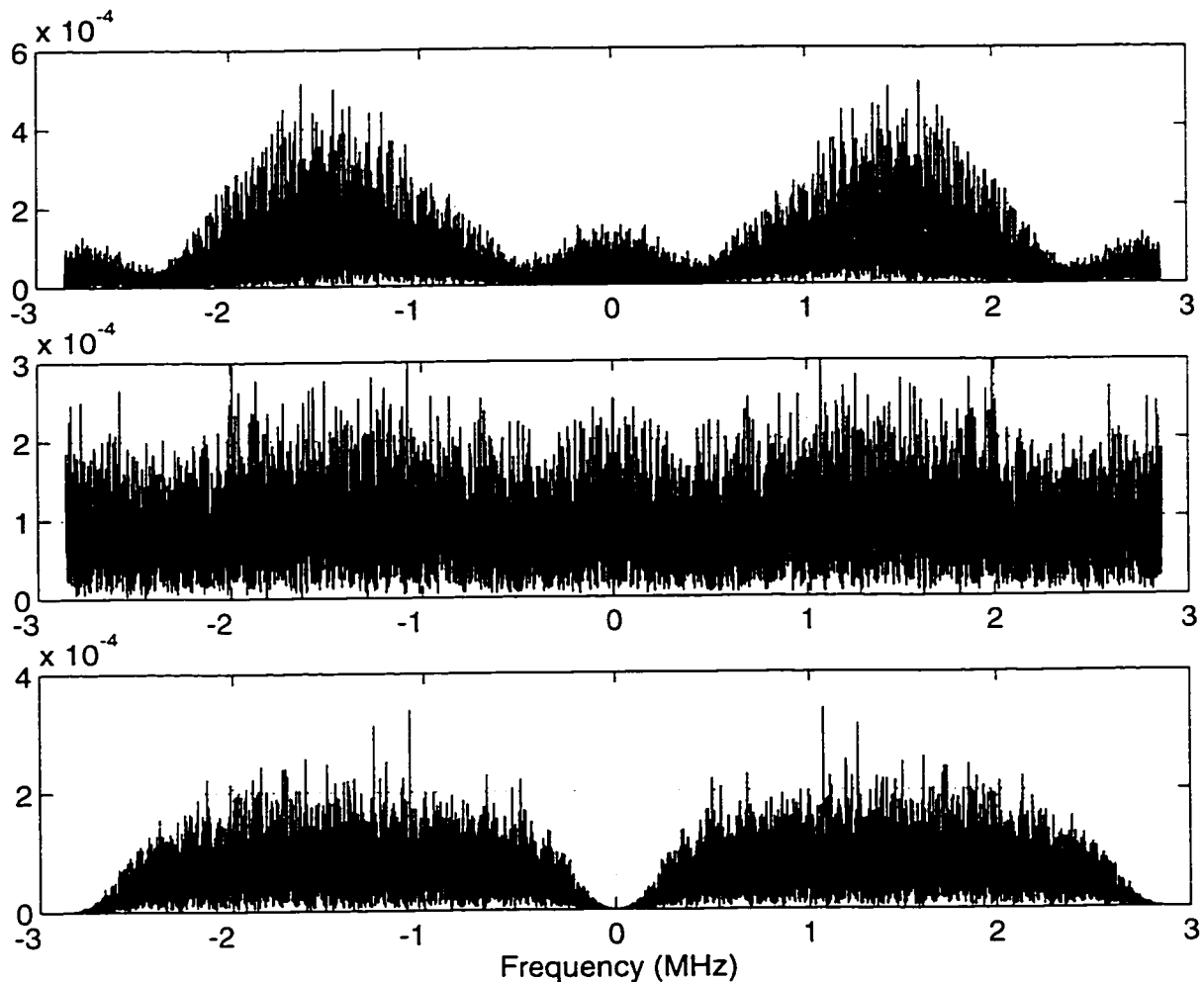


Figure 20: Bandpass filtering an IF4 signal (center frequency = 1.41 MHz)

- (a) Spectrum of single noise free satellite signal.
- (b) Spectrum of GPS signal plus 8 dB of AWGN prior to filtering
- (c) Spectrum of GPS signal plus 8 dB of AWGN after filtering with 1 MHz bandwidth 4th order Butterworth bandpass filter.

Figure 20a shows the spectrum of a single noise-free GPS satellite at intermediate frequency IF4. The sinc-function nature of this spectrum is clearly observable. After the addition of 8 dB of AWGN, the GPS signal becomes submerged in the noise floor (Figure 20b) as noise power dominates the signal. Bandpass filtering reduces the effective noise power as shown in Figure 20c.

The remaining in-band interference power is further suppressed through spread spectrum decorrelation inherent in the GPS signal design. This action is described as follows. Prior to transmission by the GPS satellites, the GPS 50 Hz data stream is first spread by modulating it with a 1 MHz spreading signal (Figure 21). This spreading signal is a 1023-chip long pseudorandom sequence called a Gold code, with orthogonal correlation properties.

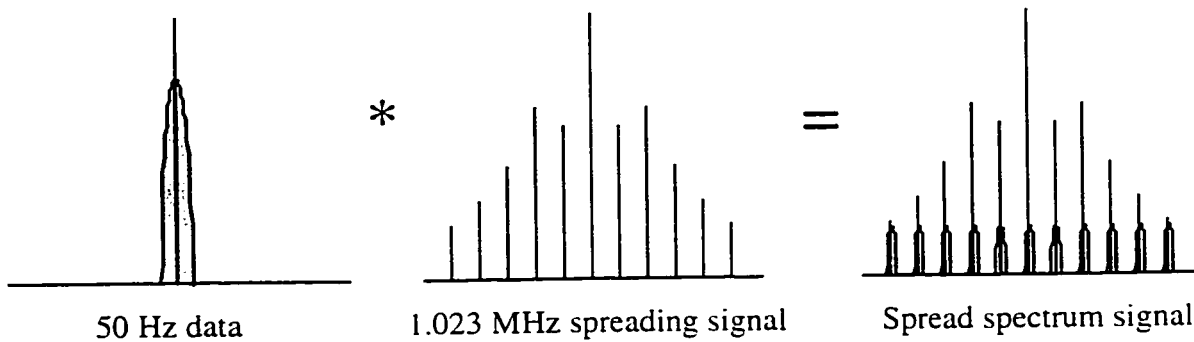


Figure 21: Spreading of 50 Hz GPS data

This action can be represented by the equation :

$$s_1(t) = s(t)D(t) \quad (3.3.1)$$

where $D(t)$ = 50 Hz binary data stream

and $s(t)$ = spreading signal

$$= \sum_{n=-\infty}^{\infty} S_n p(t - nT_c)$$

$$= \sum_{n=0}^{1023} S_n p(t - nT_c) * \sum_{m=-\infty}^{\infty} \delta(t - nT_c) \quad (3.3.2)$$

$S_n = \pm 1$, a pseudorandom sequence, and $p(t)$ is a rectangular unit pulse over the interval $\{0, T_c = 1/f_c\}$ (12).

Let $B_s = 1.023 \text{ MHz}$ be the single sided bandwidth of the spreading signal $s(t)$.

and $B_d = 50 \text{ Hz}$ be the single sided bandwidth of the data signal $D(t)$.

Observe from Figure 21 that now the GPS data is spread from its original 50 Hz bandwidth to a much wider 1 MHz single sided bandwidth. The resulting power density is also lower, keeping total power constant. This composite signal is further modulated by a carrier at L1, and broadcast by the satellites. Figure 22 shows the addition of noise and interference to the signal as it is transmitted from the satellites via the atmosphere to the antenna, and on down to the precorrelation point.

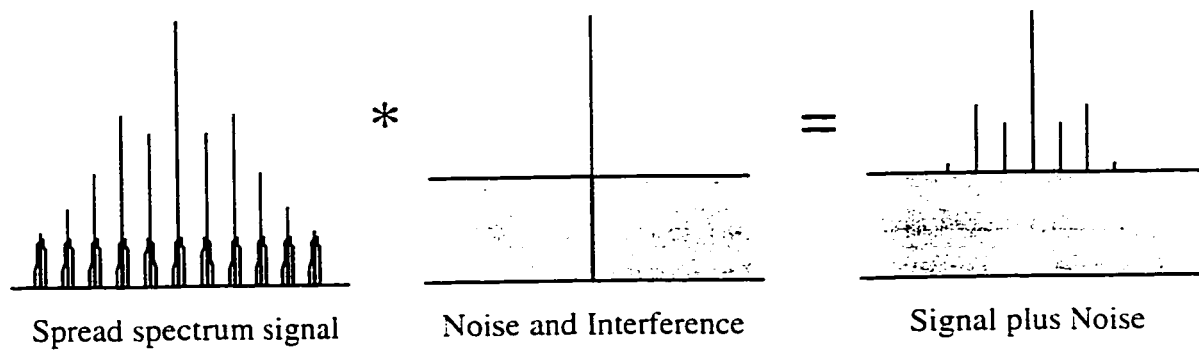


Figure 22: Noise and Interference in the GPS signal

Assuming the transmission channel has AWGN, $n(t)$, with power spectral density (PSD) N_t , and interference $b(t)$. The result as shown in Figure 22 is given by:

GPS signal combined with noise and interference = $r(t)$:

$$\text{with } r(t) = s(t)D(t) + n(t) + b(t) \quad (3.3.3)$$

where $b(t)$ is a narrowband interference with power P_b (12).

The power spectral density for the receiver thermal noise floor is -200 dBW/Hz. The total noise power in a 2 MHz bandwidth would therefore be:

$$\begin{aligned}\text{Noise power in 2 MHz bandwidth} &= -200 \text{ dBW/Hz} + 60 \text{ dB-Hz} \\ &= -140 \text{ dBW}\end{aligned}$$

However the GPS signal has a total receive power of approximately -160 dBW. Receiver thermal noise is therefore about 20 dB stronger than the GPS signal and completely buries it.

Finally during correlation, the composite signal is multiplied by an identical replica of the original spreading signal, as shown in Figure 23.

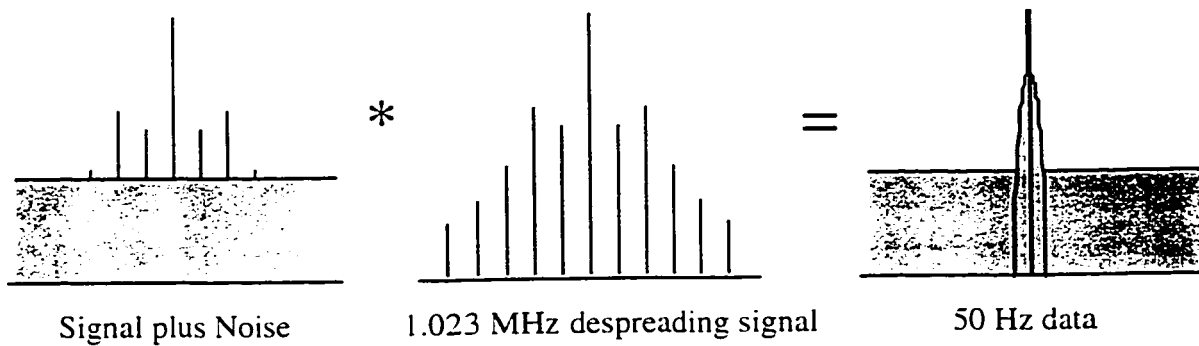


Figure 23: Despreading and Data Recovery

Despreading of 50 Hz data occurs, as shown by the following equation:

Despreading:

$$\begin{aligned}r(t)s(t) &= s(t)\{s(t)D(t) + n(t) + b(t)\} \\ &= s^2(t)D(t) + s(t)n(t) + s(t)b(t)\end{aligned}\tag{3.3.4}$$

which, after filtering to remove the high frequency terms, becomes:

$$r(t)s(t) = D(t) + s(t)n(t) + s(t)b(t)\tag{3.3.5}$$

since $s^2(t) = 1$.

At this point, as shown in Figure 23 and equation 3.3.5, the GPS data is recovered from the noise floor. Consider closely the second and third right hand side terms of equation 3.3.5:

$s(t)n(t)$: Represents the spreading of the AWGN. Post correlation noise PSD will still be N_0 , since correlation of white noise with a spread spectrum will still result in white noise.

$s(t)b(t)$: This term represents the spreading of the narrow interferer. Interference power is spread from its original bandwidth to the bandwidth of the spreading sequence, B_s , (in much the same way as $D(t)$ was originally spread).

Passing the output signal (3.3.5) through a bandpass filter with bandwidth equivalent to the bandwidth of the data signal B_d , will pass all the power in the data signal $D(t)$, but only a fraction of noise and interference power:

$$\text{noise + interference power after filtering} = N_t B_s + P_b B_d / B_s \quad (3.3.6)$$

Equation 3.3.6 implies that spread spectrum has in effect reduced the post correlation power of the narrow band interference by the ratio B_d/B_s (bandwidth of data signal divided by bandwidth of spreading signal). This ratio is called the *processing gain* of the spread spectrum system. GPS has a processing gain $= 43.1 \text{ dB} = 10 \log_{10} \left[\frac{50 \text{ Hz}}{1 \text{ MHz}} \right]$.

Note also that equation 3.3.6 shows that spread spectrum does not affect the PSD of white noise.

3.4 Expected Interference Signal Power for Loss of lock

To estimate the levels of interference power that would cause a receiver to loose lock, noise plus interference density, N_o , is given by:

$$N_o = N_t + N_i \quad (3.4.1)$$

where N_t = receiver thermal noise floor.

N_i = interference noise density:

N_o , N_t and N_i are expressed in W/Hz, not dBW/Hz.

Dividing (3.4.1) by GPS carrier power C gives:

$$\frac{N_o}{C} = \frac{N_t}{C} + \frac{N_i}{C} \quad (\text{expressed in Hz}) \quad (3.4.2)$$

or

$$\left[\frac{C}{N_o} \right]^{-1} = \left[\frac{C}{N_t} \right]^{-1} + \left[\frac{C}{N_i} \right]^{-1} \quad (\text{Hz}) \quad (3.4.3)$$

Assuming interference is absent ($N_i = 0$ W/Hz), using value of $N_t = -200$ dBW/Hz for receiver thermal noise floor, and GPS signal strength $C = -160$ dBW, equation (3.4.3) will result in:

$$\left[\frac{C}{N_o} \right]^{-1} = \left[\frac{-160\text{dBW}}{-200\text{dBW/Hz}} \right]^{-1} + [0] = -40 \text{ dB - Hz}$$

or

$$\frac{C}{N_o} = 40 \text{ dB - Hz} \quad (3.4.4)$$

In actual operation the value of C/N_o can be as high as 55 dB-Hz in the absence of interference, due to lower receiver noise floors (-205 dBW/Hz), and the GPS satellites transmitting better-than-published signal power.

Assuming that a receiver requires a C/N_o value of at least 20 dB-Hz to maintain lock on the GPS signal. (With a noise bandwidth of 50 Hz, a C/N_o value of 20 dB-Hz will produce a post correlation signal-to-noise ratio, $SNR = 20 \text{ dB-Hz} - \{50\text{Hz, in dB}\} = 3.0 \text{ dB}$, which is just enough to maintain lock, depending on receiver settings for acquisition thresholds.) The corresponding level of interference power can be evaluated by rearranging equation (3.4.3):

$$\left[\frac{C}{N_i} \right]^{-1} = \left[\frac{C}{N_o} \right]^{-1} - \left[\frac{C}{N_t} \right]^{-1} \quad (\text{Hz}) \quad (3.4.5)$$

Substituting Hz values for C/N_o of 20 dB-Hz, and C/N_t of 40 dB-Hz, and converting back to dB-Hz gives:

$$\left[\frac{C}{N_i} \right]^{-1} = -20.04 \text{ dB - Hz}$$

or

$$\frac{C}{N_i} = 20.04 \text{ dB - Hz} \quad (3.4.6)$$

For an interference spread over bandwidth of $B_j = 1.023 \text{ MHz}$ (60.1 dB-Hz), equation (3.4.6) yields a worst case interference to signal power ratio (ISR):

$$\text{ISR} = J_s/S = 40 \text{ dB} \quad (3.4.7)$$

(where J_s = interference power, and S = signal power)

since

$$\frac{C}{N_i} = \frac{C}{J_s/B_j} = 20.04 \text{ dB - Hz}$$

therefore

$$\frac{J_s}{S} = \frac{J_s}{C} = B_j - 20.04 \text{ dB - Hz} = 40 \text{ dB}$$

Equation (3.4.7) was verified in both simulation and bench tests: the value $J_{\text{c}}/S = 40$ dB was used as the high end for the various types of interference applied to the receiver. As we expected, the receiver lost lock before this high interference value was attained.

3.5 Coherent CW Interference

Consider the despreading process in equation (3.3.5). The last term of this equation, $s(t)b(t)$, captures the spreading of the interferer. For CW interference, computation of this term is simplified, and can be visualized from Figure 23 (or Figure 21, replacing GPS data with the CW interference). Let the pure tone interference fall directly on L1 for a satellite signal with zero Doppler. Note that the DC term for the spreading signal is minimal, since the Gold codes chosen for GPS are balanced. (A balanced code has a sum of -1 [12].) This implies that the C/A code spectral line directly on L1 has power of only -60 dB ($= 20 \log \left[\frac{1}{1023} \right]$).

The resulting PSD would look similar to the PSD for the spreading sequence, scaled by a constant, as shown in Figure 24. Note that this figure shows convolution in the frequency domain, which is equivalent to multiplication in the time domain.

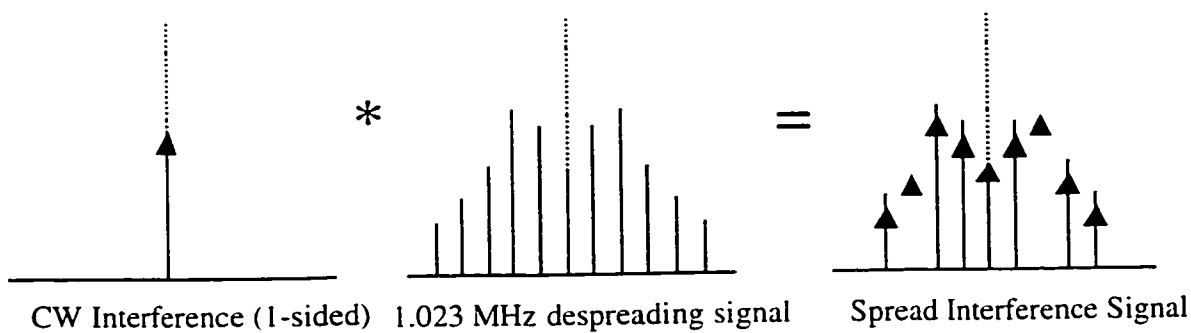


Figure 24: CW Interference on L1 Before and After Spreading by Receiver

Notice from Figure 24 that the higher power spectral lines modulate the CW interferer off from the center GPS 50 Hz band, thereby ensuring that minimal interference power is left

after low-pass filtering for data recovery.

On the other hand, consider a case where the CW interferer falls directly on a high power (worst case) spectral line. Figure 25 shows that a larger portion of the CW interferer's power will be translated by the spreading signal to fall directly in the center of GPS band, and a larger amount of interference power would therefore show up post correlation.

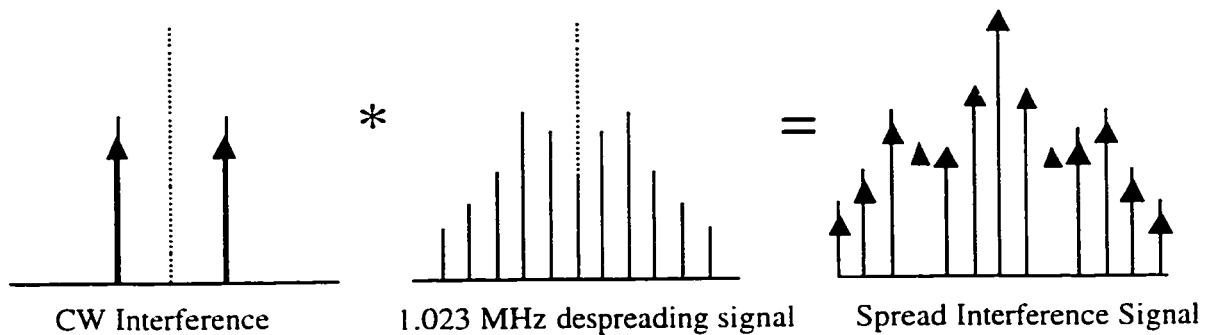


Figure 25: CW Interference on worst case C/A spectral line before and after spreading by receiver

The severity of coherent CW interference coinciding with worst case spectral lines can be seen from the resulting modulation PSD of Figure 25. Indeed coherent CW is the most severe form of interference to GPS. Figure 26 shows correlator output power and pseudorange error as a function of CW interference spectral line offset for a receiver tracking a GPS signal³. The CW interference is successively moved in 1 KHz steps off L1 to coincide with different spectral lines, and its effect in the receiver is captured in this figure. It can be seen that by placing the CW interference on a 'hot' spectral line at 7 KHz offset from L1, there is a sharp drop in correlator output power from above 12 dB to 2 dB (Figure 26), with a corresponding increase in pseudorange error, indicating the severity of coherent interference at this frequency. Indeed loss of lock occurs within 3 seconds for this particular interference scenario.

³ Results from software simulation described in chapter 5.

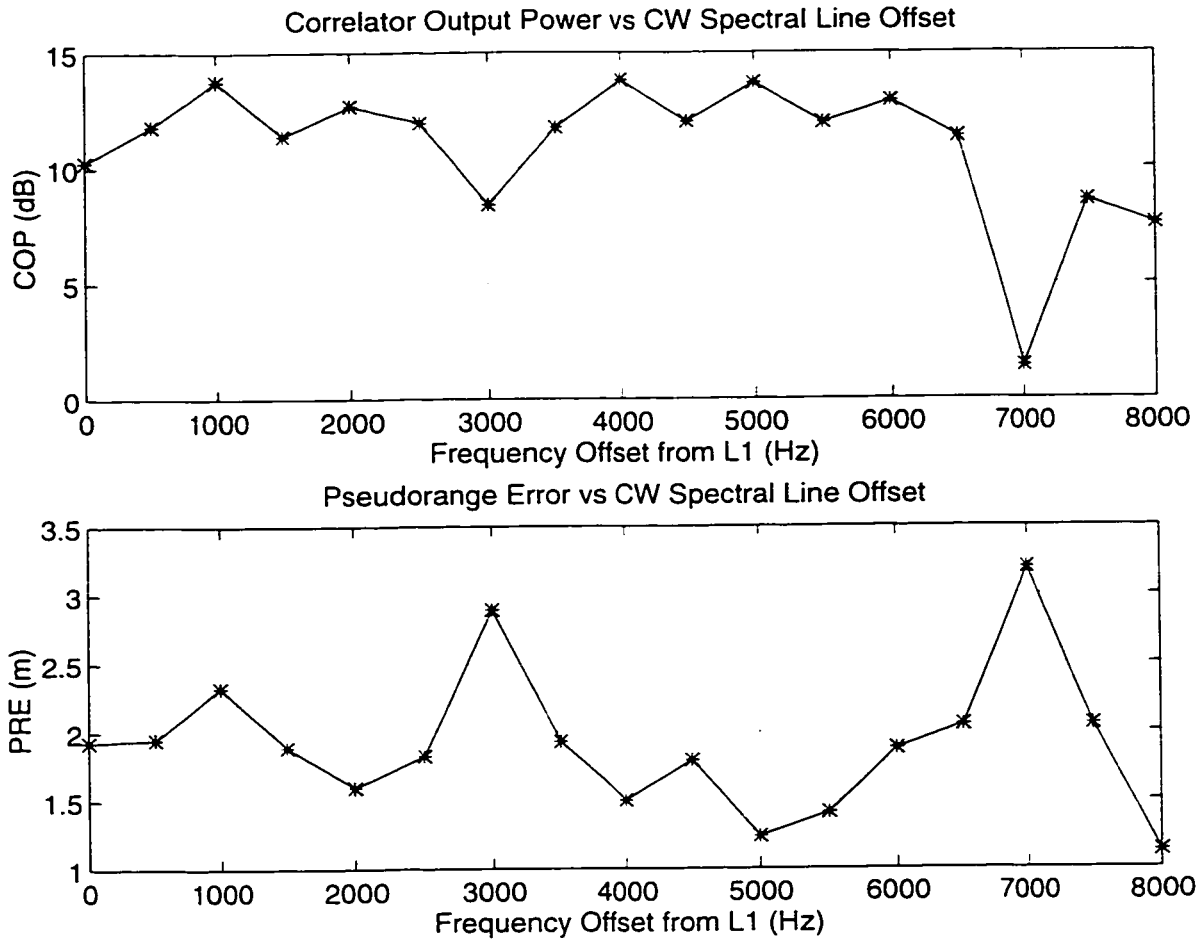


Figure 26: Correlator Output Power and Pseudorange Error as a function of CW Interference frequency offset

Frequency steps: $\Delta f = 1\text{KHz}$ spectral lines.

Fortunately it can be shown that this type of interference is rare in practice. Firstly, the position of worst case spectral lines vary from one C/A code to the next. An offset of 7 KHz from L1 corresponds to a powerful spectral line for the satellite shown in Figure 26, but would not correspond to strong lines for all the other GPS satellites. Therefore it is unlikely that more than one satellite will suffer worst case spectral line interference by the same CW interferer at any one time. Furthermore, the Doppler frequencies of the GPS satellites are always changing, because the satellites are in motion. Figure 27 shows the Doppler shift of the GPS constellation over 12 hours. Observe the positive Doppler at the

start of each frequency curve, associated with rising satellites. The value diminishes as the satellite reached its maximum elevation, at zero Doppler. Doppler goes negative as the satellite descends. The shorter lines in Figure 27 are for satellites with a low maximum elevation angles. These satellites transit briefly near the horizon .

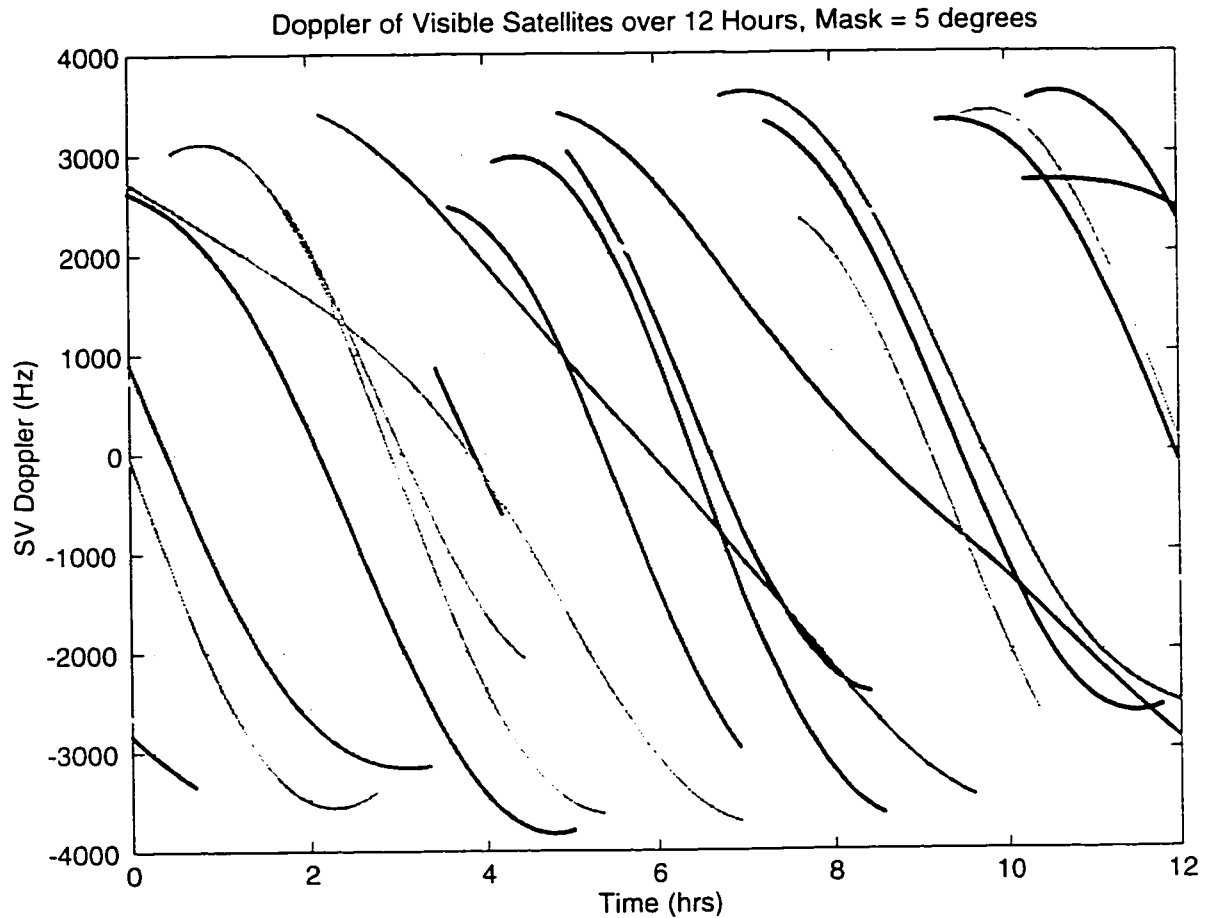


Figure 27: Doppler of GPS Constellation over 12 hours

Figure 27 shows that there are few cross-over points where two or more satellites have the same Doppler. Also observe the rapid changes in frequency for higher elevation satellites (Doppler = 0 Hz), and the slower changes for lower elevation satellites. Figure 28 captures the elevation dependency of Doppler rate, for satellite PRN 14 over a 12 hour period.

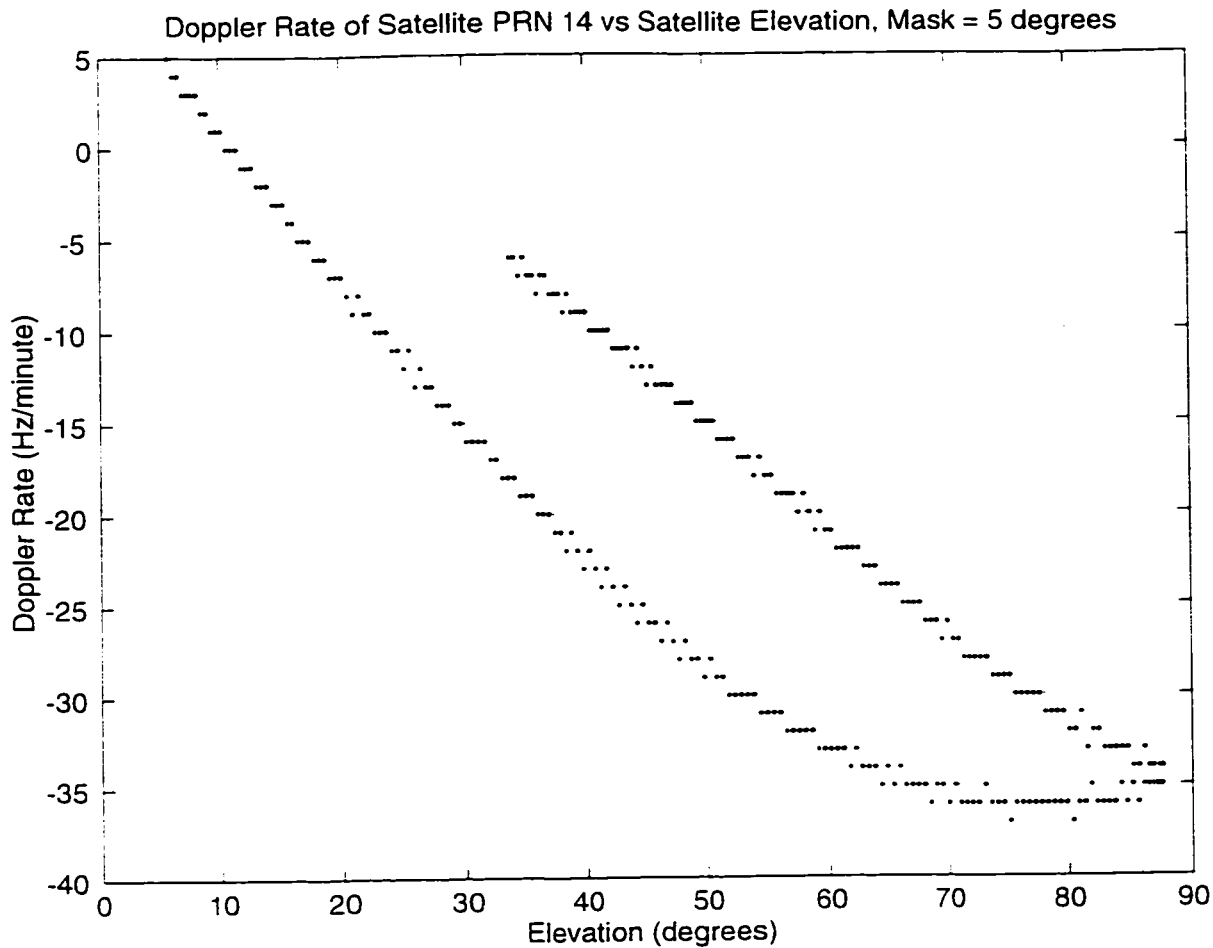


Figure 28: Doppler Rate of Satellite PRN 14 vs. Elevation

The figure shows Doppler rates as fast as 37 Hz/minute. A CW interferer with a fixed frequency would therefore coincide with any specified spectral line for only a few seconds at a time.

Dynamic jammers would not only have to follow the Doppler of moving satellites, but also track the Doppler of the receiver platform. In summary, worst case coherent CW is rare, when it does occur it would likely affect only one satellite, and that for a brief duration in time.

3.6 Impact of Interference on GPS

Interference adversely affects all four key systems requirements of accuracy, integrity, availability and continuity. The following sections quantify these effects.

3.6.1 Impact on Accuracy

The delay lock loop (DLL) of a GPS receiver, used to directly track the C/A code, determines the accuracy of the range measurements, since the error of the DLL directly translates to pseudorange error. Its accuracy is affected by noise power according to the following equation for a non-coherent DLL with an envelope detector [20]:

$$\sigma_k^2 = \frac{d(cT_c)^2 BW_{L,1-sided}}{2 \left(\frac{C(el^k)}{N_o + I_o} \right)} \left[1 + \frac{2}{(2-d) \left(\frac{C(el^k)}{N_o + I_o} \right) T_{square}} \right] \quad (3.6.1.1)$$

where d = correlator spacing;

T_c = C/A code chip width = $1\mu s$;

$BW_{L,1-sided}$ = 1-sided tracking loop bandwidth;

T_{square} = squaring loss, from early-late power detector;

$C(el^k)$ = carrier power of the k^{th} satellite, which is a function of its elevation angle el^k .

The 2-sigma vertical accuracy can then be obtained from [20]:

$$2\sigma_v = 2 \sqrt{\left[(G^T P_p^{-1} G)^{-1} \right]_{3,3}} \quad (3.6.1.2)$$

where G = geometry matrix of visible satellites;

and P_p is the covariance matrix of range measurements;

For uncorrelated satellite errors, with n visible satellites, P_p is the diagonal matrix given by:

$$P_p = \begin{bmatrix} \sigma_1^2 & & & 0 \\ & \sigma_2^2 & & \\ & & \ddots & \\ 0 & & & \sigma_n^2 \end{bmatrix} \quad (3.6.1.3)$$

Figure 29 shows the vertical error ($2\sigma_v$) as a function of interference power for squaring loss: $T_{\text{square}} = 1/50$, $BW_{L,1\text{-sided}} = 1/50$ Hz (50 seconds of carrier smoothing), and correlator spacing ranging from 0.1 (narrow) to 0.5 (wide).

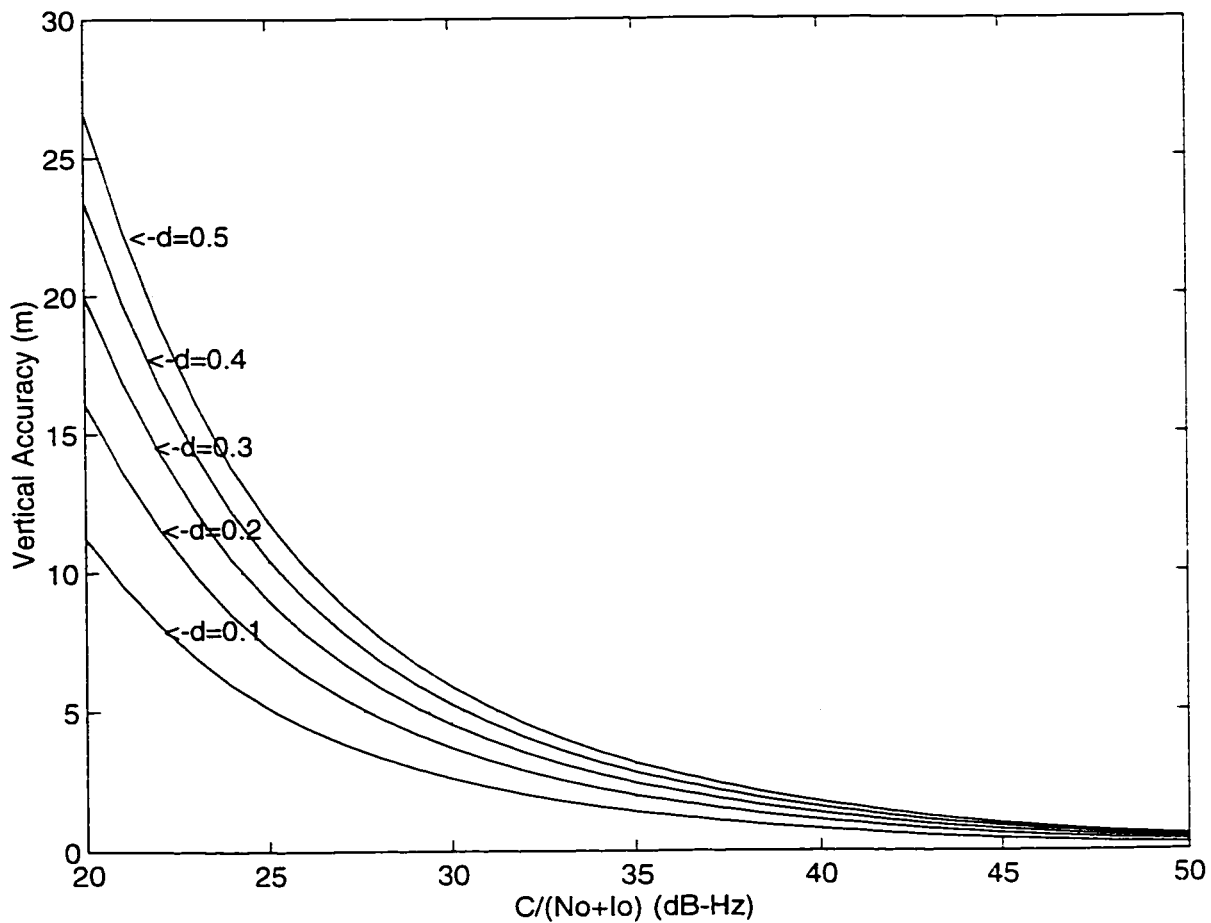


Figure 29: Vertical Position Error of a Receiver as a Function of Noise plus Interference Power

Figure 29 shows increasing vertical position errors for increasing interference power, or decreasing $C/(N_0+I_0)$ in all cases, with wider correlator spacings having larger errors, up to 27 meters for a correlator spacing of 0.5 and $C/(N_0+I_0)$ of 20 dB-Hz. With a narrow correlator spacing of 0.1, vertical position accuracy degrades above 2 meters for interference levels beyond $C/(N_0+I_0) = 32$ dB-Hz.

3.6.2 Impact on Integrity and Continuity

Integrity is only lost if pseudorange or position error is larger than we think. Increasing interference levels cause increasing pseudorange error, and would eventually cause a receiver to lose lock, resulting in a breach in continuity. As an example, consider an aircraft on approach, on a 3-degree glide slope. A CW interference source is located 1000 m from runway threshold, with power adjusted to equal broadcast GPS signal power for an aircraft 50 km away. Figure 30 shows the increase in received signal power as a function of aircraft distance from runway threshold. As the aircraft overflies the CW source, received interference power ranges from 0 dB to 60 dB, which from earlier analysis would jam the receiver, causing a loss of continuity.

3.6.3 Impact on Availability

High levels of interference would also prevent a receiver from acquiring any satellites. Acquisition occurs in a receiver when the correlator output power rises above a preset *detection* threshold, as described in section 2.2.5. The detection threshold for most receivers ranges from 3 dB to 6 dB above the noise floor (signal to noise ratio). As interference power levels increase, SNR decreases, and signal acquisition is impossible.

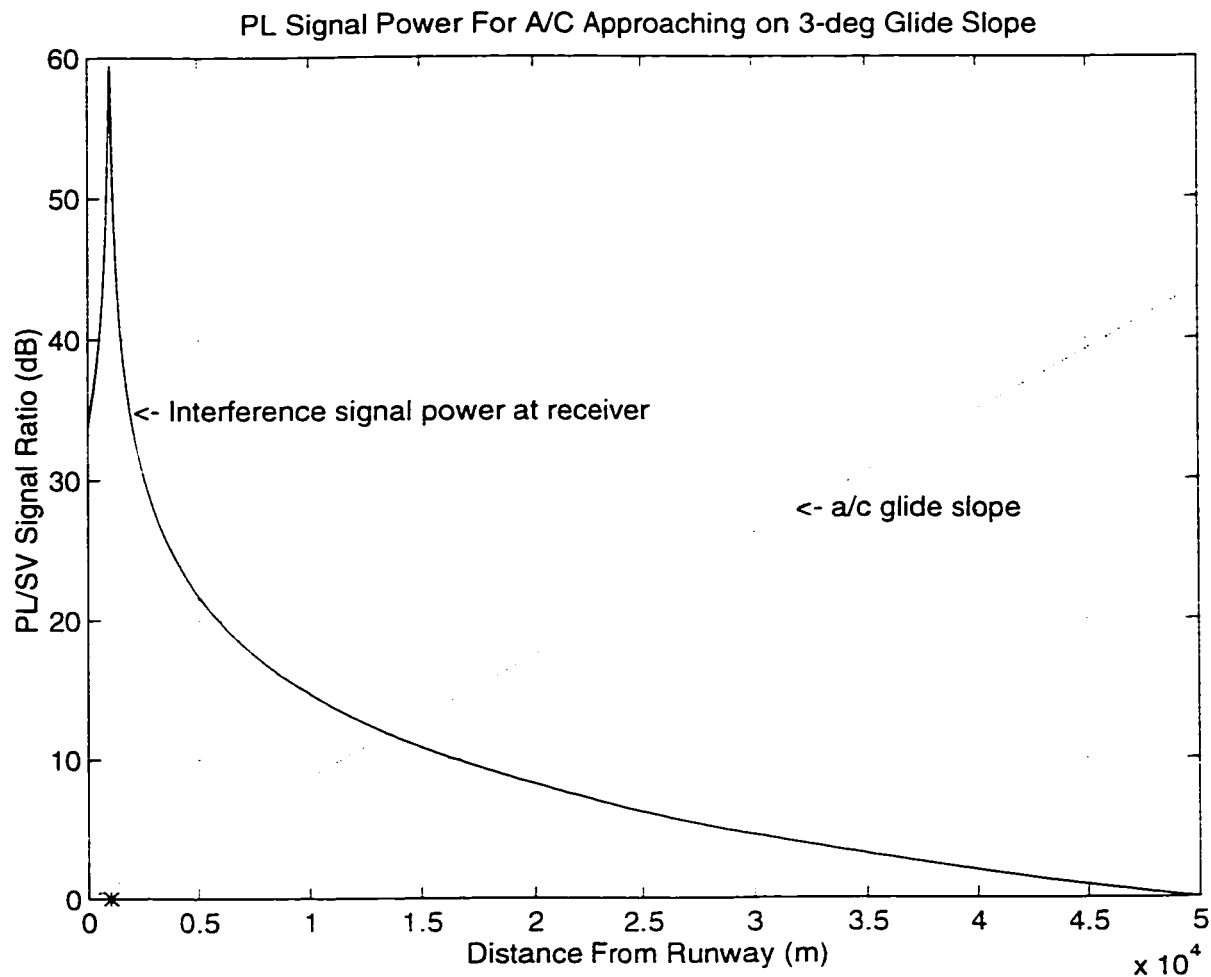


Figure 30: Interference Power Level as a Function of Aircraft Distance from Runway Threshold

A recent incident occurred at Stanford where an inadvertent source of RFI, a time-tagged remote camera transmitting images on in-band frequencies, disabled GPS operation across Stanford GPS labs, preventing receivers from acquiring any signal (zero availability). This installation was illegal and quickly modified by moving the frequency band, once the owners were notified.

Chapter 4

SIMULATION AND BENCH TEST SETUP

4.1 Introduction

A software simulation of GPS was developed to study the impact of interference on raw receiver measurements. This simulation generates GPS signals for all satellites in view including thermal noise and interference. Models were also developed to simulate the RF and digital processing modules of a GPS receiver. A simulation approach was chosen for the following reasons:

- i. A software receiver simulation provides access to all desired receiver measurements, which are not readily available in current receivers without hardware modification.
- ii. Truth data is readily available with a simulation. It was possible to monitor true pseudorange in software, and therefore error.
- iii. The GPS receiver-in-software allows for ease of configuration. One can readily change receiver components by modifying the receiver configuration input file.
- iv. The simulation provides a controlled environment which allows for accurate repeatability of test environments. This was necessary in order to compare the effects of various interference types on input signal.
- v. The software simulation facilitated the observation of receiver response to a wide range of interference scenarios, and the effect of signal attenuation.
- vi. New concepts in receiver design for integrity monitoring are readily implementable within software.

Bench testing was done using a real GPS receiver to validate our simulation. The following sections describes both the simulation and bench tests.

4.2 GPS Simulation

This software simulation captures with high fidelity the generation of GPS satellite signals, addition of thermal noise and interference, down conversion and filtering in the RF front end of a GPS receiver, and finally the digital processing for search, acquisition, and tracking of the GPS signals. The simulation performs 23×10^6 computation cycles to capture 1 second of real time.

The receiver-in-software is easily configurable to handle as many channels as the memory and processing power of the host computer can handle. However for this study, it was configured as a single channel receiver, as our interest lies in understanding fundamental receiver / interference interactions at the level of the correlator, code and carrier loops, and quantizer. In general the software simulation was designed to provide maximum flexibility for receiver configuration. By changing values in an input configuration file, a number of key receiver hardware and GPS parameters can be configured. These parameters include number of satellite signals generated, receiver thermal noise floor, correlator spacing, code tracking loop bandwidth, correlator integration time, detection and loss-of-lock thresholds, sampling rate of digitizer, and AGC loop bandwidth. An example of a configuration file is shown below in Table 4. A complete list of configurable parameters, and values used for each set of simulation runs are shown in appendix D. To facilitate validation, defaults for the receiver were set to simulate the GEC Plessey open architecture receiver hardware, used for bench tests.

Table 4: Sample Receiver Hardware Configuration Input File

```

//=====//
// Receiver Specifications      rcv_spec.in      //
// Awele Ndili, November 1996      //
// This file is read at the start of a run to      //
// set the properties of the receiver.      //
// Format:      //
// #define TOKENNAME      TOKENVALUE      //Optional comments      //
//=====//

//----Generic Constants
#define RUNID      501      // Unique Id for this Run
#define RUNTIME      6.000      // Total run time (seconds)
#define PRINT2SCREEN      NO      // Display summary to screen at 1kHz

//----Noise & Interference Input
#define INCLUDE_AWGN      YES      // Add AWGN? options are YES, NO
#define DELAY_B4_AWGN      3.0      // Delay before addition of AWGN,
// seconds
#define INCLUDE_BG_AWGN      YES      // Add Background AWGN (rcv noise)?
// options are YES, NO
#define BG_AWGNSR_DB      3      // Background AWGN level, in dB
#define INCLUDE_CW      NO      // Add CW interference? options are
// YES, NO
#define DELAY_B4_CW      3.0      // Delay before addition of CW int.,
// seconds
#define INCLUDE_PULSE      NO      // Add Pulsed (CW) interference?
// options are YES, NO
#define PULSING_SCHEME      RANDOM      // Pulse Scheme (pulse position) -
// options are FIXED, RANDOM
#define INCLUDE_SVATTENUATION      NO      // Attenuate SV signal? (simulates
// fading, blockages, multipath,
// etc) options are YES, NO
#define DELAY_B4_SVATTENUATION      3.0      // Delay before SV signal
// attenuation, seconds

//----Incoming Signal Properties
#define IF3_INIT_CAOFFSET      40      // Initial CA Code Offset
#define Init_CarrFreqErr      100.0      // Initial Carrier Freq. Error
// (Sig - DCO)
#define MAX_ALMANAC_SIZE      25      // Maximum number of GPS Satellites
// in almanac, default=25

//----Receiver Properties
#define IF4_Sample_Rate      5.714285714285714e+06      // Samples per second
#define MAXCHANNELS      1      // Number of channels this receiver has

```



```

//----Correlator Properties
#define INTEGRATIONTIME 0.001 // Sample n Dump time (seconds)
#define EarlyLateCorrSpace 0.5 // Chip Width (Note: 0.5=>.25 each
                                // side, for GEC Plessey)

//----Acquisition Properties
#define DetectionThreshold 971420 // (10dB) Signal acquisition
                                // threshold
#define LockLossThreshold 191437 // Signal loss-of-lock threshold
#define DCO_CA_Code_Srate 1.02325e6 // dco ca code rate during search
                                // (cycles per second)
#define DCO_CA_Code_Rate 1.023e6 // dco ca code rate after lock
                                // (cycles per second)
#define DCO_INIT_CAOFFSET 0 // Initial dco ca code offset
#define FreqSearchBinWidth 300 // Hz
#define DCOMaxDoppler 6000 // Hz
#define CARRIERSMOOTHINGON YES // Carrier smoothing on? on code DLL
                                // (YES OR NO) 980417

//----Tracking Loop Properties
#define COASTING_PERIOD 20 // Coasting period, in seconds
#define EMLREADINGS 40 // Number of EML readings to
                        // integrate for code tracking

//----Adaptive Quantizer Properties
#define ADAPTIVEQUANTIZERPERIOD 0.0005 // Time to reevaluate
                                // quantizer levels, in secs
#define A_QUANTIZER_INCREMENT 0.01 // Adaptive Quantizer
                                // adjustment increments

```

The entire simulation code is attached as appendix C of this thesis. The following sections give an overview of the process from signal generation to tracking.

4.2.1 Signal Generation and Downconversion

At startup, the signal generation module computes and stores C/A codes for PRN numbers 1 through 32 - all possible GPS C/A codes. This archive provides easy access to C/A codes for subsequent computation. Satellites in view are then determined based on a GPS almanac downloaded from a real receiver, and on a user located at San Francisco International airport. Mask angle is determined from the configuration input file. Satellite positions and velocities are then computed, which determines the correct Doppler frequency

for each satellite. A correction is applied to transmitted power of each satellite as a function of elevation to correct for the attenuation experienced by satellites signals in lower elevations due to the ionosphere and troposphere. This correction is determined empirically, from a curve fit of real signal strength data collected over a period of months at Stanford. This curve fit is shown below in Figure 31.

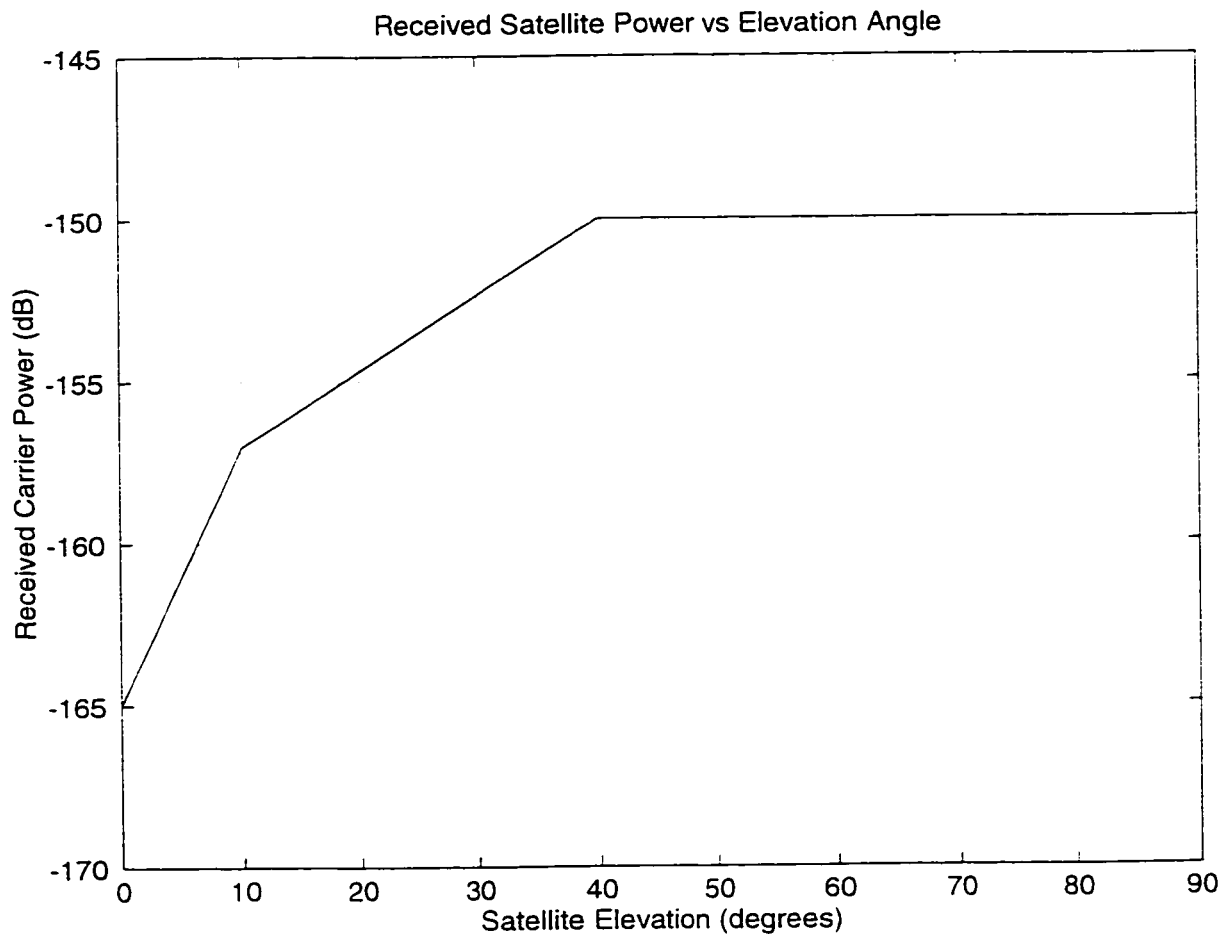


Figure 31: Received Satellite Signal Strength as a function of Satellite Elevation – a Curve Fit based on Empirical Data

The C/A code is then modulated with the carrier, at the appropriate computed signal power for each GPS satellite in view. As simulation time progresses carrier and code phases are updated for all simulated satellite signals. The 50 Hz GPS data stream is not included in

this simulation, as this data was not needed to obtain true pseudorange.

Three-stage down analog downconversion process of the GEC Plessey card is simulated in a single step: all computed satellite carrier frequencies are adjusted by an amount equal to the sum of the three local oscillator frequencies, as shown in Figure 32.

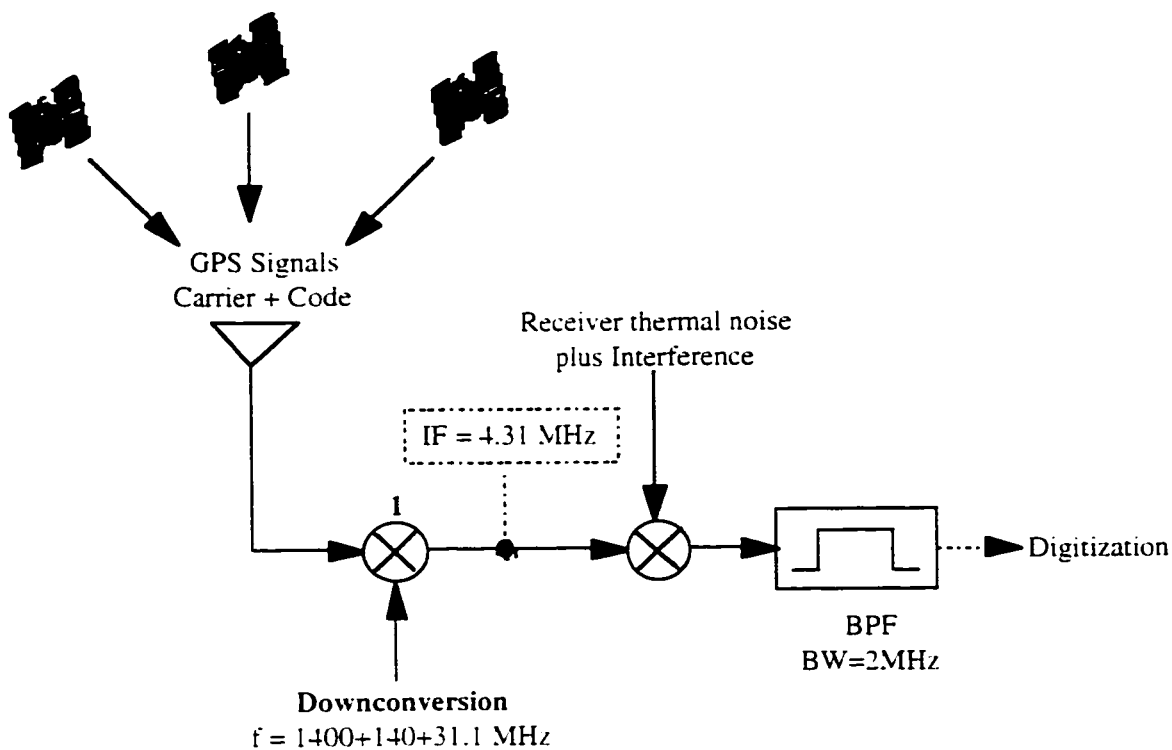


Figure 32: GPS Simulation Signal Generation and Downconversion

Downconversion is simulated by directly computing the GPS signals at the intermediate frequency $IF_3 = 4.31$ MHz plus Doppler, sampled at 22.86 MHz to provide continuity. Figure 33 shows the computed signal for a single satellite. With the time axis in microseconds, Figure 33 shows not only the 4.31 MHz carrier, but also the 1.023 MHz modulating CA code. The composite RF signal, consisting of the summation of signals from all satellites in view is shown in Figure 34. At this point, thermal noise and interference have not yet been added to the signal.

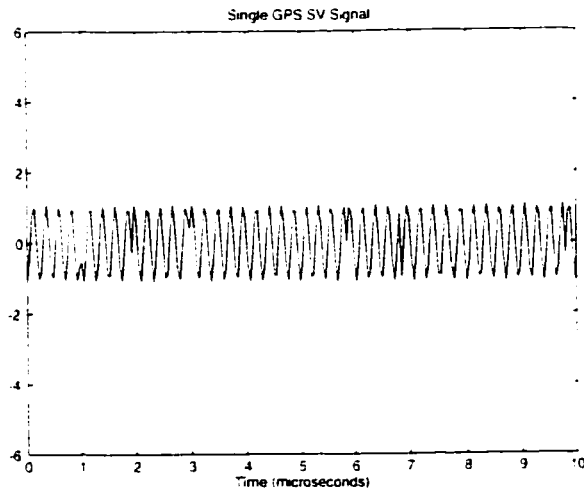


Figure 33: RF Signal from Single GPS Satellite at $IF_1 = 4.31$ MHz

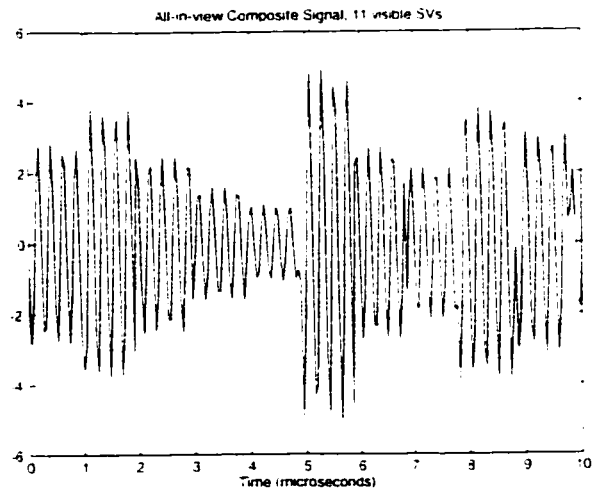


Figure 34: Composite Signal from All (11) In-View Satellites at $IF_1 = 4.31$ MHz

Receiver thermal noise is computed as AWGN using a Box-Muller normal random variate generator [61]. The noise power level is specified by the input configuration file. Figure 35 shows a plot of receiver thermal noise, at a power level of 8 dB relative to the GPS satellite signal being tracked, or equivalently at a noise power density of -205 dBW/Hz in a 2 MHz bandwidth.

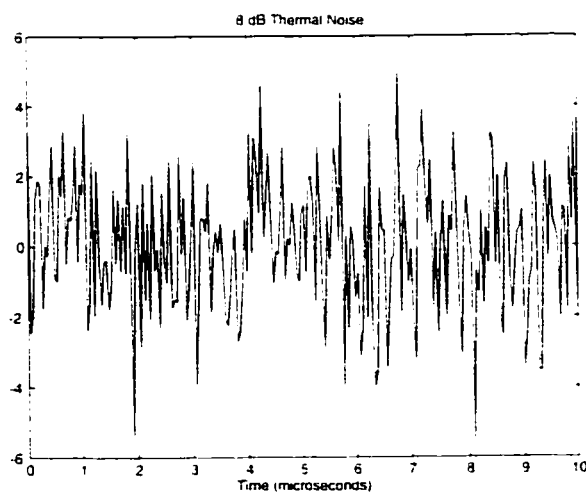


Figure 35: 8 dB Thermal Noise at IF_1

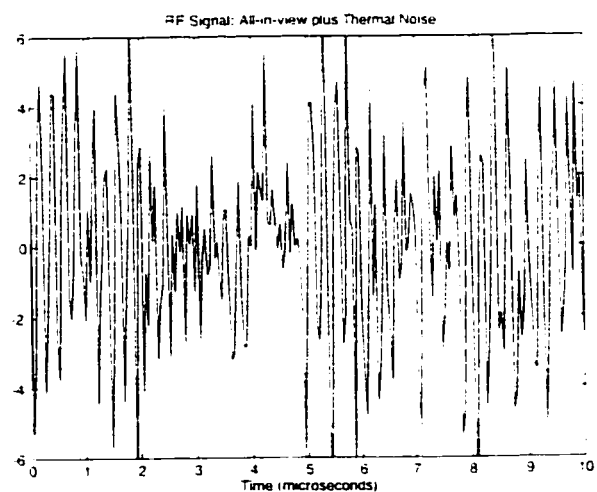


Figure 36: Thermal Noise plus All-in-View RF Signal prior to Filtering

The computed thermal noise is then added to the satellite signals at the intermediate frequency IF_1 . Interference of the type and power level specified by run input file is also computed and added to the composite signal at this stage, as shown in Figure 32. The composite signal consisting of all satellites in view plus thermal noise is shown in Figure 36

This signal is then passed through a 4th order Butterworth bandpass filter with a 1 MHz one sided bandwidth. The output from this filter is a signal at a nominal RF frequency of 4.31 MHz, sampled at 22.86 MHz, which is the input to the analog to digital converter (ADC).

4.2.2 Analog to Digital Converter

Sampling at an IF_4 sampling frequency of 5.714 MHz is achieved by subsampling the output of the bandpass filter. This is achieved in software by passing only one out of every four samples from the filtered 22.86 MHz signal. Oversampling produces a phase reversal on the resulting signal, which has nominal frequency = 1.4 MHz. This action is demonstrated by the equation:

$$4.31 \text{ MHz} - 5.71 \text{ MHz} = -1.4 \text{ MHz.} \quad (4.2.2.1)$$

where the negative sign indicates the occurrence of a phase reversal.

Figure 37 shows the resulting signal, derived from sampling the composite all-in-view plus thermal noise IF_3 signal. The nominal frequency of 1.4 MHz can be clearly seen. At this stage the output is still maintained as a real number within software, pending quantization.

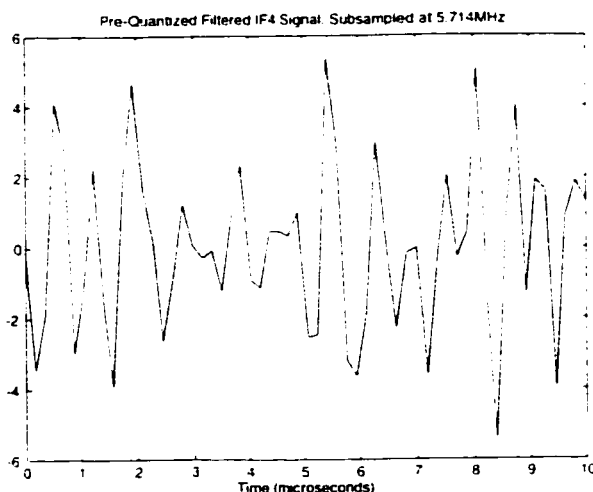


Figure 37: Filtered IF_4 Signal sampled at 5.714 MHz. Nominal Freq. = 1.41 MHz

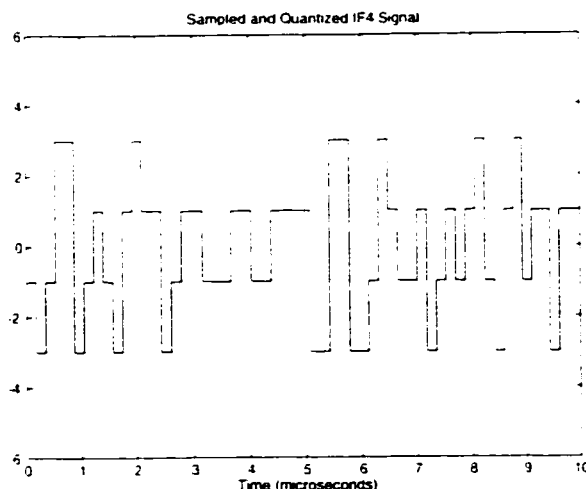


Figure 38: Sampled and Quantized Composite IF_4 Signal

The 'real' signal is finally passed through the quantizer module, which emulates an adaptive 2-bit quantizer, converting the signal into the four distinct values $\pm 1, \pm 3$. Figure 38 shows the digital output of the quantizer. This digital signal will always have its signal level maintained within $-3, -1, 1, 3$ by the 2 bit analog to digital converter, irrespective of the level of the input signal or noise.

The AGC tracking loop, which sets the quantization thresholds (Figure 18) is implemented in the quantizer module using integrators. Counts for each of the four distinct output levels are accumulated in four bins. Periodically the bins are compared to determine the ratios between each quantizer level. The interval between comparisons, AGC loop bandwidth, is set within the configuration input file. Adjustments are made to the thresholds (Figure 18) to maintain output levels in the ratio $-3, -1, +1, +3 : 15\%, 35\%, 35\%, 15\%$.

4.2.3 Correlation

The signal IF_4 which contains GPS code at a nominal frequency of 1.41 MHz is fed into a bank of correlators. A local signal for the satellite being tracked is generated having a

carrier also at 1.41 MHz. This local signal has the same C/A code as the satellite to be tracked, however the code frequency is initially set 0.25 chips/second faster than expected, to achieve the sliding search described in section 2.2.5. Four versions of this local signal are computed and applied to the incoming signal: early inphase, early quadrature, late inphase and late quadrature signals, as shown in Figure 39. The products are then summed over 5,714 epochs (1 ms). Integration outputs, designated at I_e , Q_e , I_l , Q_l , are used to compute the virtual prompt signal I_p , Q_p , according to equation 2.2.4.1. I_p and Q_p are used to determine signal acquisition, as described in section 2.2.5.

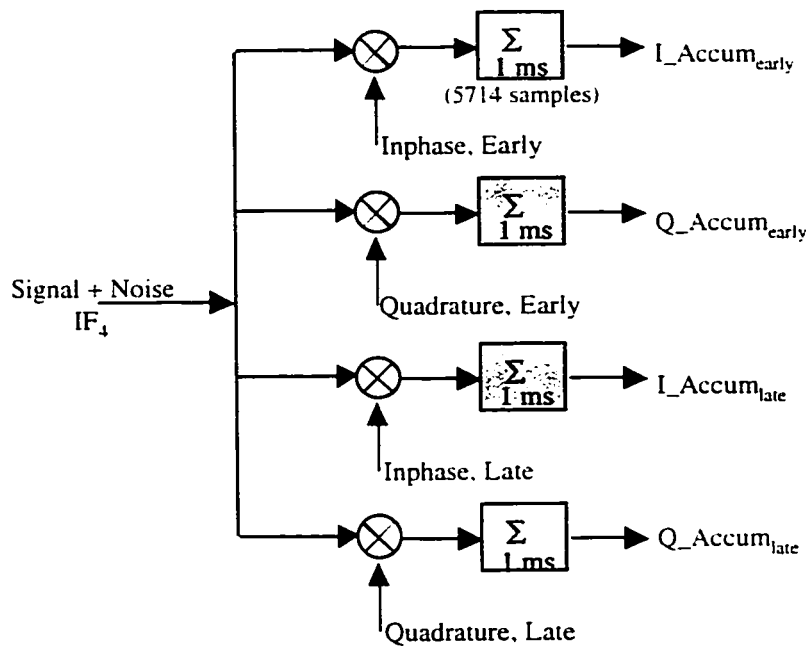


Figure 39: Correlation on Single Channel

4.2.4 Tracking

Two digitally controlled oscillators (DCOs), also called NCOs for numerically controlled oscillators, are maintained in software to generate the phases of the local code and carrier. The output frequency, f_o , of a DCO is determined by a 'control word' written to its register. This frequency is given by:

$$f_o = w * \frac{f_c}{2^N} \quad (4.2.4.1)$$

where f_c = DCO clock frequency, a constant. = 5.714 MHz

w = DCO control word

N = Number of bits in DCO

The control word, w , may range from 0 to a maximum of 2^N , giving an output frequency range, f_o , of 0 to 5.714 MHz.

Corrections are applied to code and carrier DCOs by the code and carrier tracking loops respectively, according to the transfer functions described in section 2.2.6. The conversion gain, K_o , for each DCO, translates input control word into frequency changes as shown in Figure 14. Values for K_o are set to be same as for GEC Plessey GP2021 correlator hardware.

4.3 Bench Test Setup

The bench test layout is shown in Figure 40. A GEC Plessey receiver card was used, as this gave easy access to most of the measurements being studied. This card was housed in a Pentium 166 MHz computer, and a WelNavigate GPS Signal Generator was used to generate all of the GPS signals in view. Broadband noise was generated using a WelNavigate noise generator, which had the capability to generate broadband and narrow band noise over a 60 dB power range, adjustable in 1 and 10 dB steps. CW Interference was generated using an HP 8648B signal generator capable of generating CW signals in the frequency range 0 - 2 GHz, with power output up to +20 dBm (100 milliwatts). To generate pulsed signals, a WaveTek FG3B pulse generator was used to drive both the HP signal generator and the WelNavigate noise generator. GPS signals were combined with noise and interference using a MiniCircuits zero-phase power combiner (part number

ZESC-2-11), which has an insertion loss of 3.5dB.

Figure 40 shows the CW generator (HP8648B), the broadband noise generator, and the pulse signal generator, in the configuration for generating pulsed interference. Continuous interference was generated by disabling on/off gating by the pulse generator. Actual hardware is shown in the photo (Figure 41).

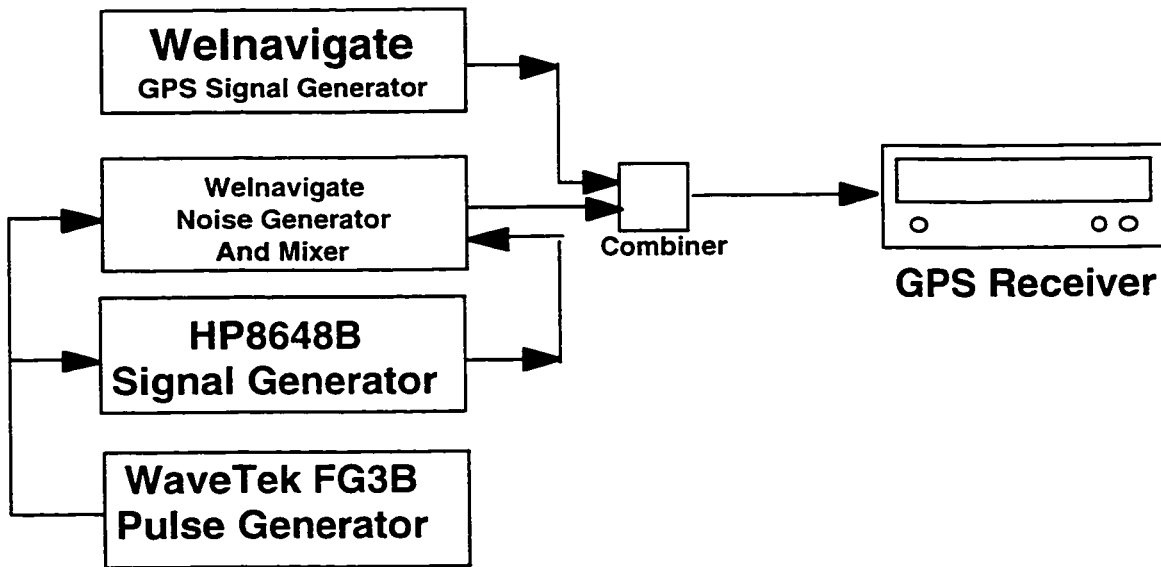


Figure 40: Bench Test Layout

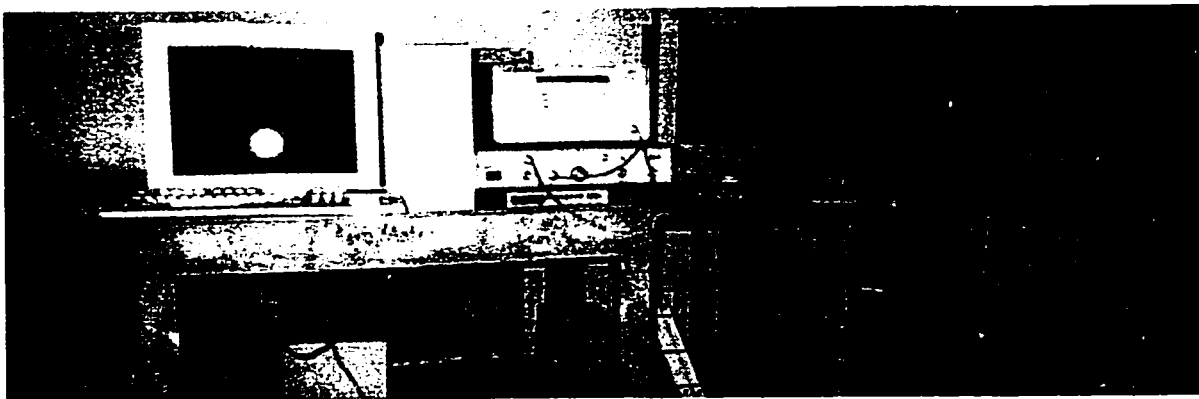


Figure 41: Bench Test Hardware showing the receiver-under-test (leftmost), CW broadband and pulse generator (stacked in middle), and GPS Signal Generator (right).

4.4 Test Procedures

The test procedures for both software simulation and bench tests were similar. The receiver-under-test was first allowed enough time to search, acquire and track the GPS signals before the introduction of interference, since our focus is on integrity – if there is no acquisition, then there exists no integrity risk.

In software, search time was reduced by setting the initial C/A code offset of the tracked satellite to a small enough value, 40 chips, that allowed acquisition within 160 ms of simulation time. Initial frequency error was also set a small value (100 Hz). For bench tests, the time-to-first fix could not be rigidly set, therefore 3 minutes was allowed for acquisition.

After acquisition, enough time was given for the receiver tracking loops to settle before the introduction of interference. This takes 3 seconds in software, however, for bench testing 2 minutes was allowed due to the unpredictability of time-to-first fix. With the receiver in steady state tracking mode a specified type and level of interference is then introduced and sustained for the duration of the run (6 seconds for software or 35 seconds for bench tests). During the entire software simulation pseudorange error, AGC gain, and raw I/Q receiver measurements were captured to file, for post-analysis. For bench tests data was captured only during the 35 seconds of interference testing, including 3 seconds before the introduction of interference, excluding the prior 5 minutes (acquisition plus loop settle time), due to the huge amounts of data that would result, and limitations in memory. Also AGC gain was not captured for bench tests, since there was no access to the AGC in bench test hardware.

Two types of input files were used to control the simulation. The first, "Hardware Configuration File", specifies properties of the receiver such as tracking loop bandwidths

**Table 5: Sample Run Configuration File for Continuous CW Interference,
with Power Level ranging from 0 to 40 dB**

AWGN (dB)	CW (dB)	Pulse Cycle	Duty (%)	CW Offset	Doppler (Hz)	SV Attenuation (dB)
0	0	0		0		0
0	1	0		0		0
0	2	0		0		0
0	3	0		0		0
0	4	0		0		0
0	5	0		0		0
0	6	0		0		0
0	7	0		0		0
0	8	0		0		0
0	9	0		0		0
0	10	0		0		0
0	11	0		0		0
0	12	0		0		0
0	13	0		0		0
0	14	0		0		0
0	15	0		0		0
0	16	0		0		0
0	17	0		0		0
0	18	0		0		0
0	19	0		0		0
0	20	0		0		0
0	21	0		0		0
0	22	0		0		0
0	23	0		0		0
0	24	0		0		0
0	25	0		0		0
0	26	0		0		0
0	27	0		0		0
0	28	0		0		0
0	29	0		0		0
0	30	0		0		0
0	31	0		0		0
0	32	0		0		0
0	33	0		0		0
0	34	0		0		0
0	35	0		0		0
0	36	0		0		0
0	37	0		0		0
0	38	0		0		0
0	39	0		0		0
0	40	0		0		0
-1	-1	-1		-1		-1
AWGN_DB	CW_DB	PULSE_DC		DOPPLER_OFFSET		SV_ATTENUATION_DB

and correlator spacing. Also specified in this file are receiver noise floor and type of interference (CW, pulsed, fading, etc.). A sample file is shown in Table 4. Configurations used for all groups of test are shown in appendix D.

The second input file called the "Run Configuration File" defines a "group" of runs. Runs were grouped according to interference type. The Run Configuration Input files were developed for each interference-type group to permit the scheduling of a series of similar runs, with same hardware specifications, but varying interference power levels. For example, Table 5 shows the input file for the CW tests (at zero Doppler offset), consisting of 41 different runs, each run having a different CW interference-to-signal power ratio, spanning 0 through 40 dB. All Run Configuration Input files for all interference groups are shown in appendix E

The simulation was run on a Digital DEC 4100 3-cpu machine operating at 410 MHz. It typically took about 30 minutes to complete a single run, or 20 hours to complete a single group of 40 runs. (Pulsed interference tests, in groups of 100, would take 50 hours to complete).

Figure 42 shows sample software runs, for 26 dB of AWGN and 18 dB of CW interference. Signal acquisition occurs 160 ms into the run. This is indicated by the boost in correlator output power as shown in the COP plots, and the sudden reduction in carrier phase jitter. For the AWGN run, the tracking loops stabilize 2 seconds into the run, as shown by the settling of the delay lock loop, shown in fractions of a C/A code chip (top plot of Figure 42a). With CW interference, code tracking loop shows more instability (top plot of Figure 42b). Also shown is the recovery of COP from its severe drop due to strong CW interference, as the AGC gain increases to suppress interference (bottom two plots of Figure 42b).

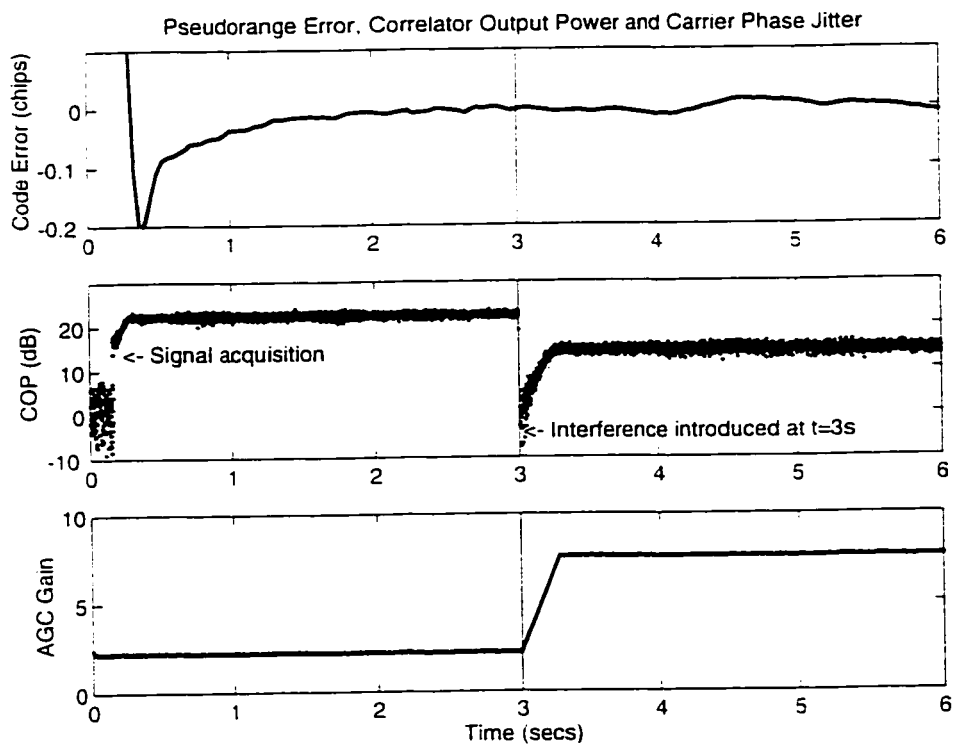
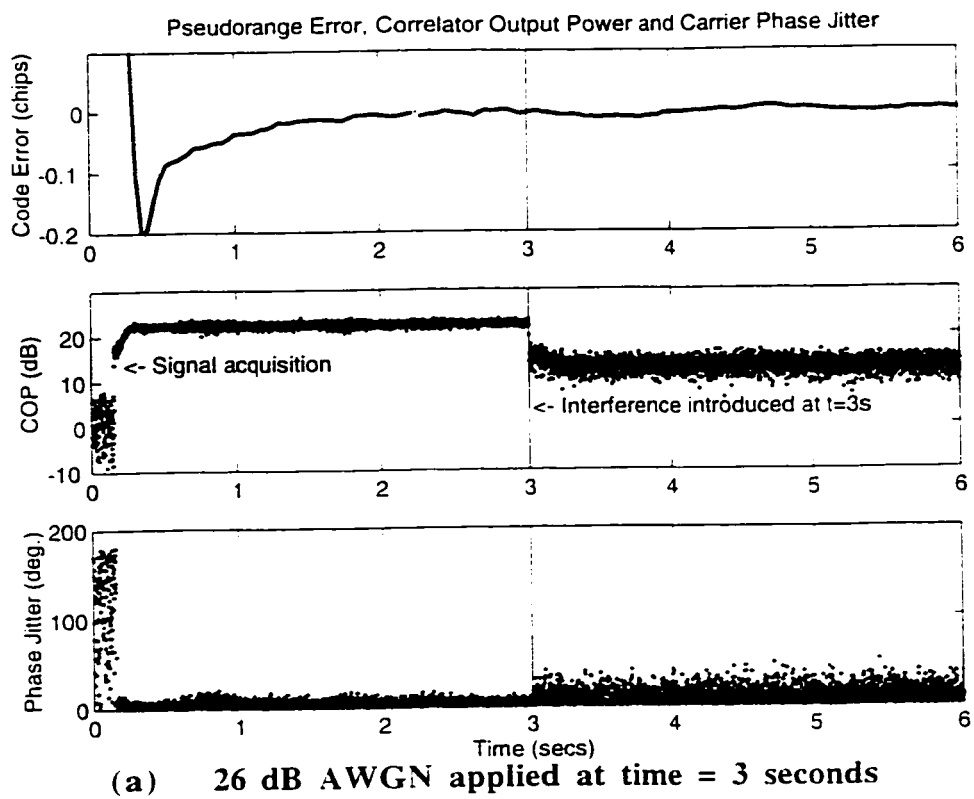


Figure 42: Sample Simulation Runs: 26 dB AWGN, 18 dB CW Interference

Chapter 5

INTEGRITY MONITORING RESULTS

5.1 Introduction

Table 6 below gives an overview of results presented in this chapter. The first column shows different types of interference applied. Coherent CW was introduced at two frequencies: $L1 + 1$ KHz, which represents the first and mildest spectral line, and $L1 + 7$ KHz, which represents the most severe spectral line for satellite PRN #1 (Figure 26). The second column shows the integrity test statistics used to predict the pseudorange error (third column).

Table 6: Overview of Presented Results

Causes	Effects	
	Observable Test Statistics	Non-Observable Parameter to Protect
<ul style="list-style-type: none"> - AWGN - CW Interference - Pulsed CW - Pulsed AWGN - Coherent CW hot/cold spectral lines - Multipath, Fading 	<ul style="list-style-type: none"> - Correlator Output Power - COP Variance - Carrier Phase Vacillation - AGC Gain 	<ul style="list-style-type: none"> - Pseudorange Error

The ultimate goal of integrity monitoring is to reliably predict degradation in pseudorange accuracy caused by the broad range of interference types shown in column 1 of Table 6, by observing the test statistics in column 2. It is therefore desirable that the selected test statistic be sensitive to interference, while at the same time maintain minimal sensitivity to *variations in types* of interference for robust performance.

In the next section, results are presented which show the pseudorange accuracy dependency on interference. Section 5.3 shows the correlation between our selected test statistic and interference. Section 5.4 shows results for integrity monitoring using selected test statistics. Results for bench test validation are presented and discussed in section 5.5.

5.2 Impact of Interference on Code and Carrier Tracking Loops

5.2.1 Impact of Continuous RF Interference on Code and Carrier Tracking Loops

Figure 43 shows the degradation in pseudorange accuracy as a function of interference power, for the following types of interference: AWGN, CW at L1, CW at L1 + 1 KHz, CW at L1 + 7 KHz. For these types of interference, as interference power increases - denoted by decreasing C/N_0 , there is initially no noticeable increase in pseudorange error. This is due to fact that interference is buried under the receiver noise floor at low interference power levels in the regime from $C/N_0 = 55$ dB-Hz to 45 dB-Hz. As interference power increases beyond the C/N_0 knee at 45 dB-Hz, pseudorange error increases, until loss of lock occurs. For the same values of interference power, Figure 43 shows that CW interference produces more severe results than AWGN, as to be expected. Agreement with theory can be observed from the solid line in Figure 43, which shows the theoretical prediction of pseudorange error degradation, according to equation 3.6.1.1 [20]:

$$\sigma_k^2 = \frac{d(cT_c)^2 BW_{L1\text{-sided}}}{2 \left(\frac{C(eI^k)}{N_0 + I_0} \right)} \left[1 + \frac{2}{(2-d) \left(\frac{C(eI^k)}{N_0 + I_0} \right) T_{\text{square}}} \right] \quad (3.6.1.1)$$

where d = correlator spacing;

T_c = C/A code chip width = 1 μ s;

$BW_{L,1\text{-sided}}$ = 1-sided tracking loop bandwidth:

T_{square} = squaring loss, from early-late power detector:

$C(e l^k)$ = carrier power of the k^{th} satellite, which is a function of its elevation angle $e l^k$.

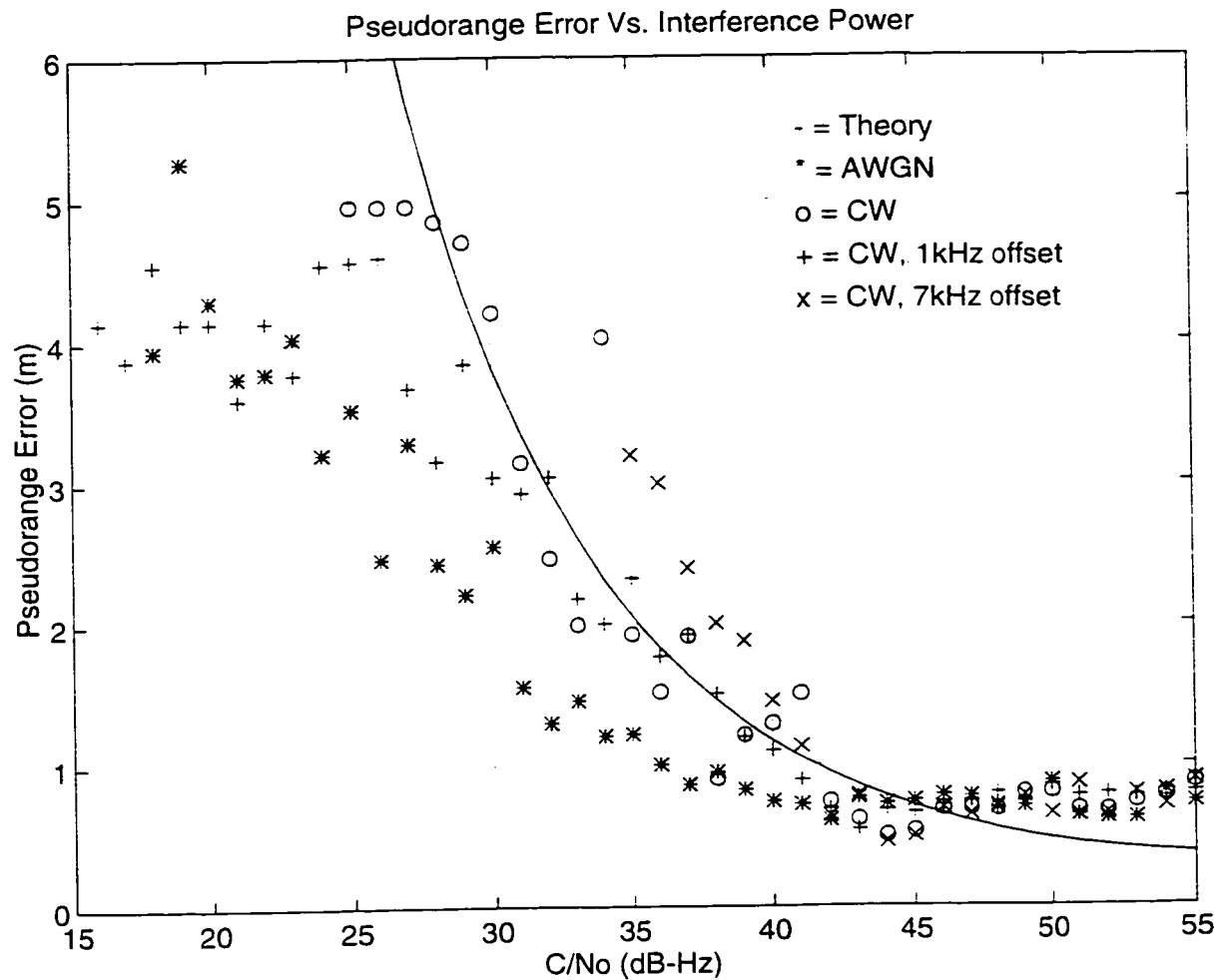


Figure 43: Code Tracking Loop Error as function of Continuous RF Interference Power: Pseudorange Error vs. $C/(N_o + I_o)$

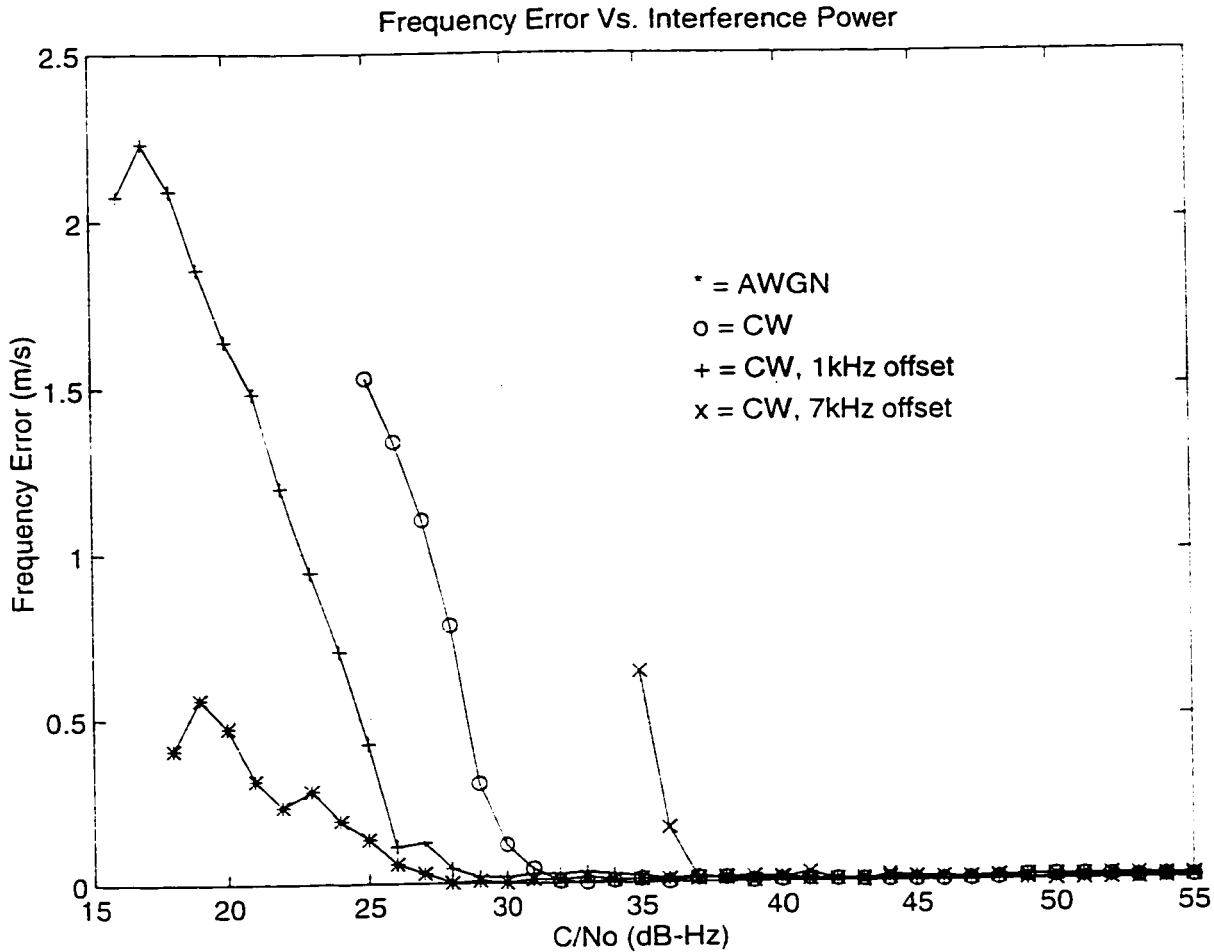


Figure 44: Carrier Tracking Loop Error as function of Continuous RF Interference Power: Mean Carrier Frequency Error vs. $C/(N_o + I_o)$

Figure 44 shows the mean frequency error as a function of interference for the same four types of interference. It can be seen that the frequency locked loop is less sensitive to RF interference. Most severe is CW at L1+7 KHz, which causes a loss of lock to occur earlier, at a $C/(N_o + I_o) = 35$ dB-Hz. Next sensitive is CW at L1, then at L1+1 KHz, which corresponds to a mild C/A code spectral line. Figure 44 shows that the frequency locked loop performs best against AWGN.

5.2.2 Impact of Pulsed RF Interference on Code and Carrier Tracking Loops

The effect of pulsed interference on pseudorange error can be seen from Figure 45. For

pulse duty cycles below 20%, there is no significant impact on pseudorange accuracy. The minimal impact of low duty cycles on GPS accuracy bears significance for applications such as pseudolite signal design, and will be utilized later in this thesis. Beyond 20% duty cycle, pseudorange accuracy degrades in a linear manner, until loss of lock occurs at a duty cycle of 68% for CW interference, and 90% for AWGN. The impact of pulsed

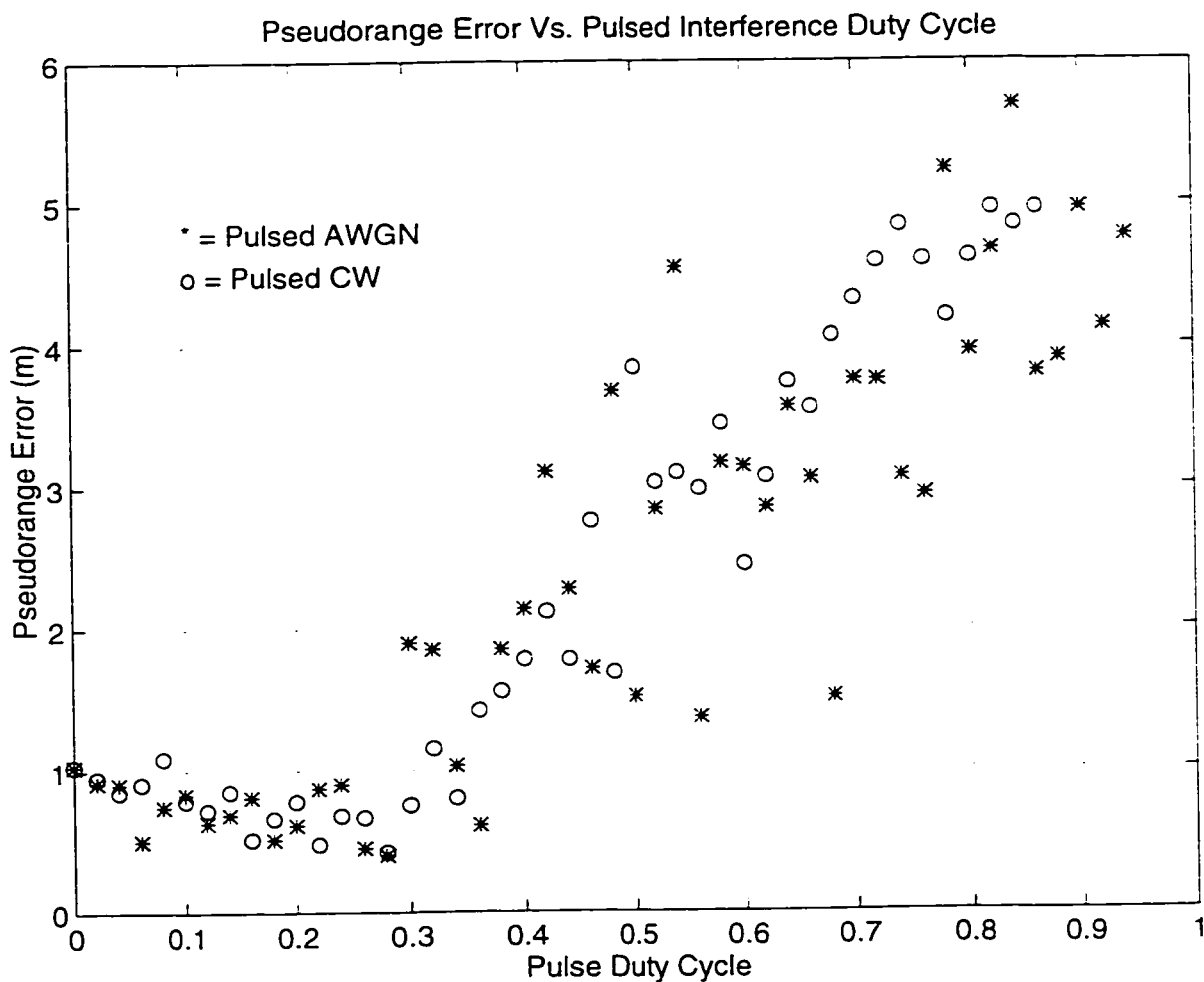


Figure 45: Code Tracking Loop Error as function of Pulsed RF Interference Power: Pseudorange Error vs. Pulse Duty Cycle

interference on pseudorange accuracy is directly related to the bandwidth of the AGC. A fast AGC in effect suppresses pulsed interference [15].

Pulsing produces little effect on the carrier tracking loop below duty cycles of 70%, as can be seen from Figure 46. Beyond 70% duty cycle, frequency error grows rapidly until loss of lock occurs. In general, frequency locked loops will show better performance than phase locked loops in interference [9].

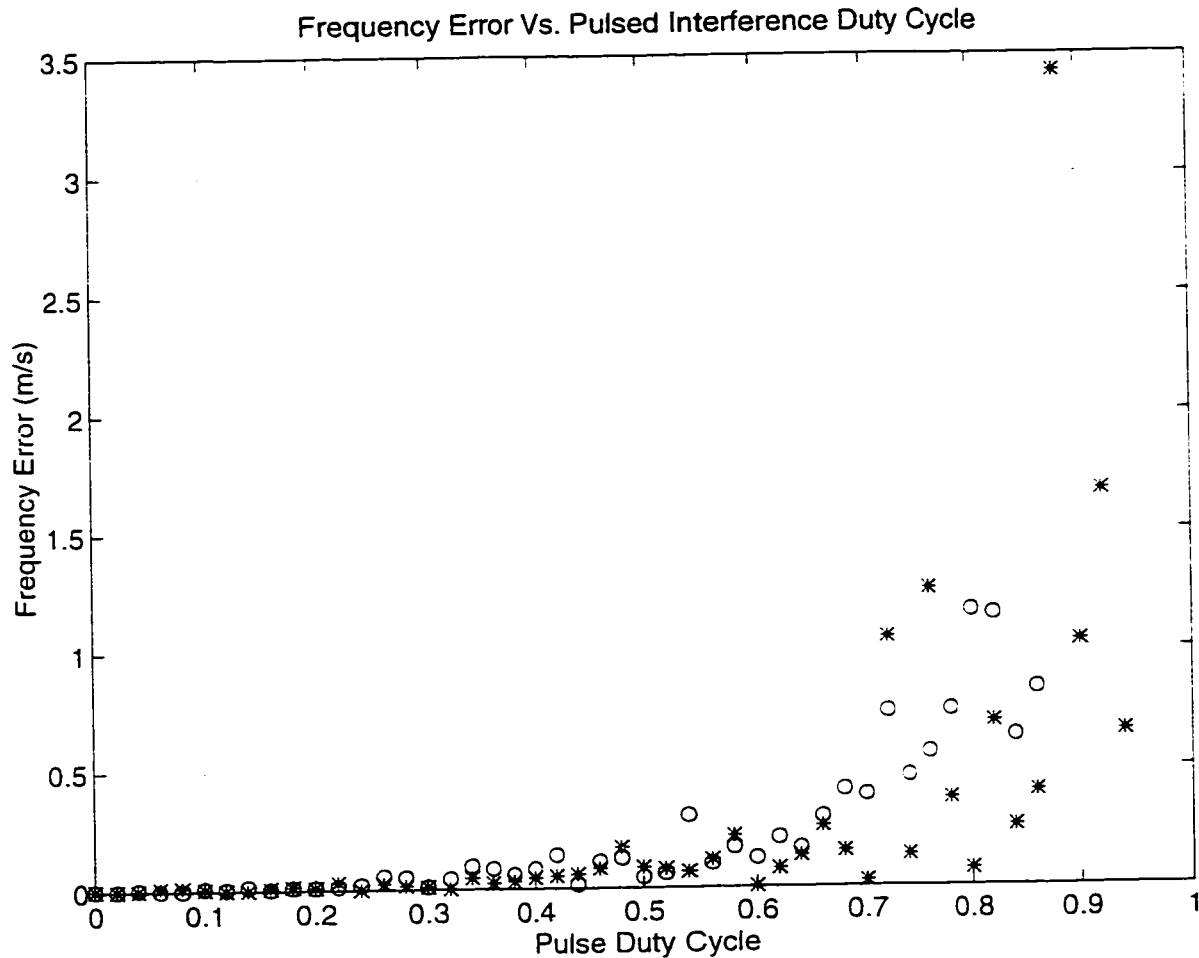


Figure 46: Carrier Tracking Loop Error as function of Pulsed RF Interference Power: Mean Carrier Frequency Error vs. Pulse Duty Cycle

5.2.3 Impact of Signal Attenuation on Code and Carrier Tracking Loops

The effect of signal blockage on a receiver's code and carrier tracking loops can be seen from Figures 47 and 48. The initial value of C/N_0 corresponding to zero attenuation is 55

dB-Hz. Below 8 dB of attenuation, there is little effect on the code and carrier loops, as seen from the figures, with frequency error less than 0.1 m/s, and pseudorange error below

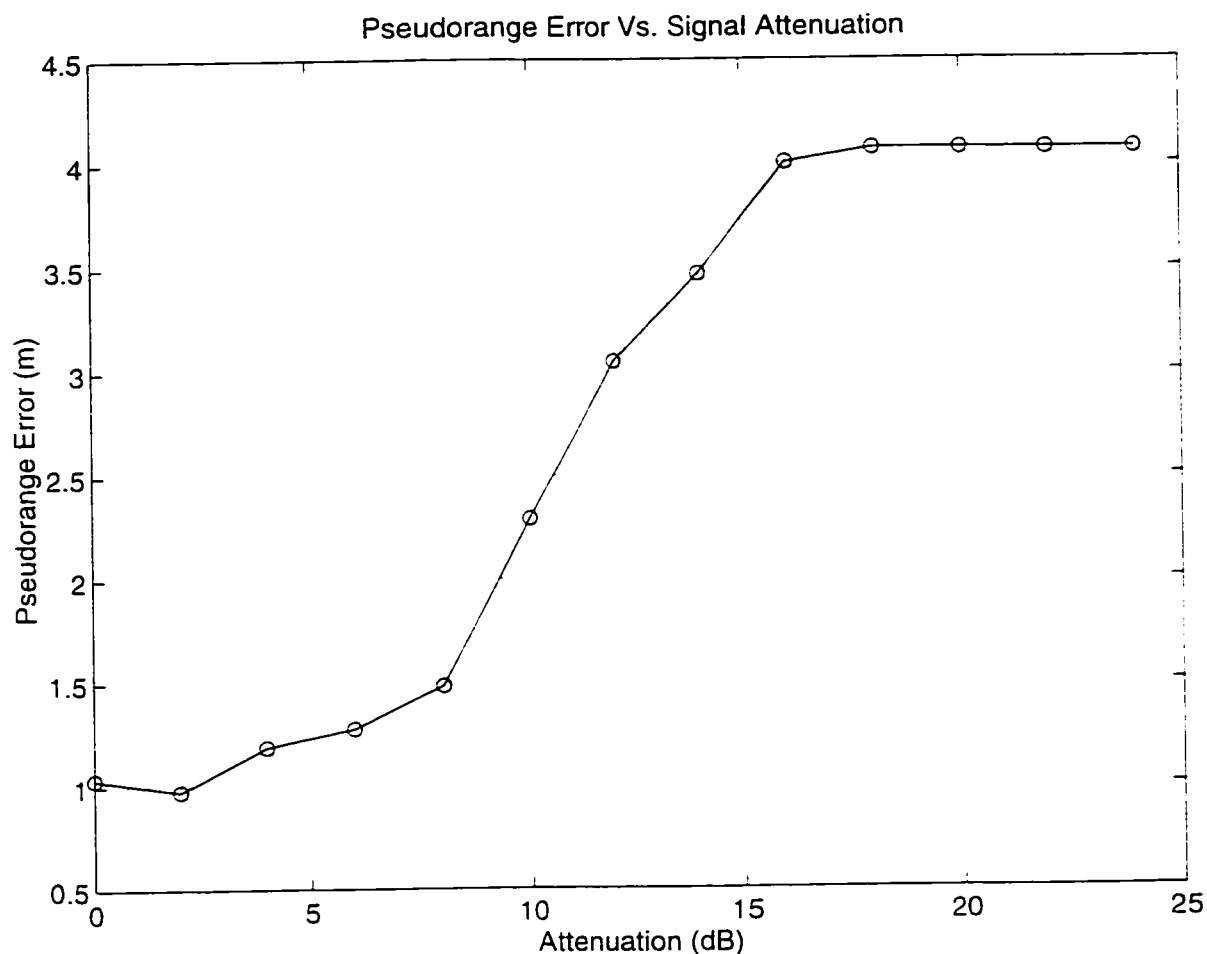


Figure 47: Code Tracking Loop Error as function of Signal Attenuation: Pseudorange Error vs. Signal Attenuation

1.5 m. As attenuation increases beyond 8 dB, both frequency and pseudorange accuracy degrade rapidly. A maximum pseudorange error of 4.1 m, and frequency error of 2.75 m/s, is attained at the onset of loss of lock. In practice the response of a receiver to attenuation of a tracked signal will depend on the receiver noise floor.

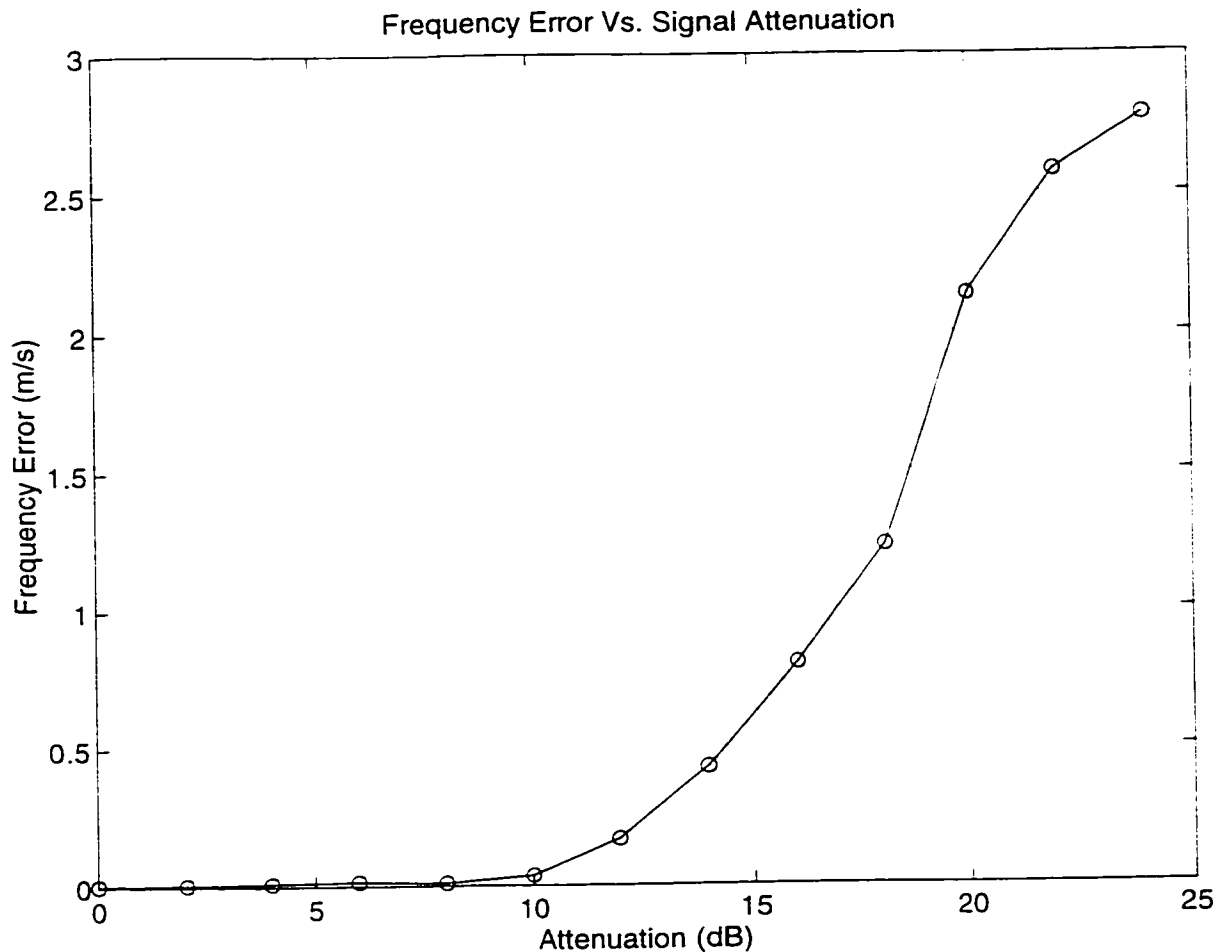


Figure 48: Carrier Tracking Loop Error as function of Signal Attenuation: Mean Carrier Frequency Error vs. Signal Attenuation

5.3 Impact of Interference on Test Statistics

5.3.1 Impact of Continuous RF Interference on Test Statistics

Figure 49 shows a composite plot of the effect of continuous RF interference on all four selected test statistics. As with pseudorange error, there is little noticeable effect of interference on our test statistics in the C/N_0 from regime 55 dB-Hz to 47 dB-Hz. Beyond this region COP decreases monotonically with increasing interference (or decreasing C/N_0), while the variance of the correlator output power, carrier phase jitter and AGC gain

increase. The increase in AGC gain is identical for all three kinds of CW interference. This increase is to be expected since the AGC is sensitive to total noise power in the signal, and all three kinds of CW interference fall within the bandwidth of RF front end band pass filter, therefore the same interference power, equivalent to the total applied interference power, is observed by the AGC.

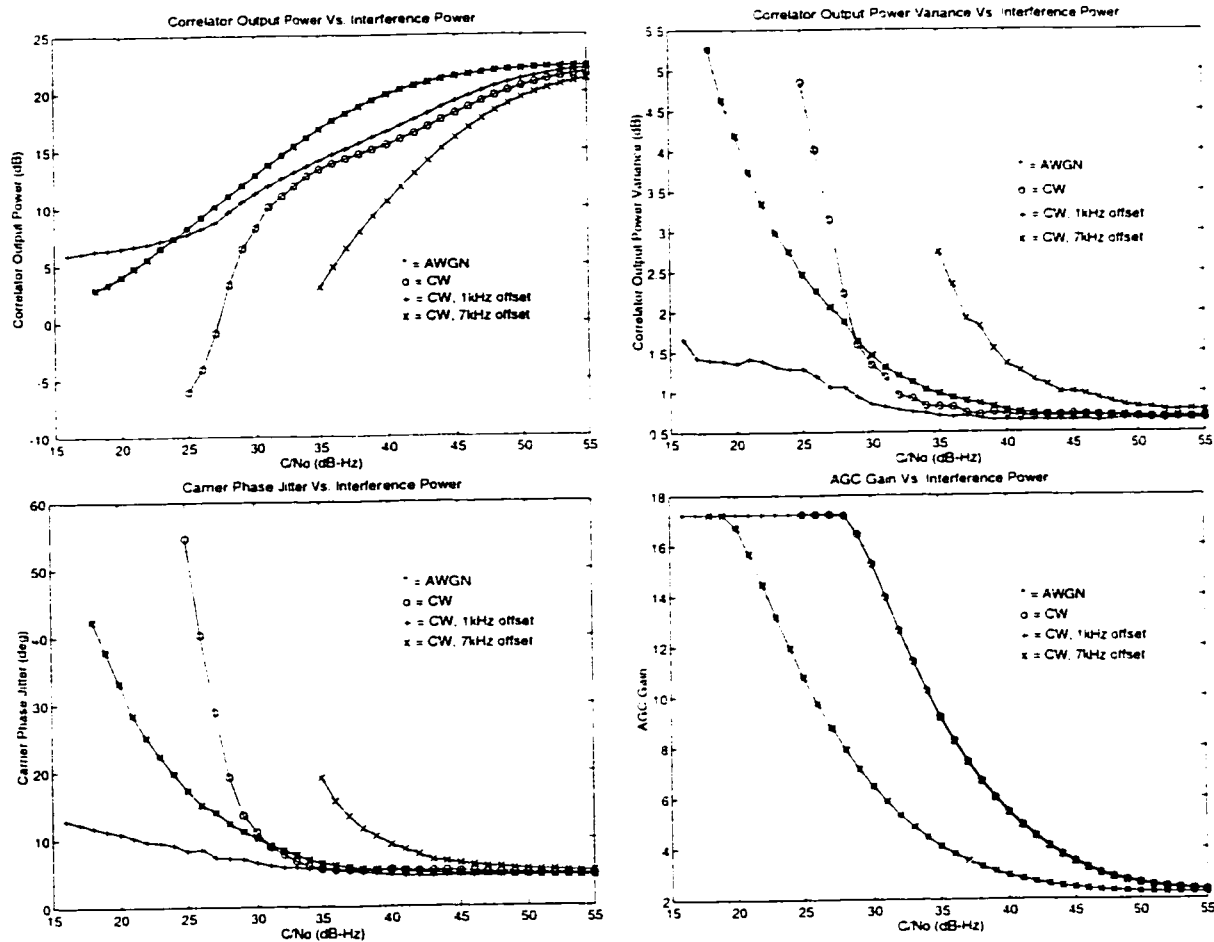


Figure 49: Impact of Continuous RF Interference on Test Statistics

Continuous RF Interference consists of AWGN, CW at L1, CW at L1 + 1 KHz, and CW at L1 + 7 KHz. (a) Impact of Continuous RFI on Correlator Output Power; (b) Impact of Continuous RFI on Correlator Output Power Variance; (c) Impact of Continuous RFI on Carrier Phase Jitter; (d) Impact of Continuous RFI on AGC Gain

On the other hand, AWGN shows a lower magnitude AGC response, since its energy is spread across the a wider bandwidth, and therefore some AWGN interference energy gets filtered out at the receiver RF front end. AGC gain peaks at its maximum 1-second-after value of 17 units. The quantizer thresholds continue to increase, after the 4th second into the test, when the values displayed in Figure 49 are taken.

AWGN causes the largest swing in variance of correlator output power variance, while CW interference causes the largest drop in COP : -6 dB, from Figure 49a. Note that this dip is temporal - the receiver recovers quickly. The receiver does not lose lock, since the time-averaged correlator output power used as a lock indicator does not dip below the loss of lock threshold.

It is significant to note the difference in severity across types of interference. As we expect, CW interference in general produces the most severe degradation in COP: CW at L1 + 7 KHz, a strong C/A code spectral, produces the most severe impact, and CW at L1 + 1KHz (a mild spectral line) produces the weakest CW effect. Severe CW at L1 + 7 KHz causes the receiver to loose lock fastest, while CW at the mild L1 + 1 KHz spectral line least impacts the all four test statistics. This behavior is similar to the interference characterization of pseudorange error, which lends confidence to the suitability of the selected test statistics.

5.3.2 Impact of Pulsed RF Interference on Test Statistics

With a pulse peak power of +150 dB relative to the tracked GPS signal, equivalent to 1 watt, the pulsed interference in effect completely blanks out the tracked GPS signal for the duration of the pulse. As a result the effective GPS signal power as perceived by the receiver is a linear function of pulse duty cycle. This relationship for the correlator output power can be observed from Figure 50a. From an initial value of 22 dB at 0 % duty cycle

(pulse off), the correlator output power degrades linearly with duty cycle; at 50 % duty cycle COP is about 11 dB, half of its original value. Further increase in pulse duty cycle results in loss of lock beyond 86% and 92% for CW and AWGN respectively, as the detected signal power drops below the preset receiver threshold.

Pulsed AWGN interference produces greater jitter in carrier phase, and variance in COP than pulsed CW for duty cycles greater than 40 %. The effects of pulsed CW and AWGN on AGC gain are identical, as seen from Figure 50d, since the AGC is sensitive to the total power in the signal. Note that AGC gain continues to increase, beyond the 4th second values shown in Figure 50d.

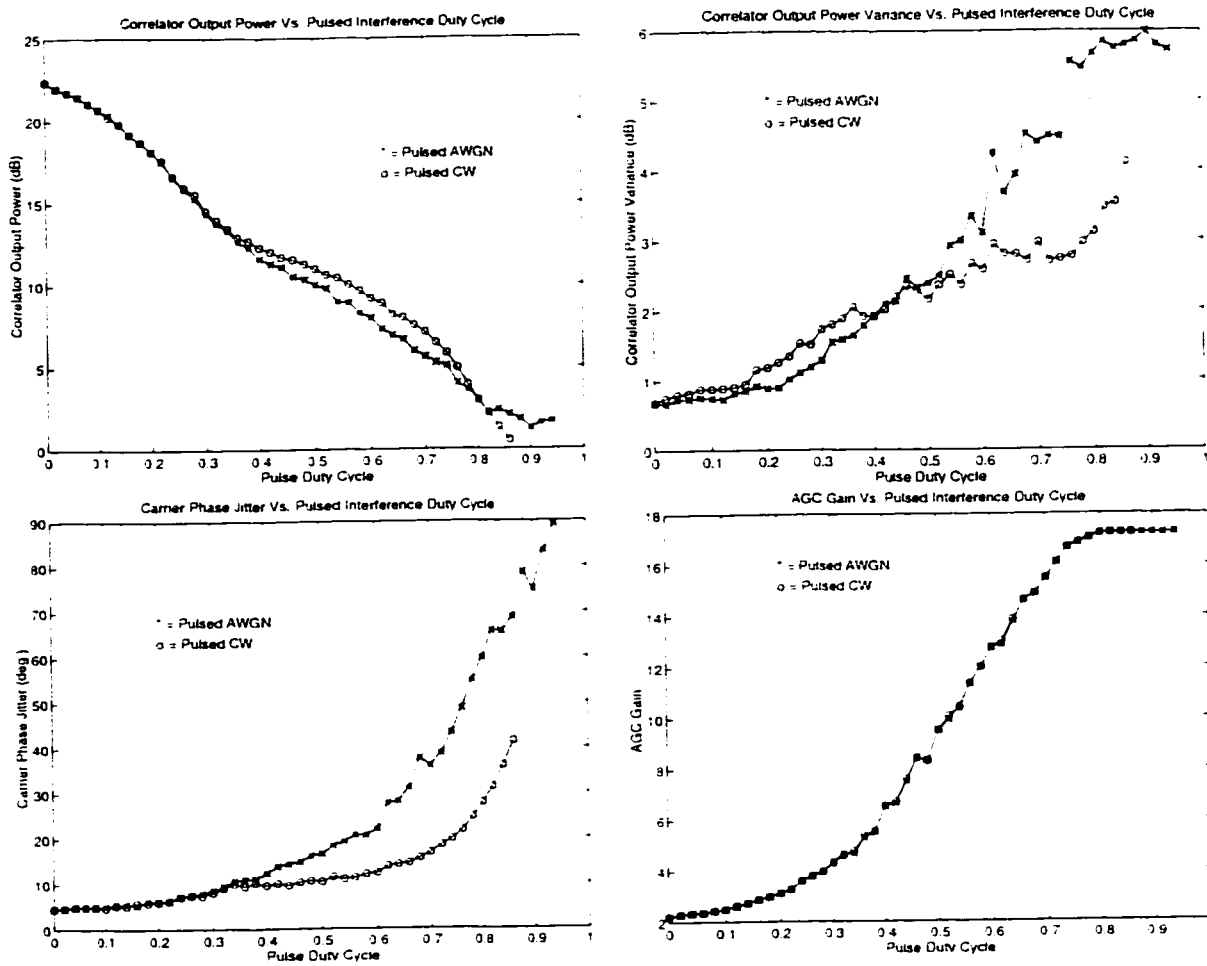


Figure 50: Impact of Pulsed RF Interference on Test Statistics

Pulsed RF interference consists of Pulsed AWGN and Pulsed CW interference. Duty Cycle is varied from 0% until loss of lock occurs.
 (a) Impact of Pulsed RFI on Correlator Output Power; (b) Impact of Pulsed RFI on Correlator Output Power Variance; (c) Impact of Pulsed RFI on Carrier Phase Jitter; (d) Impact of Pulsed RFI on AGC Gain

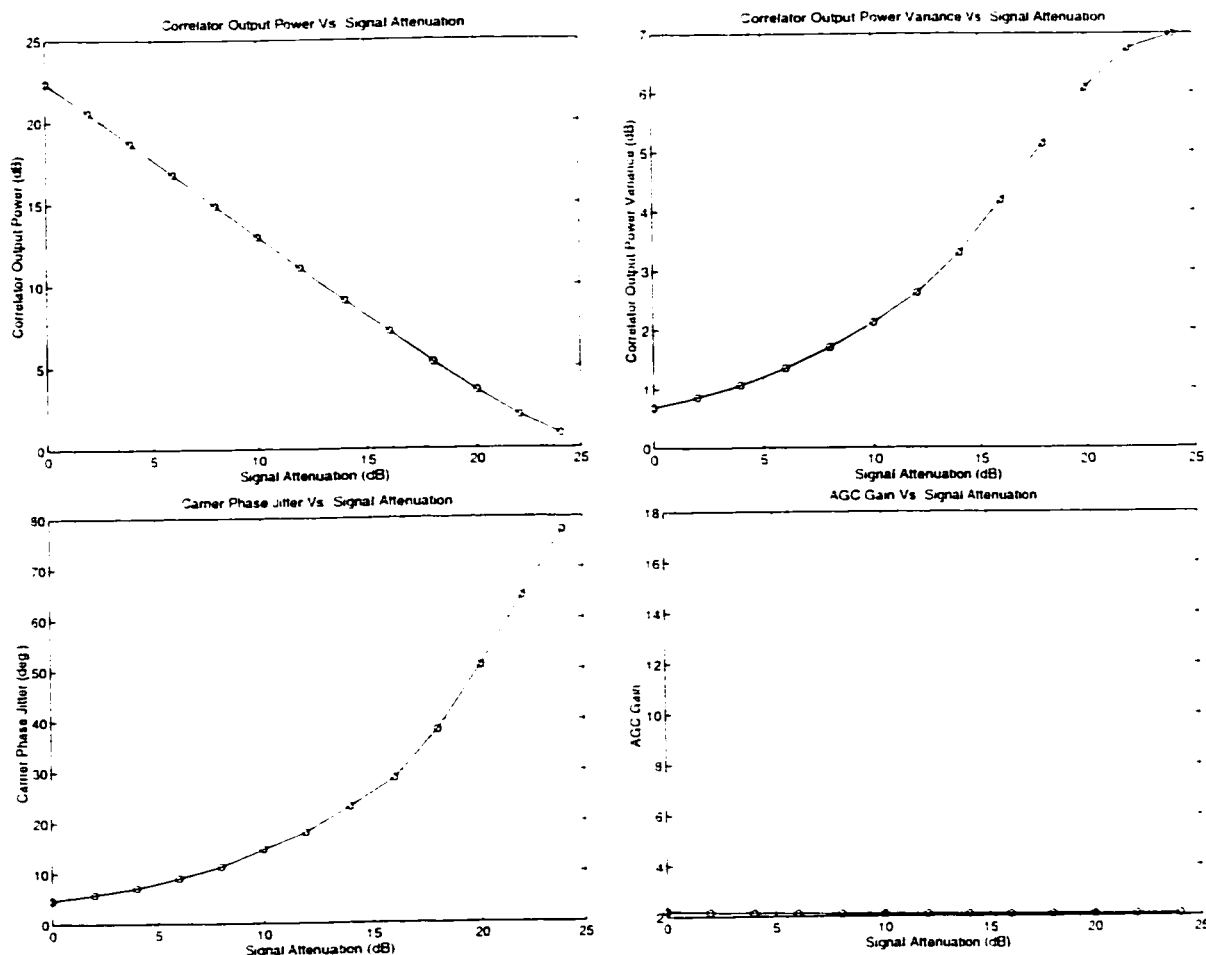


Figure 51: Impact of Signal Attenuation on Test Statistics

The tracked GPS signal is attenuated from 0 dB until loss of lock, in 1 dB steps.

(a) Impact of Signal Attenuation on Correlator Output Power; (b) Impact of Signal Attenuation on Correlator Output Power Variance; (c) Impact of Signal Attenuation on Carrier Phase Jitter; (d) Impact of Signal Attenuation on AGC Gain

5.3.3 Impact of Signal Attenuation on Test Statistics

Signal attenuation should have a linear relationship with correlator output power. Indeed, Figure 51a shows COP drop linearly with attenuation, in dB, until loss of lock occurs. Both the variance of COP and carrier phase jitter also grow with signal attenuation, at a slower pace for attenuation values less than 10 dB, and then more rapidly as the signal gets attenuated beyond 10 dB.

AGC gain acts in a completely different manner for signal attenuation, as compared with other forms of interference. It is least sensitive to signal attenuation, actually decreasing with increased attenuation. Total drop in AGC is a mere 0.14 AGC-units (Figure 51d), compared with a 15 AGC-units increase for AWGN (Figure 49d). This is because the tracked signal makes up only a small percentage of total signal power - which is comprised of all other satellites in view, and dominated by receiver thermal noise. The AGC registers the slight decrease in total signal power as the tracked signal is attenuated.

5.4 Coherent Interference and Loop Capture

CW interference on a strong spectral line presents the most potent form of interference to GPS receivers. A substantial portion of the interference energy is translated into the central GPS frequency by the correlator, resulting in the maximum amount of post correlation interference power. This type of interference is so severe it can actually cause the receiver to falsely lock onto the interference, losing track of the real signal. This type of false lock is also known as *loop capture*.

The phenomenon of loop capture is a consequence of the way GPS receivers track signals. The presence or absence of a signal is detected from the value of the correlator output power: COP values above the 'detection threshold' indicate the presence of a GPS signal, and signal the receiver to declare a successful acquisition. A persistent value of COP above

this threshold indicates a successful lock of the tracking loops. When CW interference falls on a strong C/A code spectral line, a large amount of interference power is transmitted directly through the correlators, and can fool the receiver into thinking it still has lock on the signal. This effect can be seen from the extensions of the CW interference run at L1 + 7 KHz (Figure 52). As interference power is increased, $C/(N_0+I_0)$ drops from 55 dB-Hz, down to 35 dB-Hz, corresponding to an applied I_0 of 20 dB. Correlator power output (used as the basis for loop lock indication) decreases monotonically through this interference regime, as can be seen from Figure 52. An increase in CW interference beyond this value causes the receiver to lose lock. The circled data points of Figure 52

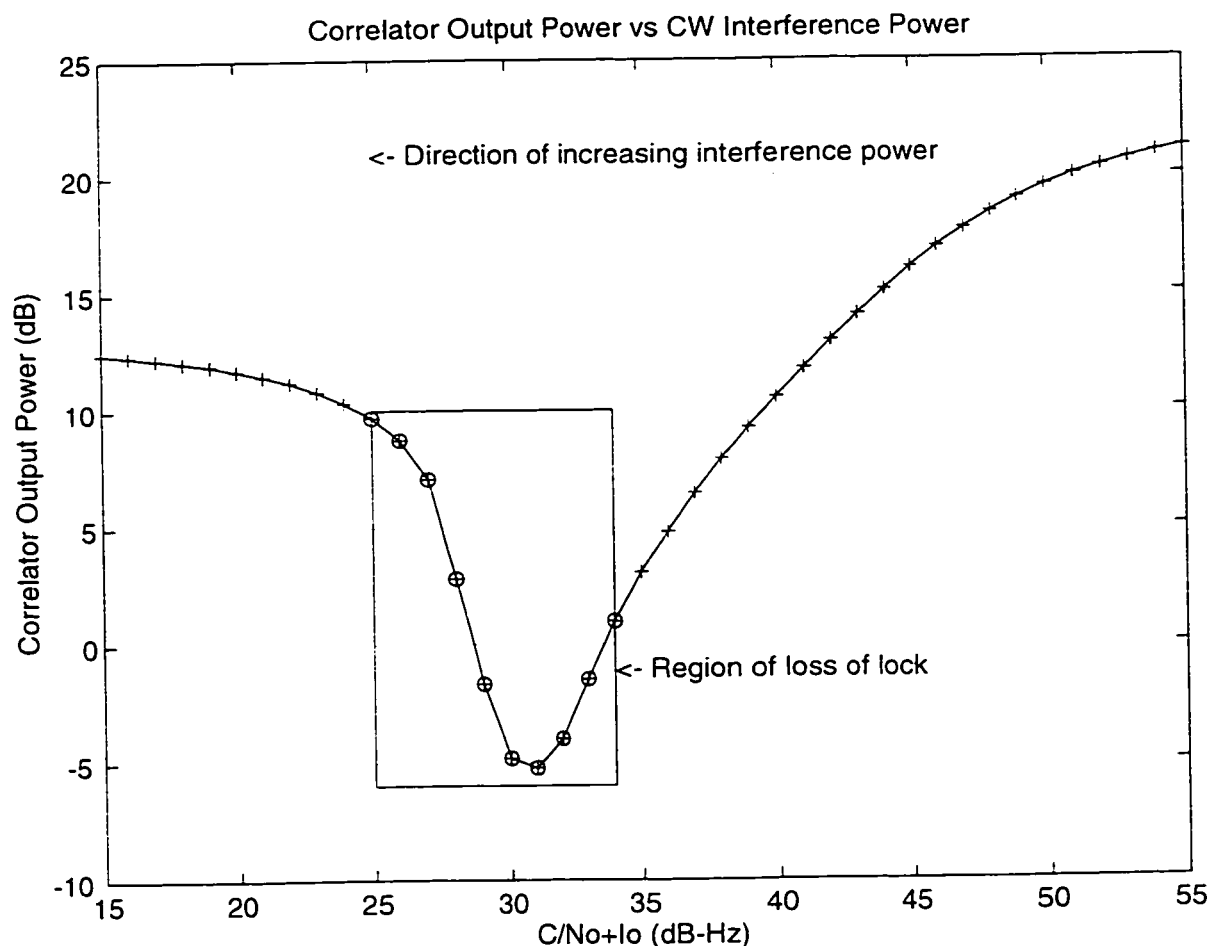


Figure 52: Loop Capture with Strong Coherent CW Interference: Correlator Output Power vs. CW Interference for CW at L1 + 7 KHz, a strong spectral line. Points in box indicate receiver has lost lock.

indicate that the receiver has lost lock.

Ignoring this loss of lock and further increasing the interference power, an interesting phenomenon occurs: correlator output power begins to increase. For CW interference loading with $C/(N_0+I_0)$ less than 25 dB-Hz, the loop loss flag in the receiver never gets triggered, although the receiver is now tracking the CW interference, and not the GPS signal. Figure 53a, which shows the pseudorange error for an extended run, confirms the loop capture: pseudorange error grows beyond the 1-chip correlation peak range, as the tracking loop wanders, while correlator output power does not drop to zero (Figure 53b). The drop in correlator output power reduces as larger values of CW are applied, as the attenuation caused by the AGC is balanced by the increased post correlation interference power.

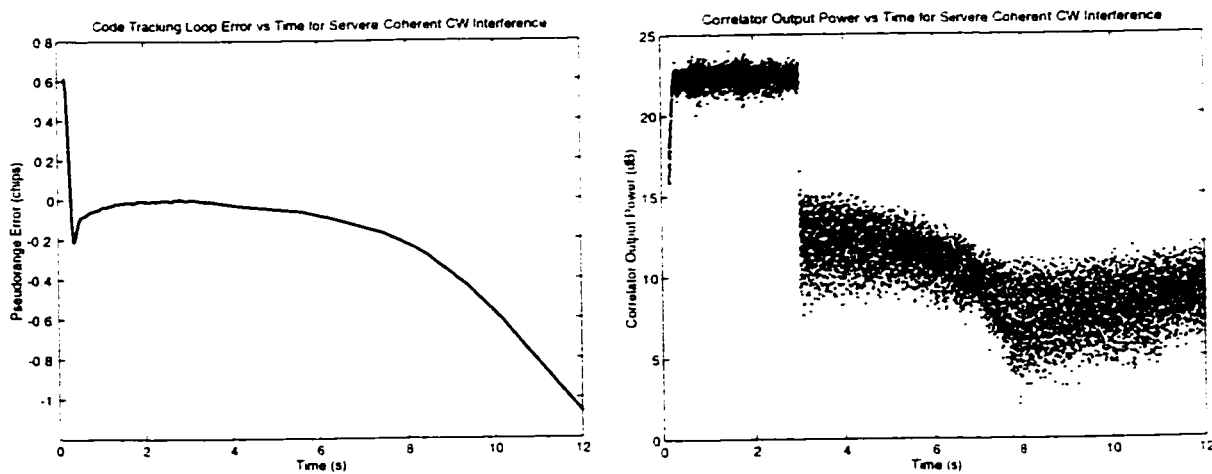


Figure 53: Loop capture with Strong Coherent CW at L1 + 7 KHz. CW Interference Power to Signal ratio is 40 dB for a $C/(N_0+I_0)$ of 15 dB-Hz. (a) Pseudorange Error over Time, (b) Correlator Output Power over Time.

The phenomenon of loop capture underscores the need for data parity and CRC checks to accompany any integrity monitoring scheme.

5.5 Test Statistics Performance

Since COP, unlike the three other selected test statistics, actually decreases with increasing interference (and thus with increasing pseudorange error), the negative slope decision matrix, shown in Figure 54a, is used to analyze COP performance. In this mode, the regions of missed detection, false alarm, normal operation and normal detection described in chapter one are redefined by a reflection across the vertical. A positive slope matrix, shown in Figure 54b, is used for analyses of the variance of COP, carrier phase jitter, and AGC.

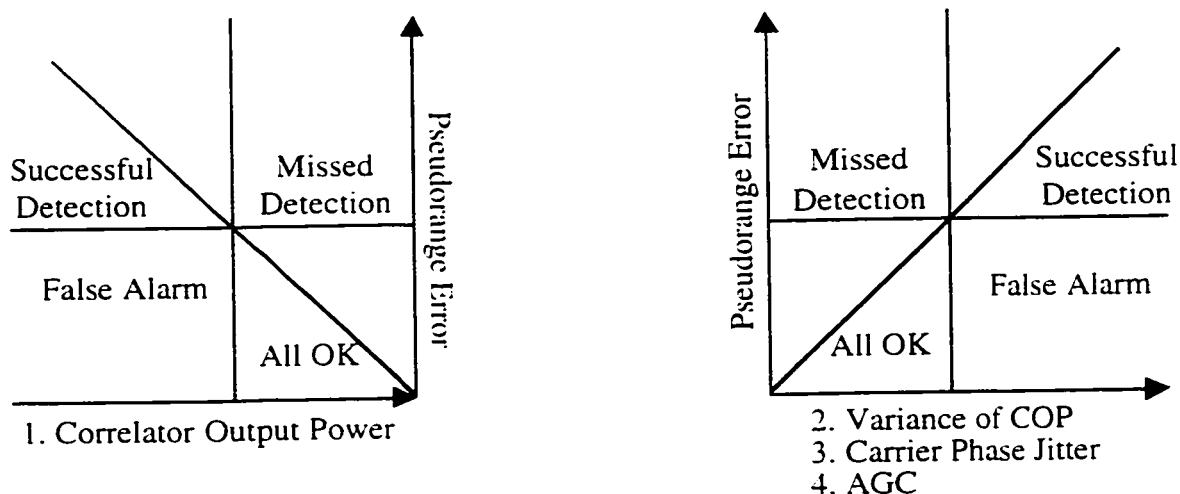


Figure 54: Test Statistics Decision Matrix showing (a) Negative Slope Matrix and (b) Positive Slope Matrix

In both positive and negative slope matrices, the missed detection region would contain data points for which pseudorange error exceeded its preset alarm limit while our test statistic remained below detection threshold. Conversely data points in the false alarm region correspond to incidents where our test statistic exceeds its threshold while pseudorange error remained below its alarm limit. Unlike missed detection, false alarm is not a direct threat to integrity. However they are a nuisance, because they adversely affect the continuity of the system.

To detect the degradation in accuracy due to interference, a 2 m pseudorange alarm limit is assumed. Errors greater than 2 m are to be detected. Test statistics thresholds are set to ensure zero missed detection for the case of AWGN and CW at L1 only. This serves as a basis to evaluate the robustness of the test statistics over the different types of interference.

To evaluate test statistic robustness, least square error linear fits are plotted to emulate the detection statistic lines in Figures 54a and 54b. For each test statistic, seven linear fits are plotted for each of the seven interference types tested. The following quantity is defined:

NSR = Normalized Slope Range, where

$$\text{Normalized Slope Range} = \left| \frac{\text{Maximum Slope} - \text{Minimum Slope}}{\text{Average Slope}} \right| \quad (5.5.1)$$

where maximum and minimum slopes apply to the maximum and minimum linear fits for a test statistic. NSR gives an indication of the robustness of a test statistic.

Note that a maximum⁴ negative slope indicates fastest degradation of the test statistic for a fixed degradation in pseudorange error - which implies maximum sensitivity of the test statistic to the specific interference with the maximum slope, while a maximum positive slope indicates slowest degradation in test statistic for a fixed degradation in pseudorange error, implying minimum sensitivity of test statistic for the specific interference type. The former uses the negative slope matrix of Figure 54a, and latter, Figure 54b.

5.5.1 Performance of Correlator Output Power

Figure 55 shows the detection of pseudorange error using COP for AWGN, CW at L1.

⁴ 'Maximum' as used here always means closest to positive infinity, while 'minimum' always implies closest to negative infinity. Therefore a slope of -0.12 is larger than a slope of -0.25.

pulsed AWGN, pulsed CW, signal attenuation, CW at $L1 + 1$ KHz and CW at $L1 + 7$ KHz. For each interference type, a minimum square error best linear fit line is plotted, in the same color as the plot points for the interference type. The slope of this line, and the standard deviation of the data points from the line are shown in Table 7 for each type of interference.

Table 7: Least Squares Error Linear Fit Slopes and RMS Data Spreads from Linear Fit, for Correlator Output Power When Used as an Integrity Monitoring Test Statistic

	AWGN	CW @L1	Pulsed AWGN	Pulsed CW	Signal Attenuation	CW at L1+1KHz	CW at L1+7KHz
Slope	-0.1988	-0.1994	-0.2006	-0.2354	-0.1793	-0.2530	-0.1262
Spread	0.3510	0.6332	0.7512	0.6478	0.3537	0.4130	0.3881

A strong linear relationship can be observed from Figure 55 : the bulk of data points are clustered in the 'All OK' and 'Detection OK' region, with a narrow spread along the linear fit. A linear fit slope of -0.20 exists for AWGN, CW at L1 and Pulsed AWGN, with slopes for signal attenuation at -0.18 and -0.24. The maximum negative slope of -0.13 occurs for CW at $L1 + 7$ KHz, coherent CW interference on a strong spectral line. This slope is to be expected since the largest amount of interference power is translated directly into post-correlation noise power. Therefore the fastest degradation in correlator output power occurs for this type of interference. Strong coherent CW interference would therefore likely increase the occurrence of false alarms using correlator output power as a detection statistic. Fortunately this type of interference is rare, and is unlikely to persist when it does occur, as described in section 3.5. Observe from Figure 55 that the linear fit line for coherent CW at $L1 + 7$ KHz is the shortest of the seven linear fits, indicating that the receiver loses lock fastest with this type of interference.

Table 7 also shows that the linear fit with the minimum slope of -0.25 occurs for coherent

CW interference at $L1 + 1$ KHz, which is a particularly mild spectral line. This also is to be expected, since the maximum amount of spread spectrum interference spreading occurs for CW interference on this spectral line. Therefore correlator output power degrades least for this type of interference. Average spreading from linear fit is 0.5054, with pulsed interference generating the largest spread, and AWGN generating the minimum spread. Correlator output power has a normalized slope range (NSR) of 0.64.

5.5.2 Performance of Correlator Output Power Variance

Figure 56 shows pseudorange error as a function of correlator output power variance for all seven types of interference tested. As with correlator output power, least squares error best linear fit lines have been plotted, with the slopes and rms spread tabulated in Table 8 for each type of interference. Slopes range from 0.5394 to 4.2153 (for CW at $L1 + 1$ KHz), a range of 3.6759, compared to 0.4002 for COP, showing that the variance of correlator output power is less precise than correlator output power in robust detection of pseudorange error for a variety of interference types. Notice however that coherent CW interference at $L1 + 7$ KHz has a slope of 1.4, close to the average slope of 1.5, and also that this type of interference produces the least spread from its linear fit, indicating that the variance of correlator output power is a reliable test statistic for use even with coherent CW interference on strong spectral lines. This would suggest the combined use of both correlator output power and its variance to form a reliable and robust test statistic.

Table 8: Least Squares Error Linear Fit Slopes and RMS Data Spreads from Linear Fit, for Correlator Output Power Variance When Used as an Integrity Monitoring Test Statistic

	AWGN	CW @L1	Pulsed AWGN	Pulsed CW	Signal Attenuation	CW at L1+1KHz	CW at L1+7KHz
Slope	1.0232	1.2417	0.7094	1.6236	0.5394	4.2153	1.4148
Spread	0.4304	0.9860	0.7424	0.6862	0.4970	0.6457	0.2301

The maximum slope of 4.21 occurs for coherent CW at $L1 + 1$ KHz, a weak spectral line,

which is to be expected since the minimum amount of interference energy is passed through the correlator for this type of interference and therefore the variance of the correlator output power is least affected by this type of interference. The minimum slope of 0.54 occurs for signal attenuation, indicating that the variance of correlator output power is most sensitive to signal attenuation. Correlator output power variance has a normalized slope range of 2.39. Hence it is significantly more sensitive to the type of interference, and therefore less robust.

5.5.3 Performance of Carrier Phase Jitter

Figure 57 shows pseudorange error as a function of carrier phase jitter for all seven types of tested interference. Slope values range from a minimum of 0.05 for signal attenuation to a maximum of 0.51 for CW at $L1 + 1$ KHz. This indicates carrier phase jitter is most sensitive to signal attenuation, and least sensitive to CW at $L1 + 1$ KHz. The minimal impact of CW at $L1 + 1$ KHz is consistent with results for correlator output power and its variance. The largest spreading of points occurs for CW at $L1$, with a spread of 1.06.

Table 9: Least Squares Error Linear Fit Slopes and RMS Data Spreads from Linear Fit, for Carrier Phase Jitter When Used as an Integrity Monitoring Test Statistic

	AWGN	CW @L1	Pulsed AWGN	Pulsed CW	Signal Attenuation	CW at L1+1KHz	CW at L1+7KHz
Slope	0.1247	0.1061	0.0510	0.1560	0.0467	0.5069	0.2055
Spread	0.4849	1.0610	0.8716	0.8393	0.7226	0.6957	0.2184

Carrier phase jitter has a normalized slope range (NSR) of 2.69. This NSR is largest of all test statistics, if the effect of signal attenuation on AGC Gain is not taken into account, as discussed in the next section.

5.5.4 Performance of AGC Gain

The relationship between pseudorange error and AGC gain is shown in Figure 58 for all seven types of tested interference. Slopes for all interference types, except signal attenuation, are positive, indicating an increase in AGC gain with increase in interference. Signal attenuation has a negative slope of -26.28, the minimum value by a large margin. This large value is due to the fact that the AGC, unlike other test statistics, is sensitive to the total power in the incoming signal, which includes receiver thermal noise, interference, and other satellites in view. Attenuation of the tracked GPS signal would degrade the pseudorange accuracy but would barely affect the AGC gain. Therefore AGC gain is not suitable for detecting the presence of signal attenuation. However when used in conjunction with the other test statistics, AGC gain can identify the *type* of interference to be signal attenuation. The minimum positive slope of 0.2294 occurs for pulsed AWGN, which indicates AGC Gain is most sensitive to pulsed AWGN. In general AGCs are most sensitive to pulsed interference. Fast acting AGCs act as pulse suppressants [15], and therefore would show large gains for pulsed interference. This can be seen by showing non-low-pass filtered AGC values at the end of each 6 second run. These values are plotted in Figure 59. Slopes of, and rms spreads from, the lines of best fit are shown in Table 10. Notice the increase in the maximum AGC Gain values, from a maximum of 17 units (Figure 58) to a maximum of 65 units (Figure 59), demonstrating the pulse suppression action of a fast AGC.

Table 10: Least Squares Error Linear Fit Slopes and RMS Data Spreads from Linear Fit, for AGC Gain When Used as an Integrity Monitoring Test Statistic

	AWGN	CW @L1	Pulsed AWGN	Pulsed CW	Signal Attenuation	CW at L1+1KHz	CW at L1+7KHz
Slope	0.2656	0.2815	0.2294	0.2743	-26.2804	0.2327	0.3685
Spread	0.2995	0.4623	0.7470	0.3810	0.8095	0.3182	0.2837

Table 11: Least Squares Error Linear Fit Slopes and RMS Data Spreads from Linear Fit, for AGC Gain Using a Fast Acting AGC, When Used as an Integrity Monitoring Test Statistic

	AWGN	CW @L1	Pulsed AWGN	Pulsed CW	Signal Attenuation	CW at L1+1KHz	CW at L1+7KHz
Slope	0.2060	0.1898	0.0569	0.0684	-29.5476	0.0637	0.3553
Spread	0.4175	0.5047	0.7292	0.3595	0.8377	0.7607	0.2795

The maximum positive slope of 0.37 (or 0.35 for the fast acting AGC) occurs for CW at L1 + 7 KHz, which implies that next to attenuation, AGC gain is least sensitive to coherent CW interference on L1 + 7 KHz.

AGC gain has an average test statistic slope of -3.52, and a normalized slope range of 7.57, the largest of all four test statistics, due to the large negative contribution from the signal attenuation term. . If this term were omitted the average slope would be 0.27, with a NSR of 0.50.

5.6 Frequency Error Monitoring using Test Statistics

Carrier smoothing or aiding is used to reduce noise on the code tracking loop (pseudorange) using input from the carrier tracking loop. The carrier has a smaller wavelength (19cm) compared to code (300m), and therefore is more accurate. However there is an integer ambiguity in carrier cycles, and as a result range measurements cannot be obtained directly from the carrier tracking loop.

While there are numerous ways to resolve this ambiguity and attain centimeter level accuracies, the carrier can also be used without integer ambiguity resolution to 'smooth' or aid the code tracking loop. When used in this mode, errors in the carrier tracking loop (the FLL in our case) will affect the code tracking loop, and pseudorange errors will be directly

related to frequency errors.

Carrier smoothing is not used in this simulation, and therefore frequency errors are not directly coupled to pseudorange errors. However we examine the relationship between carrier tracking loop errors and our selected test statistics, with a goal to verifying the performance of our integrity monitoring scheme for a receiver using a carrier aided code tracking loop.

Our objective is to verify that pseudorange error detection using the decision statistics *thresholds* selected in the previous section for a non-carrier aided receiver, will work with carrier aiding. Therefore identical thresholds of COP, COP- σ , carrier phase jitter and AGC are applied to the detection of frequency error. The results are shown in Figures 60, 61, 62 and 63. Frequency error protection level is set at 1.5 m/s, given that the nominal frequency error with no interference is about 1.1 m/s.

Frequency Error and Correlator Power Output:

Figure 60 shows frequency error as a function of COP for all 7 interference types. Frequency errors remain below the 1.5 m/s threshold as COP decreases, until after the COP threshold at 12.9 dB. Beyond this point, frequency error increasingly grows with decreasing COP. The fastest rate of increase occurs for CW at L1 + 1 KHz, indicating that frequency error responds faster to increasing CW while COP does not respond as fast. This is to be expected since CW at L1 + 1 KHz is coherent interference on a weak spectral line, and therefore does not contribute much to the post correlation interference power (see discussion in section 3.5). The carrier tracking loop is nonetheless particularly susceptible to CW interference, and hence rapid growth in the frequency error versus COP curve.

Note from the figure that using the same threshold as with pseudorange error monitoring,

there are no missed detections. This result verifies that our earlier conclusions for pseudorange error monitoring using COP will indeed perform successfully for a receiver with a carrier aided tracking loop, since frequency errors remain below the protection limit within the same regime of COP values as with pseudorange errors.

Frequency Error and the Variance of Correlator Power Output:

Figure 61 shows frequency error as a function of the variance of COP for all 7 interference types. This figure follows a similar pattern as with COP, however with a larger cluster of points in the OK and false alarm regions. Again, CW at $L1 + 1\text{KHz}$ draws the most significant response. Notice again that there are no false alarms using the same threshold for $\text{COP-}\sigma$ as with pseudorange error monitoring.

Frequency Error and Carrier Phase Jitter:

Carrier phase jitter shows similar characteristics with $\text{COP-}\sigma$, with a dense cluster in the OK region, and an maximum slope for CW at $L1 + 1\text{KHz}$ (Figure 62). More importantly there are again no missed detections using the same carrier phase jitter threshold as with pseudorange error monitoring.

Frequency Error and AGC Gain:

Frequency error as a function of AGC gain is shown in Figure 63, for all 7 interference types. Notice the vertical line for signal attenuation, indicating that AGC gain is not sensitive to attenuation in a single signal, as discussed in section 5.5. This behavior is similar to that for pseudorange. The other 6 interference types all remain below the frequency error protection limit, until more than 1 AGC unit after the selected AGC threshold.

This set of results verify the performance of our test statistics in a carrier aided code loop

environment, thereby lending confidence to the use of our test statistic in both non- and carrier aided code tracking loops.

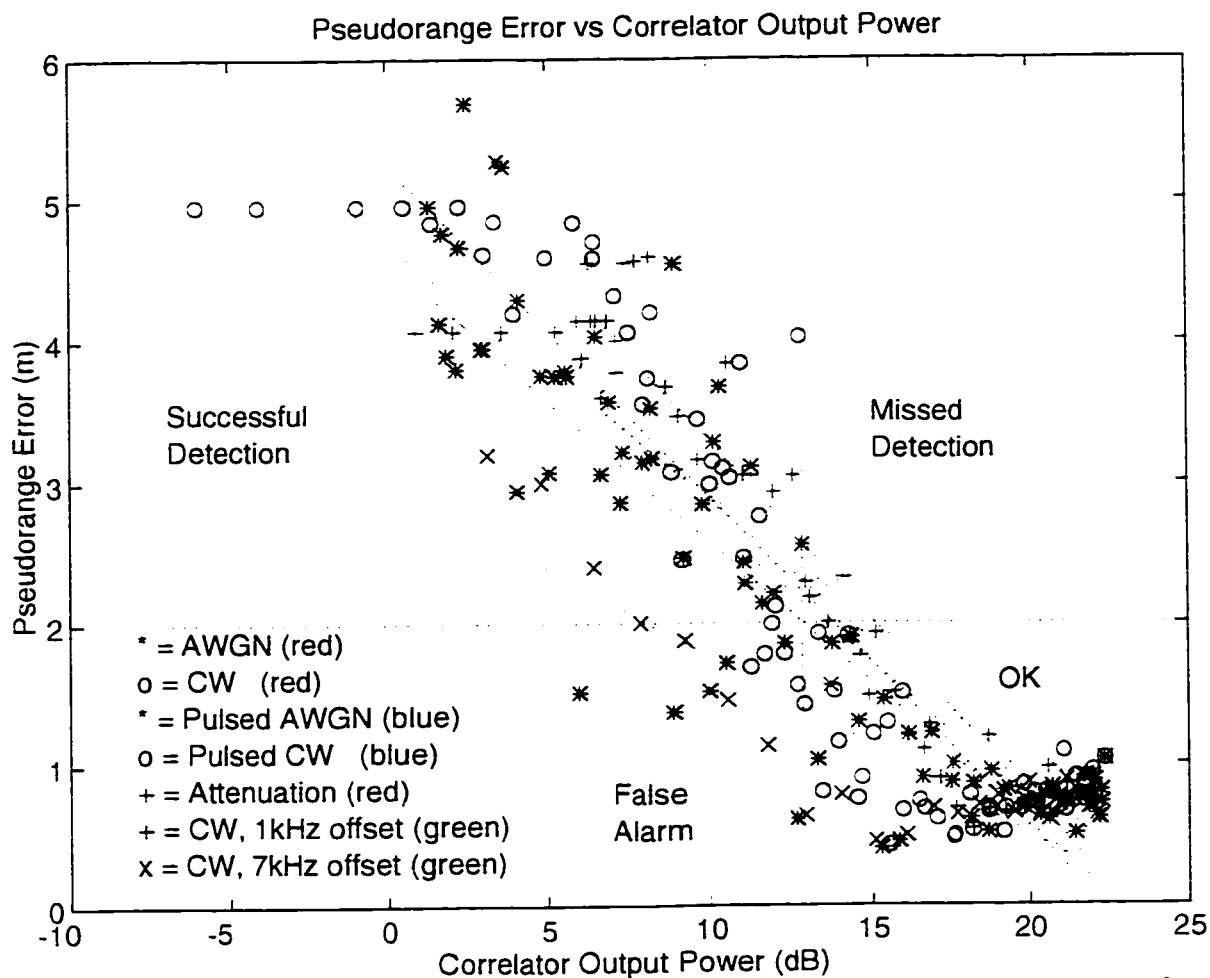


Figure 55: Pseudorange Error as a function of Correlator Output Power for AWGN, CW, Pulsed AWGN, Pulsed CW, Signal Attenuation, CW at L1 + 1 KHz, and CW at L1 + 7 KHz.

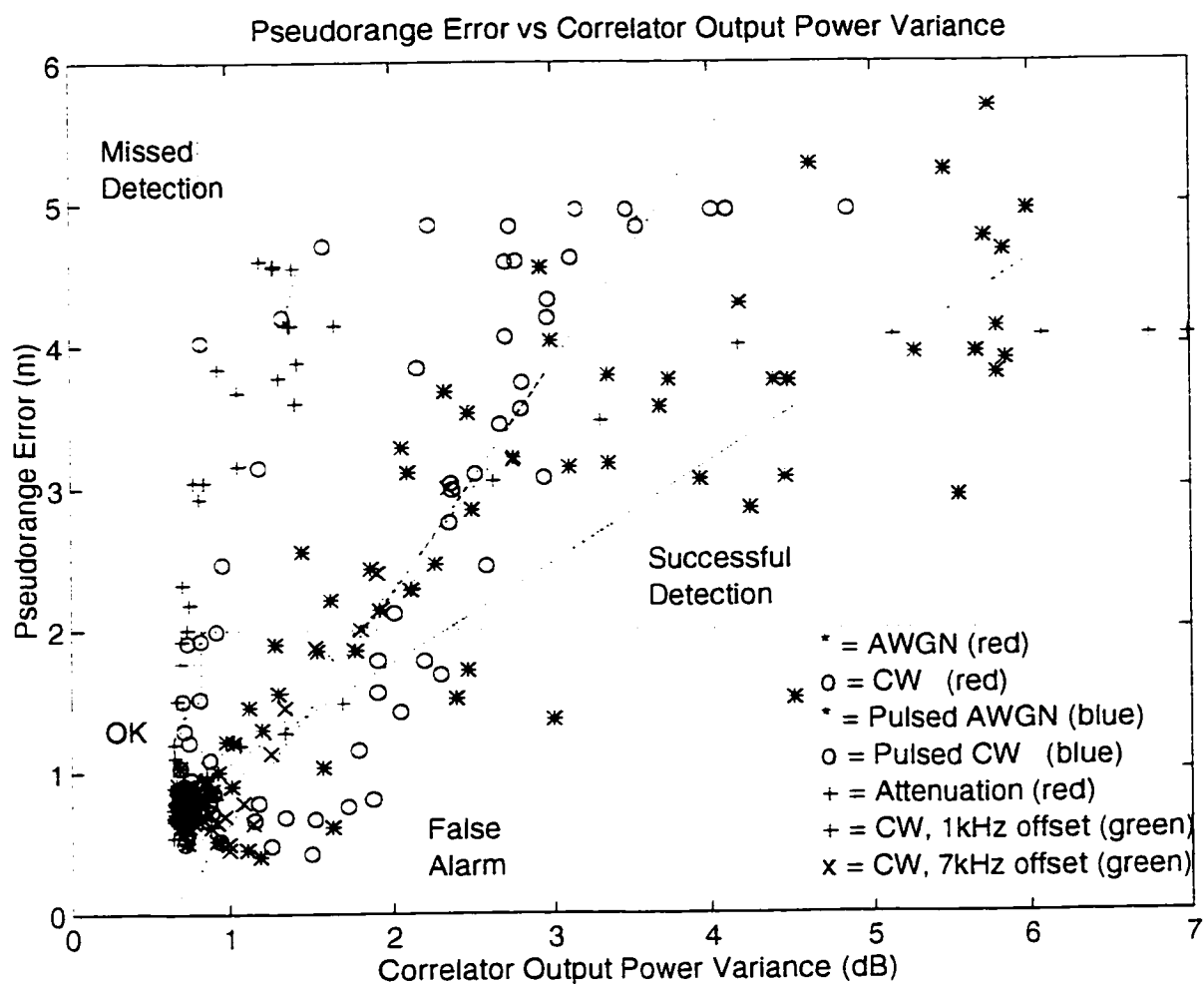


Figure 56: Pseudorange Error as a function of Correlator Output Power Variance for AWGN, CW, Pulsed AWGN, Pulsed CW, Signal Attenuation, CW at L1 + 1 KHz, and CW at L1 + 7 KHz.

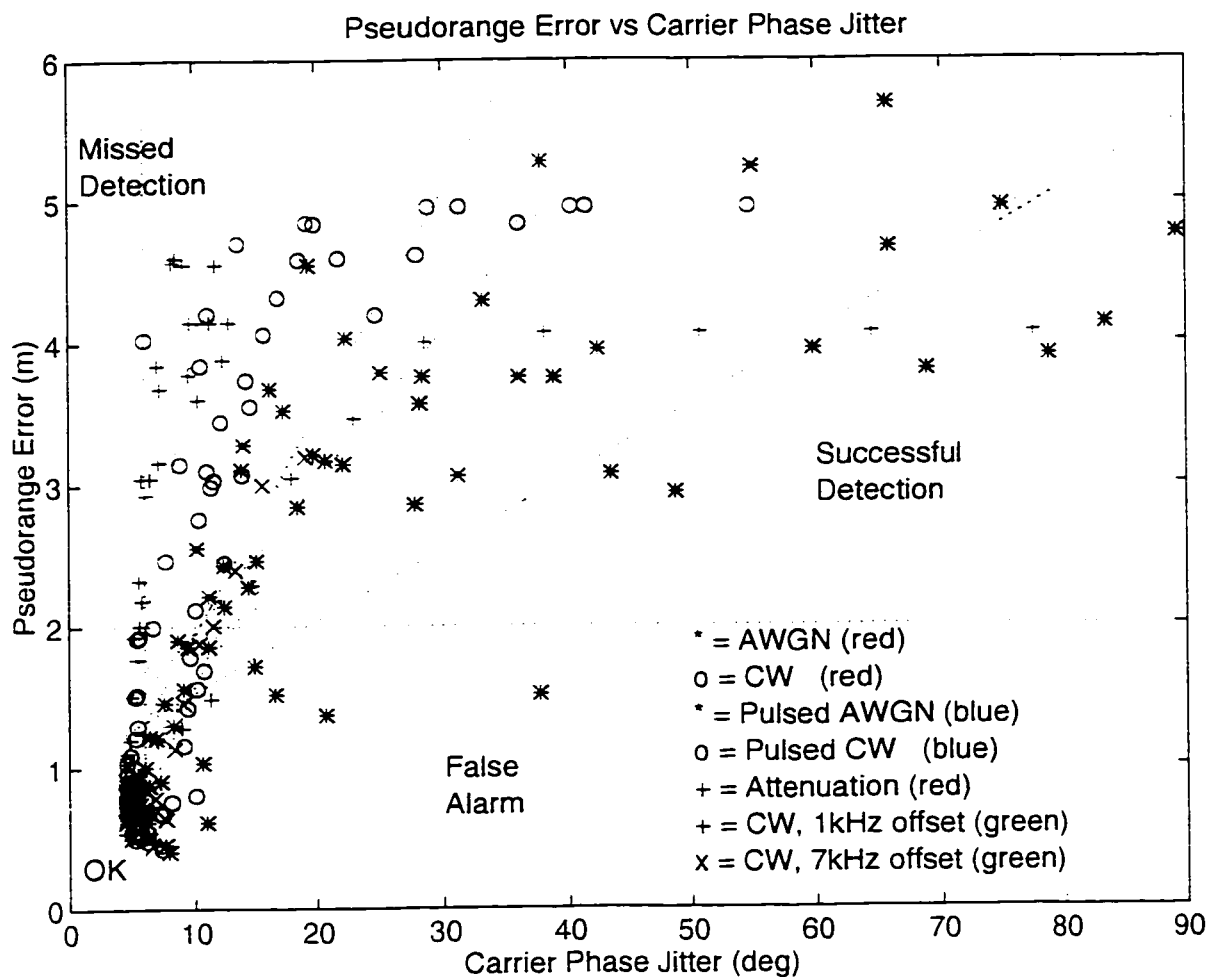


Figure 57: Pseudorange Error as a function of Carrier Phase Jitter for AWGN, CW, Pulsed AWGN, Pulsed CW, Signal Attenuation, CW at L1 + 1 KHz, and CW at L1 + 7 KHz.

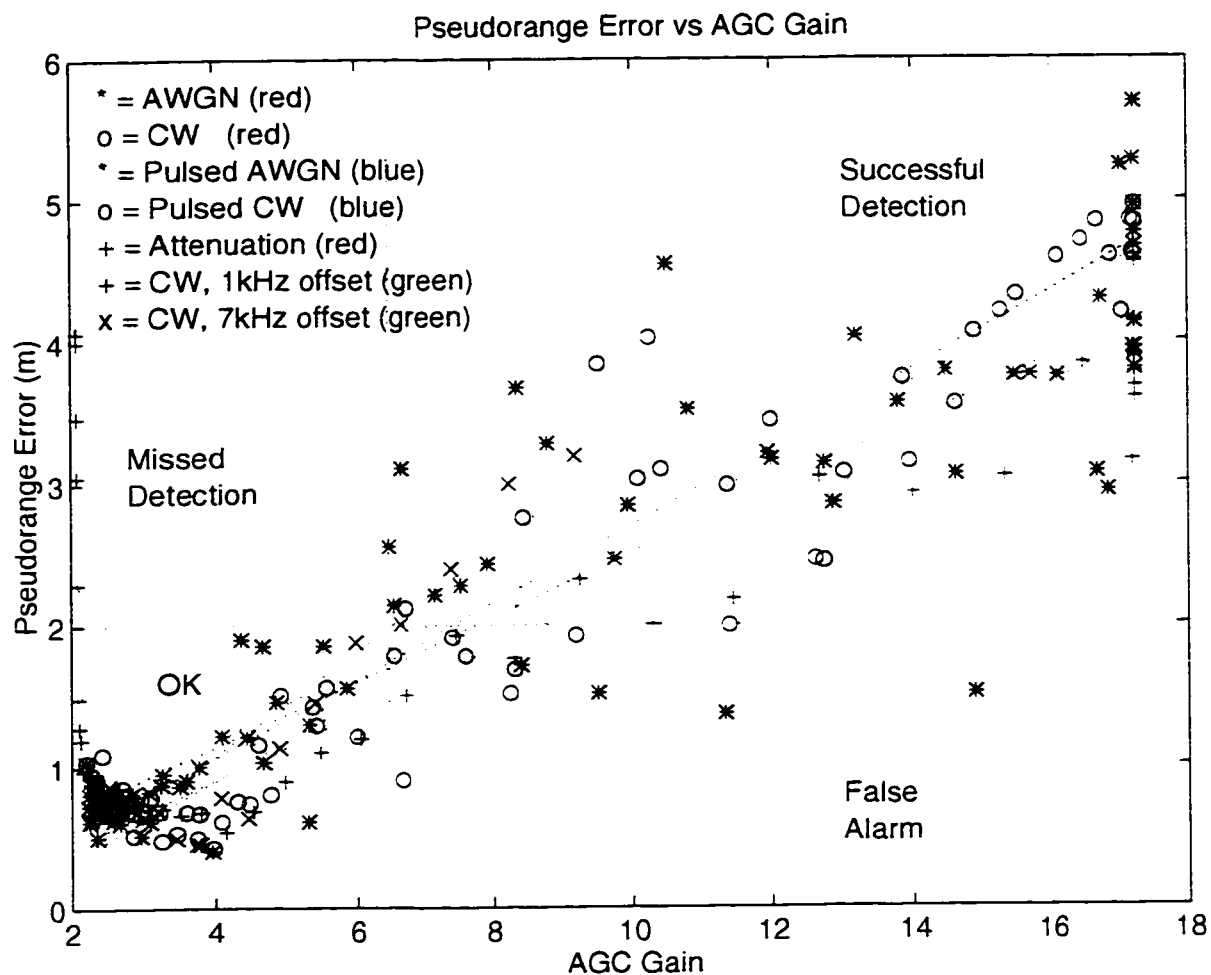


Figure 58: Pseudorange Error as a function of AGC Gain for AWGN, CW, Pulsed AWGN, Pulsed CW, Signal Attenuation, CW at L1 + 1 KHz, and CW at L1 + 7 KHz.

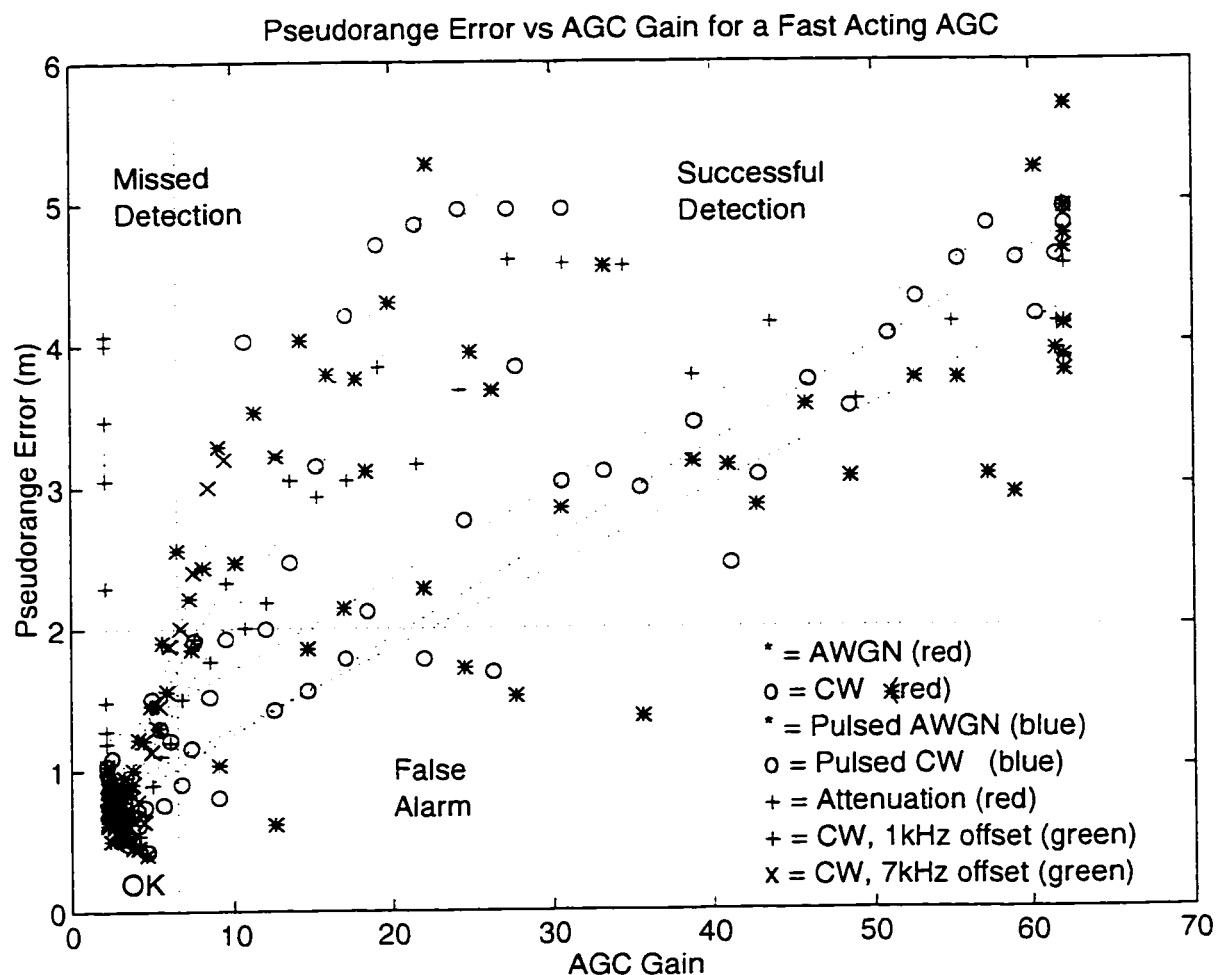


Figure 59: Pseudorange Error as a function of AGC Gain for a fast acting AGC, for AWGN, CW, Pulsed AWGN, Pulsed CW, Signal Attenuation, CW at L1 + 1 KHz, and CW at L1 + 7 KHz.

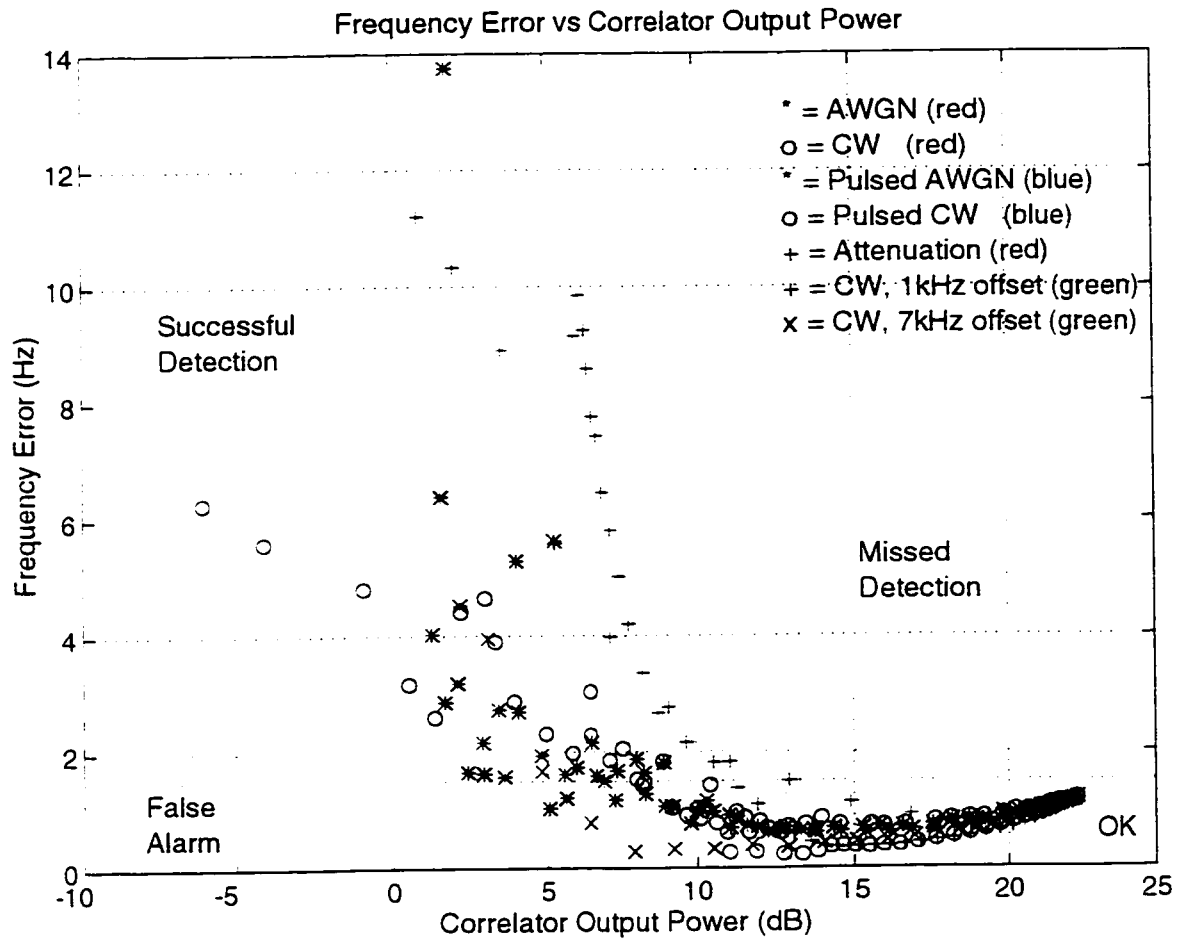


Figure 60: Frequency Error as a function of Correlator Output Power for AWGN, CW, Pulsed AWGN, Pulsed CW, Signal Attenuation, CW at L1 + 1 KHz, and CW at L1 + 7 KHz

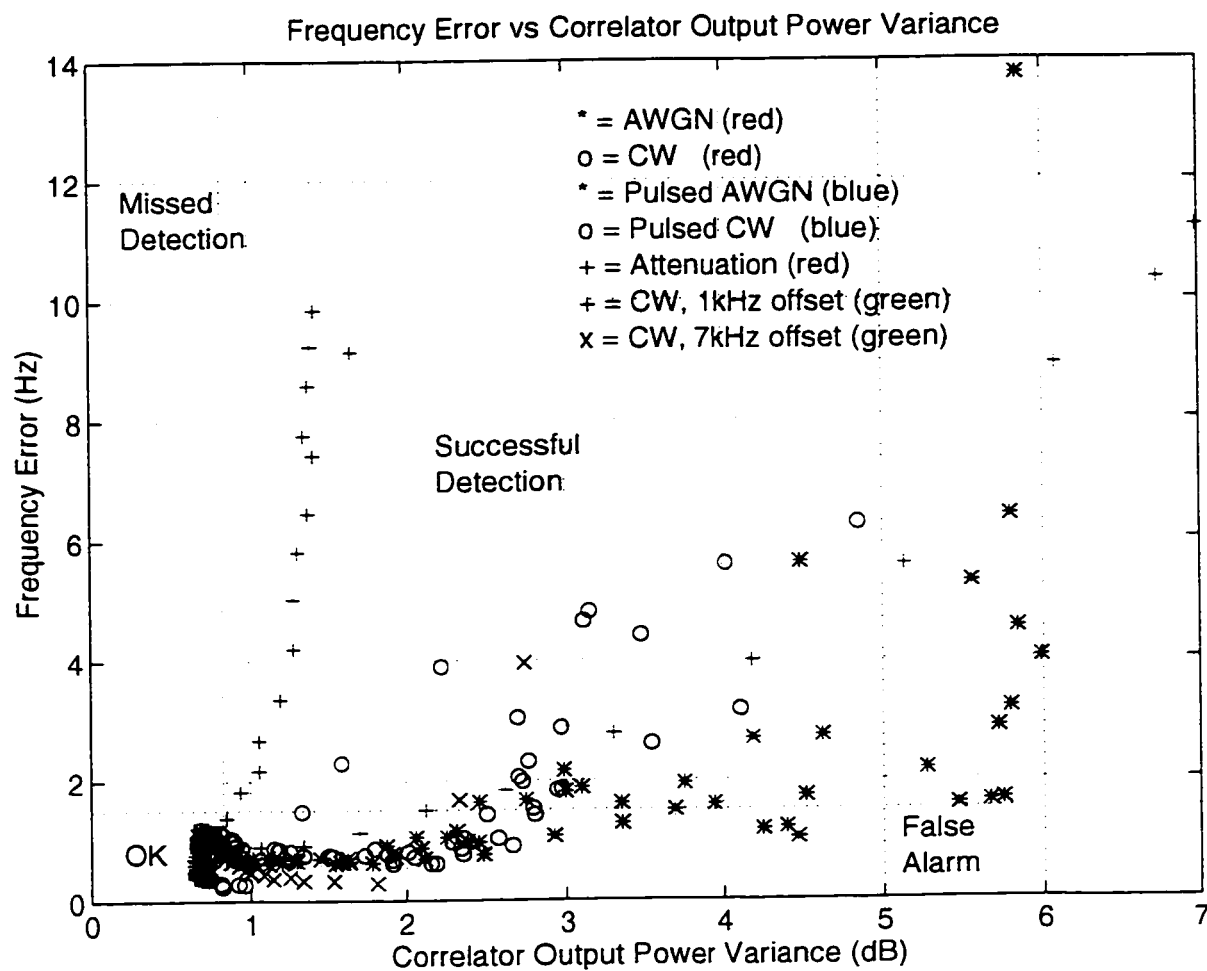


Figure 61: Frequency Error as a function of Correlator Output Power Variance for AWGN, CW, Pulsed AWGN, Pulsed CW, Signal Attenuation, CW at L1 + 1 KHz, and CW at L1 + 7 KHz

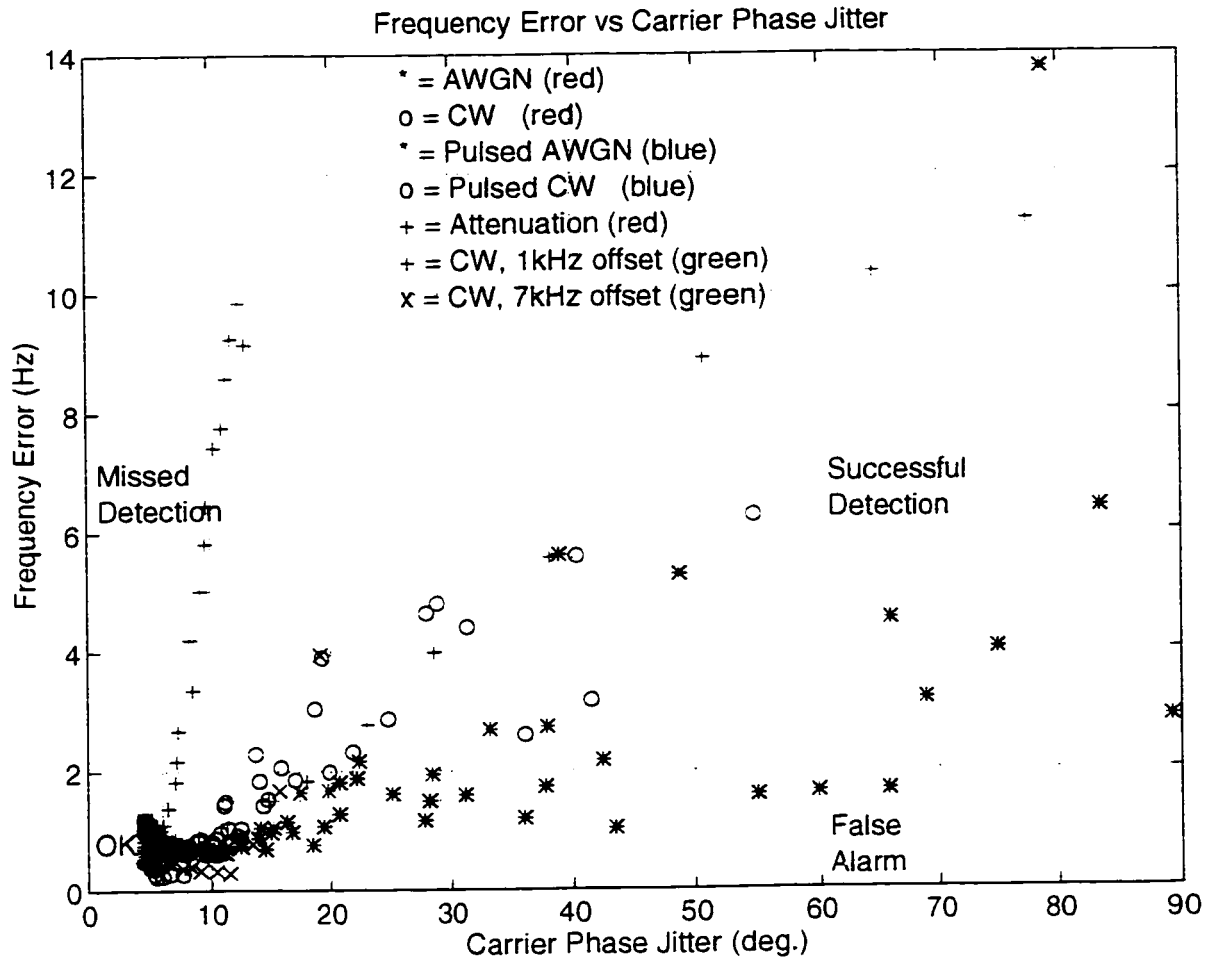


Figure 62: Frequency Error as a function of Carrier Phase Jitter for AWGN, CW, Pulsed AWGN, Pulsed CW, Signal Attenuation, CW at L1 + 1 KHz, and CW at L1 + 7 KHz

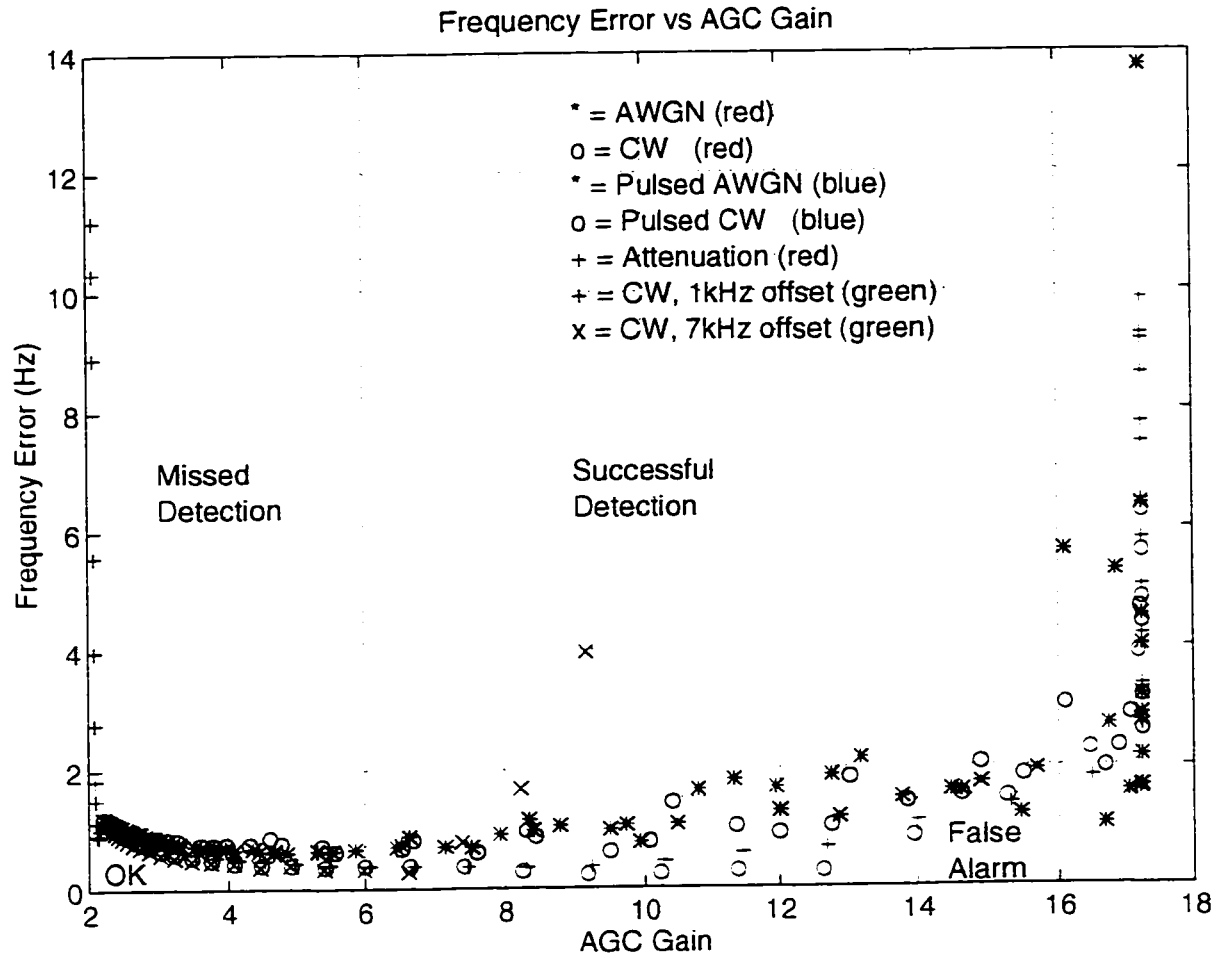


Figure 63: Frequency Error as a function of AGC Gain for AWGN, CW, Pulsed AWGN, Pulsed CW, Signal Attenuation, CW at L1 + 1 KHz, and CW at L1 + 7 KHz

5.7 Bench Test Results

Bench testing was performed to verify the software simulation. With hardware it was not possible to have the same degree of accessibility to truth data as with the software simulation. The receiver under test was a GEC Plessey GPSCard, which, while providing excellent access to correlator outputs, does not provide access to the AGC gain values. Therefore the validity of AGC gains was not verified with the bench tests.

True pseudorange error was also not readily observable with a real receiver. Therefore to validate the simulation, comparisons were made of three of our candidate test statistics: correlator output power, variance of correlator output power, and carrier phase jitter, as a function of interference power.

A challenge with hardware tests is to determine the true thermal noise floor of the receiver, and therefore to determine the correct C/N_0 . The GPS simulator used provides the ability to scale the output power of the GPS signal, however it has no absolute reference, all scaling is relative. In addition, the true power of the interference signal at the input to the GPS receiver RF front end was difficult to determine with precision, since the equipment used to generate interference did not have power calibration, and also due to signal attenuation through a number of subcomponents, including RF cables, signal combiner, switches and amplifiers.

To overcome these challenges, the thermal noise floor of the receiver under test was inferred from its correlator power output plots, as the power level in the absence of interference. The thermal noise floor of the software simulation was then adjusted to this observed noise floor. Thus the nominal correlator output power, in the absence of interference, were equivalent for simulation and bench test.

Also the absolute power of the applied interference was inferred from: the interference power levels required to cause loss of lock of the receiver under test, and the interference power levels required to cause a deviation of observed test statistics from nominal noise floor values. The latter data point marks the $C/(N_o+I_o)$ value at which interference power level is comparable to the noise floor. This enabled the evaluation of $C/(N_o+I_o)$ for bench test results. The complete set of bench test results are shown in Appendix B.

Figures 64, 65 and 66 show bench test results for AWGN and CW. Superimposed on these plots is a single software simulation run for the same type of interference, with noise floor similar to that for the receiver under test. Simulation results are shown as the solid line with asterisk points for AWGN tests (Figures 64a, 65a and 66a), and the solid line with circled data points for CW tests (Figures 64b, 65b and 66b). Receiver thermal noise floor and absolute interference levels can be inferred from the nominal COP values and the knee in these plots.

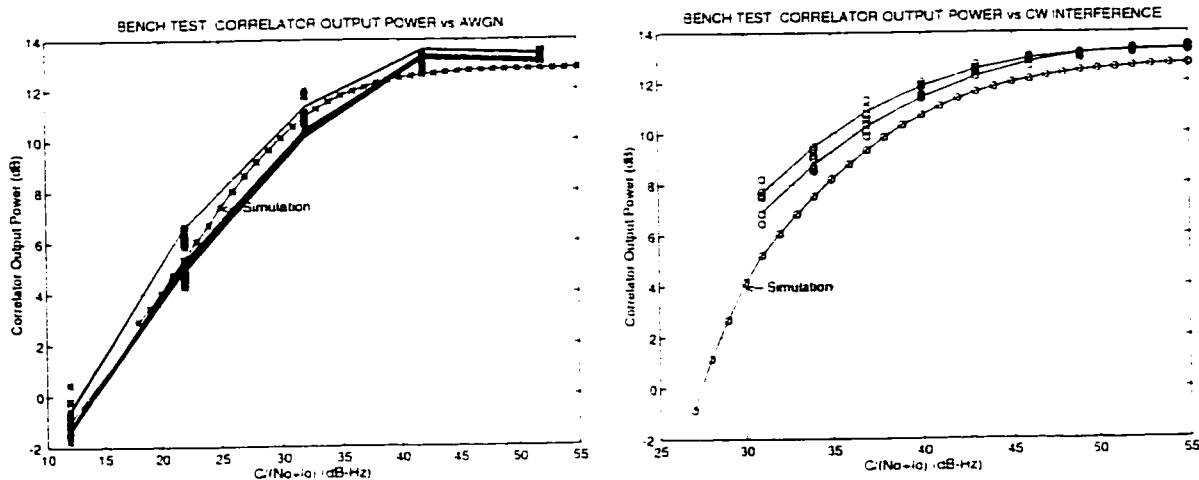


Figure 64: Bench test results for Correlator Output Power vs. $C/(N_o+I_o)$

- (a) Correlator Output Power vs. $C/(N_o+I_o)$ for AWGN. 16 Bench test runs with 1 simulation run superimposed.
- (b) Correlator Output Power vs. $C/(N_o+I_o)$ for CW at L1. 2 Bench test runs with 1 simulation run superimposed.

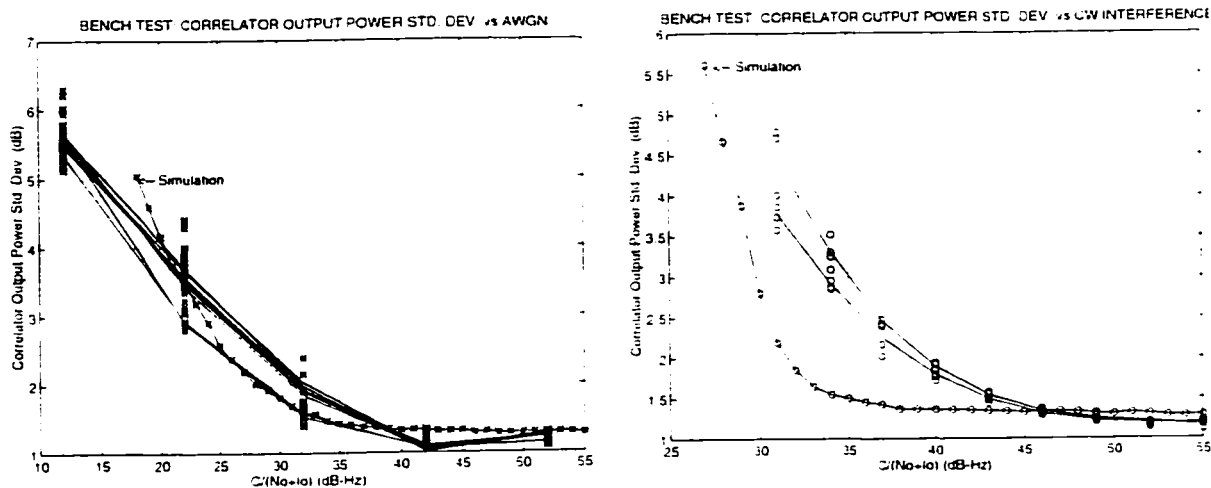


Figure 65: Bench test results for Correlator Output Power Variance vs. $C/(N_0+I_0)$

- (a) Correlator Output Power Variance vs. $C/(N_0+I_0)$ for AWGN. 16 Bench test runs with 1 simulation run superimposed.
 (b) Correlator Output Power Variance vs. $C/(N_0+I_0)$ for CW at L1. 2 Bench test runs with 1 simulation run superimposed.

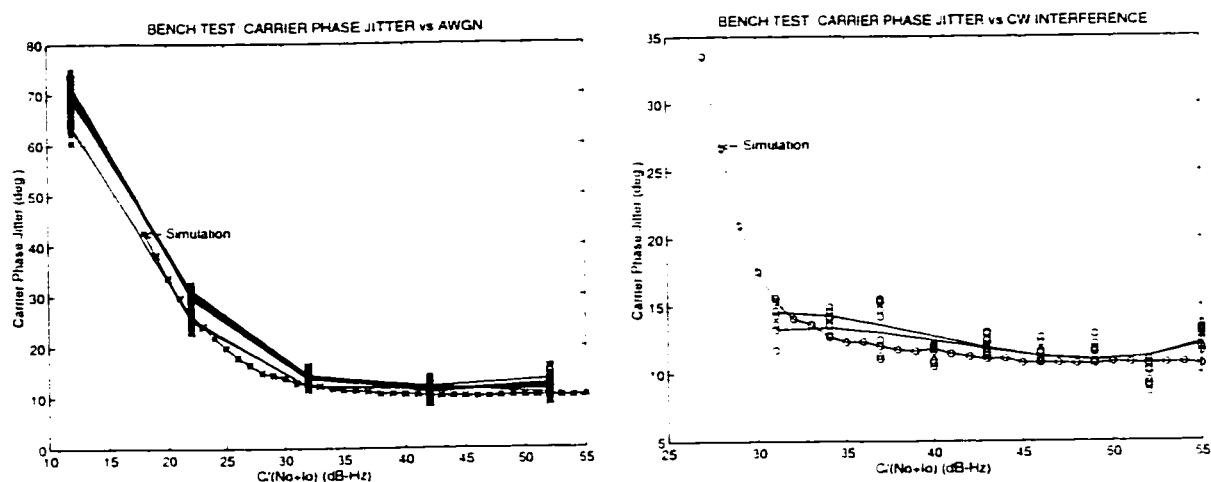


Figure 66: Bench test results for Carrier Phase Jitter vs. $C/(N_0+I_0)$

- (a) Carrier Phase Jitter vs. $C/(N_0+I_0)$ for AWGN. 16 Bench test runs with 1 simulation run superimposed.
 (b) Carrier Phase Jitter vs. $C/(N_0+I_0)$ for CW at L1. 2 Bench test runs with 1 simulation run superimposed.

The results show a close match between simulation and bench test results. Simulation results for COP in the AWGN tests lie within bench test values. The simulation values for COP variance and carrier phase jitter also follow bench test results closely, and coincide with a number of bench test points. AWGN tests show the simulation as losing lock earlier than the bench tests. However due to the step in bench test loading, the precise point of loss of lock for the receiver under test is known loosely between a $C/(N_o+I_o)$ value of 23 dB-Hz and 12 dB-Hz, which is consistent with the simulation.

CW tests also show a similarity between simulation and bench tests. Correlator output power degrades with increasing CW interference until loss of lock occurs at a $C/(N_o+I_o)$ value of 31 dB-Hz for bench tests. Simulation results continue to degrade until loss of lock at a $C/(N_o+I_o)$ of 26 dB-Hz. Variance in COP for bench test results shows a higher value of about 2 dB than for simulation, above the noise floor. This may be due to inaccuracies in modeling the noise floor, as may be seen from the slightly different noise floor levels (Figures 64b and 65b).

Carrier phase jitter shows a close match between simulation and bench tests for CW interference: bench test data points follow closely with simulation values until the receiver under test loses lock at a $C/(N_o+I_o)$ value of 31 dB-Hz.

5.8 Conclusions

The performance of four selected test statistics is summarized in Table 12. Correlator output power has been shown to demonstrate the most robustness to variations in *type* of interference, with a normalized slope range of 0.64. It is most sensitive to coherent CW on strong spectral lines. Conversely, it is least sensitive to CW interference on weak spectral lines. The correlator output power variance and carrier phase jitter both have a large NSR.

Table 12: Comparison of Test Statistic Performance

	Correlator Output Power	Correlator Output Power Variance	Carrier Phase Jitter	AGC Gain
Normalized Slope Range	0.64	2.39	2.69	7.57 (0.51 ⁵)
Greatest sensitivity	CW at L1+7KHz	Signal Attenuation	Signal Attenuation	Pulsed AWGN Interference
Least sensitivity	CW at L1+1KHz	CW at L1+1KHz	CW at L1+1KHz	Signal Attenuation

indicating that they are less robust than correlator output power. However they are most sensitive to a different form of interference : multipath and signal blockage which cause attenuation in the tracked signal power. AGC gain shows minimal sensitivity to signal attenuation. It has an NSR of 7.57, which drops to 0.51 when the insensitivity to signal attenuation is excluded from the computation. AGC gain shows the largest sensitivity for pulsed interference.

The greatest reliability can be achieved, with maximum robustness by using a combination of correlator output power and its variance, along with data bit parity checks to detect loop capture. Inclusion of carrier phase jitter adds some redundancy and thus extra safety, but may also increase the occurrence of false alarms. The simultaneous use of multiple test statistics in this manner provides an overall solution with greater reliability and robustness than with the use a single statistic, since some statistics exhibit maximum sensitivity to types of interference to which others are least sensitive.

AGC Gain, due to its minimal sensitivity to signal attenuation has the potential to provide the additional information on type of interference. This is achieved by comparing its output

to the other three test statistics. A large deviation for the other three test statistics occurring coincidentally with minimum or no deviation in AGC gain would indicate the tracked signal is being attenuated. A fast AGC would also discern pulsed interference for this same reason.

⁵ AGC Gain has normalized slope range of 0.505 when its minimal sensitivity to signal attenuation is excluded from the computation.

Chapter 6

PSEUDOLITES TO MITIGATION SERVICE OUTAGES DUE TO INTERFERENCE

6.1 Introduction

Up to this point methods for detecting interference have been discussed. By early detection of interference, the integrity of GPS is enhanced. However integrity monitoring does not enhance signal availability. Indeed it may actually adversely affect continuity, given the false alarms that accompany any detection scheme. For continuity-critical applications such as aircraft landing, it becomes necessary to enhance the continuity of GPS, in addition to protecting system integrity.

In this section we address the availability and continuity of the GPS signal. Interference mitigation techniques are presented based on the use of pseudolites. Airport-located pseudolites (APLs) mitigate interference by providing a strong clean signal robust in the face of interference. APLs also enhance the geometry of the GPS system, providing for a better navigation solution that is robust against satellite outages [20].

Use of pseudolites is not without its drawbacks. Chief among these is the near-far problem. The range of a user to the pseudolite varies greatly, whilst range to GPS satellites remains fairly constant. Therefore power received from the pseudolite will also vary greatly, and will jam the receiver at close ranges. The very pseudolite that was intended to mitigate interference could therefore become a source of interference. This chapter also presents signal designs for pseudolites that would mitigate the near-far problem through the use of spread spectrum and time division multiple access methods.

6.2 Interference Mitigation Through Use of Airport Pseudolites

Pseudolites are ground-based transmitters, emitting a GPS-like signal usually on L1. The concept of using pseudolites is certainly not new. There has been a lot of recent interest in pseudolites, usually as a form of augmentation for differential GPS. Lawrence, et. al. and Pervan, et. al. proposed the use of hyperbolic APLs [16], [17] and [18]. Integrity beacons have been proposed by Cohen, et. al. [19] to provide high precision with integrity. A. J. Van Dierendonck suggests the use of pseudolites for marine navigation [36].

While the role of pseudolites in augmenting GPS accuracy is well established, it is also true that pseudolites play a role in mitigating interference. Airport located pseudolites will not only extend the accuracy and integrity of GPS, but also show the potential to enhance system availability and continuity. The following covariance analysis illustrates the benefits of APLs in this role.

6.2.1 APL Covariance Analyses

This analysis demonstrates the interference mitigation benefits from one and two APLs. Consider an aircraft on approach to an airport on a 3-degree glide slope, as shown in Figure 67. The aircraft is located at a horizontal distance of 600 m from runway threshold, and at a height of 30 m. The first pseudolite, APL1, is located at the runway threshold. APL2 is located 1200m further down the runway. The latitude and longitude of this airport are selected to coincide with the coordinates of San Francisco International airport. This analysis will consider GPS availability and system accuracy, under varying interference conditions, over a period for 24 hours, for a healthy 24 satellite GPS constellation, and for a 21-satellite constellation (3 failed satellites).

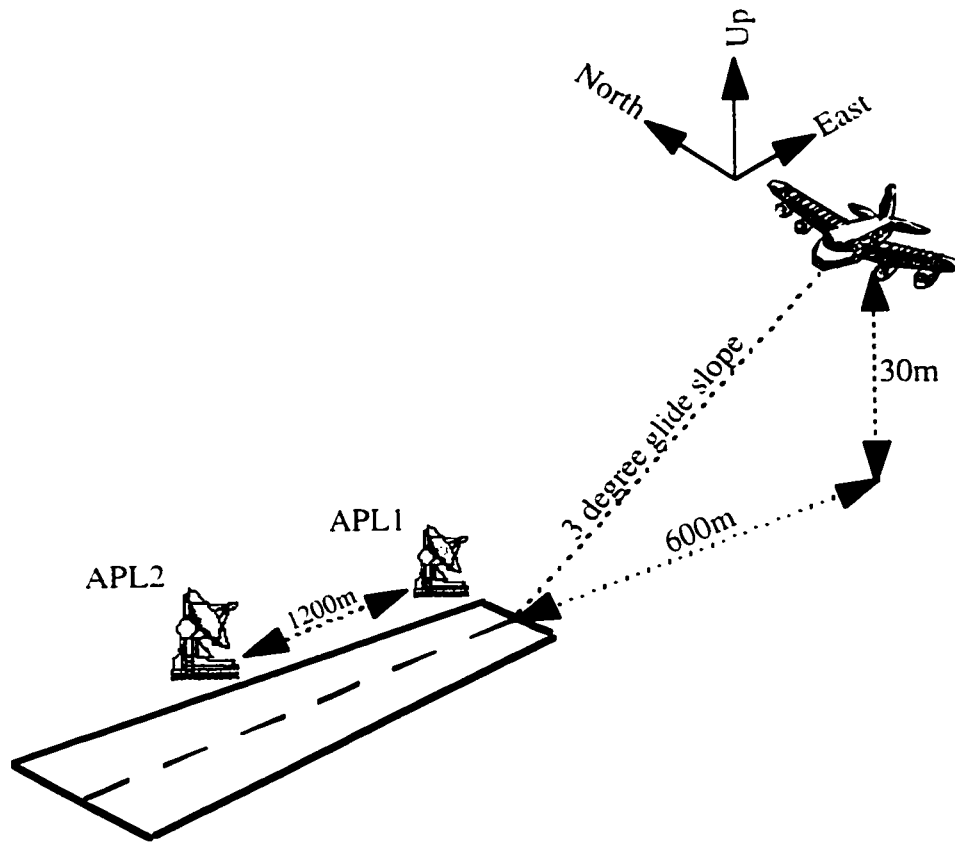


Figure 67: Interference Mitigation Scenario: An Aircraft on a 3-degree approach path

The following definitions are made:

az^i : azimuth of i^{th} satellite, as perceived from aircraft;

el^i : elevation of i^{th} satellite, as perceived from aircraft;

The geometry matrix for visible satellites at any snapshot in time is given by [20]:

$$G = \begin{bmatrix} \sin(az^1)\cos(el^1) & \cos(az^1)\cos(el^1) & \sin(el^1) & 1 \\ \vdots & \vdots & \vdots & \vdots \\ \sin(az^k)\cos(el^k) & \cos(az^k)\cos(el^k) & \sin(el^k) & 1 \\ \frac{x^1}{r^1} & \frac{y^1}{r^1} & \frac{z^1}{r^1} & 1 \\ \frac{x^2}{r^2} & \frac{y^2}{r^2} & \frac{z^2}{r^2} & 1 \end{bmatrix} \quad (6.2.1.1)$$

where: x^i , y^i , and z^i are the along-track, cross-track and vertical displacements of APL_i from the aircraft, and r^i is the range of APL_i from the aircraft:

$$r^i = \sqrt{(x^i)^2 + (y^i)^2 + (z^i)^2} \quad (6.2.1.2)$$

From Figure 67, for an east-north-up coordinate axis centered on the aircraft, the pseudolite positions are given by:

$$\begin{aligned} [x^1 \ y^1 \ z^1] &= [-600, 0, -30] \\ [x^2 \ y^2 \ z^2] &= [-1800, 0, -30] \end{aligned} \quad (6.2.1.3)$$

The 2-sigma vertical accuracy is given by [1]:

$$2\sigma_v = 2\sqrt{\left[\left(G^T P_p^{-1} G \right)^{-1} \right]_{3,3}} \quad (6.2.1.4)$$

where P_p is the covariance matrix of range measurements.

Assuming that measurement errors are independent across satellites, and no APLs are operating, P_p is given by the diagonal matrix with elements σ_I^2 [20]:

$$P_p = \left[\sigma_I^2 = \frac{d(cT_c)^2 BW_{L,1-sided}}{2 \left(\frac{C_x(el^k)}{N_o + I_o} \right)} \left[1 + \frac{2}{(2-d) \left(\frac{C_x(el^k)}{N_o + I_o} \right) T_{square}} \right] \right] \quad (6.2.1.5)$$

where d = correlator spacing = 0.1 for this analysis:

T_c = C/A code chip width, = $1\mu s$

$BW_{L,1-sided}$ = 1-sided tracking loop bandwidth, = 1/50 Hz

T_{square} = squaring loss, from early minus late power detector, = 1/50

$C_x(el^k)$ = carrier power of satellite k , which is a function of the elevation angle el^k .

C_x is obtained empirically from a curve fit to observed data. This curve fit is shown in Figure 31.

With one APL turned on, with a pulse duty cycle, α , of 10%, the measurement covariance matrix is given by [20]:

$$P_p = \text{Diag} \left[\begin{array}{l} \text{Diagonal elements for all visible GPS satellites:} \\ \sigma_I^2 = \frac{d(cT_c)^2 \frac{BW_{L,1-sided}}{1-\alpha}}{2 \left(\frac{C_x(el^k)}{N_o + I_o} \right)} \left[1 + \frac{2}{(2-d) \left(\frac{C_x(el^k)}{N_o + I_o} \right) T_{square}} \right] \\ \text{Diagonal element for single APL:} \\ \sigma_I^2 = \frac{d(cT_c)^2 \frac{BW_{L,1-sided}}{\alpha}}{2 \left(\frac{C^{APL}(el^k)}{N_o + I_o} \right)} \left[1 + \frac{2}{(2-d) \left(\frac{C^{APL}(el^k)}{N_o + I_o} \right) T_{square}} \right] + \frac{1^2}{3} \end{array} \right] \quad (6.2.1.6)$$

C^{APL} is the pseudolite signal power, assumed to have equal strength with GPS satellites for a user 10 km away. The additional $1/3$ term accounts for pseudolite signal multipath.

When both APLs are in use, it is assumed carrier phase ranging to the APLs is used, as two APLs can be configured to provide hyperbolic lines of position [1], [18]. Each APL has a duty cycle of 10%, resulting in a 20% total duty cycle as observed by GPS signals. The measurement covariance with both APLs in use is given by [20]:

$$P_p = \text{Diag} \left[\begin{array}{l} \text{Diagonal elements for all visible GPS satellites:} \\ \sigma_I^2 = \frac{d(cT_c)^2 \frac{BW_{L,1\text{-sided}}}{1-2\alpha}}{2 \left(\frac{C_x(\text{el}^k)}{N_o + I_o} \right)} \left[1 + \frac{2}{(2-d) \left(\frac{C_x(\text{el}^k)}{N_o + I_o} \right) T_{\text{square}}} \right] \\ \\ \text{Diagonal elements for each APL:} \\ \sigma_0^2 = \frac{\left(\frac{\lambda}{2\pi} \right)^2 \frac{BW_{C,1\text{-sided}}}{\alpha}}{2 \left(\frac{C^{\text{APLi}}(\text{el}^k)}{N_o + I_o} \right)} \left[1 + \frac{1}{2 \left(\frac{C^{\text{APLi}}(\text{el}^k)}{N_o + I_o} \right) T_{\text{square}}} \right] + 0.022^2 \end{array} \right] \quad (6.2.1.7)$$

with $BW_{c,1\text{-sided}}$ = Costas loop carrier tracking bandwidth, = 10 Hz

With carrier tracking, multipath for the APLs is set at 0.022m standard deviation.

Interference is varied from the nominal receiver noise floor value of $N_o = -200.5$ dBW/Hz up to a peak value $N_o + I_o = -170.5$ dBW/Hz. to give a $C_{\text{zenith}}/(N_o + I_o)$ range of 50 dB-Hz to 20 dB-Hz for $C_{\text{zenith}} = -150.5$ dBW.

Satellite orbits are propagated using ephemerides from a real GPS almanac file. A 5 degree elevation mask is used to determine visible satellites. Results are shown in Figures 68, 69 and 70 for a healthy GPS constellation of 24 satellites, and in Figures 71, 72 and 73 for the case of 3 satellite failures (21 satellites).

6.2.2 Results of APL Covariance Analyses

For an aircraft on approach, we will use the same 2 m protection level discussed in the earlier sections of this work: if GPS accuracy degrades beyond 2 meters, the system is declared unavailable. Figure 68 shows vertical accuracy for a DGPS-only user (no APLs), with 24 healthy satellites. With minimal interference, the general solution is good, over the entire 24 hours. Accuracy does not degrade beyond 2m, which gives 100% availability.

As interference power increases however, GPS-only accuracies degrade rapidly, until at an interference level equivalent to $C_{zenith}/N_0 = 27$ dB-Hz. (C/N_0 at satellite zenith), GPS only accuracy is degraded beyond the 2 m level 100% of the time, giving a 0% availability. Comparing this with Figure 69 (1APL + GPS), and Figure 70 (2APLs + GPS) shows the APL enhancement in GPS accuracy and availability : as interference increases, there is less degradation in accuracy for the case of 1 APL, and even better performance is obtained with 2 APLs. The enhancement is highlighted in Figure 74, which gives GPS availability as a function of interference for this same 24 satellite 24 hour configuration. This figure shows that at $C_{zenith}/N_0 = 27$ dB-Hz, augmentation of stand-alone GPS with 1 APL boosts system availability from zero percent to 82%. Adding a second APL will result in 100 % system availability! This enhancement results from: the added accuracy available with the carrier tracked APLs, the augmented GPS geometry, and the peak amplitude of the APL signals, relative to interference.

2Sigma-V over 24hrs at SFO : DGPS; 24 SV Constellation

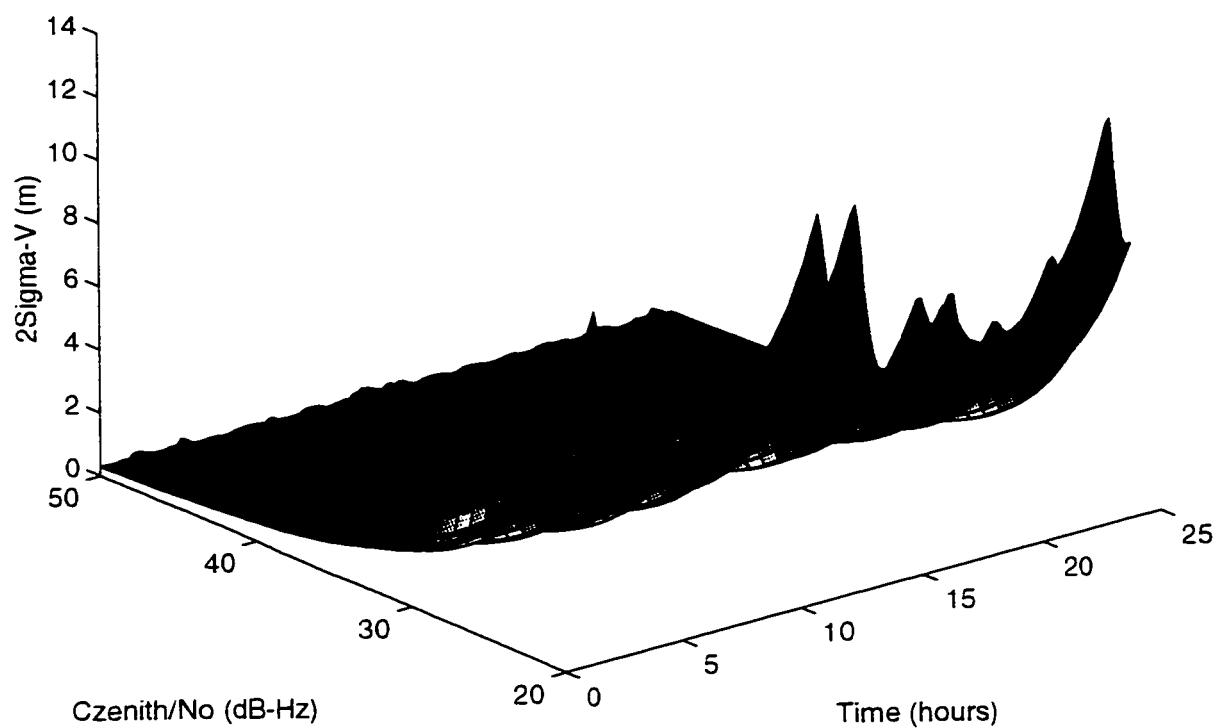


Figure 68: Vertical Accuracy over 24 hours and over 30 dB interference power range, for a DGPS-only User at San Francisco International Airport. Full GPS Constellation of 24 satellites are available

2Sigma-V over 24hrs at SFO : 1 APL; 24 SV Constellation

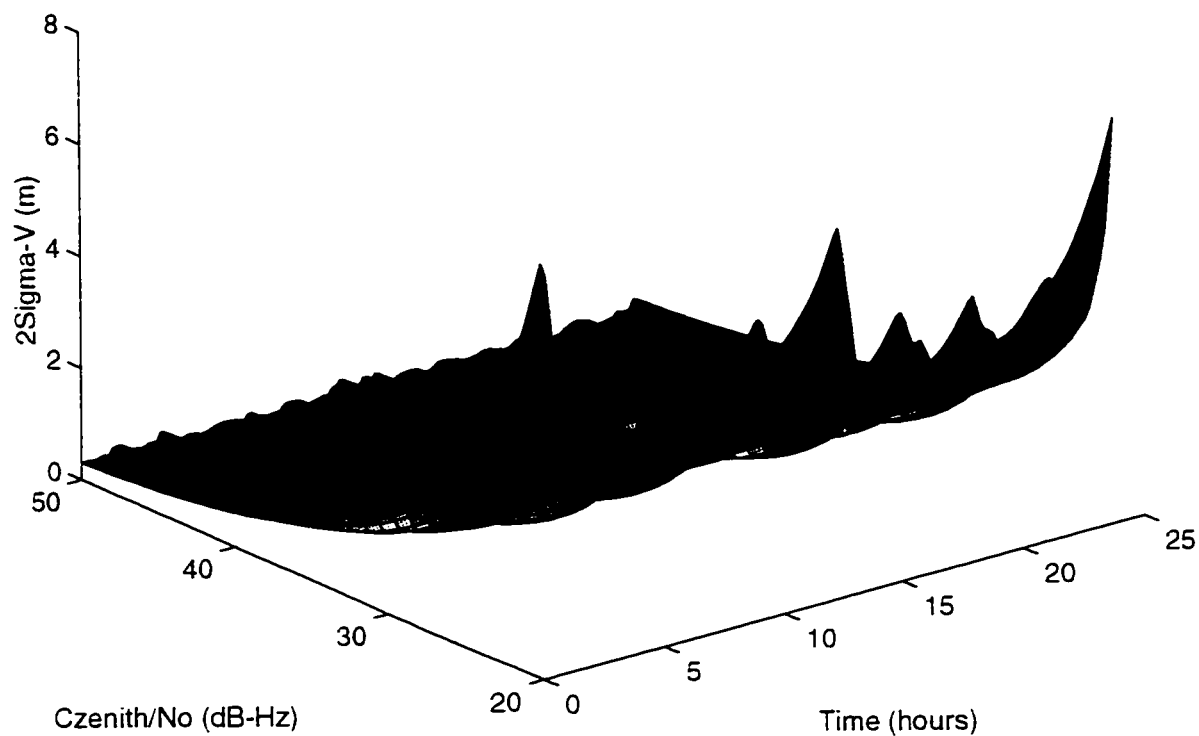


Figure 69: Vertical Accuracy over 24 hours and over 30 dB interference power range, for a 1 APL + GPS User at San Francisco International Airport. Full GPS Constellation of 24 satellites are available

2Sigma-V over 24hrs at SFO : 2 APLs; 24 SV Constellation

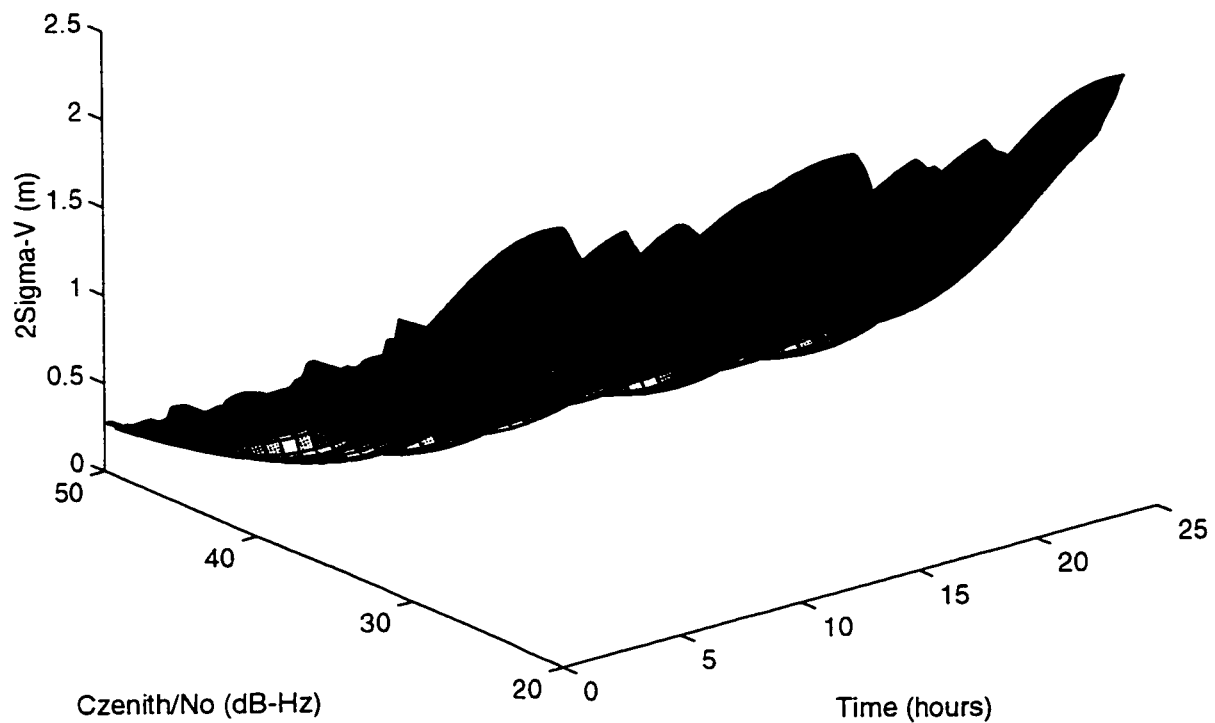


Figure 70: Vertical Accuracy over 24 hours and over 30 dB interference power range, for a 2 APLS + GPS User at San Francisco International Airport. Full GPS Constellation of 24 satellites are available

2Sigma-V over 24hrs at SFO : DGPS; 21 SV Constellation

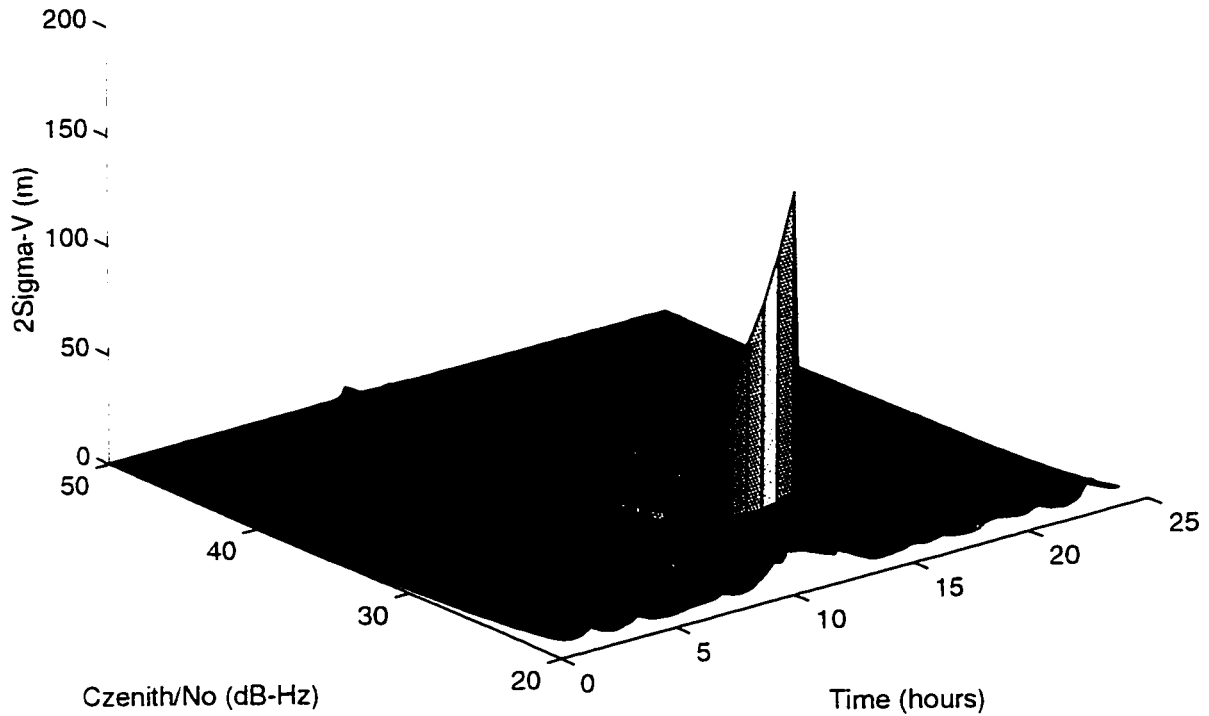


Figure 71: Vertical Accuracy over 24 hours and over 30 dB interference power range, for a DGPS-only User at San Francisco International Airport. Analysis assumes a 3 GPS satellite failure, leaving only 21 healthy satellites

2Sigma-V over 24hrs at SFO : 1 APL; 21 SV Constellation

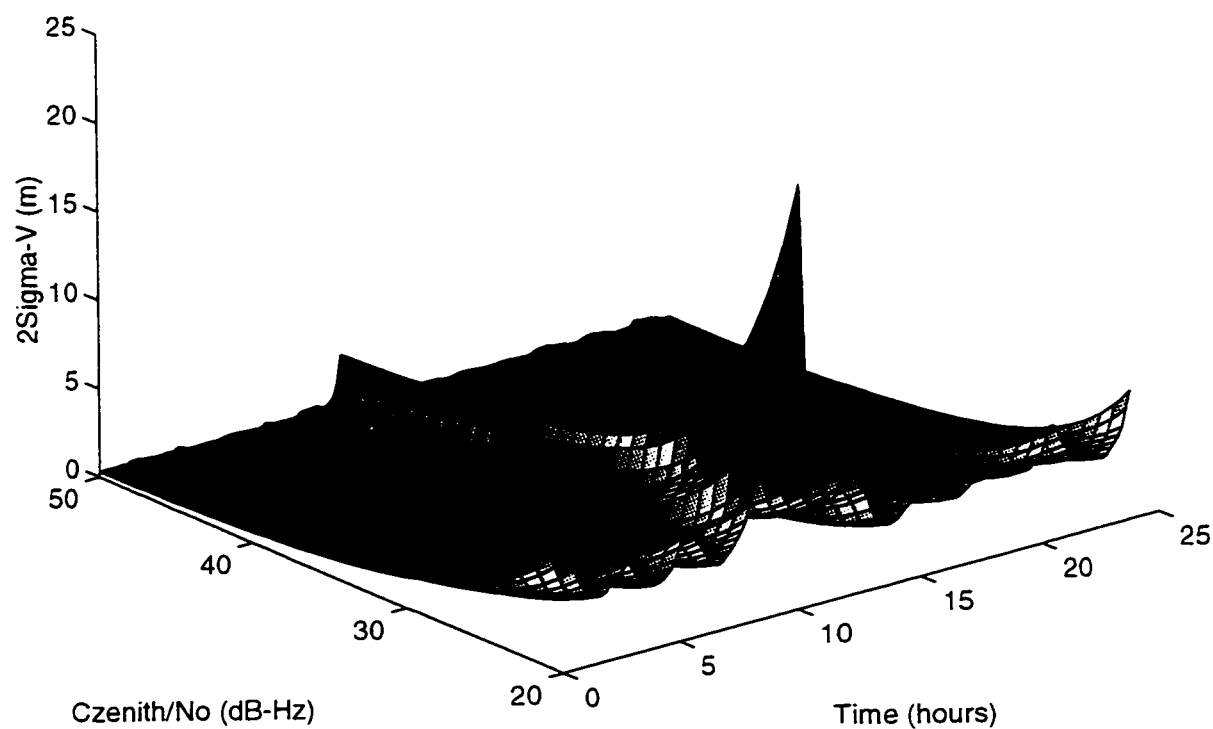


Figure 72: Vertical Accuracy over 24 hours and over 30 dB interference power range, for a 1 APL + GPS User at San Francisco International Airport. Analysis assumes a 3 GPS satellite failure, leaving only 21 healthy satellites

2Sigma-V over 24hrs at SFO : 2 APLs: 21 SV Constellation

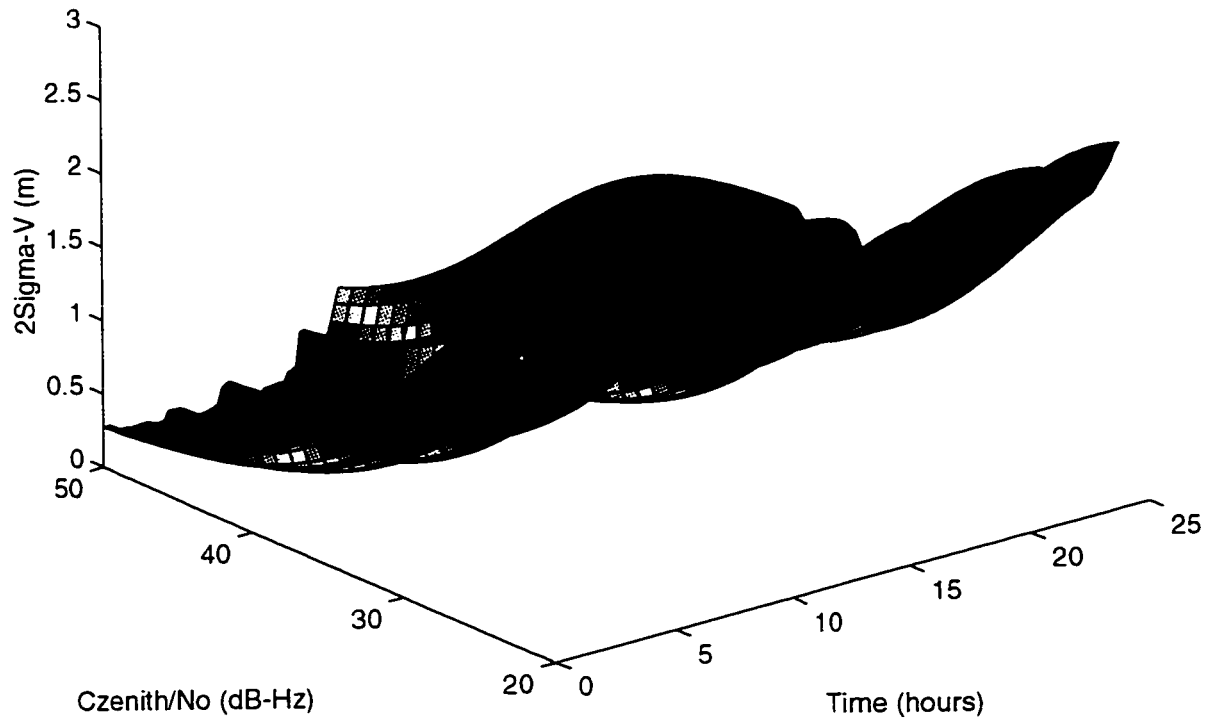


Figure 73: Vertical Accuracy over 24 hours and over 30 dB interference power range, for a 2 APLS + GPS User at San Francisco International Airport. Analysis assumes a 3 GPS satellite failure, leaving only 21 healthy satellites

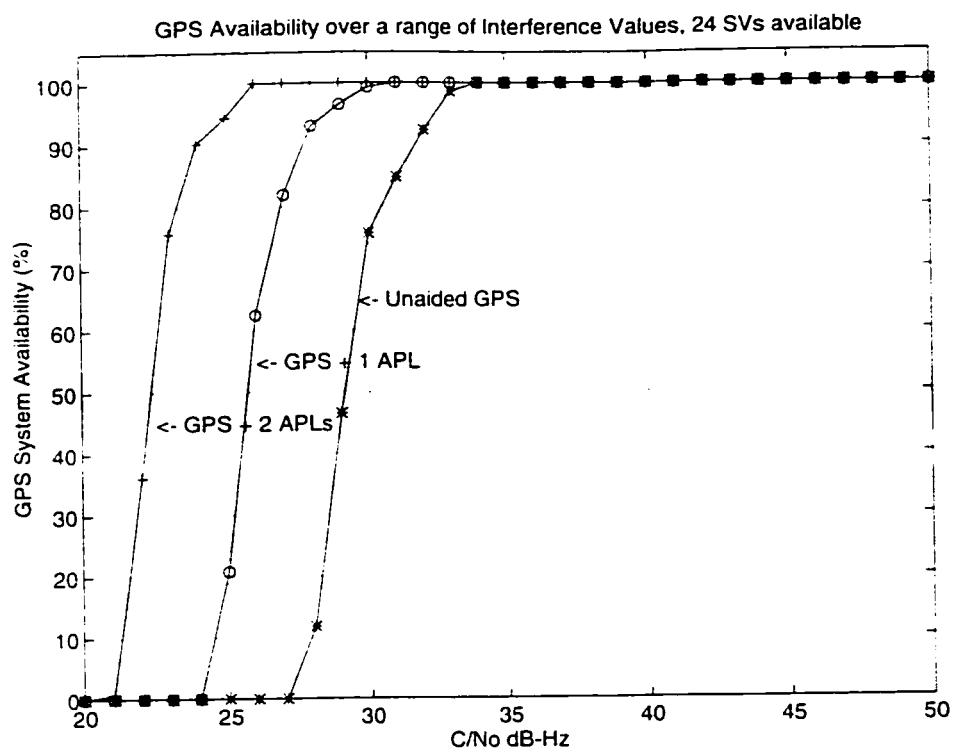


Figure 74: GPS Availability over 24 hours for a User located at San Francisco, 24 Healthy Satellites

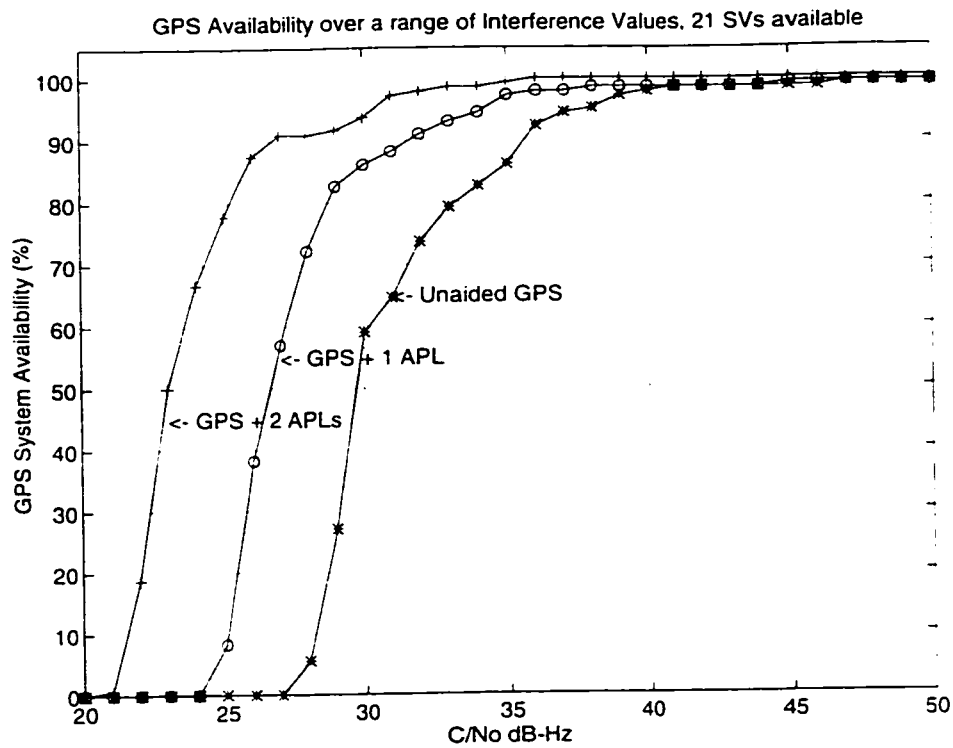


Figure 75: GPS Availability over 24 hours for a User located at San Francisco, 3 satellite failure (21 Healthy Satellites)

With a triple satellite failure, standalone GPS degrades rapid with interference, as can be seen from Figure 71. Although the three failed satellites are selected randomly, the failures produce a period around noon with only four satellites in view that have poor geometry. During this brief period the vertical accuracy degrades beyond 100 m. Comparing this with the worst case values for the 1 APL (Figure 72) and the 2 APL case (Figure 73), worst case vertical accuracies are 24m and 2.7m respectively. GPS availability plots are shown in Figure 75, for all slices of interference across the 24 hours plots for this 21 satellite constellation case. Once again the APL augmented cases hold out with higher levels of GPS availability over severe interference, even when stand-alone GPS availability drops to zero

This demonstrates the APL enhancements to GPS accuracy, continuity and availability, in the face of strong interference, and also in the event of satellite outages.

6.3 Pseudolite Signal Design

6.3.1 Introduction

We have seen that APLs mitigate interference by virtue of their large peak power relative to the interference power. However we must still design the APL signal such that it does not interfere with GPS.

The major problem with the use of pseudolites, the 'near-far' problem, arises because of the large variation of the user-to-APL range. The average power being received from the GPS satellites remains approximately constant due to the large distance of the satellites from users. The pseudolite power on the other hand, varies a great deal, inversely proportional to the square of the user's distance from the pseudolite, and can overwhelm incoming GPS satellite signals.

The near/far problem varies in severity for different applications of pseudolites. It is most severe when pseudolites are used for terminal-area-data-broadcast, since its service area is widest. A signal being received by a user 50km away from the pseudolite would become 60 dB stronger when the user is only 50 m away. On the other hand an aircraft on a precision approach would experience a 45 dB APL/satellite power ratio, if the APL signal were acquired with equal strength as the GPS satellite's, 10km from start of runway, and the aircraft flew a 3-degree glide slope directly over an APL located 1000m from runway base (see Figure 76 below).

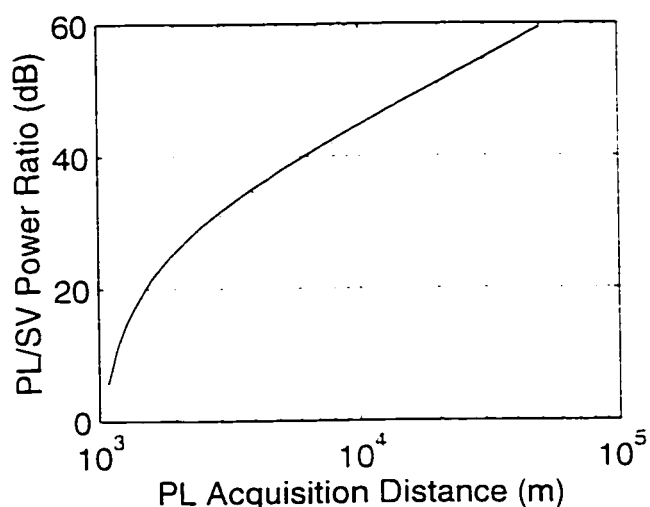


Figure 76: Pseudolite/Satellite Maximum Power ratio vs. Pseudolite Acquisition Distance for an Aircraft on a 3-degree Glide Slope assuming Closest Distance is 50 meters

If the same aircraft were to acquire the APL signal 5 km from runway threshold, the maximum APL/satellite power ratio experienced would be 38 dB.

For APL-aided carrier phase ambiguity resolution in the Integrity Beacon Landing System (IBLS) implementation of aircraft high precision approach [19], the APL provides a

'bubble' through which the aircraft must pass to resolve carrier cycle integer ambiguities [25].

The size of this bubble is limited by the near/far problem. Figure 77 shows the maximum size of an APL 'bubble' if the APL transmits a C/A code. This figure is based on the 24dB signal-to-interference (S/I) margin inherent in length 1023 Gold codes, and on receivers that require a 10dB minimum S/I margin for operation. For the IBLS system, it is desirable to increase the size of the APL bubble.

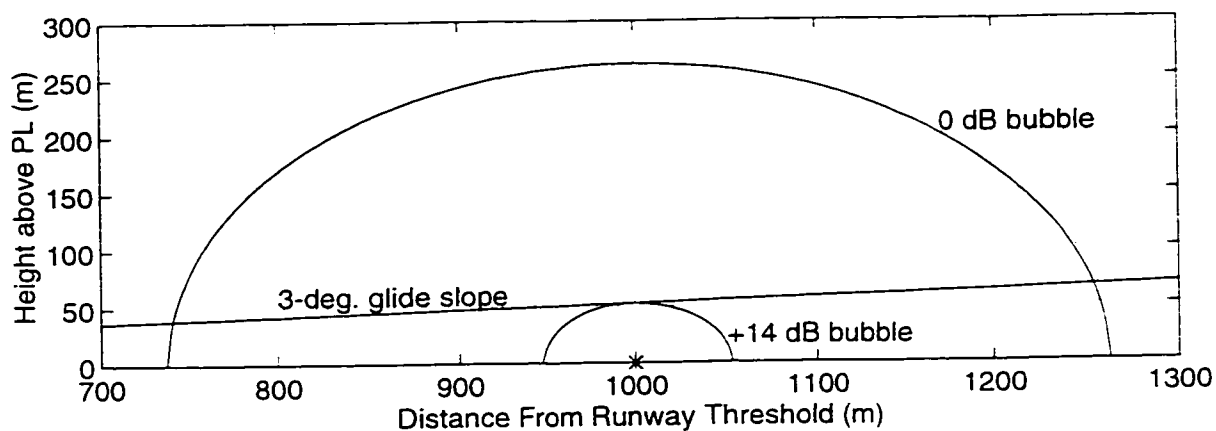


Figure 77: Near-Far Problem with Pseudolite located 1KM from Runway Threshold

The APL signal should be designed to minimize interference to non-participating receivers, as well as to participating receivers. A non-participating receiver is a GPS receiver which is not re-designed or modified in any way to receive or ignore the APL signal. Therefore to guarantee the efficient operation of this class of receivers in the vicinity of an APL, effort should be focused primarily on the design of the APL signal, rather than the redesign of specific receivers. This approach is adopted for this work. It is desirable also to have the APL signal compatible with existing receiver designs, and minimize cost of integration into participating receivers.

Figure 7 shows the signal processing path for a generic GPS receiver. Incoming signals are downconverted, digitized, and fed to the correlators. A locally generated signal with the C/A code being searched for is generated by the receiver and correlated with the incoming signal.

C/A codes, which are Gold codes of length 1023, have a 4-valued autocorrelation function: 0 dB (perfect match), -23.9 dB, -24.2 dB, and -60.2 dB. The cross correlation function, a measure of the level of interference from other signals, is 3-valued, with values -23.9 dB, -24.2 dB, and -60.2 dB. With small differences in the carrier frequencies between two signals, say due to Doppler effect, we should expect no more than a 3 dB degeneration in cross correlation performance [32]. To minimize the near-far interference from the APL signal, one should therefore seek to minimize the cross correlation level of the APL signal with the GPS Gold codes.

There have been various methods proposed to solve the near-far problem. These generally fall under three categories: FDMA (frequency division multiple access), TDMA (time division multiple access), and CDMA (code division multiple access) methods.

FDMA involves placing the APL signal on a carrier frequency different from that being used by the GPS signal (1575.42 MHz). The frequency offset may be large or small. Alison Brown [24] proposes a frequency offset of about 40 MHz by placing the APL signal in the 1610-1626.5 MHz band. A. J. Van Dierendonck [37] suggests placing the APL signal on the first null of the spectrum of the GPS satellite signal, an offset of 1.023MHz from the L1 center frequency. This minimizes interference from the APL signal. However this also requires modification to current receiver designs. TDMA involves gating the APL signal for fractions of a full duty cycle. Thomas Stansell [34]

proposes using a 10% duty cycle by transmitting 93 out of 1023 chips each cycle of the APL signal.

This work focuses on CDMA solutions, combined with TDMA (pulsing). CDMA methods have the advantage of lower costs of implementation due to their high degree of compatibility with current receiver design. In addition, they do not introduce any inter-frequency biases in the receiver hardware. Codes of length 4092 (4×1023) at the L1 frequency band are explored. These codes are formed using length-1023 Gold codes at the C/A code rate, 1.023MHz. Use of P-Code rate code (10.23MHz) is also investigated. It is shown that one can achieve greater levels of signal-to-interference margin gains, while maintaining reasonable compatibility with current receiver design.

The following section provides an analysis of random sequences, to establish our theoretical best case CDMA performance as a function of code length.

6.3.2. Random Sequence Analyses

Given two binary sequences x and y , each of length N , consisting of +1s and -1s, we desire to know what is the RMS value of cross-correlation value, $\phi_{xy}(\tau)$, between x and y .

The following assumptions are made about the sequences x , and y :

i) x and y are purely random sequences:

x_i is independent of x_j for $i \neq j$.

y_i is independent of y_j for $i \neq j$.

ii) x and y are statistically independent:

x_i is independent of y_j for all i, j .

$$\text{RMS}(\emptyset_{xy}(\tau)) = \sqrt{E\{\{\emptyset_{xy}(\tau)\}^2\}}$$

$$\text{where } \emptyset_{xy}(\tau) = \int_{t=0}^{t=NT_c} x(i)y(i+\tau)dt \quad (6.3.2.1)$$

T_c = duration of 1 chip in seconds:

τ = time offset.

For square pulses, and $\tau = nT_c$ equation 6.3.2.1 can be replaced with:

$$\emptyset_{xy}(\tau) = \sum_{i=0}^{N-1} x(i)y(i+\tau)$$

where τ is the chip offset.

Therefore:

$$E\{(\emptyset_{xy}(\tau))^2\} = E\{x_0^2 y_{\tau}^2 + x_1^2 y_{\tau+1}^2 + \dots + x_{N-1}^2 y_{\tau+N-1}^2 + x_0 x_1 y_{\tau} y_{\tau+1} + \dots \text{ other cross terms}\}$$

Since x and y are statistically independent:

$$E\{x_i x_j y_k y_l\} = 0 \text{ for } i \neq j \text{ or } k \neq l$$

$$\text{and } E\{x_i x_j y_k y_l\} = 1 \text{ for } i=j \text{ and } k=l$$

$$\text{Therefore } \text{mean}(\emptyset_{xy}(\tau))^2 = \sum_{i=0}^{N-1} 1 = N$$

$$\text{and } \text{RMS}(\emptyset_{xy}(\tau)) = \sqrt{N}$$

Normalized by code length, N:

$$\text{RMS}(\phi_{xy}(\tau)) = \frac{\sqrt{N}}{N} = \frac{1}{\sqrt{N}}$$

This implies that the expected rms cross correlation value between two equal length random independent binary sequences is inversely proportional to the square root of the code length.

For random codes of length = 1023, the expected normalized rms cross correlation value is $1/32 = 30.1\text{dB}$.

By increasing code length to 2046 one would expect a 3dB improvement in cross correlation level. A code length of 4092 (4×1023) gives an expected rms correlation value of 36.1dB.

These results can be compared to computed values for Gold codes formed from pseudo random sequences.

$$\begin{aligned} \text{sqrt}(E\{\{\phi_{xz}(\tau)\}^2\}) &= \text{sqrt}\{.5(-1^2) + .25(-65^2) + .25(63^2)\} \\ &= 45.3 = 27.1 \text{ dB} \end{aligned}$$

This gives an expected rms value of 27.1dB, within 10% of the value predicted from purely random code analysis. In addition GPS Gold codes give a worst case cross correlation level of 23.94dB, corresponding to the -65 cross correlation level. This performance is expected to degrade by 3dB (to 20.94dB) when Doppler shifts in carrier frequency are introduced [32].

It is therefore to be expected that by using a PR code of length 4092, we would expect a 6dB improvement in cross correlation level. This is verified in the next section.

On the other hand, correlating a faster code with a slower one gives different results. Let x be the purely random binary sequence described in the paragraphs above, and let z be another purely random binary sequence of +1s and -1s, at a chipping rate 10 times the rate of sequence x . As in the previous analysis x and z are assumed to be statistically independent. The expected rms value of the cross correlation function between x and z is given by:

$$\text{RMS}(\phi_{xz}(\tau)) = \sqrt{E\{\{\phi_{xz}(\tau)\}^2\}}$$

$$\text{with } \phi_{xz}(\tau) = \sum_{i=0}^{10N-1} x(i/10)z(i+\tau)$$

As before,

$$E\{x_i x_j z_k z_l\} = 0 \text{ for } i \neq j \text{ or } k \neq l$$

$$\text{and } E\{x_i x_j z_k z_l\} = 1 \text{ for } i=j \text{ and } k=l$$

Following similar steps as in the previous case, we arrive the result:

$$E\{\text{RMS}(\phi_{xz}(\tau))\} = \frac{1}{\sqrt{10N}} \text{ normalized.} \quad (6.3.2.2)$$

Extrapolating to PR sequences, this results implies an additional 10dB cross-correlation margin may be gained by having the APL sequence at P-code rates, and correlating for 1 ms. This result gets better as the length of the APL code is extended. A 20 ms period for

the APL sequence would be an optimal choice as this corresponds to the period of the 50 bps data modulating the C/A code, and therefore is the maximum time between decisions in a receiver. Such a code would contain 204,600 chips, and would result in an additional cross correlation margin gain of 13 dB for a total of 23 dB.

6.3.3 Code Realizations

This section presents a number of actual sequences to achieve the theoretical margins discussed in the previous section.

Let x be a sequence of length $2N$, formed by concatenating a single sequence from a family of length- N Gold codes. Let y be another sequence of length $2N$ formed in a similar manner as x above, from a different sequence in the same length- N family of Gold codes. Let the sign on the second half of y be reversed (see Figure 78 below).

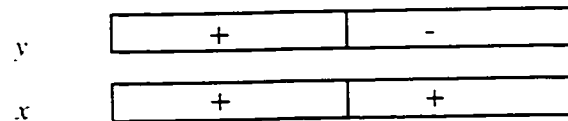


Figure 78: Length $2N$ Sequences, with one have its second half switched in sign

Then

$$\phi_{xy}(\tau) = \sum_{i=0}^{2N-1} x(i)y(i+\tau) = 0 \quad (6.3.3.1)$$

Cross-correlating x and y with a cross correlator integration time equal to twice the duration of the period of sequence- x (equal the period of sequence y) would result in zero

correlation output, or absolutely no interference (infinite dB) to signal x , assuming y is the interfering signal.

However Doppler offsets degrade this ideal performance. With differences in carrier frequencies of x and y equation 6.3.3.1 becomes:

$$\phi_{xy}(\tau) = \sum_{i=0}^{2N-1} x(i)y(i+\tau)\exp(j2\pi f_d T_c i) \neq 0 \quad (6.3.3.2)$$

where f_d is the difference in carrier frequency.

Equation 6.3.3.2, the ambiguity function, is plotted in Figure 79 using two Gold codes of order 5 (length 31) for a frequency range of 0 to 6.5 KHz. The figure shows that for zero frequency offset there is perfect cancellation - or zero correlation. However as the frequency departs from zero, so does the value of the cross correlation function.

Figure 78 introduces the concept of "Switching Sequences". A switching sequence changes the signs of segments and/or repeats, of a given code. In the example above a two bit switching sequence [+ -] was applied to a length $2N$ sequence to produce the sequence y . However as is shown in Figure 79, the perfect cancellation from this simple switching sequence degrades when differences in GPS carrier frequencies exist. Under normal circumstances one may expect up to ± 6 KHz variation in carrier frequencies due to Doppler. A good switching sequence would therefore produce longer sequences with good correlation properties over both time and frequency offsets.

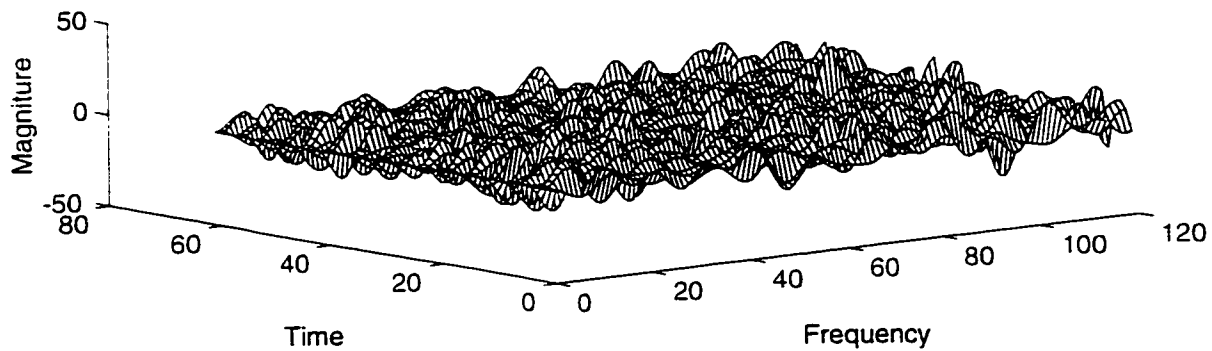


Figure 79: Ambiguity Function for 2 Gold Codes of order 5 (Length 31x2, Switch = +-)

Simulation Setup:

By using longer and more complex switching sequences it is possible to achieve low cross-correlation over time and frequency. To verify this and to ascertain characteristics of good switching sequences, we have exhaustively applied length-8 switching sequences to Gold codes of order 5 (length-31), with quadruple length ($4N = 124$). The Gold codes were generated from two maximal length pseudo random sequences (m-sequences). One m-sequence was generated from a five-stage linear feedback shift register (LFSR) with the generating polynomial (Figure 80):

$$p(x) = x^5 + x^2 + 1$$

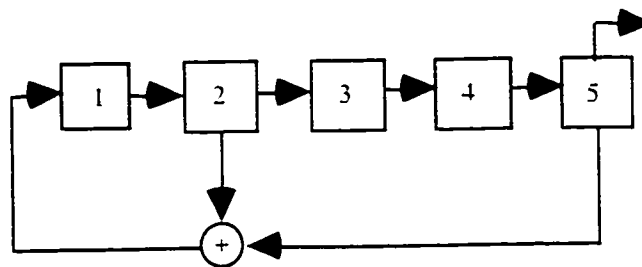


Figure 80: Five Stage Linear Feedback Shift Register

The second m-sequence was obtained by decimating the m-sequence from this polynomial with a factor of 7. The reader is referred to the paper by Robert Gold [27] for details on the characteristics and properties of Gold codes, and to refs. [28], [29] and [31] for more detailed treatment of m-sequences. Each bit in the 8-bit switching sequence switches segments of length 16 or 15 in the 124-bit sequence. The chipping rate of the composite sequence was selected to give similar characteristics as GPS codes when correlated over 0-6.5KHz. Cross-correlation was performed over all 124-chip time offsets, and over 0 - 6.5KHz in increments of 55Hz.

An exhaustive analysis of the results of all possible 8-bit switches indicate that the switching sequences which produce the best codes have low periodic autocorrelation sidelobes, as well as low sums. This combination of properties is necessary to provide good performance over both time and frequency. The simple switching sequence shown in Figure 78., '+ -', though having a low sum ($+1 - 1 = 0$), produces poor results when cross correlation is performed over shifts in frequency (Figure 79) since it has a high autocorrelation sidelobe.

A summary of the nominal performance (no switching sequence applied), the 4 best performance results of cross-correlation over time and frequency, and the best cross-correlation result over time-only, are contained in Table 13. This table shows the switching sequence, the worst-case cross correlation peaks of each switching sequence over time and frequency, as well as the corresponding cross-correlation peak over time-only (no Doppler frequency shifts). The gains over nominal performance are also shown.

Table 13: Processing Gain using Length-4x31 Codes (in dB)

Switching Sequence	Over Time Only		Time & Frequency	
	Proc.Gain	Improved	Proc.Gain	Improved
+++++++	10.7424	0	8.28981	0
+-----	15.02	4.2776	15.02	6.73019
++++-+-	16.763	6.0206	14.4853	6.19549
+--+++-	15.02	4.2776	14.4684	6.17859
+++++--	16.763	6.0206	14.3104	6.02059
++-+++-	inf	inf	10.9666	2.67679

The best time-only cross-correlation result is the zero-interference case, or infinite dB, as shown in Figure 79. However this performance drops to 2.67dB improvement only, when Doppler shifts in carrier frequency are considered.

An exhaustive search was also performed using sequences based on GPS Gold codes. A pseudolite signal was formed from four runs of an unused sequence from the GPS family of length-1023 Gold codes. This sequence was cross correlated with one of the 35 Gold codes used by the GPS satellites. The results are shown in Table 14 below. The best 'worst case' gain of 5.88dB is based on the worst case time-frequency cross correlation value, 21.3 dB, with no switch applied.

Table 14: Processing Gain using Length-4x1023 Codes (in dB)

Switching Sequence	Over Time Only		Time & Frequency	
	Proc.Gain	Improved	Proc.Gain	Improved
+++++++	23.9392	0	21.348	0
+-----	29.9598	6.0206	27.2378	5.8898
++++-+-	29.9598	6.0206	26.7108	5.3628
+--+++-	29.9598	6.0206	26.3312	4.9832
+++++--	29.9598	6.0206	26.3284	4.9804
++-+++-	inf	inf	23.585	2.237

Once more the best performing switching sequences show the same characteristics as in the previous simulation. Cross-correlation results over time only produce infinite gains in signal to interference levels with the simple switching sequence '+-----'. This

theoretically implies absolutely no interference from an APL using this code to a non-participating receiver, irrespective of range⁶, so long as there are no Doppler shifts in carrier frequencies. With as little as 300Hz difference in carrier frequencies, this performance drops to nominal (23.94dB). Worst case peak over 0-6.5KHz for code formed using this switch was 23.58 dB, 0.36 dB worst than nominal performance.

6.3.4 Pseudolite Signal Design Conclusions

Figure 81 shows the modified near-far problem for an aircraft on a 3-degree glide slope approach, with an APL located 1 km from runway threshold. This figure contains the original near-far bubble (innermost pair plotted with continuous lines), as well as the bubbles that would result from APLs utilizing length-4092 codes at C/A code rate, and from length 20460 codes at 10.23 MHz (both plotted with dashes). The figure shows that the radius of the APL bubble can be doubled by an APL using a length-4092 sequence broadcast on L1 band. A fourteen times increase in bubble radius is expected by using a length-20460 code with a chipping rate of 10MHz.

The effect of airframe attenuation of the APL signal has not be addressed in the previous discussions. Figure 82 shows the spherical antennae pattern for an antennae mounted on top of a BAC1-11 aircraft. It can be seen that the aircraft fuselage attenuates GPS signals by as much as 20dB for a transmitter directly below. This property can be an advantage in the APL application, as it provides a natural shield to the GPS-satellite antennae mounted on top an aircraft.

⁶In reality non-linearities in the receiver front end would produce interference as the APL signal power becomes very large.

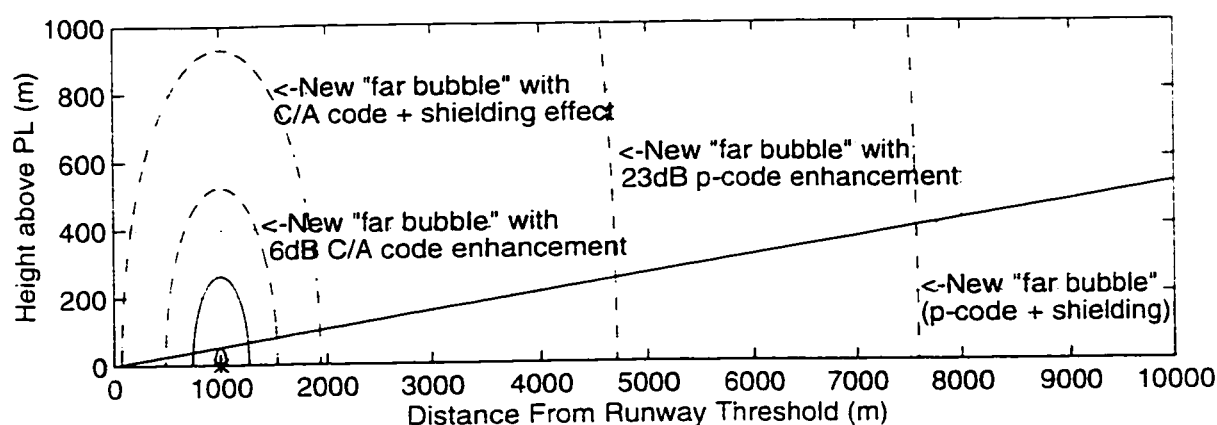


Figure 81: Modified Near-Far Problem with Pseudolite 1 km from Runway Threshold

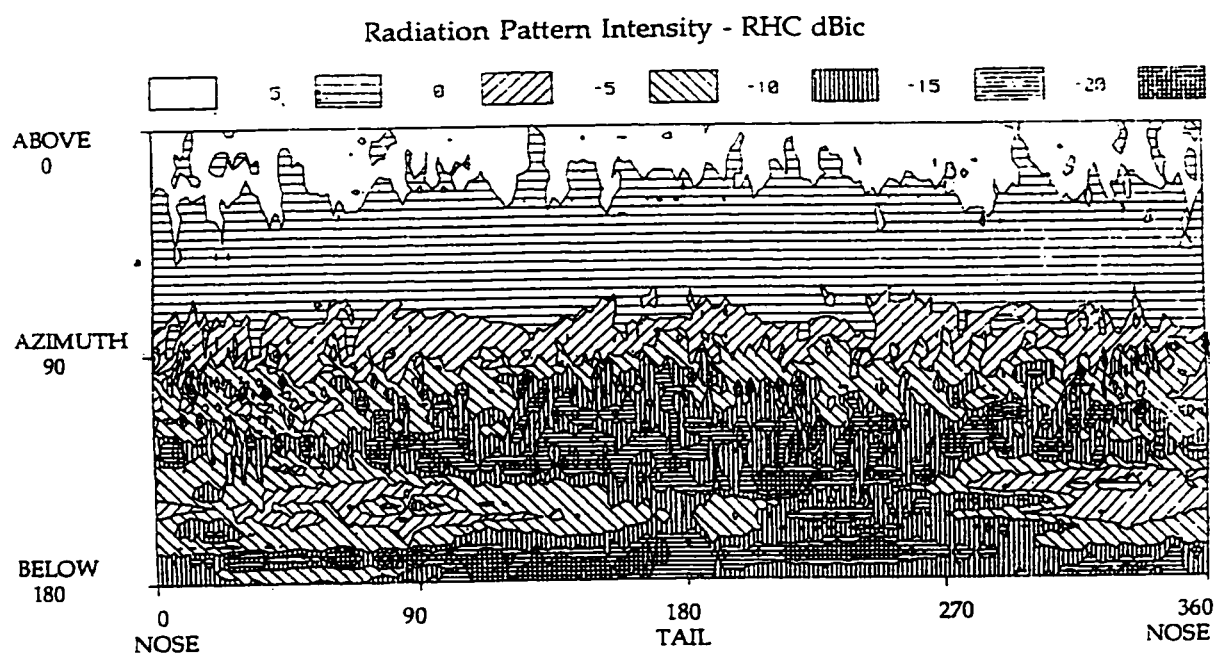


Figure 82: BAC-1-11 Radiation Pattern Spherical Coverage (Courtesy of J.I.R. Owen)

For an aircraft on a 3-degree glide slope, a top mounted antennae would experience a minimum attenuation of 5dB for the signal of an APL located as shown in Figure 77. This is the minimum expected along the chosen flight path. In reality the effect of fuselage

shielding increases as the aircraft approaches overhead the APL. Therefore shielding from the aircraft fuselage is expected to add at least an extra 5 dB gain over the performance of the proposed schemes.

Taking this shielding into account the expected minimum overall performance is plotted in Figure 81 (shown with dash-dot plot). With length-4092 sequences at *C/A* code rate, we expect a minimum gain of 11 dB with a top mounted antennae for an aircraft in the approach configuration shown in Figure 77. With length-20460 sequences at 10.23MHz we expect a 28dB improvement.

Chapter 7

CONCLUSIONS AND FUTURE WORK

The first section of this chapter presents a summary of work done in this thesis. The later section discusses options for future work.

7.1 Summary

7.1.1 Autonomous Signal Quality Monitoring

Autonomous signal quality monitoring has been demonstrated using test statistics derived from fundamental receiver measurements. Three of these quantities, correlator output power, the variance of correlator output power, and carrier phase jitter, are derived from the inphase and quadrature outputs (I, Q) from the correlators, and are therefore satellite / receiver channel specific. The fourth test statistic, AGC Gain, on the other hand is derived from the signal digitization process, and therefore is not channel specific.

These selected test statistics have been shown to respond to interference with a measure of consistency, as characterized by their decision test statistic slope. Correlator output power has been shown to have the least normalized spread in slopes across different types of interference. It therefore is least sensitive to the type of interference, and demonstrates the most robustness of all four statistics.

The selected test statistics have been shown to complement each other, especially for the case of signal attenuation as caused by multipath and physical blockage. Whereas AGC gain is barely sensitive to the presence of signal attenuation, variance of correlator output

power and carrier phase jitter are most sensitive to this type of interference. Also, whilst correlator output power is most sensitive to coherent CW on strong spectral lines, the other test statistics are not.

Correlator output power, when used concurrently with COP variance, along with data bit parity checks to detect loop capture, provides greatest reliability, with maximum robustness. Inclusion of carrier phase jitter adds some redundancy and thus extra safety. When all used concurrently, our chosen test statistics produce a triple redundant system, with a level of reliability and robustness to variations in types of interference, which surpasses the performance of any one single test statistic.

When used in conjunction with the other test statistics, AGC gain provides the ability to discern the type of interference (for pulsed interference with high peak power, and for interference caused by signal attenuation), due to its very high and very low sensitivities to these types of interference respectively.

This section addressed the need for a high level of integrity, and demonstrated that by monitoring these derived receiver parameters, the intrinsic integrity of a GPS receiver will receive a boost.

7.1.2 Interference Mitigation via Use of Airport Pseudolites

The role of airport pseudolites in the mitigation of outages due to interference has been demonstrated. It has been demonstrated by simulation and covariance analyses that by augmenting with airport pseudolites, GPS stands to gain a large improvement in system availability under interference conditions.

Airport pseudolites have been shown to also play a key role in maintaining GPS availability and accuracy during periods of GPS satellite outages. The pseudolite, by acting the role of a GPS satellite, albeit ground based, ameliorates GPS system failures. Pseudolites also enhance the geometry, providing for an overall more accurate system.

7.1.3 Pseudolite Signal Design

Designs for pseudolite signals have been presented based on code division multiple access using concatenation to produce longer codes, which show excellent correlation properties, both over time and frequency.

Realizations of this proposed design have been analyzed and shown to produce up to 6 dB of extra processing gain. By implementing these codes, the effective operating range of a pseudolite will be doubled, allowing it to service larger areas without jamming receivers.

Longer and faster sequences, P-code rate, were also proposed to further reduce cross-correlation interference. This design now forms the basis of the adopted RTCA standard for pseudolite signal design [38].

These proposed signals are, by design, friendly to non-participating receivers, as they rely on methods implicitly built into every GPS receiver to minimize interference.

7.2 Recommendations for Future Work

The following is a list of potential research ideas:

- Full implementation of the suggested signal quality monitoring algorithms would require modification to existing receiver hardware. Specifically software access to AGC

gain values would be needed. Also signal quality monitoring by computation of the proposed test statistics would likely be processor intensive if implemented directly on the receiver CPU. Therefore special digital signal processors may be needed. A project option may be to explore avenues to realize these designs using existing receivers, and/or new components.

- Due to limitations in the capability of hardware used, full calibration bench test results were not performed. A project idea may be to design a well controlled test facility to serve for the verification of this and future integrity algorithms.
- It may be possible to combine all four proposed test statistics into a single one, using some form of a truth table. The performance of this composite test statistic can then be examined and optimized to produce test metric with very high reliability and robustness.

Appendix A

Simulation Results

This section contains more detailed plots showing a cross section of simulation results for a GPS receiver subjected to various types of interference. The table below provides a guide to the nomenclature used to classify results in this section.

Interference Type Group Number	Description
Run 501	AWGN, 0 to 40 dB NSR
Run 502	CW Interference at L1, 0 to 40 dB JSR
Run 503	Pulsed AWGN, 150 dB, 0 to 100 % duty cycle
Run 504	Pulsed CW Interference at L1, 150 dB, 0 to 100 % duty cycle
Run 505	Signal Attenuation, 0 to 24 dB
Run 506	CW Interference at L1 + 1 kHz, 0 to 40 dB JSR
Run 507	CW Interference at L1 + 7 kHz, 0 to 40 dB JSR

The first section of appendix A provides a comprehensive summary of all simulation runs. for all seven interference type groups. Plots in this section are labeled in the format :

Run XYZ.all.a through f,

where XXX = interference group number = 501, 502, ...507

Within each group, plots 'a' through 'f' show COP, COP- σ , carrier phase jitter, AGC gain, pseudorange error and frequency error respectively, across the entire range of applied levels of the specified type of interference associated with the run.

The second section of appendix A provides detailed results for specific runs within each interference type group. Each page in this section shows the time history for a single run at

a single power level (or duty cycle) of a specified interference type. This permits the reader to see not only the behavior of the receiver over time as it is subjected to step loading of interference, but also to observe the degradation in performance as the level of interference is increase. To minimize bulk, only a few of these plots are included, taken at close to regular intervals of interference loading (e.g. plots are shown for AWGN interference power levels of 0 dB, 10dB, 20dB, etc). Results just prior to, and after, loss of lock are usually included.

The detailed plots are labeled as follows:

RUN XXX.YY.a through f.

where XXX = interference group number = 501, 502, ...507

YY = Interference power level index = 1, 11, 21, ... (= 0dB, 10 dB, 20dB...)

Plots a through f show time histories of COP, carrier phase, pseudorange error, carrier phase jitter, AGC, and frequency error respectively.

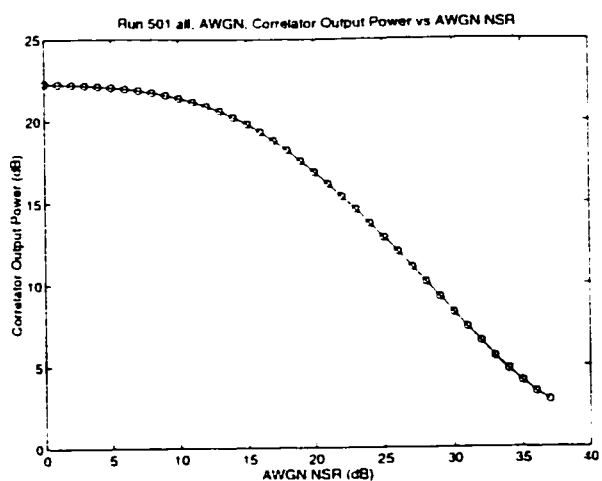


Figure 501.all.a: Correlator Output Power vs AWGN

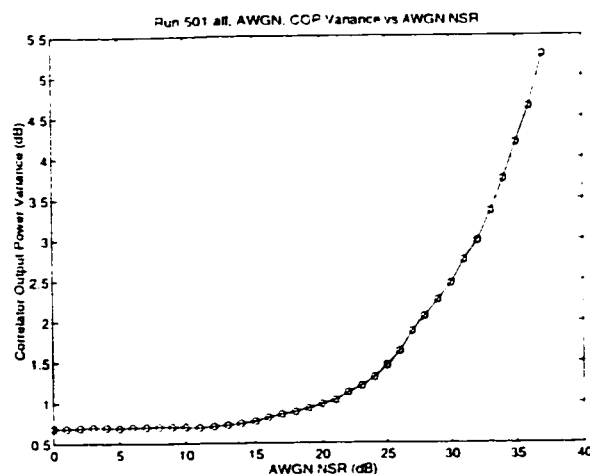


Figure 501.all.b: Correlator Power Output Variance vs AWGN

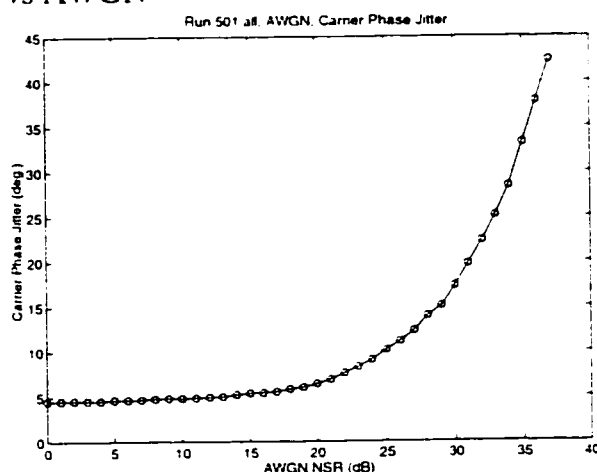


Figure 501.all.c: Carrier Phase Jitter vs AWGN

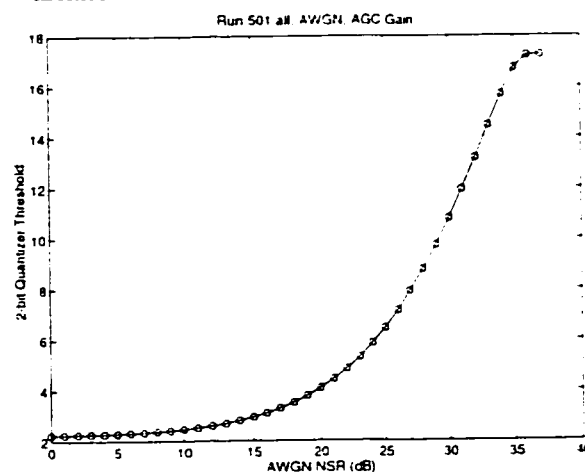


Figure 501.all.d: AGC Gain vs AWGN

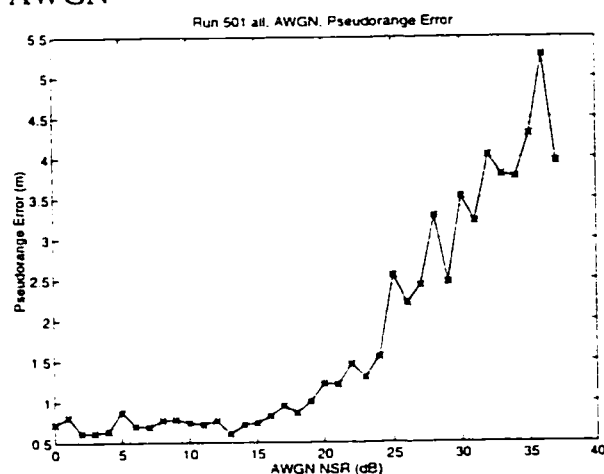


Figure 501.all.e: Pseudorange Error vs AWGN

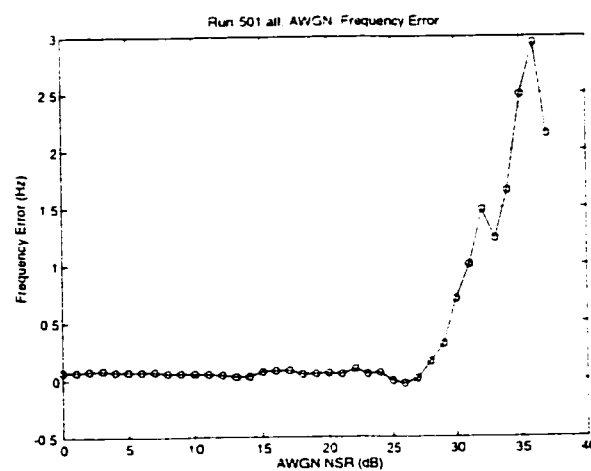


Figure 501.all.f: Frequency Error vs AWGN

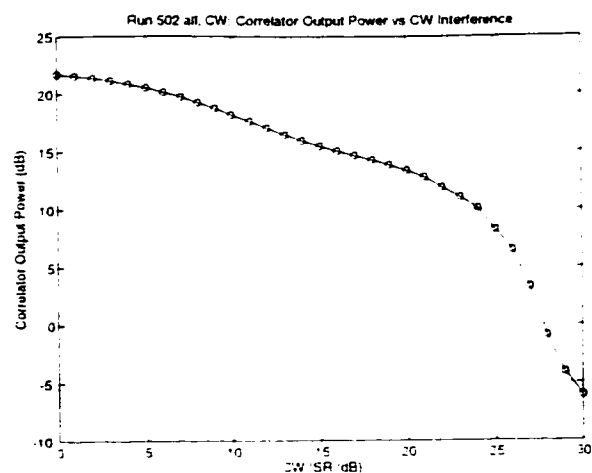


Figure 502.all.a: Correlator Output Power vs CW

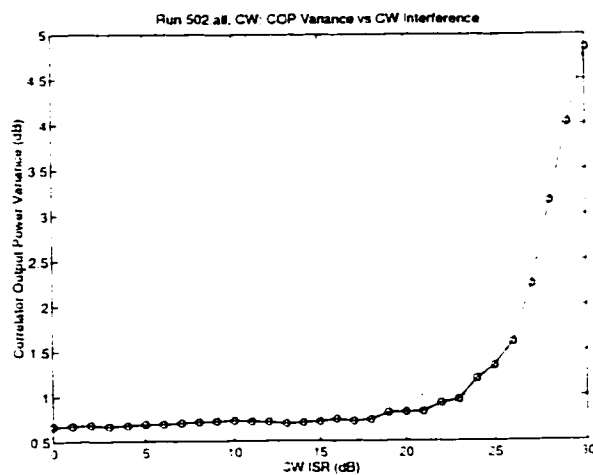


Figure 502.all.b: Correlator Power Output Variance vs CW

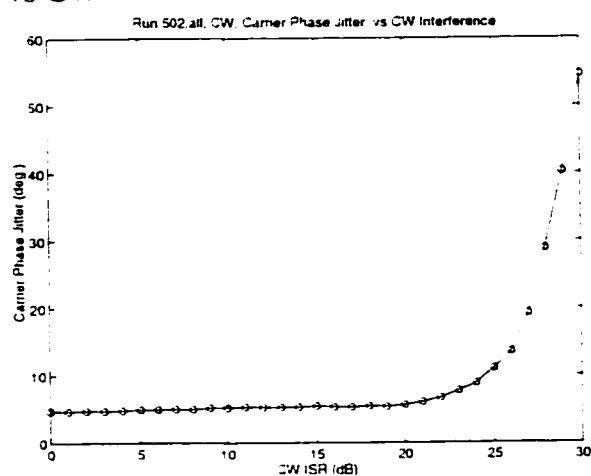


Figure 502.all.c: Carrier Phase Jitter vs CW

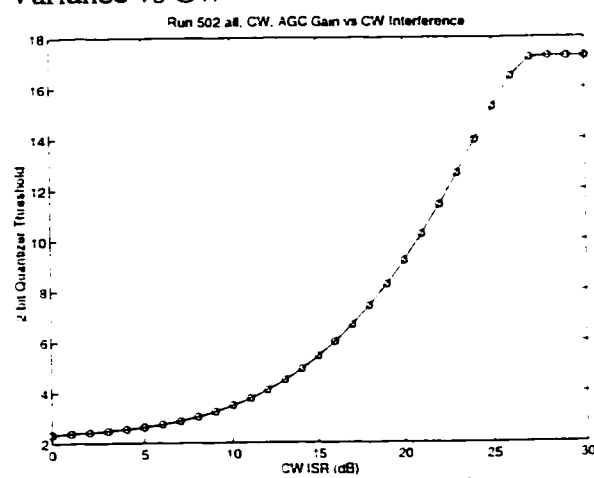


Figure 502.all.d: AGC Gain vs CW

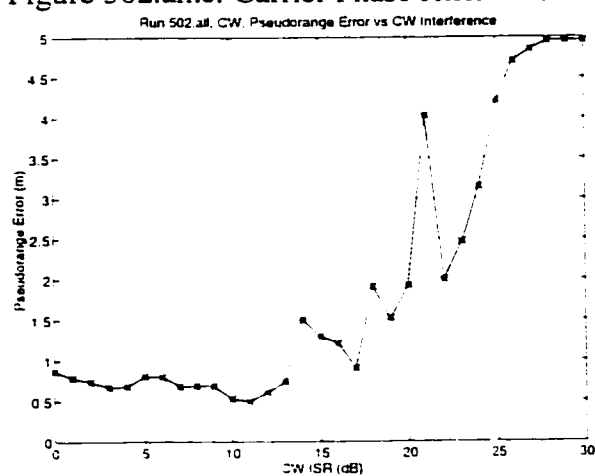


Figure 502.all.e: Pseudorange Error vs CW

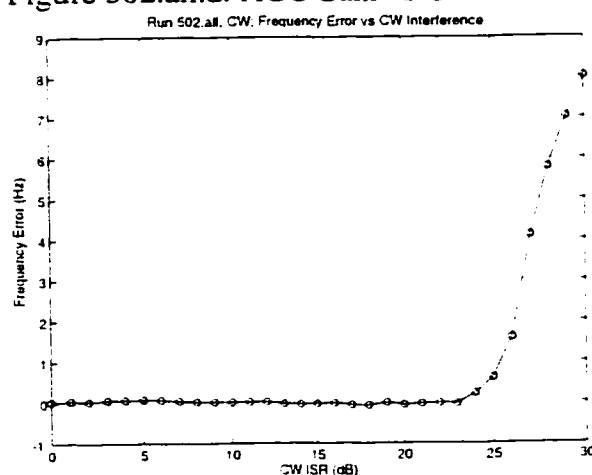


Figure 502.all.f: Frequency Error vs CW

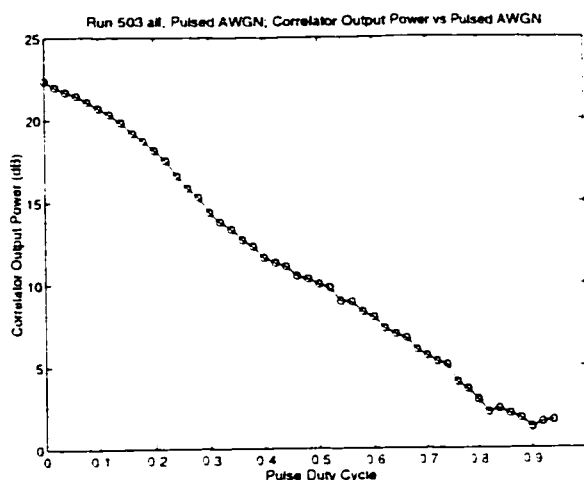


Figure 503.all.a: Correlator Output Power vs Pulsed AWGN

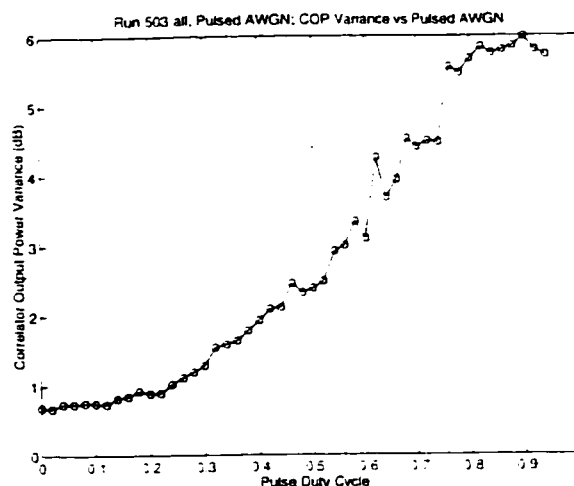


Figure 503.all.b: Correlator Power Output Variance vs Pulsed AWGN

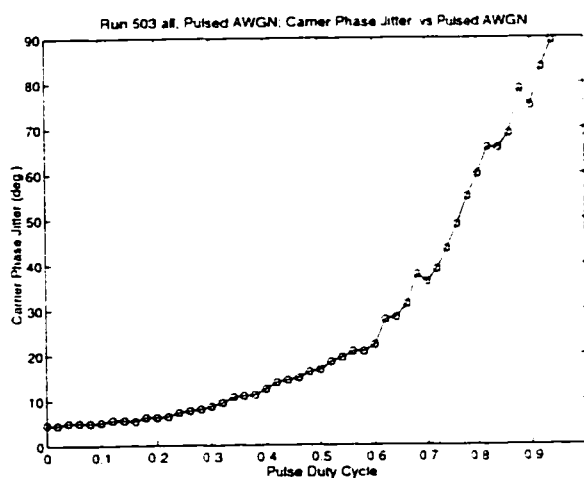


Figure 503.all.c: Carrier Phase Jitter vs Pulsed AWGN

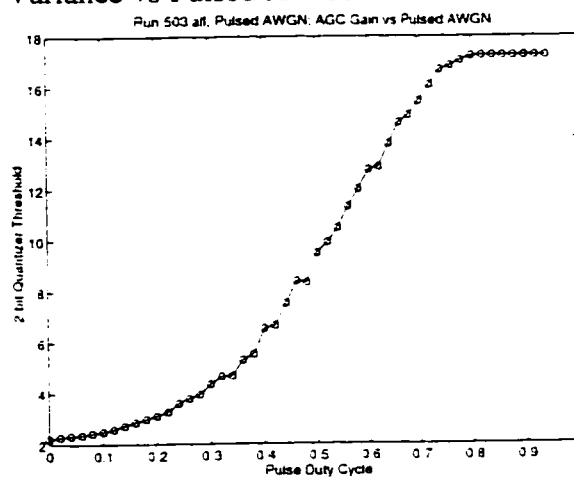


Figure 503.all.d: AGC Gain vs Pulsed AWGN

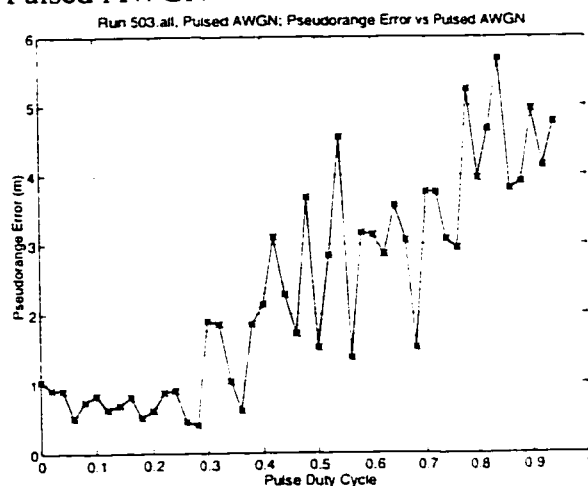


Figure 503.all.e: Pseudorange Error vs Pulsed AWGN

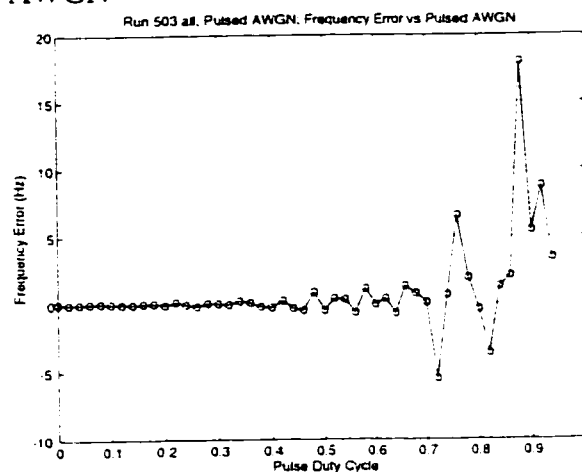


Figure 503.all.f: Frequency Error vs Pulsed AWGN

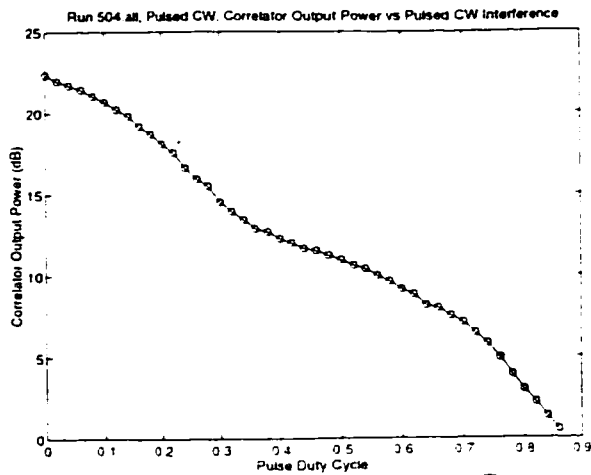


Figure 504.all.a: Correlator Output Power vs Pulsed CW

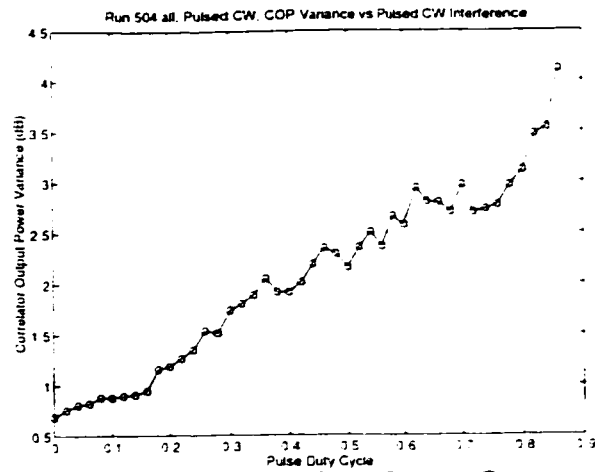


Figure 504.all.b: Correlator Power Output Variance vs Pulsed CW

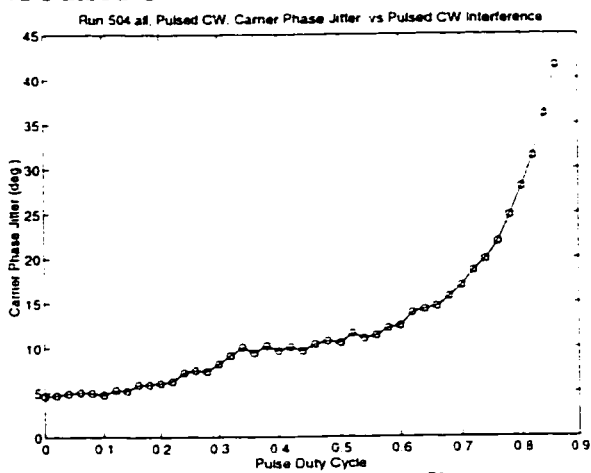


Figure 504.all.c: Carrier Phase Jitter vs Pulsed CW

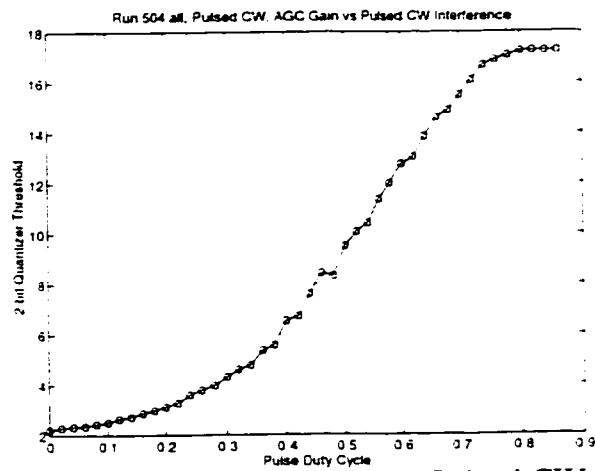


Figure 504.all.d: AGC Gain vs Pulsed CW

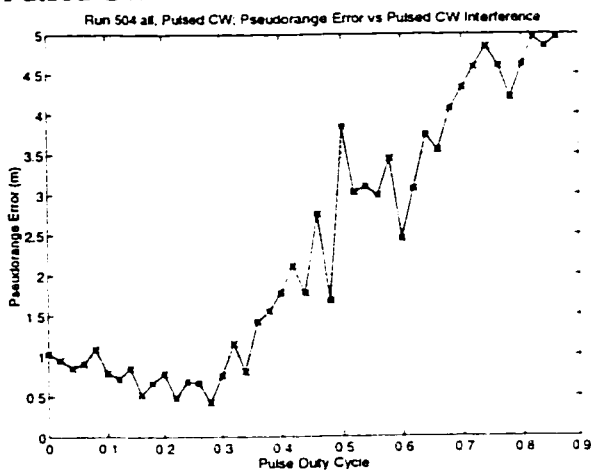


Figure 504.all.e: Pseudorange Error vs Pulsed CW

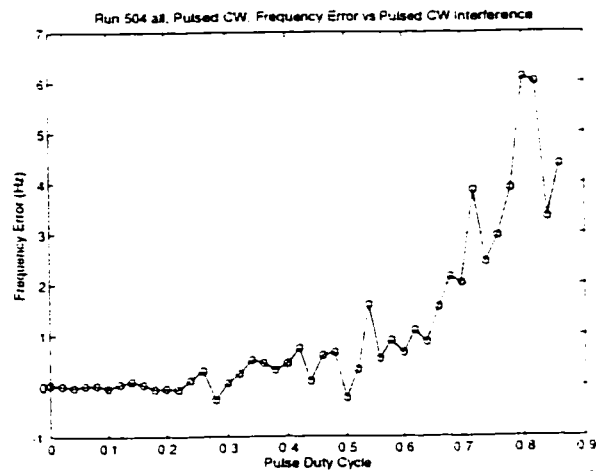


Figure 504.all.f: Frequency Error vs Pulsed CW

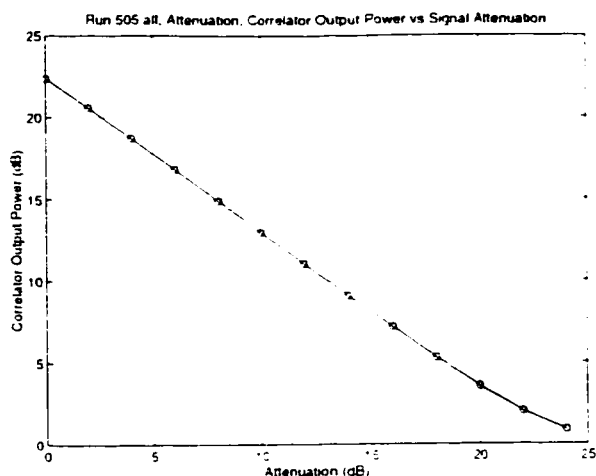


Figure 505.all.a: Correlator Output Power vs Signal Attenuation

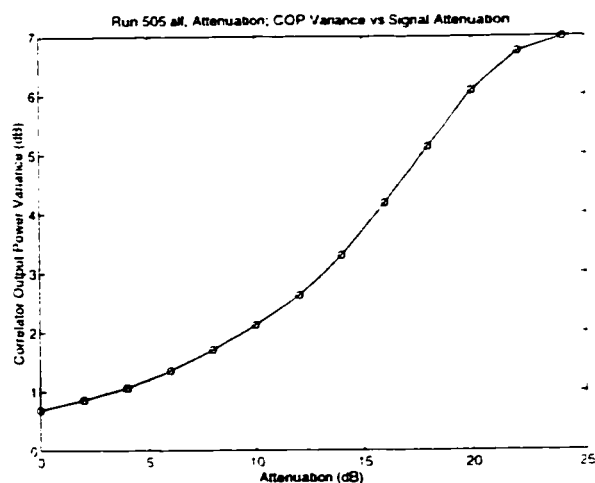


Figure 505.all.b: Correlator Power Output Variance vs Signal Attenuation

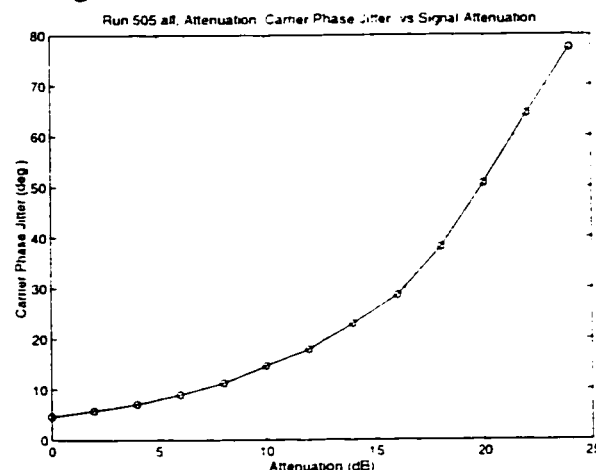


Figure 505.all.c: Carrier Phase Jitter vs Signal Attenuation

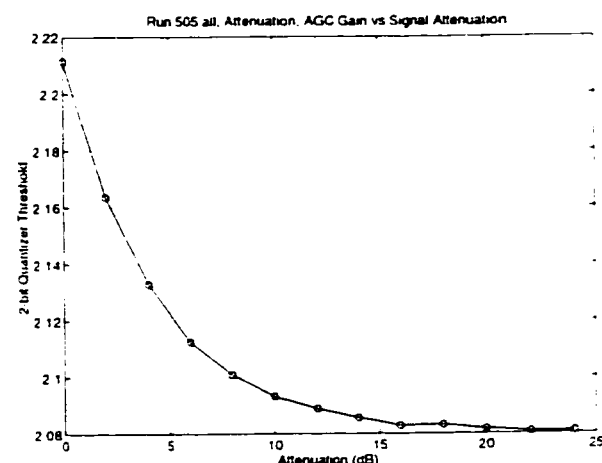


Figure 505.all.d: AGC Gain vs Signal Attenuation

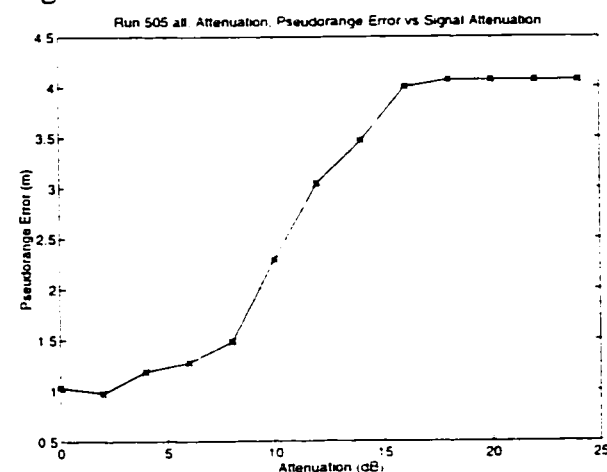


Figure 505.all.e: Pseudorange Error vs Signal Attenuation

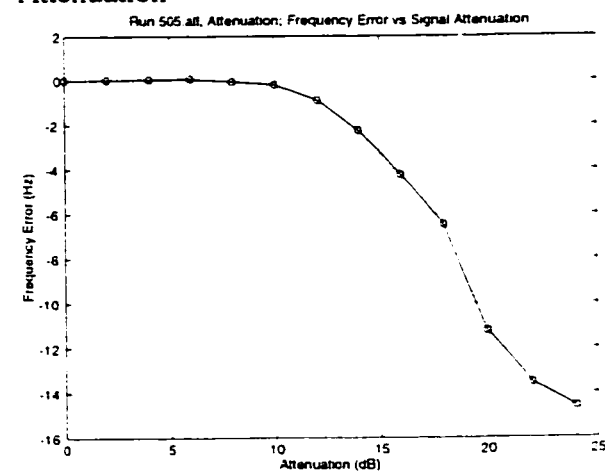


Figure 505.all.f: Frequency Error vs Signal Attenuation

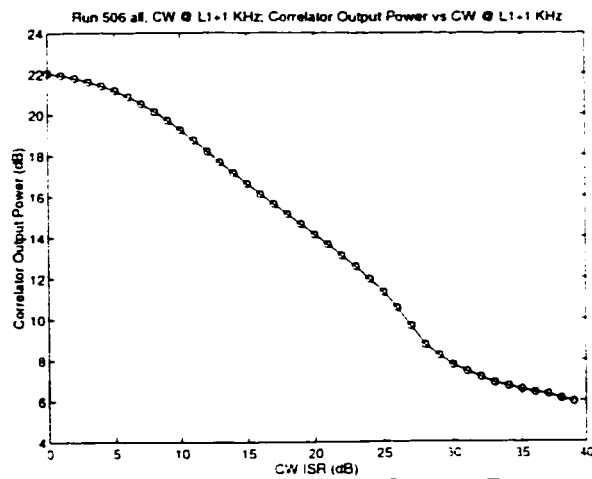


Figure 506.all.a: Correlator Output Power vs CW @ L1+1 KHz

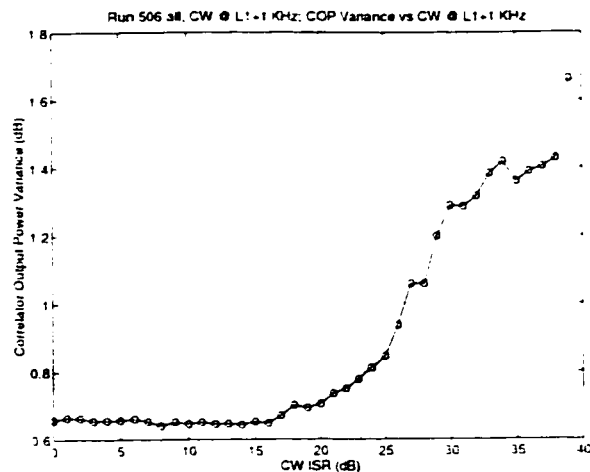


Figure 506.all.b: Correlator Power Output Variance vs CW @ L1+1 KHz

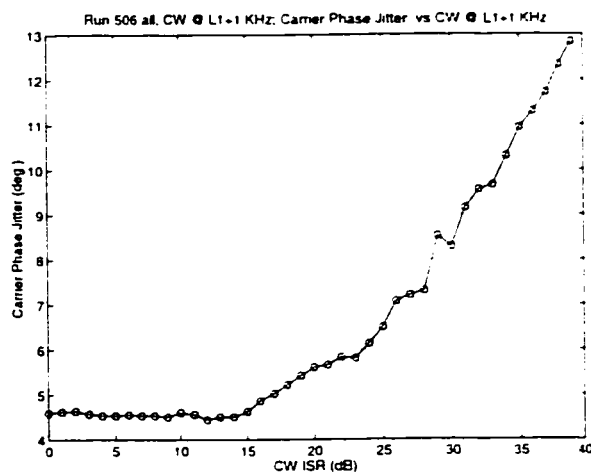


Figure 506.all.c: Carrier Phase Jitter vs CW @ L1+1 KHz

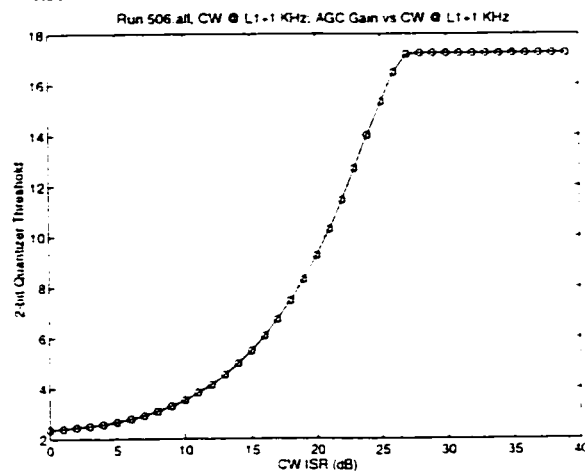


Figure 506.all.d: AGC Gain vs CW @ L1+1 KHz

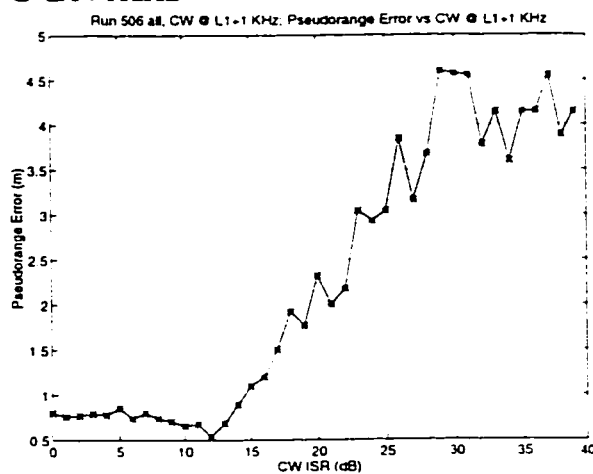


Figure 506.all.e: Pseudorange Error vs CW @ L1+1 KHz

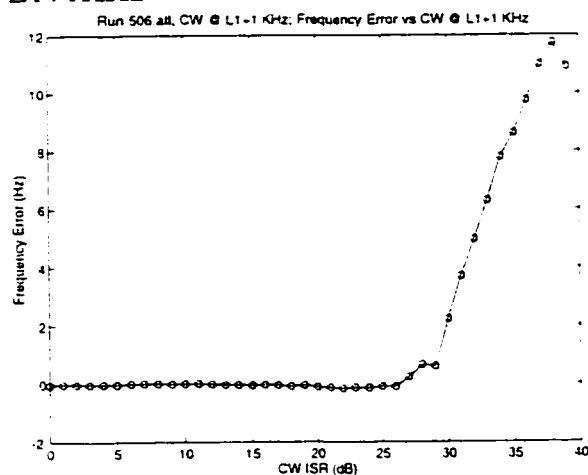


Figure 506.all.f: Frequency Error vs CW @ L1+1 KHz

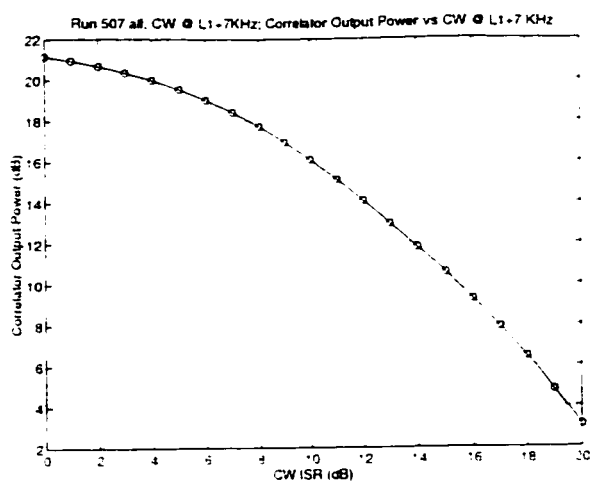


Figure 507.all.a: Correlator Output Power vs CW @ L1+7KHz

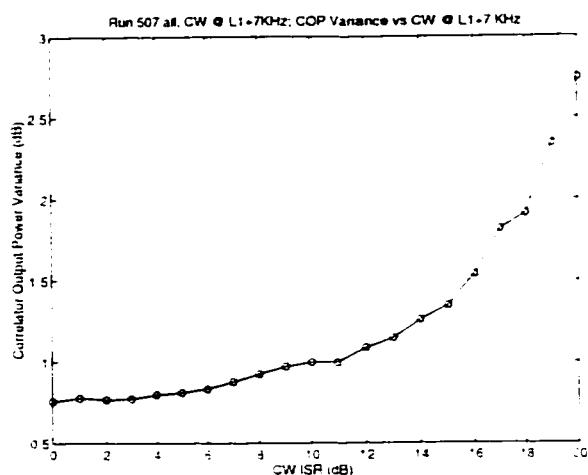


Figure 507.all.b: Correlator Power Output Variance vs CW @ L1+7KHz

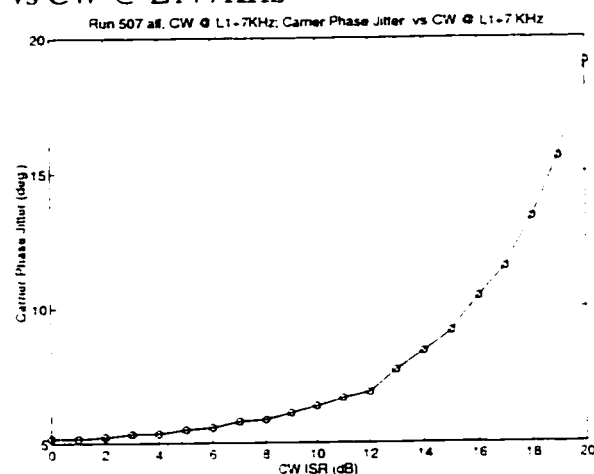


Figure 507.all.c: Carrier Phase Jitter vs CW @ L1+7KHz

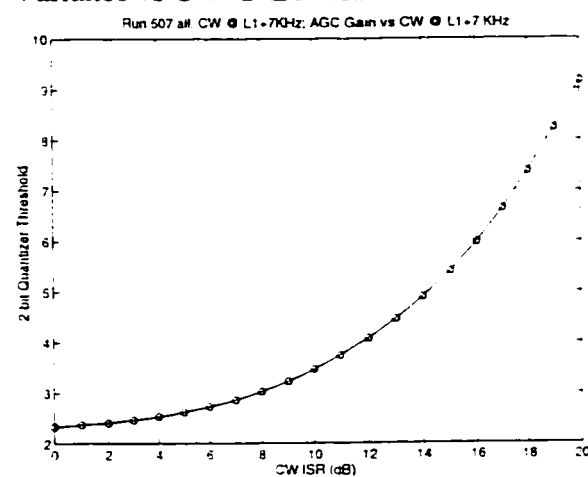


Figure 507.all.d: AGC Gain vs CW @ L1+7KHz

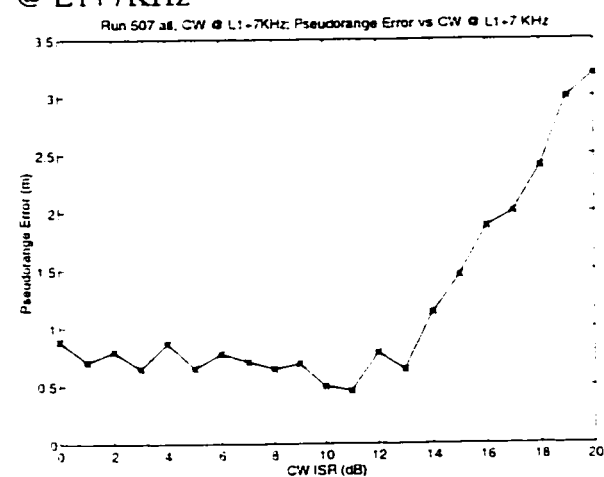


Figure 507.all.e: Pseudorange Error vs CW @ L1+7KHz

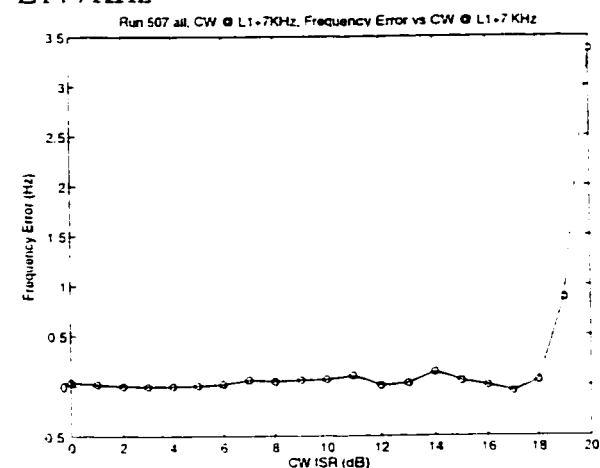


Figure 507.all.f: Frequency Error vs CW @ L1+7KHz

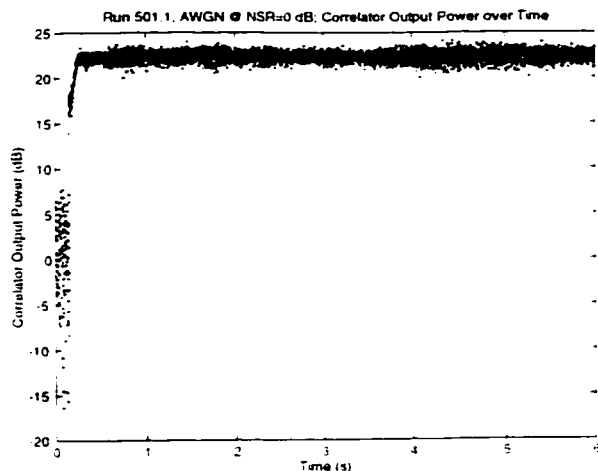


Figure 501.1.a: Correlator Output Power, 0dB AWGN

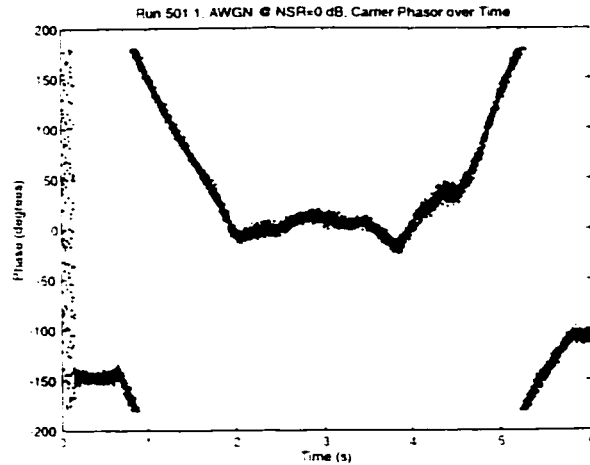


Figure 501.1.b: Carrier Phase, 0dB AWGN

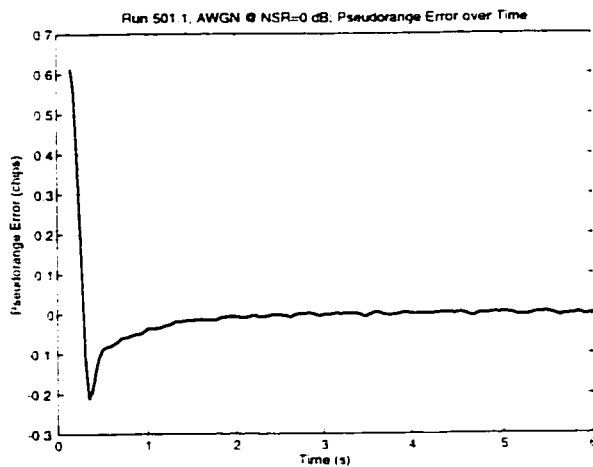


Figure 501.1.c: Pseudorange Error, 0dB AWGN

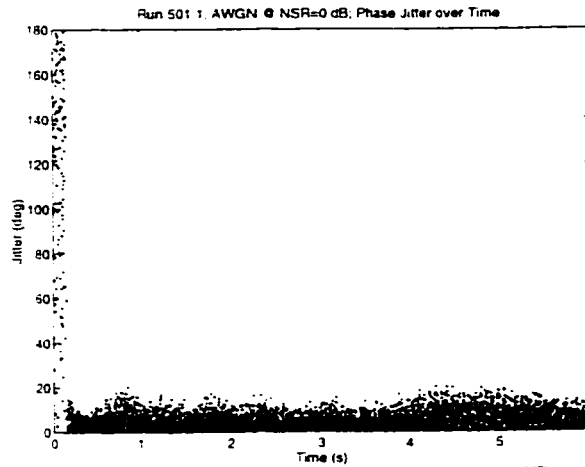


Figure 501.1.d: Carrier Phase Jitter, 0dB AWGN

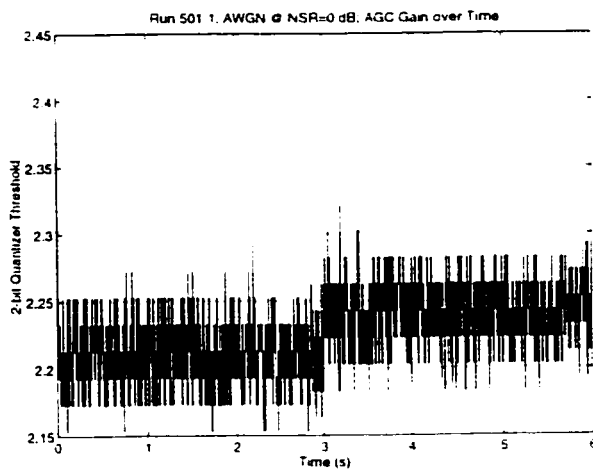


Figure 501.1.e: AGC, 0dB AWGN

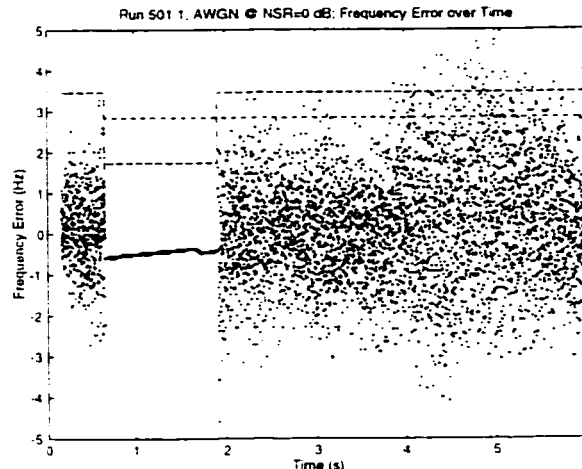


Figure 501.1.f: Frequency Error, 0dB AWGN

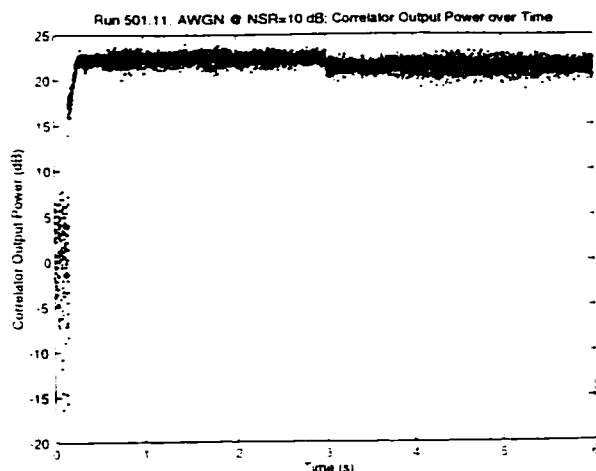


Figure 501.11.a: Correlator Output Power. 10dB AWGN

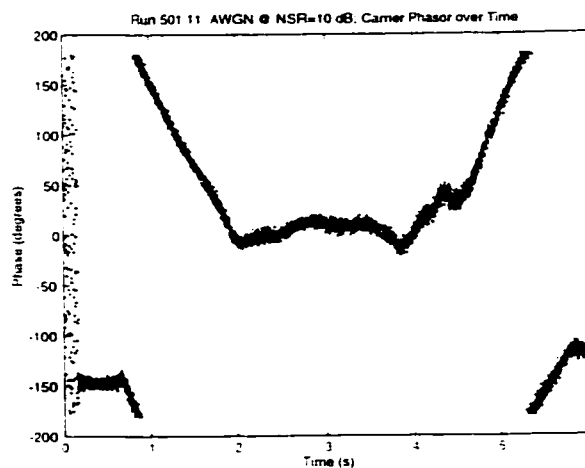


Figure 501.11.b: Carrier Phase. 10dB AWGN

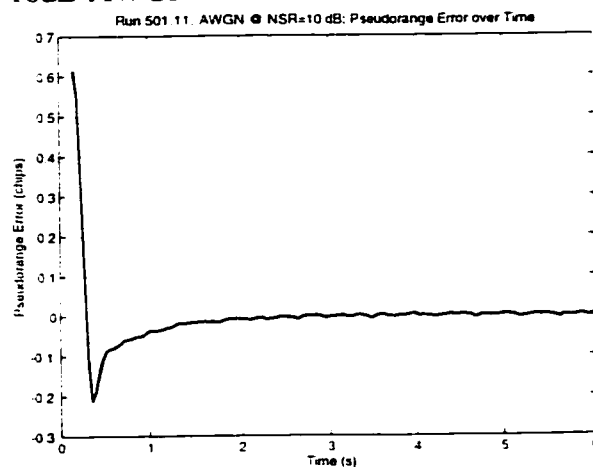


Figure 501.11.c: Pseudorange Error. 10dB AWGN

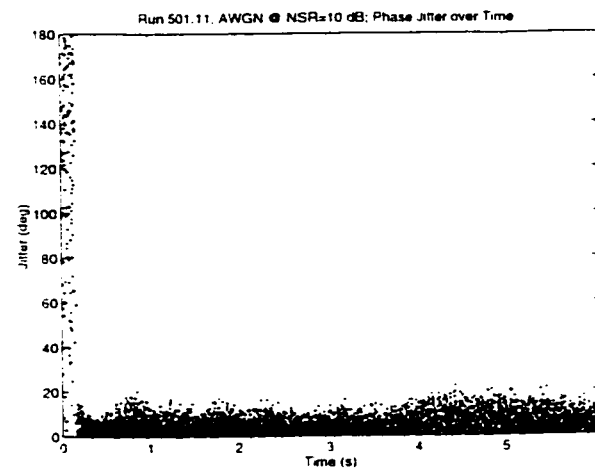


Figure 501.11.d: Carrier Phase Jitter. 10dB AWGN

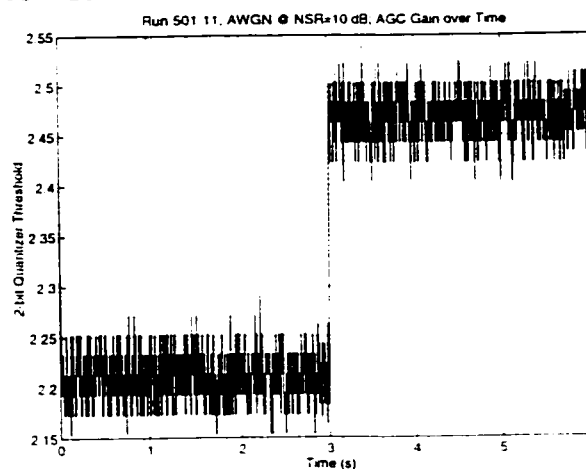


Figure 501.11.e: AGC. 10dB AWGN

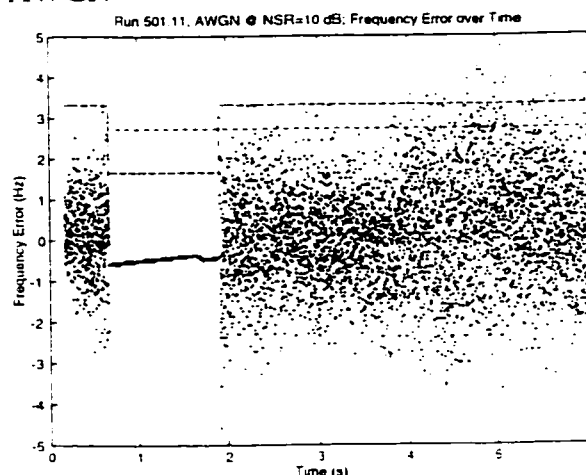


Figure 501.11.f: Frequency Error. 10dB AWGN

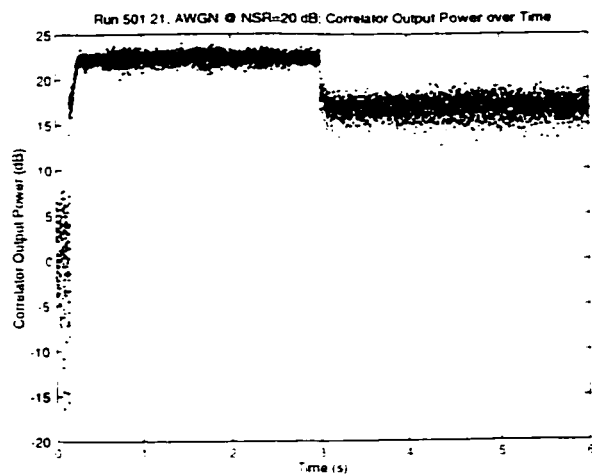


Figure 501.21.a: Correlator Output Power, 20dB AWGN

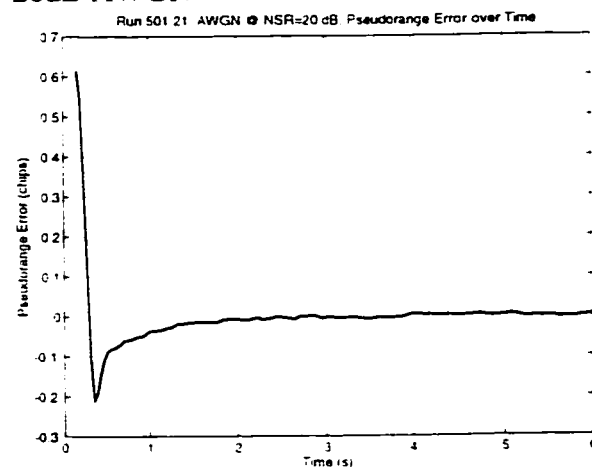


Figure 501.21.c: Pseudorange Error, 20dB AWGN

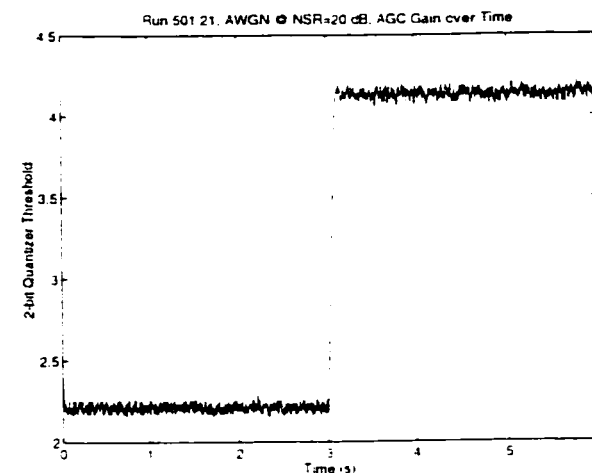


Figure 501.21.e: AGC, 20dB AWGN

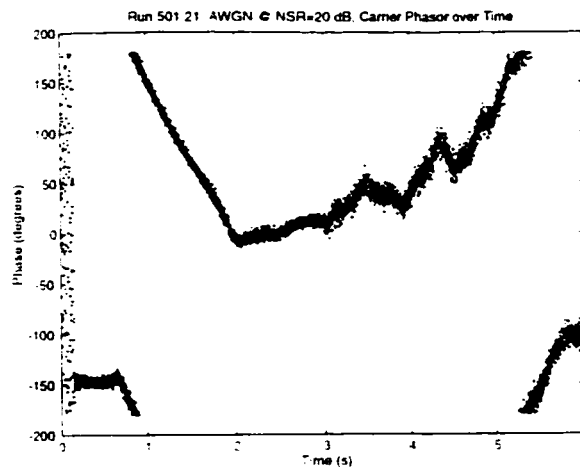


Figure 501.21.b: Carrier Phase, 20dB AWGN

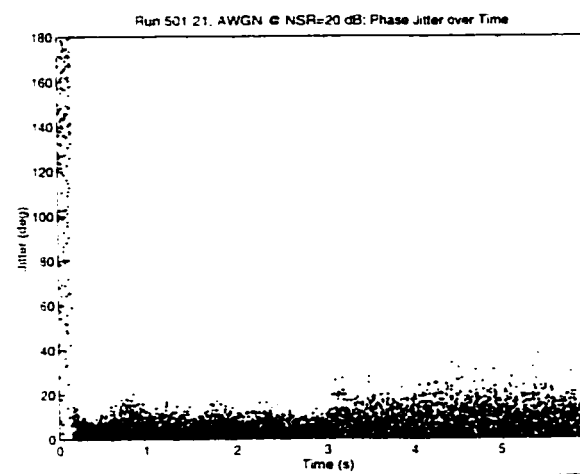


Figure 501.21.d: Carrier Phase Jitter, 20dB AWGN

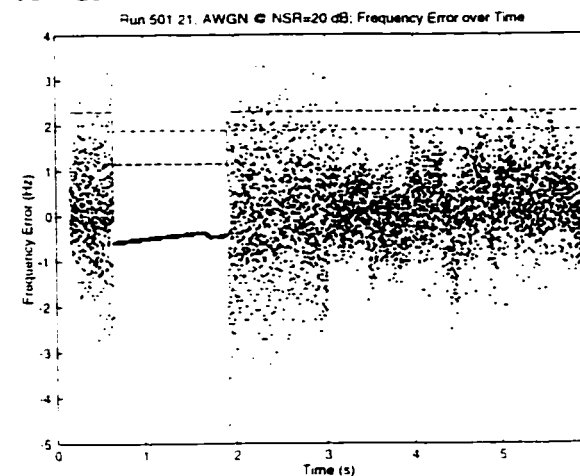


Figure 501.21.f: Frequency Error, 20dB AWGN

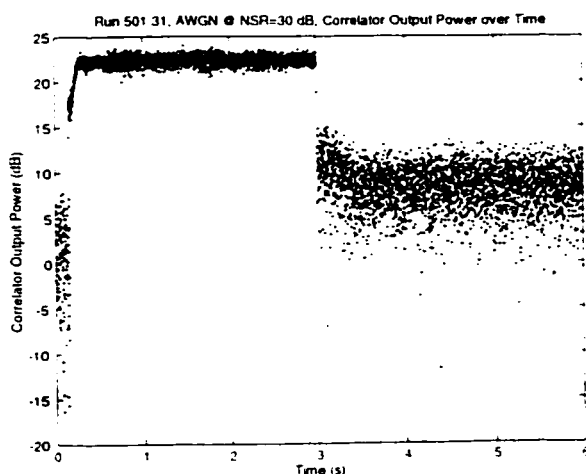


Figure 501.31.a: Correlator Output Power, 30dB AWGN

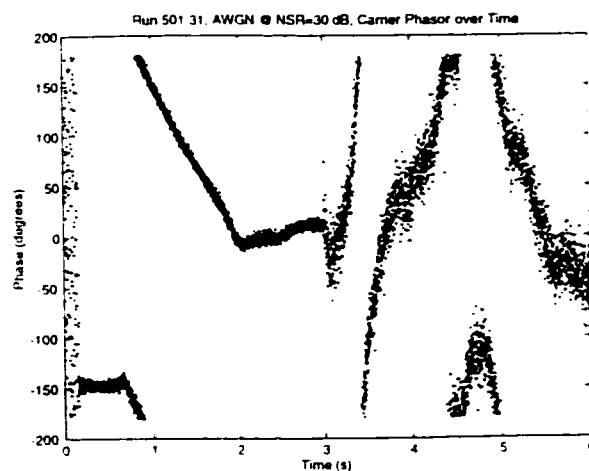


Figure 501.31.b: Carrier Phase, 30dB AWGN

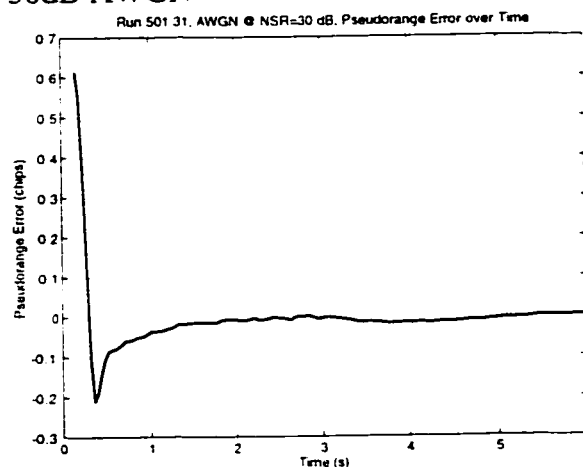


Figure 501.31.c: Pseudorange Error, 30dB AWGN

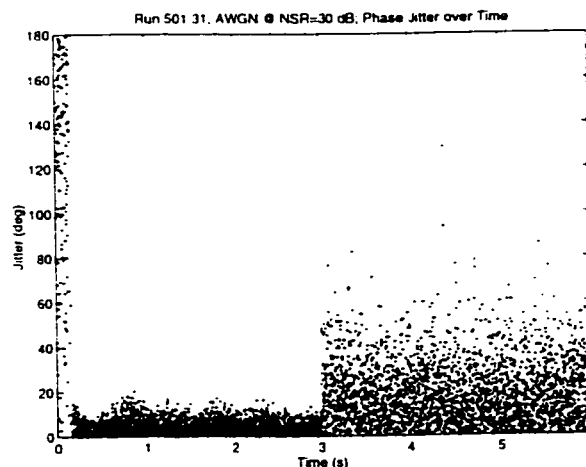


Figure 501.31.d: Carrier Phase Jitter, 30dB AWGN

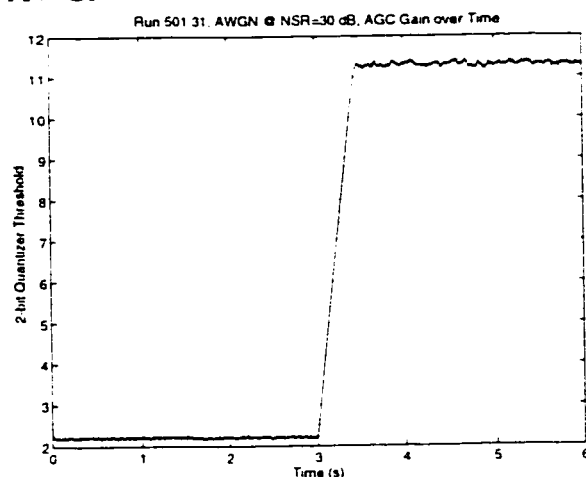


Figure 501.31.e: AGC, 30dB AWGN

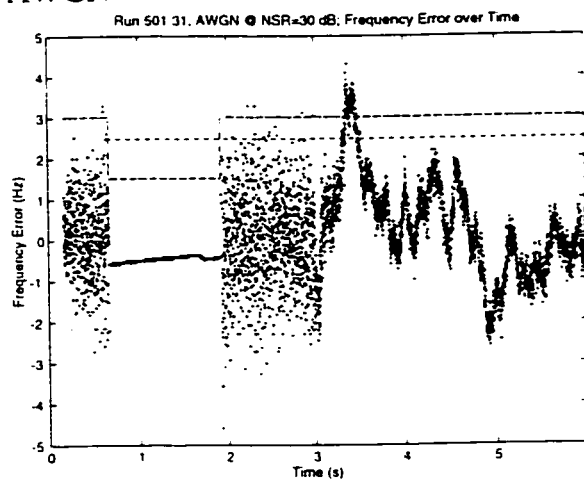


Figure 501.31.f: Frequency Error, 30dB AWGN

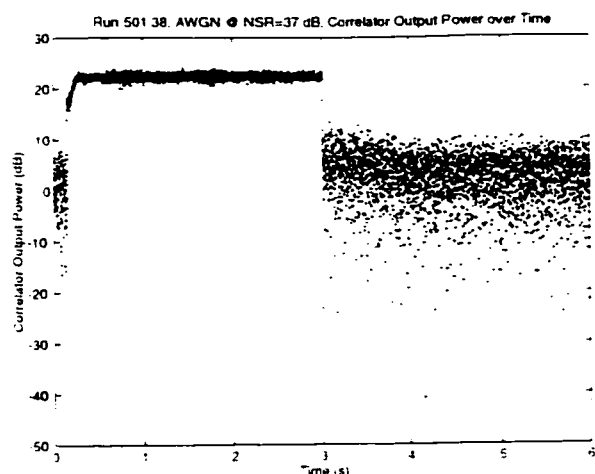


Figure 501.38.a: Correlator Output Power, 37dB AWGN

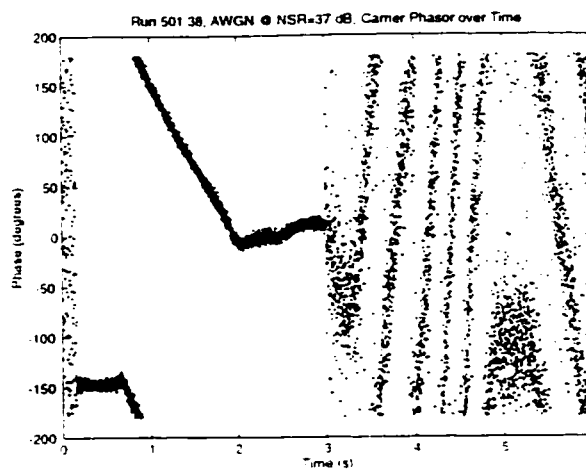


Figure 501.38.b: Carrier Phase, 37dB AWGN

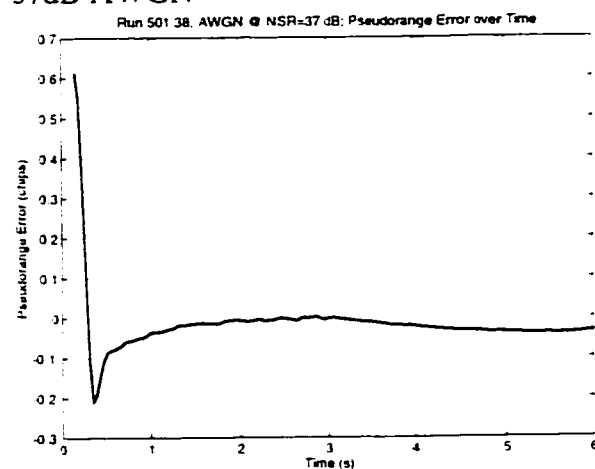


Figure 501.38.c: Pseudorange Error, 37dB AWGN

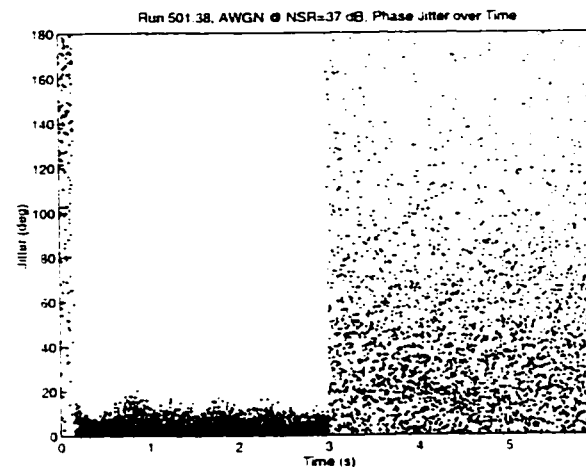


Figure 501.38.d: Carrier Phase Jitter, 37dB AWGN

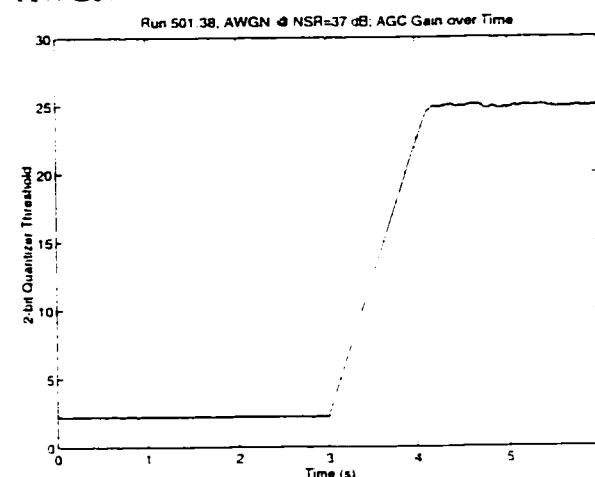


Figure 501.38.e: AGC, 37dB AWGN

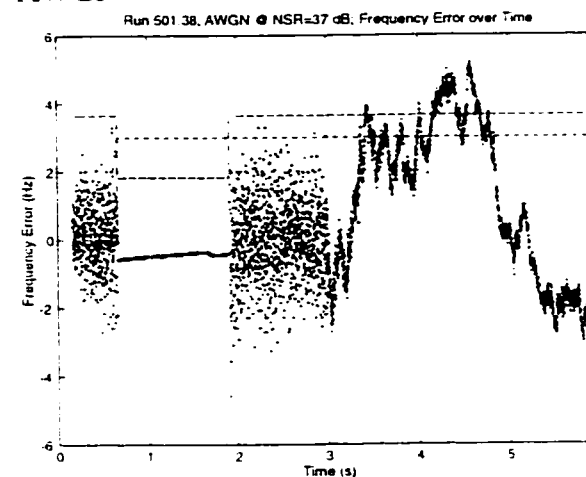


Figure 501.38.f: Frequency Error, 37dB AWGN

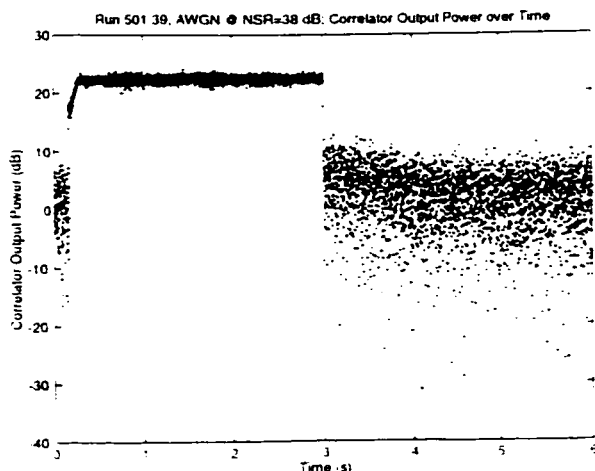


Figure 501.39.a: Correlator Output Power, 38dB AWGN

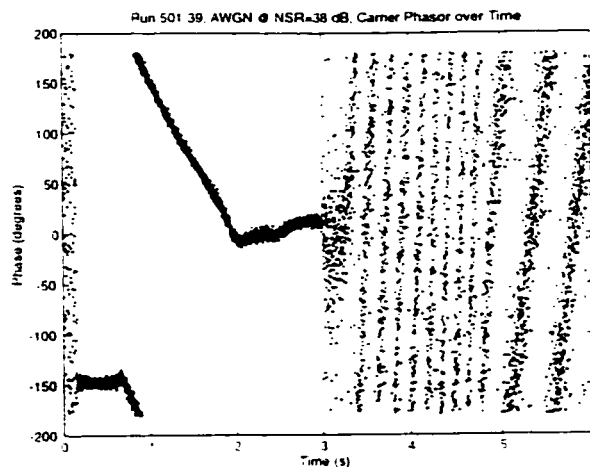


Figure 501.39.b: Carrier Phase, 38dB AWGN

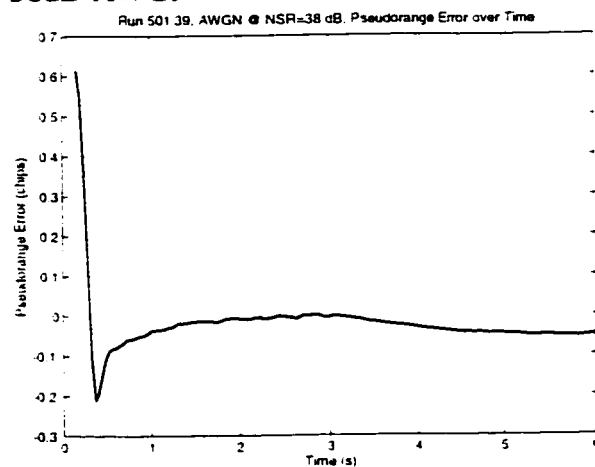


Figure 501.39.c: Pseudorange Error, 38dB AWGN

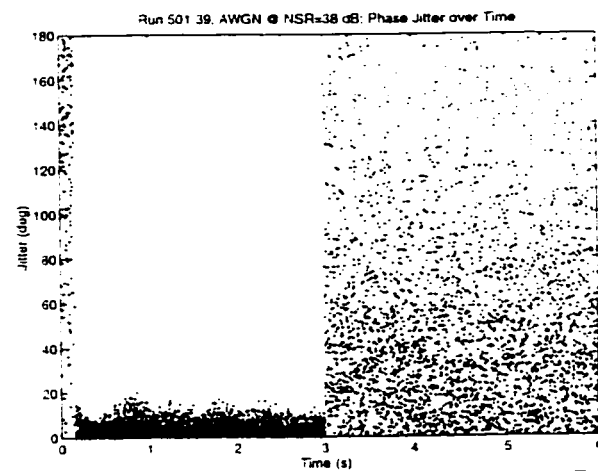


Figure 501.39.d: Carrier Phase Jitter, 38dB AWGN

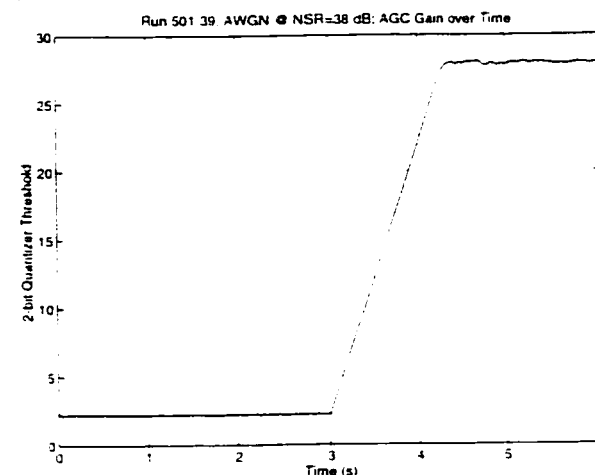


Figure 501.39.e: AGC, 38dB AWGN

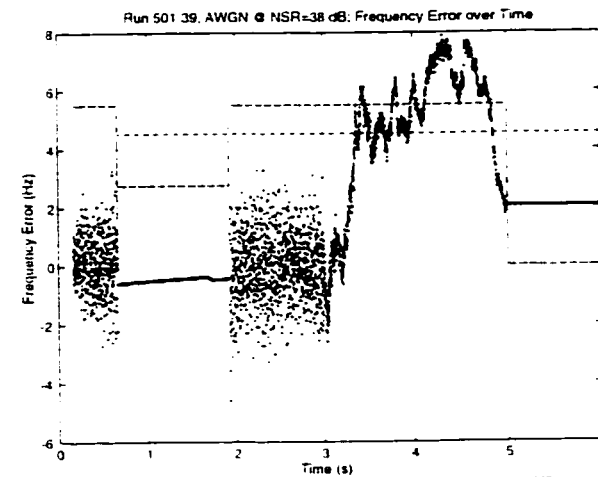


Figure 501.39.f: Frequency Error, 38dB AWGN

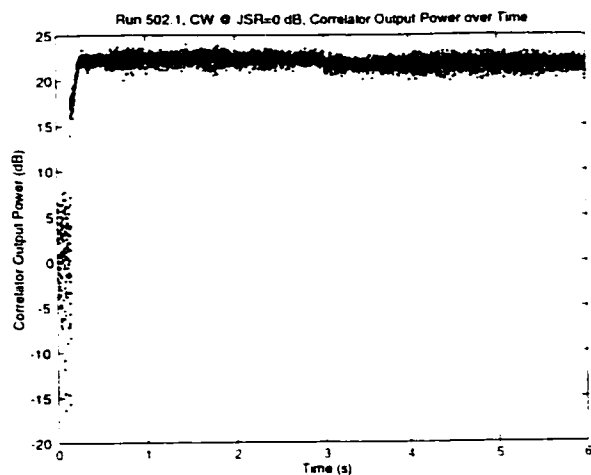


Figure 502.1.a: Correlator Output Power, 0dB CW Interference

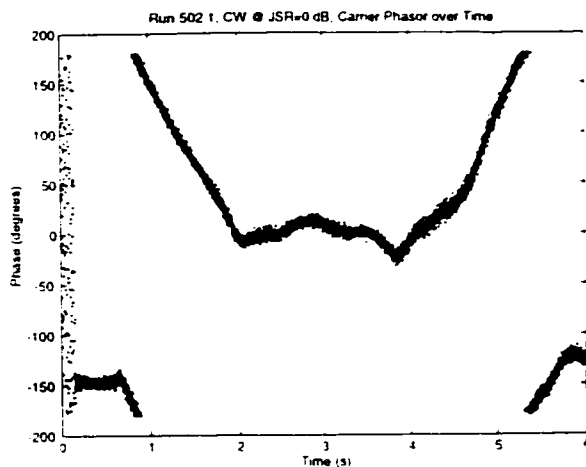


Figure 502.1.b: Carrier Phase, 0dB CW Interference

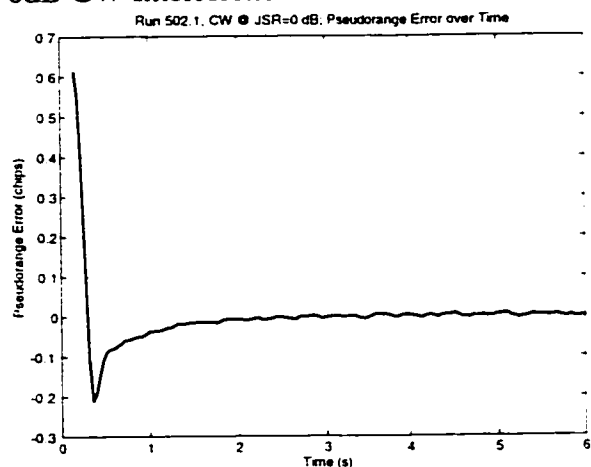


Figure 502.1.c: Pseudorange Error, 0dB CW Interference

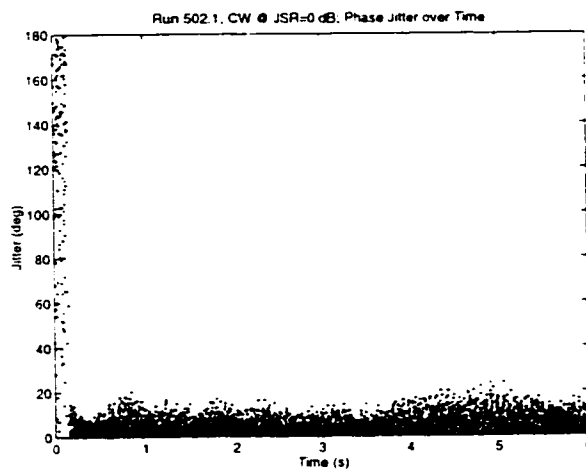


Figure 502.1.d: Carrier Phase Jitter, 0dB CW Interference

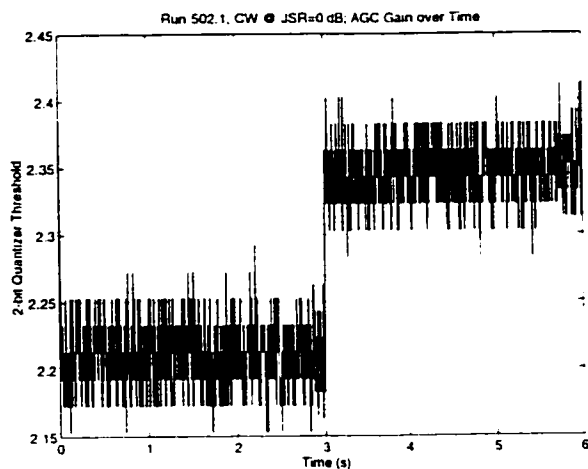


Figure 502.1.e: AGC, 0dB CW Interference

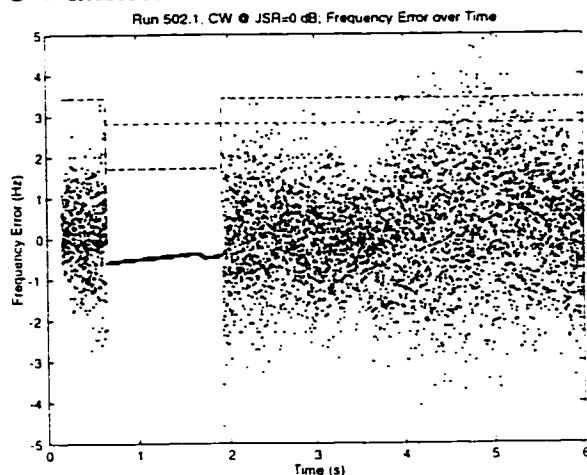


Figure 502.1.f: Frequency Error, 0dB CW Interference

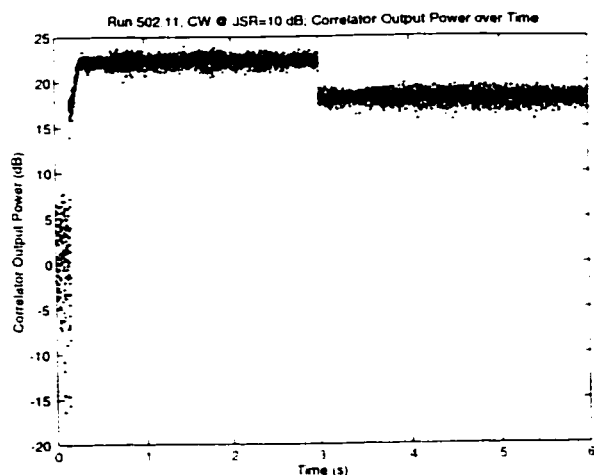


Figure 502.11.a: Correlator Output Power, 10dB CW Interference

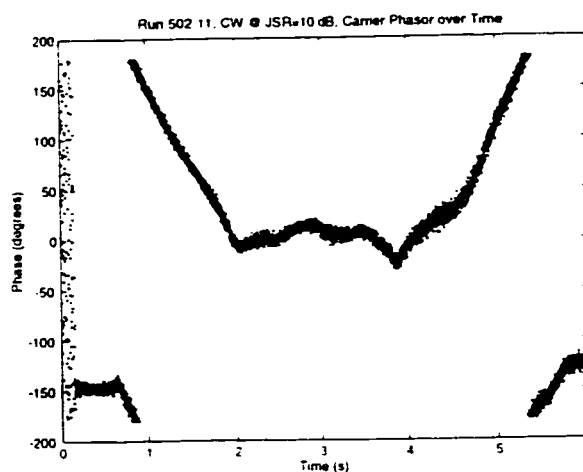


Figure 502.11.b: Carrier Phase, 10dB CW Interference

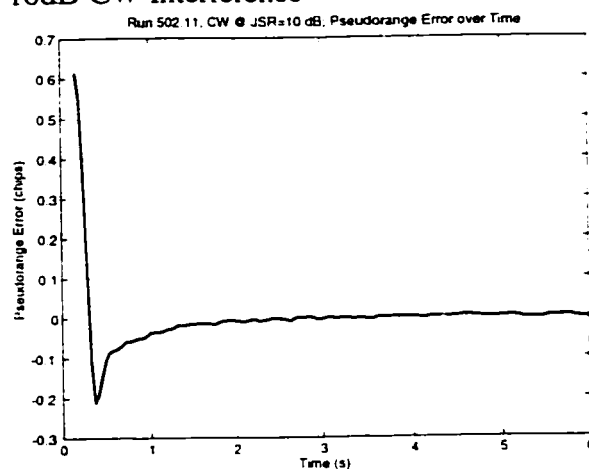


Figure 502.11.c: Pseudorange Error, 10dB CW Interference

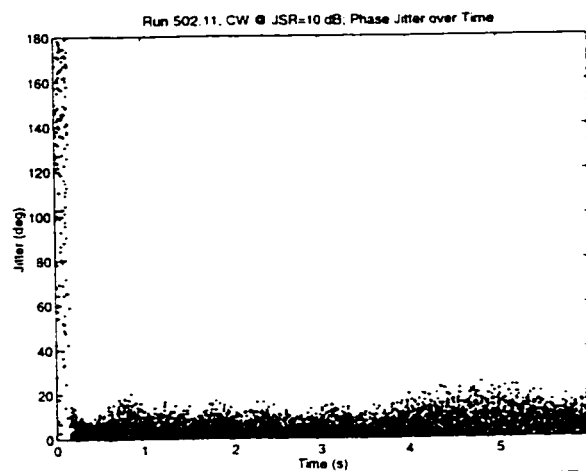


Figure 502.11.d: Carrier Phase Jitter, 10dB CW Interference

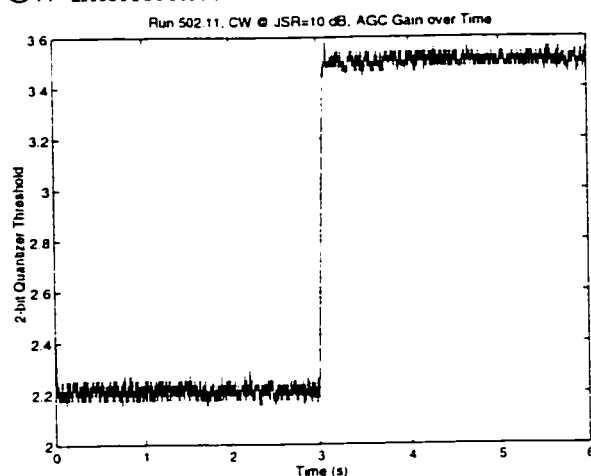


Figure 502.11.e: AGC, 10dB CW Interference

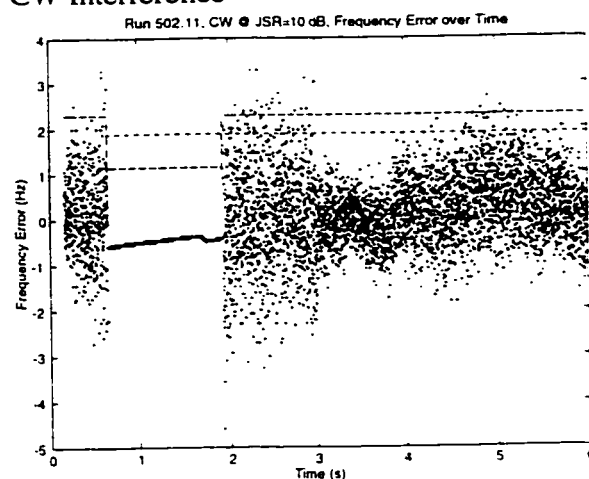


Figure 502.11.f: Frequency Error, 10dB CW Interference

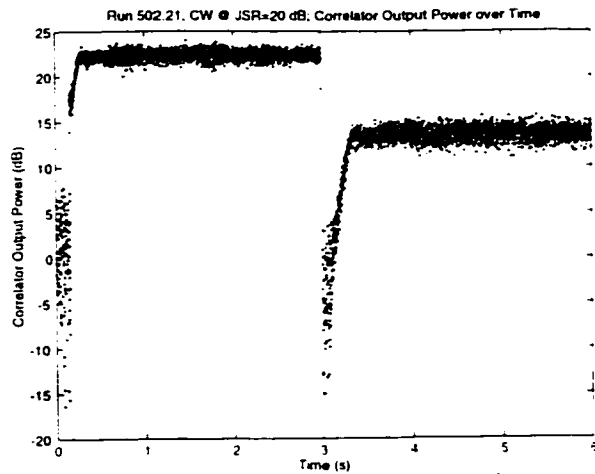


Figure 502.21.a: Correlator Output Power, 20dB CW Interference

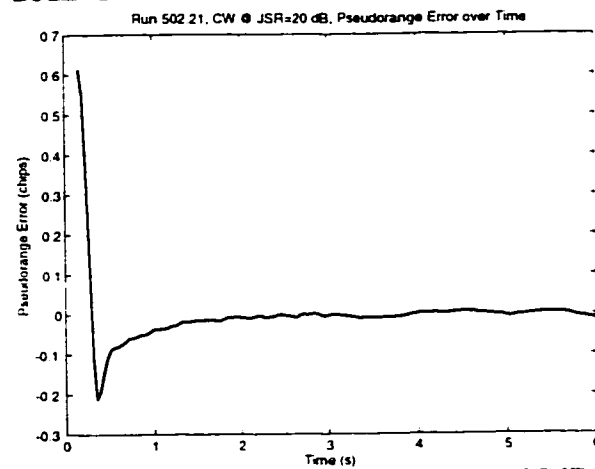


Figure 502.21.c: Pseudorange Error, 20dB CW Interference

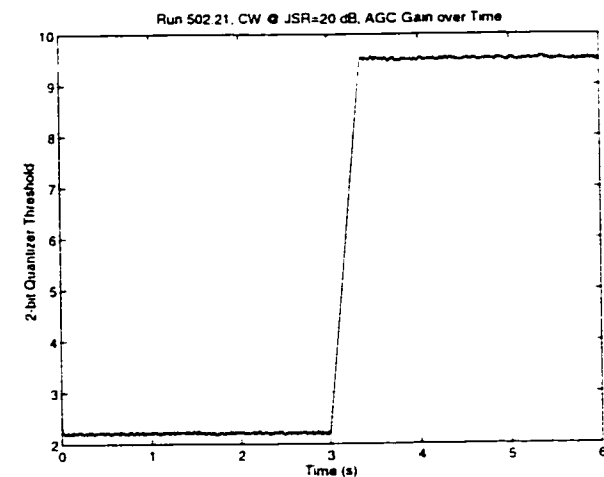


Figure 502.21.e: AGC, 20dB CW Interference

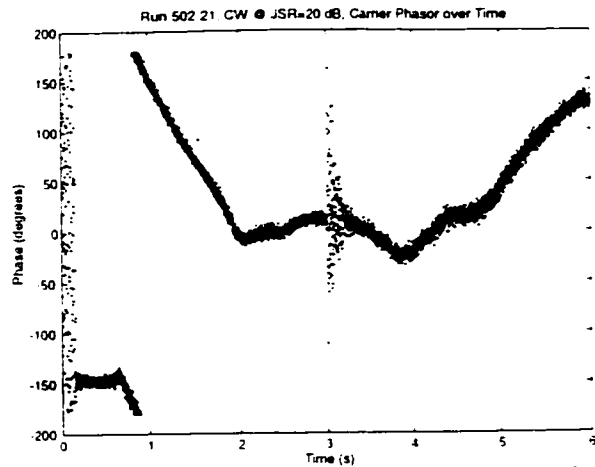


Figure 502.21.b: Carrier Phase, 20dB CW Interference

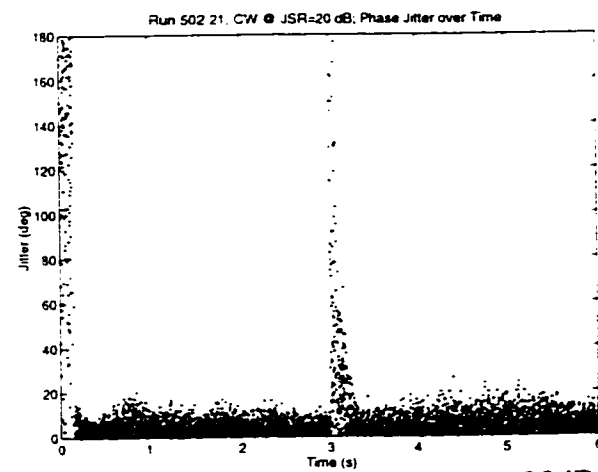


Figure 502.21.d: Carrier Phase Jitter, 20dB CW Interference

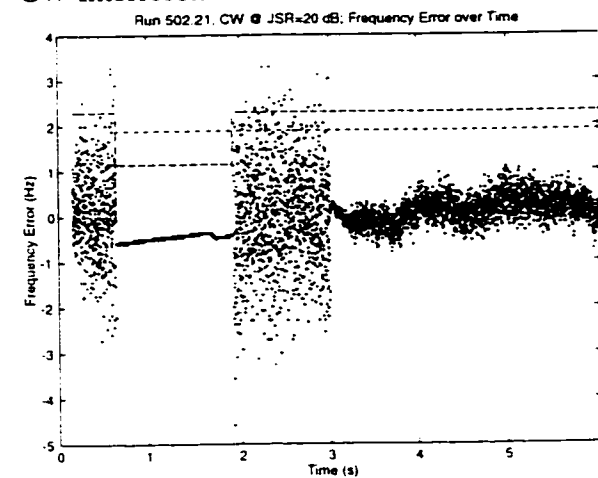


Figure 502.21.f: Frequency Error, 20dB CW Interference

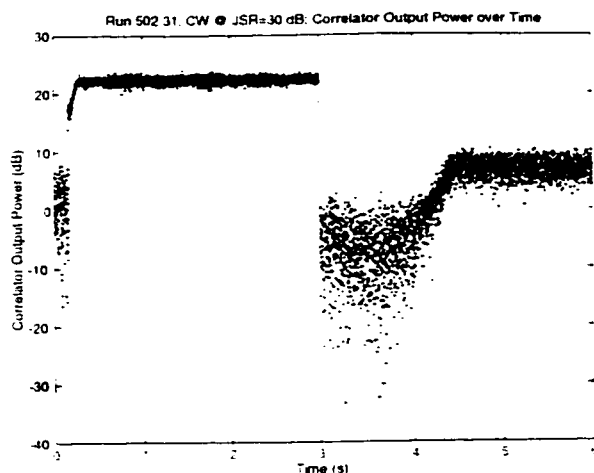


Figure 502.31.a: Correlator Output Power, 30dB CW Interference

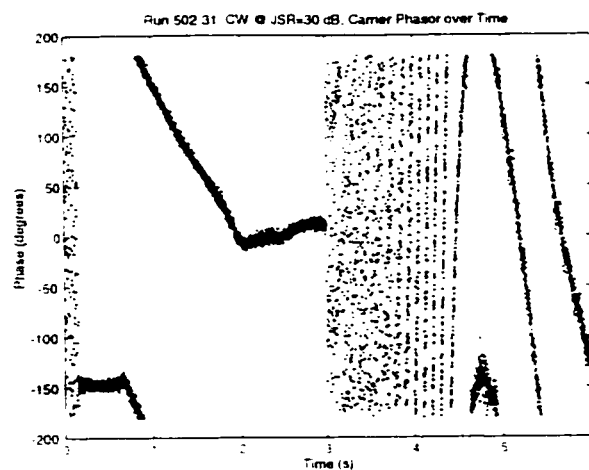


Figure 502.31.b: Carrier Phase, 30dB CW Interference

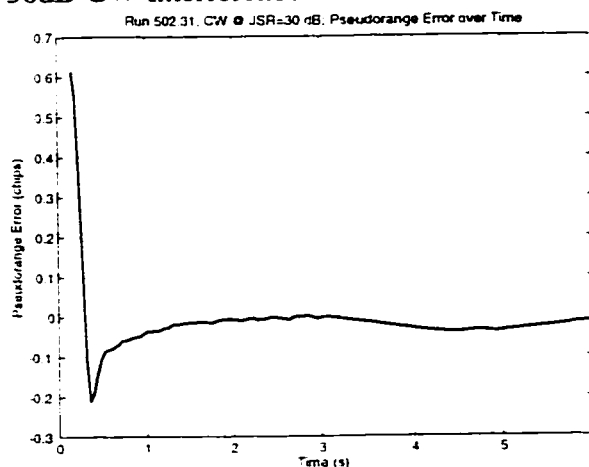


Figure 502.31.c: Pseudorange Error, 30dB CW Interference

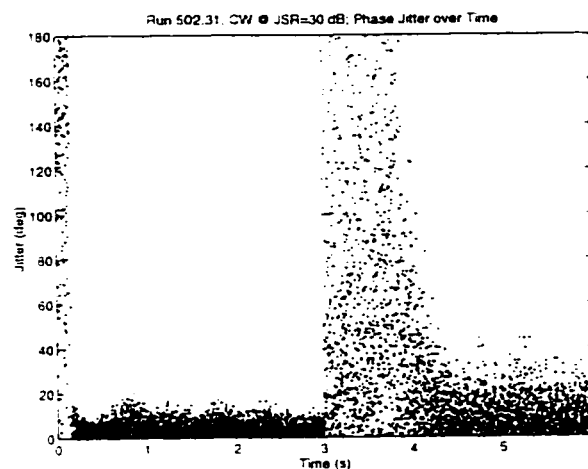


Figure 502.31.d: Carrier Phase Jitter, 30dB CW Interference

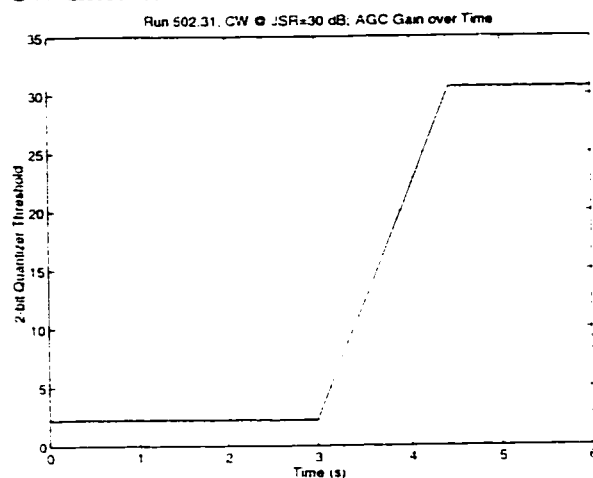


Figure 502.31.e: AGC, 30dB CW Interference

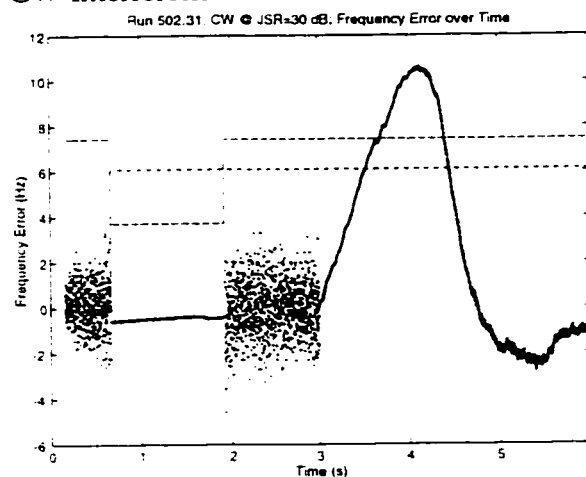


Figure 502.31.f: Frequency Error, 30dB CW Interference

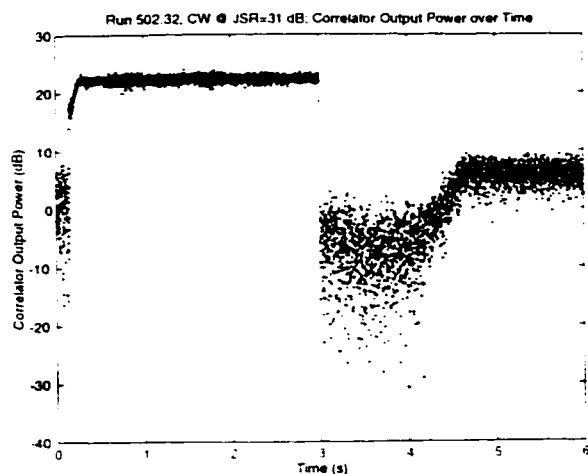


Figure 502.32.a: Correlator Output Power, 31dB CW Interference

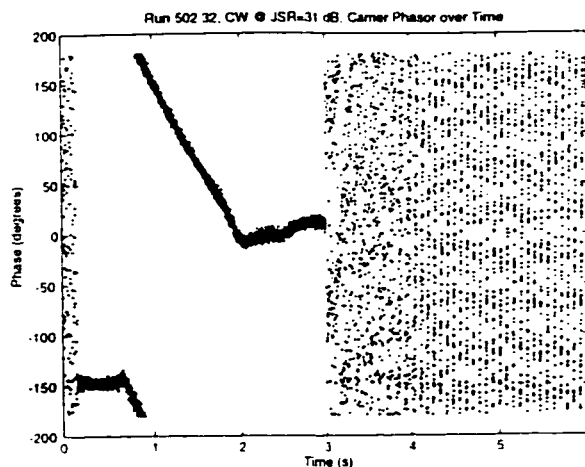


Figure 502.32.b: Carrier Phase, 31dB CW Interference

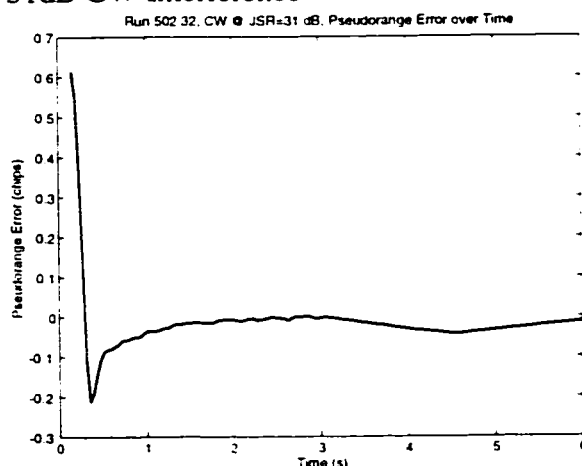


Figure 502.32.c: Pseudorange Error, 31dB CW Interference

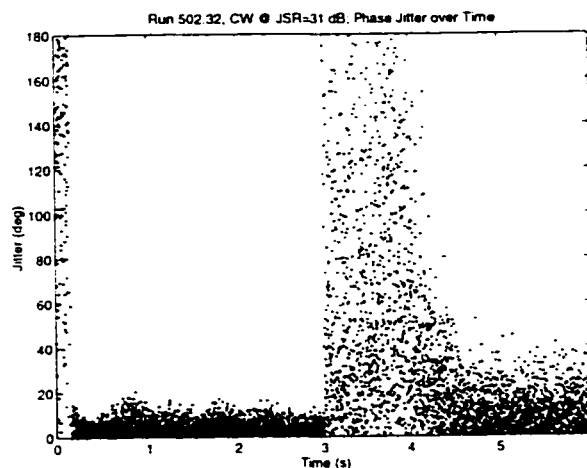


Figure 502.32.d: Carrier Phase Jitter, 31dB CW Interference

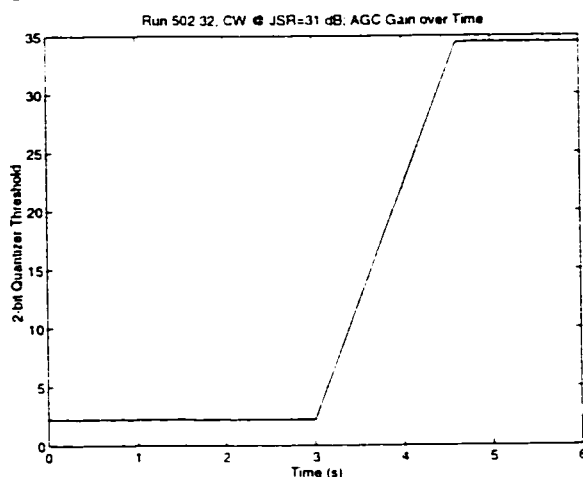


Figure 502.32.e: AGC, 31dB CW Interference

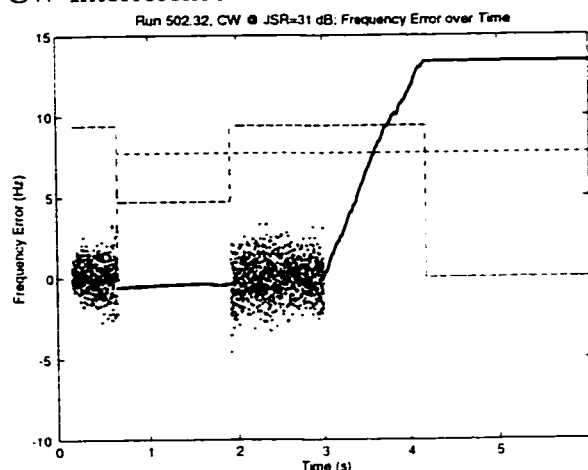


Figure 502.32.f: Frequency Error, 31dB CW Interference

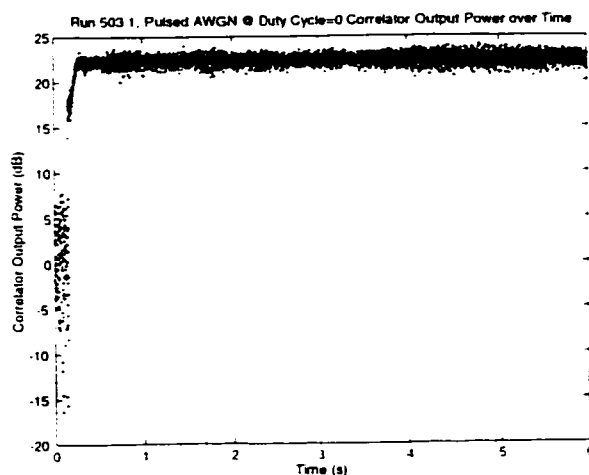


Figure 503.1.a: Correlator Output Power, 0% Pulsed AWGN Interference duty cycle

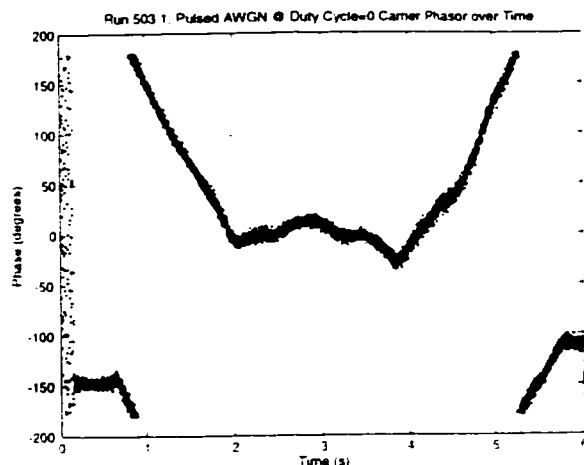


Figure 503.1.b: Carrier Phase, 0% Pulsed AWGN Interference duty cycle

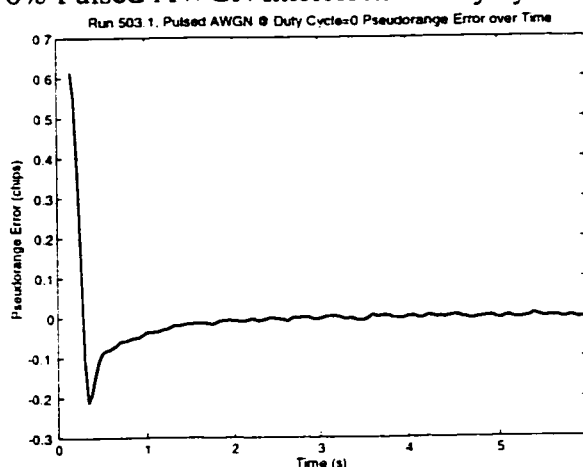


Figure 503.1.c: Pseudorange Error, 0% Pulsed AWGN Interference duty cycle

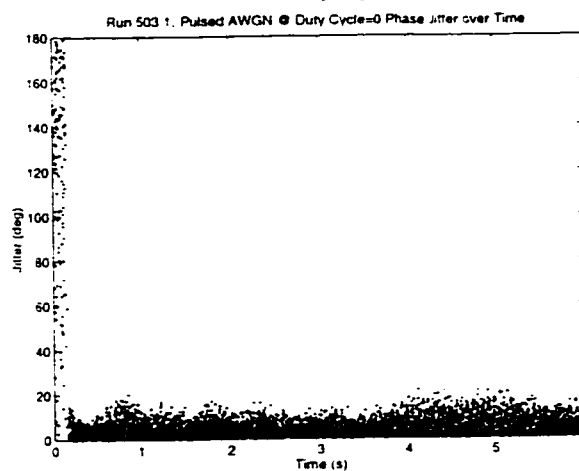


Figure 503.1.d: Carrier Phase Jitter, 0% Pulsed AWGN Interference duty cycle

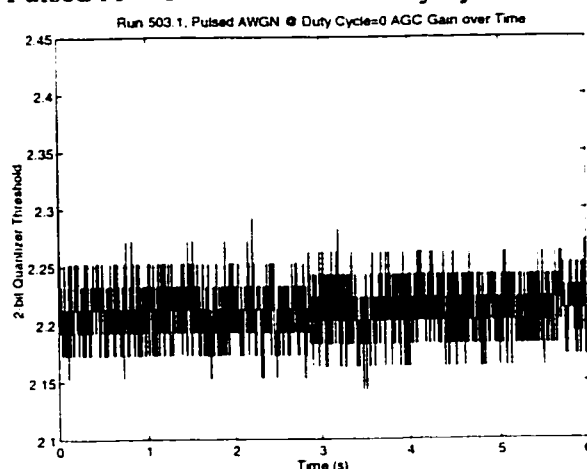


Figure 503.1.e: AGC, 0% Pulsed AWGN Interference duty cycle

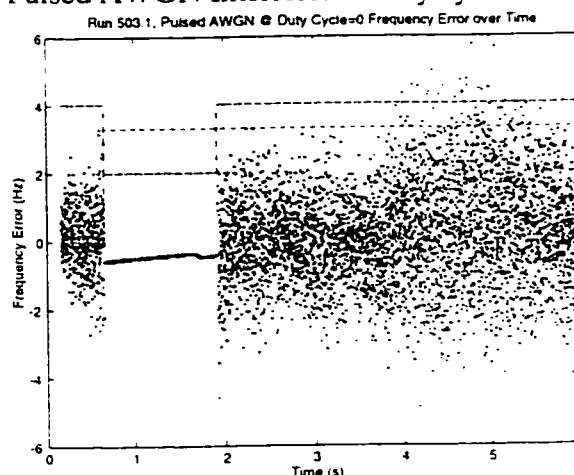


Figure 503.1.f: Frequency Error, 0% Pulsed AWGN Interference duty cycle

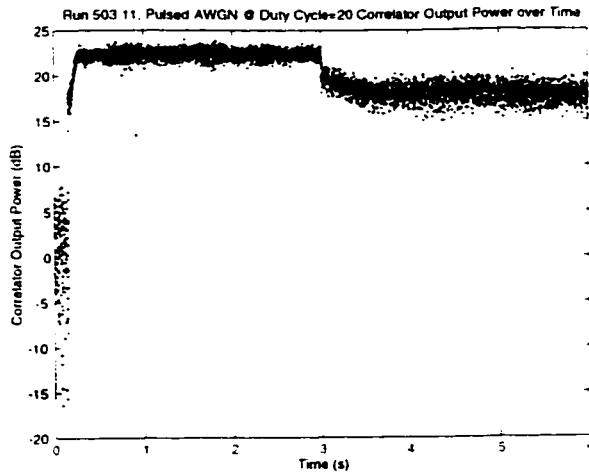


Figure 503.11.a: Correlator Output Power, 20% Pulsed AWGN Interference duty cycle

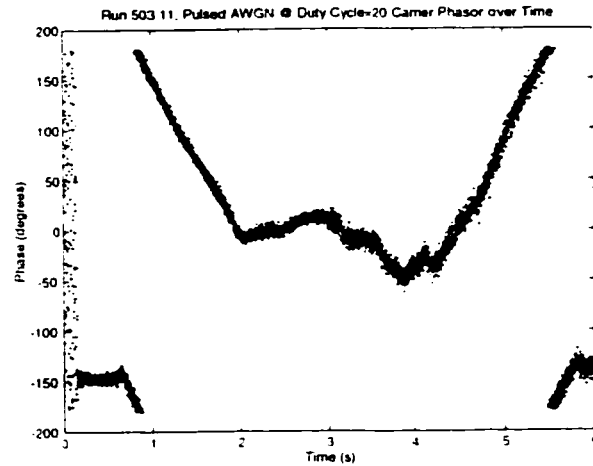


Figure 503.11.b: Carrier Phase, 20% Pulsed AWGN Interference duty cycle

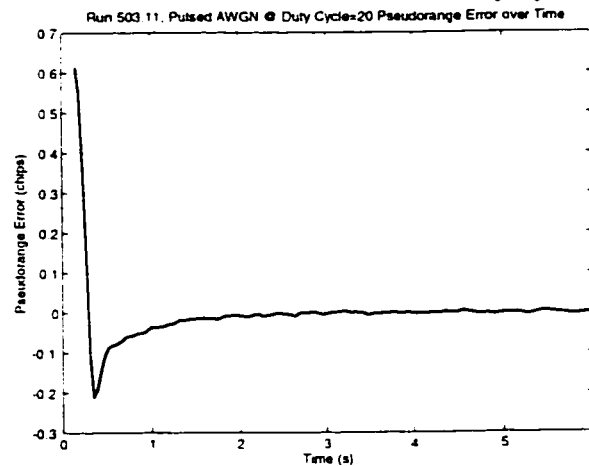


Figure 503.11.c: Pseudorange Error, 20% Pulsed AWGN Interference duty cycle

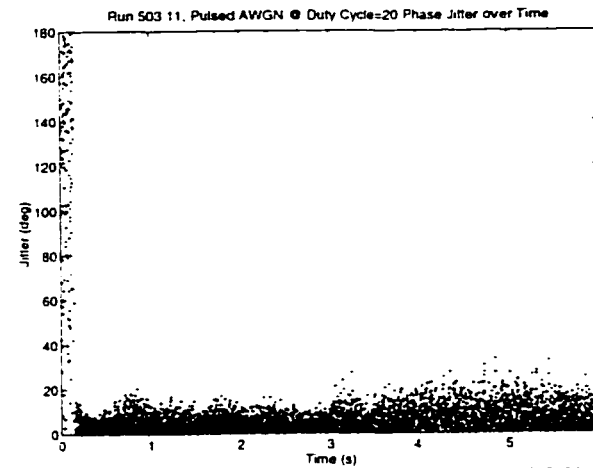


Figure 503.11.d: Carrier Phase Jitter, 20% Pulsed AWGN Interference duty cycle

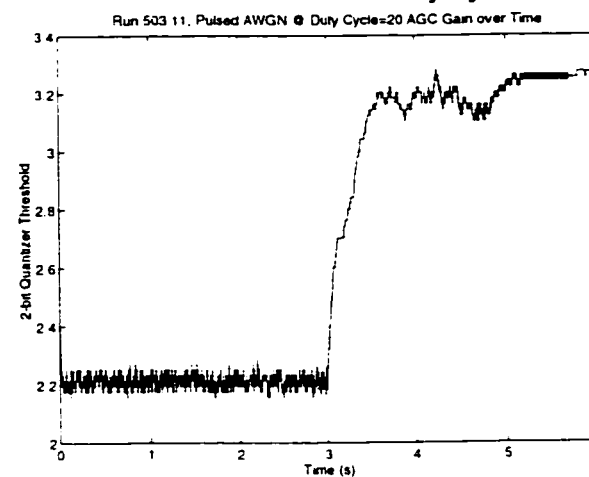


Figure 503.11.e: AGC, 20% Pulsed AWGN Interference duty cycle

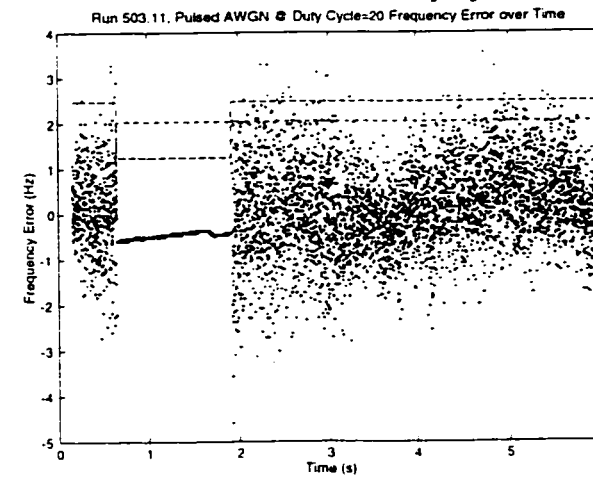


Figure 503.11.f: Frequency Error, 20% Pulsed AWGN Interference duty cycle

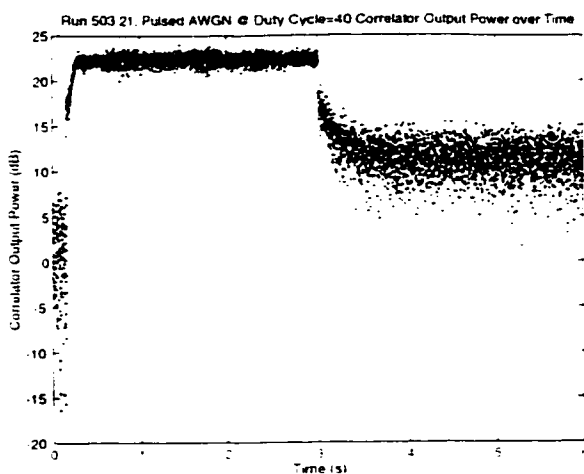


Figure 503.21.a: Correlator Output Power, 40% Pulsed AWGN Interference duty cycle

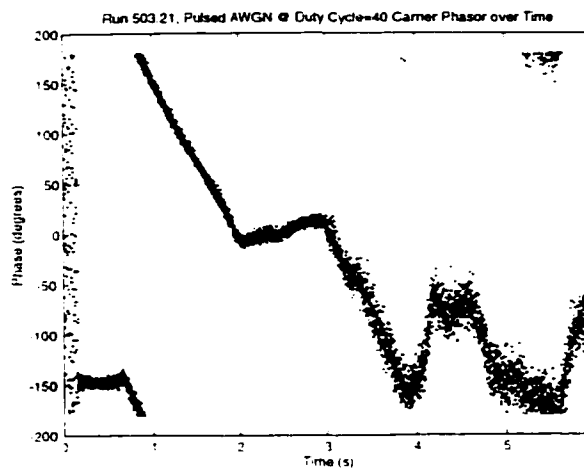


Figure 503.21.b: Carrier Phase, 40% Pulsed AWGN Interference duty cycle

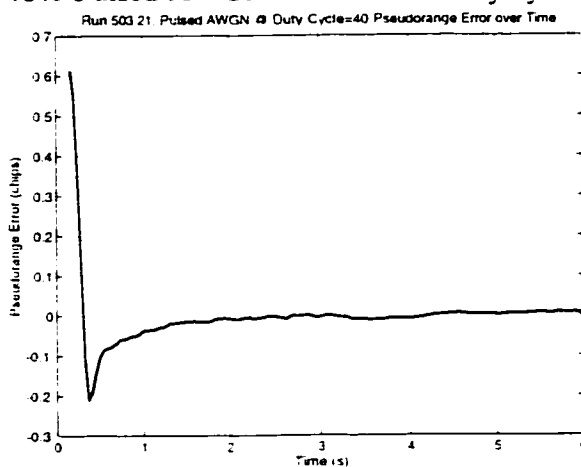


Figure 503.21.c: Pseudorange Error, 40% Pulsed AWGN Interference duty cycle

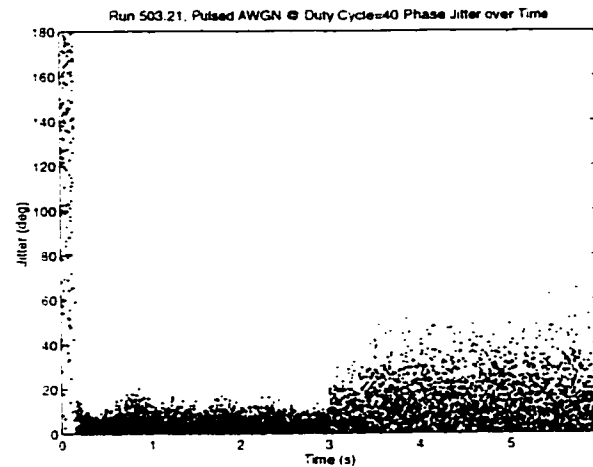


Figure 503.21.d: Carrier Phase Jitter, 40% Pulsed AWGN Interference duty cycle

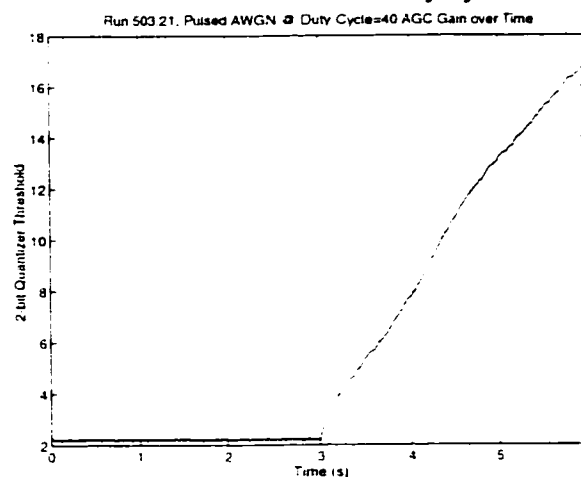


Figure 503.21.e: AGC, 40% Pulsed AWGN Interference duty cycle

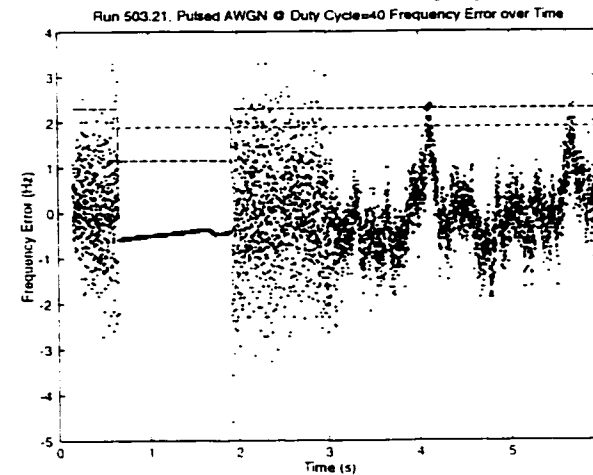


Figure 503.21.f: Frequency Error, 40% Pulsed AWGN Interference duty cycle

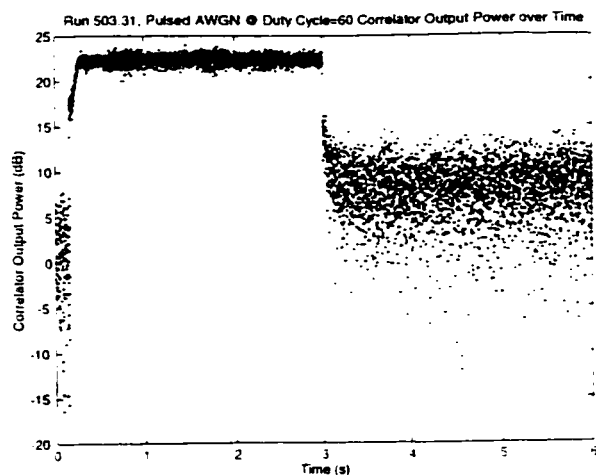


Figure 503.31.a: Correlator Output Power, 60% Pulsed AWGN Interference duty cycle

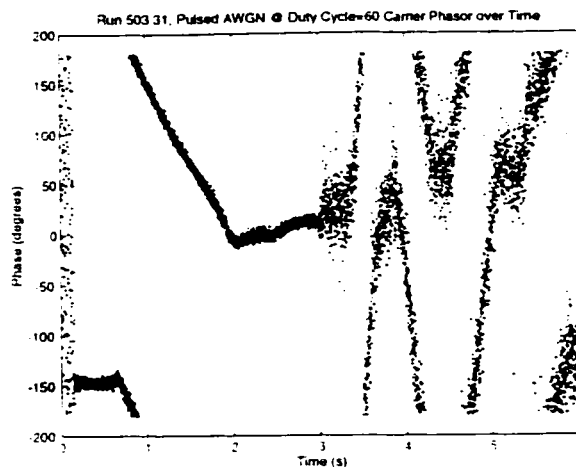


Figure 503.31.b: Carrier Phase, 60% Pulsed AWGN Interference duty cycle

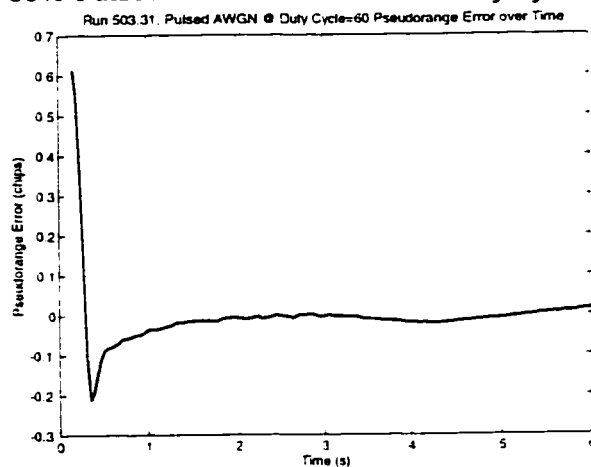


Figure 503.31.c: Pseudorange Error, 60% Pulsed AWGN Interference duty cycle

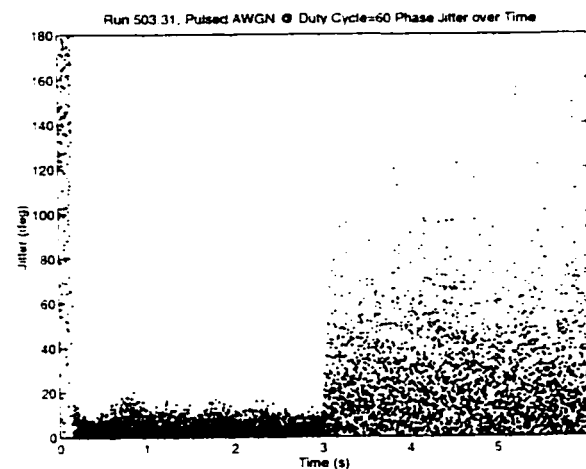


Figure 503.31.d: Carrier Phase Jitter, 60% Pulsed AWGN Interference duty cycle

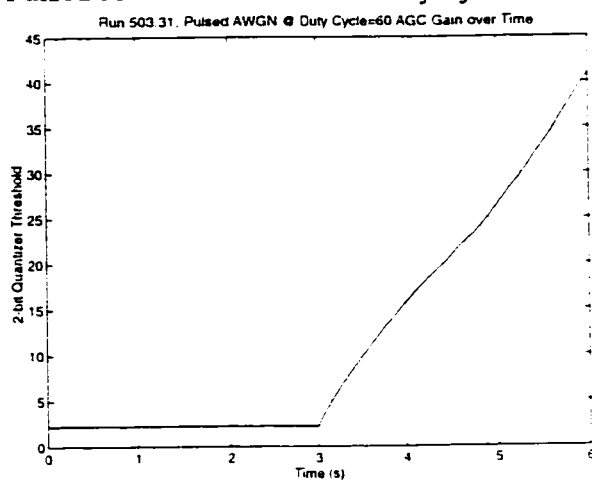


Figure 503.31.e: AGC, 60% Pulsed AWGN Interference duty cycle

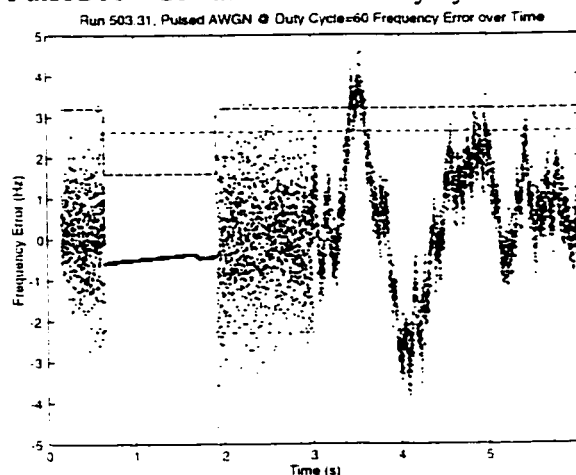


Figure 503.31.f: Frequency Error, 60% Pulsed AWGN Interference duty cycle

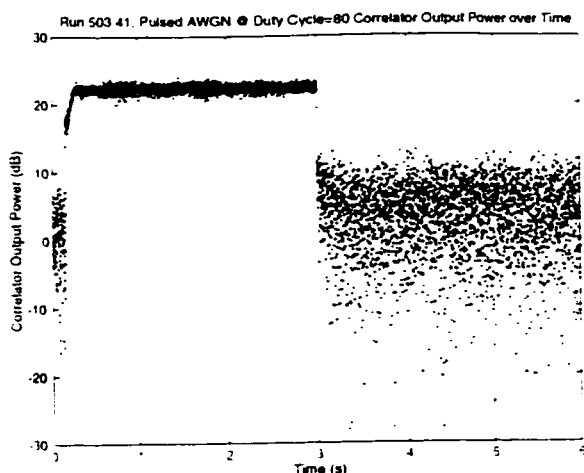


Figure 503.41.a: Correlator Output Power, 80% Pulsed AWGN Interference duty cycle

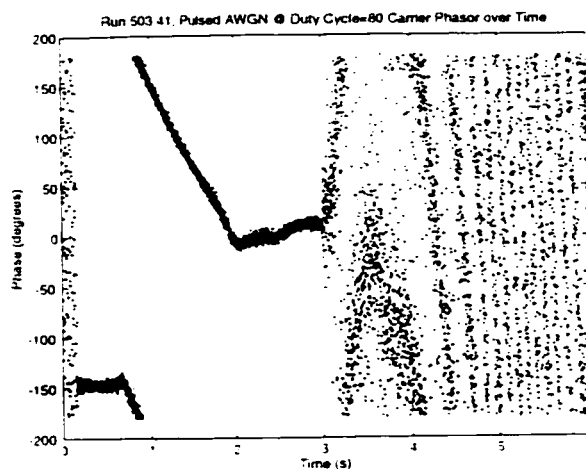


Figure 503.41.b: Carrier Phase, 80% Pulsed AWGN Interference duty cycle

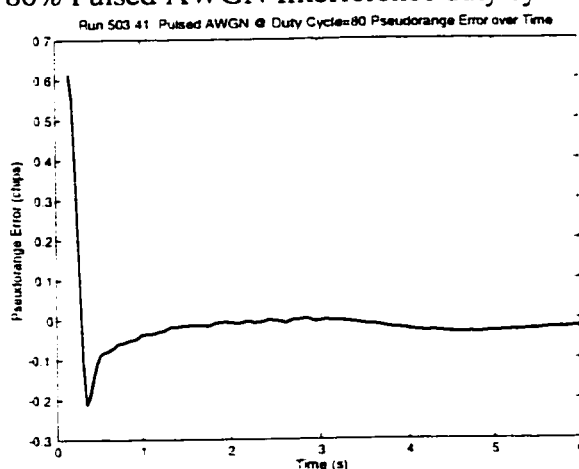


Figure 503.41.c: Pseudorange Error, 80% Pulsed AWGN Interference duty cycle

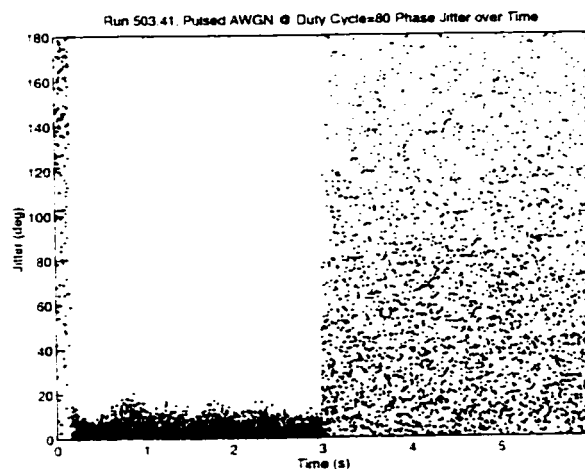


Figure 503.41.d: Carrier Phase Jitter, 80% Pulsed AWGN Interference duty cycle

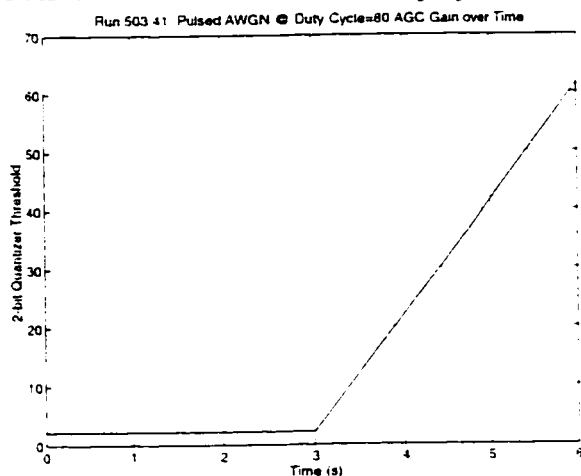


Figure 503.41.e: AGC, 80% Pulsed AWGN Interference duty cycle

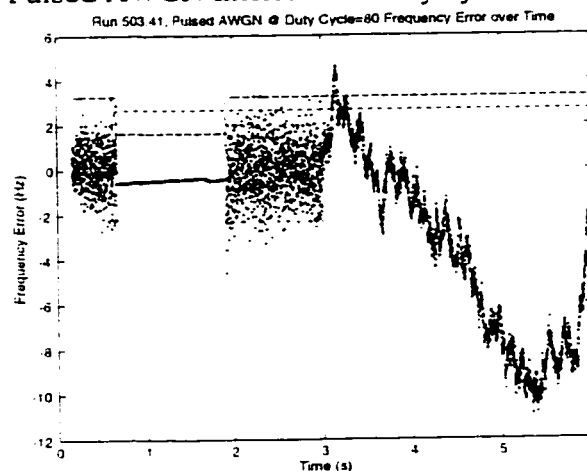


Figure 503.41.f: Frequency Error, 80% Pulsed AWGN Interference duty cycle

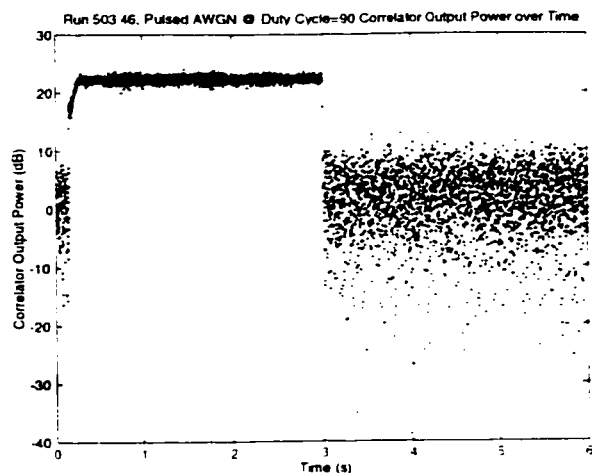


Figure 503.46.a: Correlator Output Power, 90% Pulsed AWGN Interference duty cycle

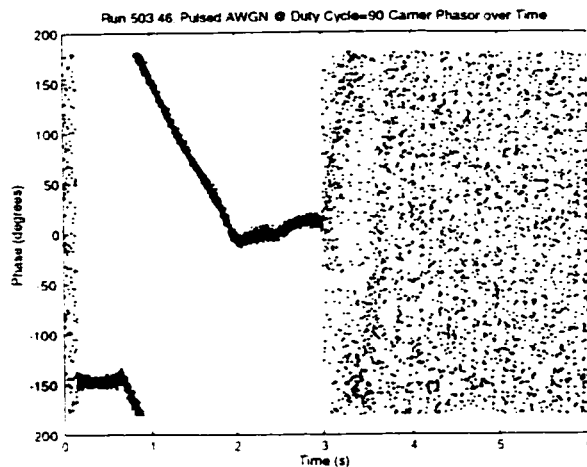


Figure 503.46.b: Carrier Phase, 90% Pulsed AWGN Interference duty cycle

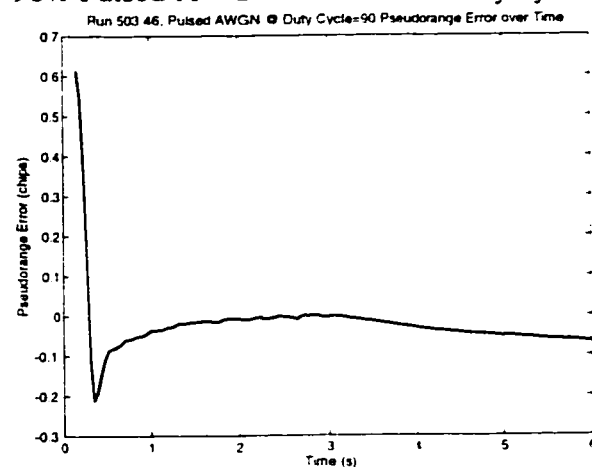


Figure 503.46.c: Pseudorange Error, 90% Pulsed AWGN Interference duty cycle

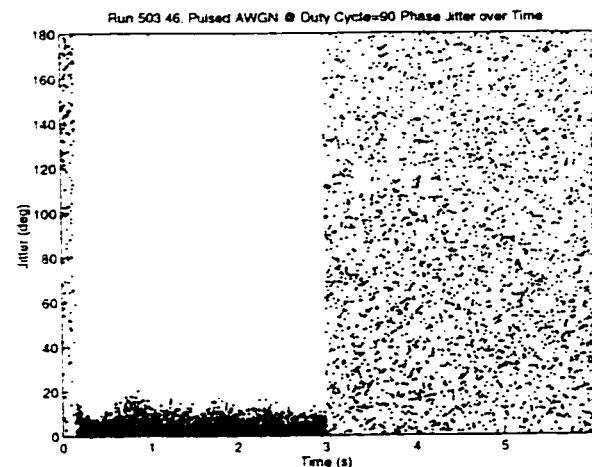


Figure 503.46.d: Carrier Phase Jitter, 90% Pulsed AWGN Interference duty cycle

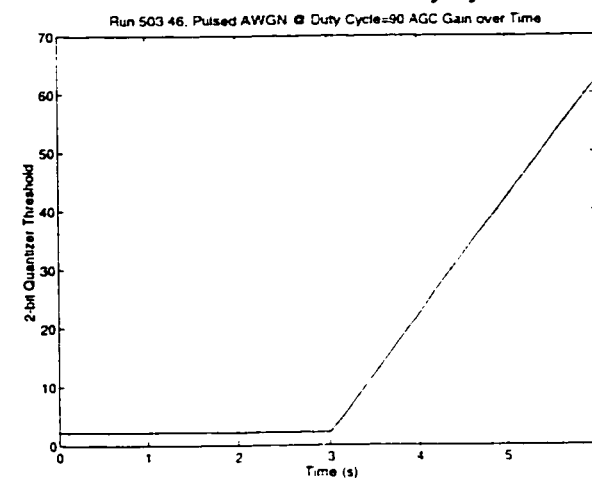


Figure 503.46.e: AGC, 90% Pulsed AWGN Interference duty cycle

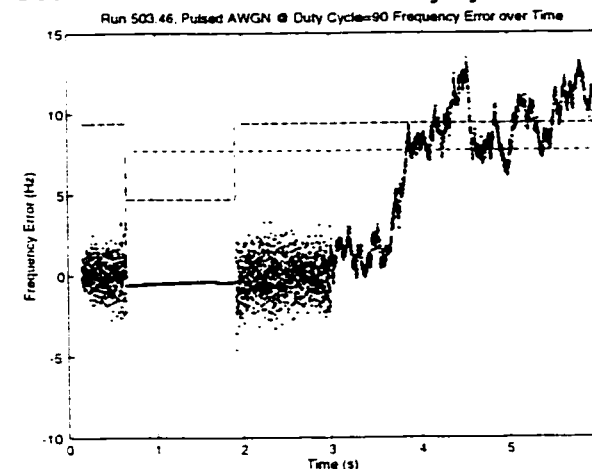


Figure 503.46.f: Frequency Error, 90% Pulsed AWGN Interference duty cycle

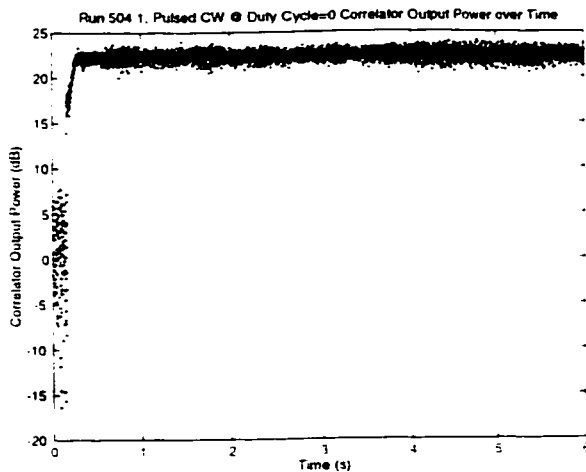


Figure 504.1.a: Correlator Output Power, 0% Pulsed CW Interference duty cycle

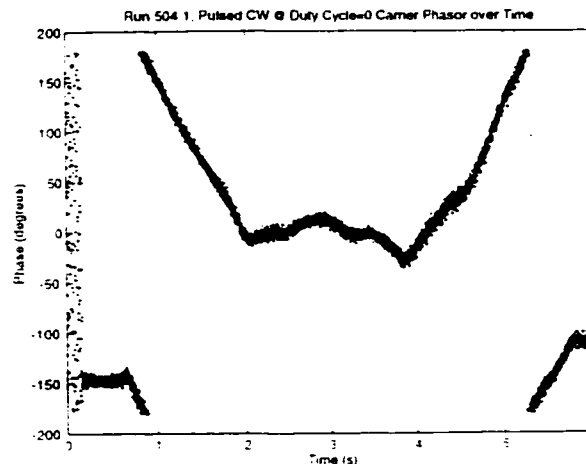


Figure 504.1.b: Carrier Phase, 0% Pulsed CW Interference duty cycle

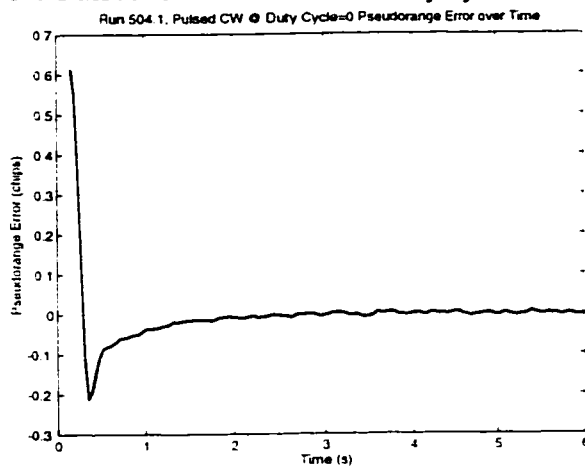


Figure 504.1.c: Pseudorange Error, 0% Pulsed CW Interference duty cycle

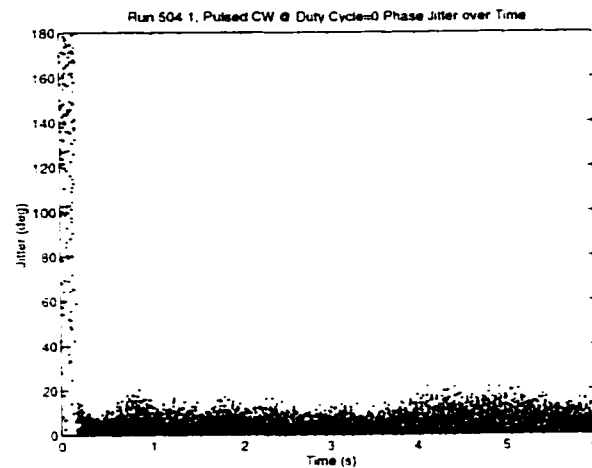


Figure 504.1.d: Carrier Phase Jitter, 0% Pulsed CW Interference duty cycle

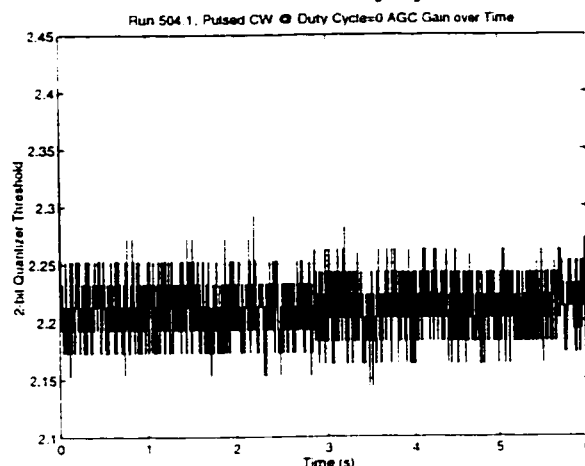


Figure 504.1.e: AGC, 0% Pulsed CW Interference duty cycle

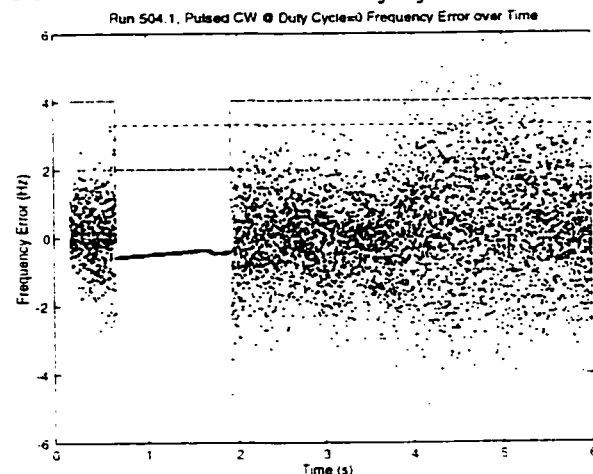


Figure 504.1.f: Frequency Error, 0% Pulsed CW Interference duty cycle

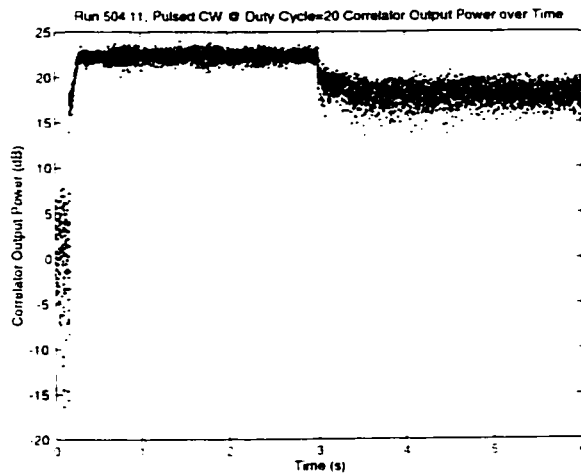


Figure 504.11.a: Correlator Output Power, 20% Pulsed CW Interference duty cycle

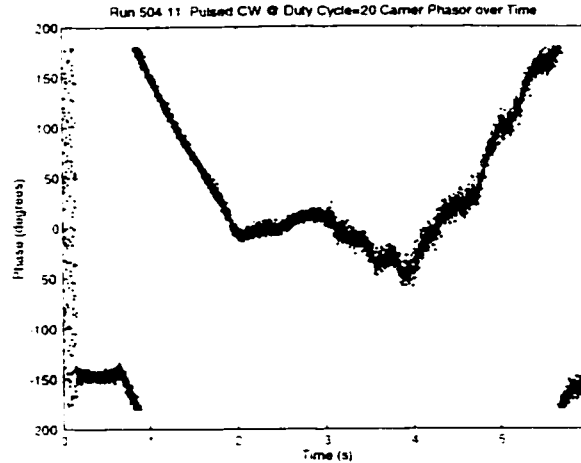


Figure 504.11.b: Carrier Phase, 20% Pulsed CW Interference duty cycle

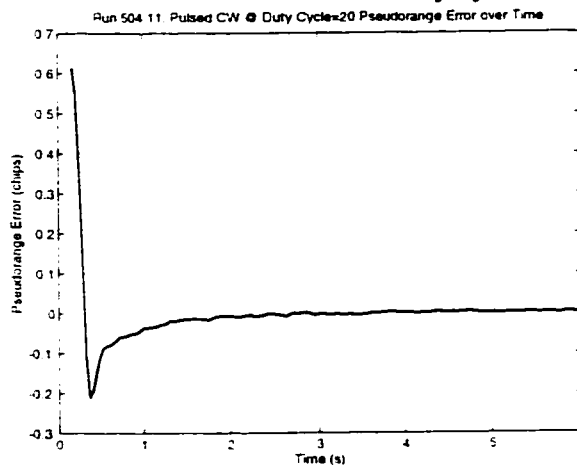


Figure 504.11.c: Pseudorange Error, 20% Pulsed CW Interference duty cycle

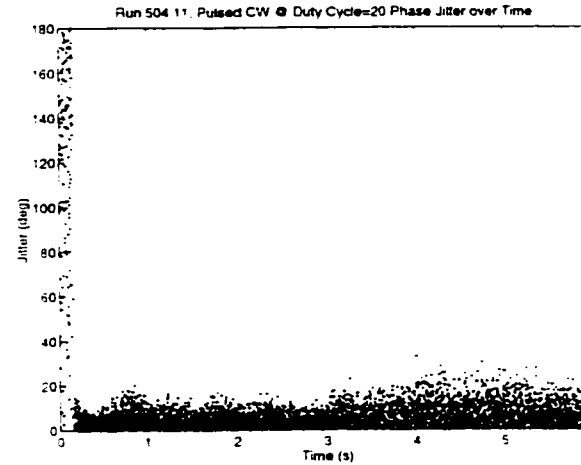


Figure 504.11.d: Carrier Phase Jitter, 20% Pulsed CW Interference duty cycle

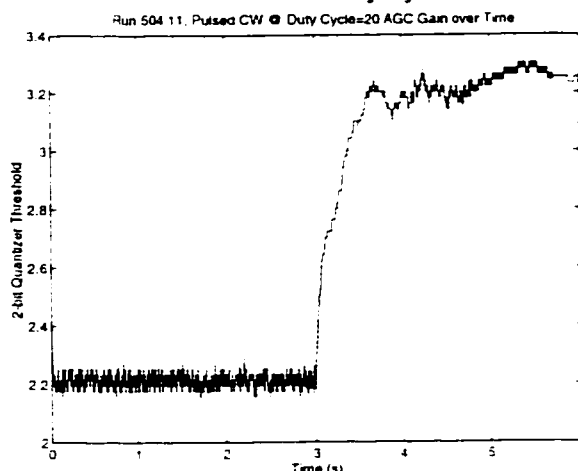


Figure 504.11.e: AGC, 20% Pulsed CW Interference duty cycle

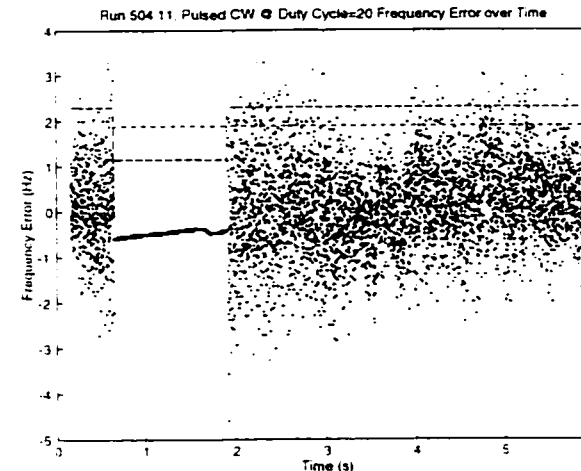


Figure 504.11.f: Frequency Error, 20% Pulsed CW Interference duty cycle

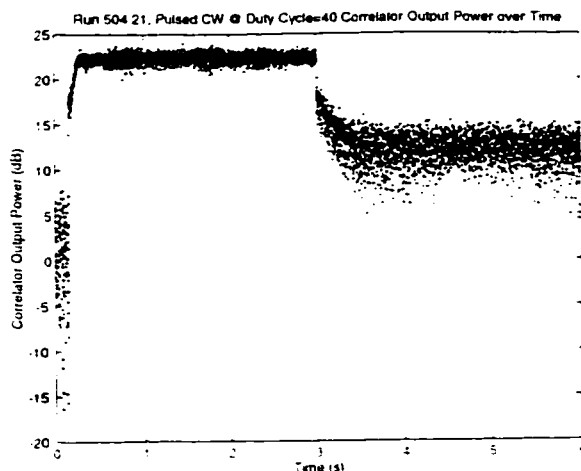


Figure 504.21.a: Correlator Output Power. 40% Pulsed CW Interference duty cycle

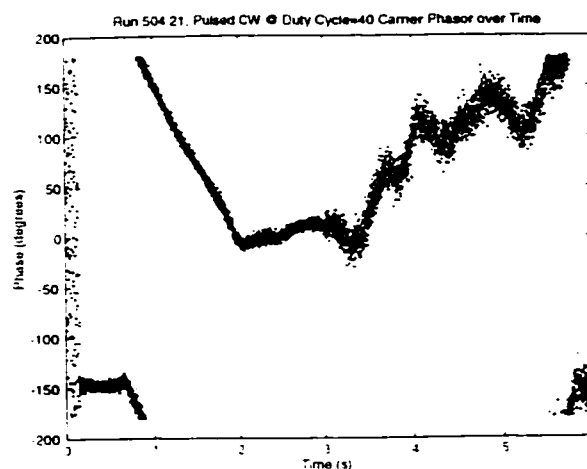


Figure 504.21.b: Carrier Phase, 40% Pulsed CW Interference duty cycle

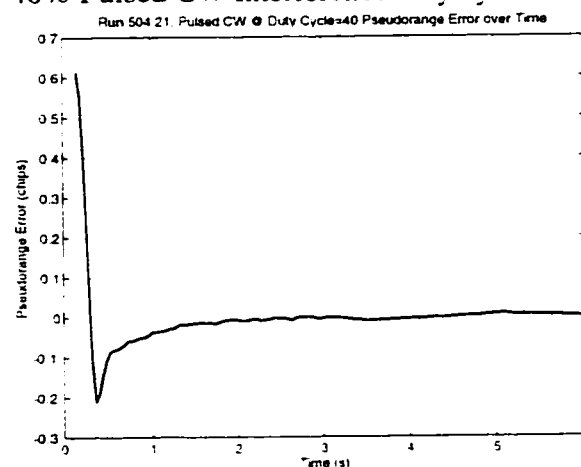


Figure 504.21.c: Pseudorange Error, 40% Pulsed CW Interference duty cycle

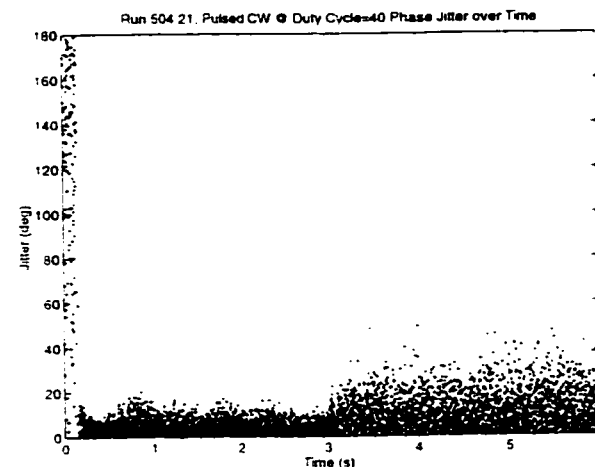


Figure 504.21.d: Carrier Phase Jitter, 40% Pulsed CW Interference duty cycle

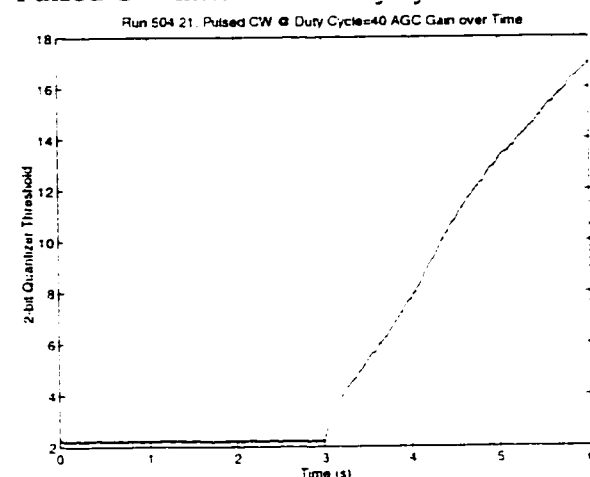


Figure 504.21.e: AGC, 40% Pulsed CW Interference duty cycle

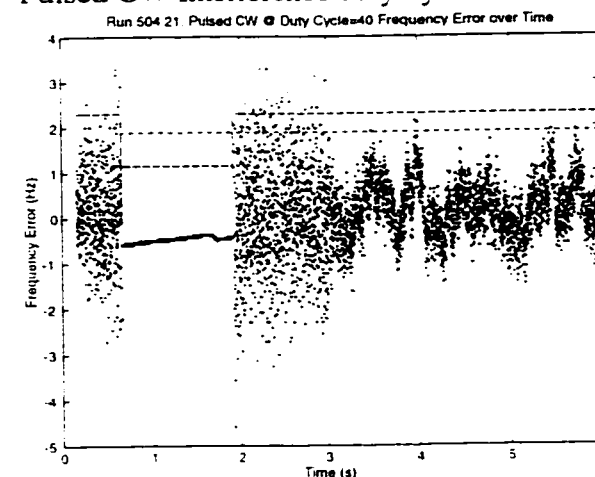


Figure 504.21.f: Frequency Error, 40% Pulsed CW Interference duty cycle

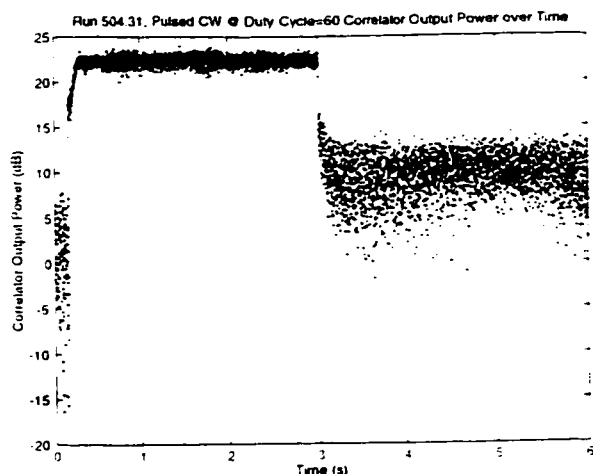


Figure 504.31.a: Correlator Output Power. 60% Pulsed CW Interference duty cycle

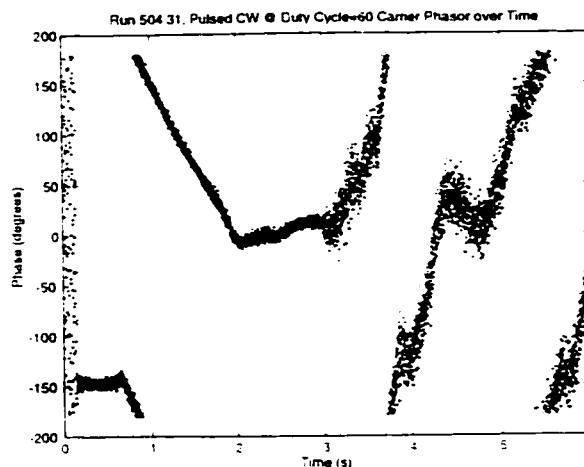


Figure 504.31.b: Carrier Phase. 60% Pulsed CW Interference duty cycle

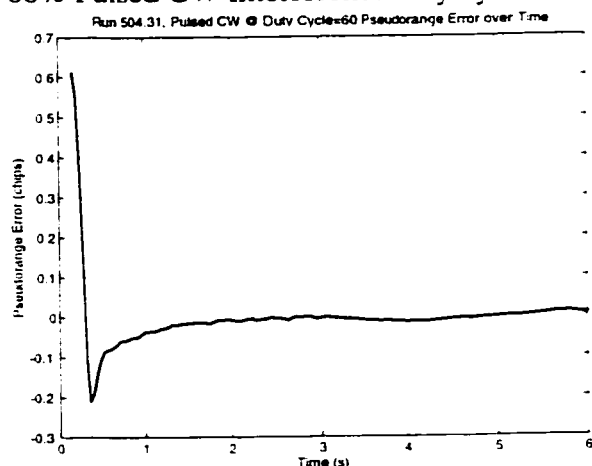


Figure 504.31.c: Pseudorange Error. 60% Pulsed CW Interference duty cycle

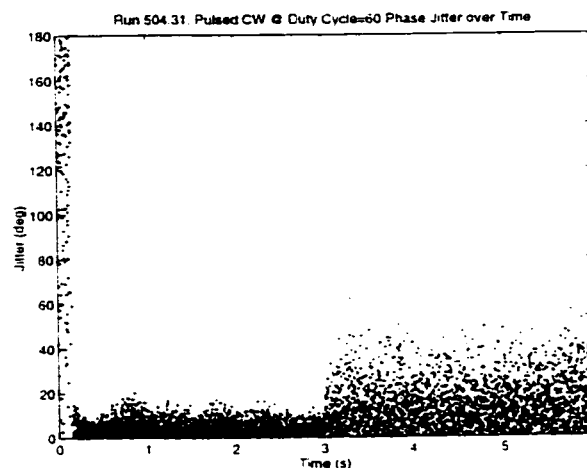


Figure 504.31.d: Carrier Phase Jitter. 60% Pulsed CW Interference duty cycle

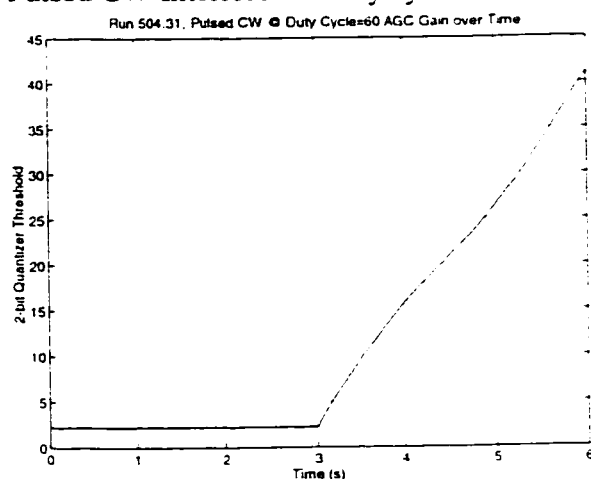


Figure 504.31.e: AGC. 60% Pulsed CW Interference duty cycle

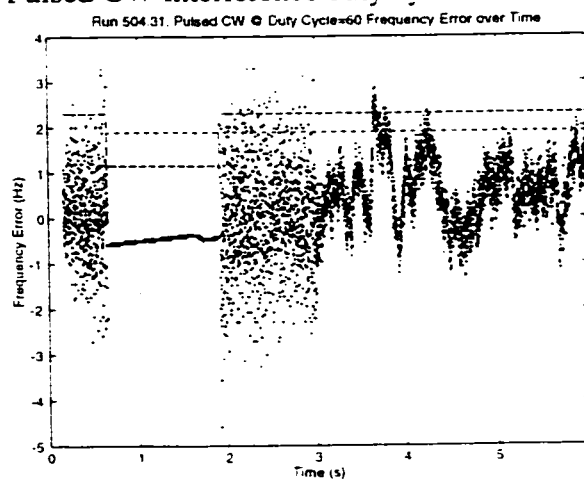


Figure 504.31.f: Frequency Error. 60% Pulsed CW Interference duty cycle

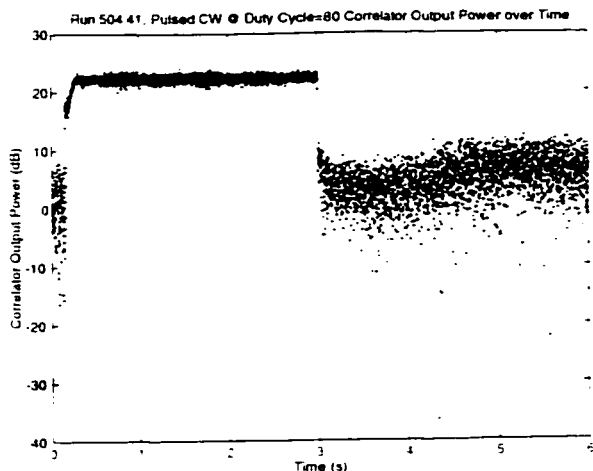


Figure 504.41.a: Correlator Output Power, 80% Pulsed CW Interference duty cycle

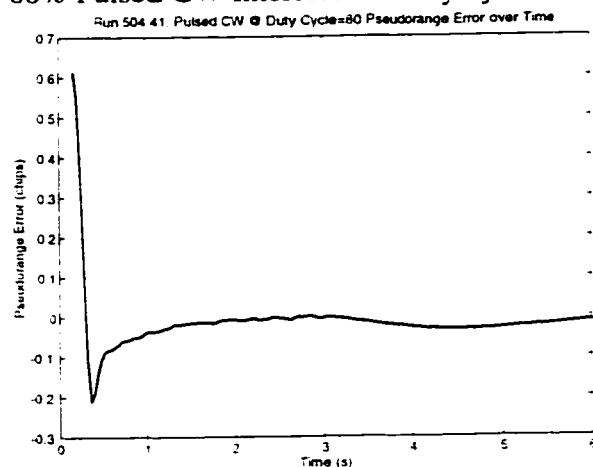


Figure 504.41.c: Pseudorange Error, 80% Pulsed CW Interference duty cycle

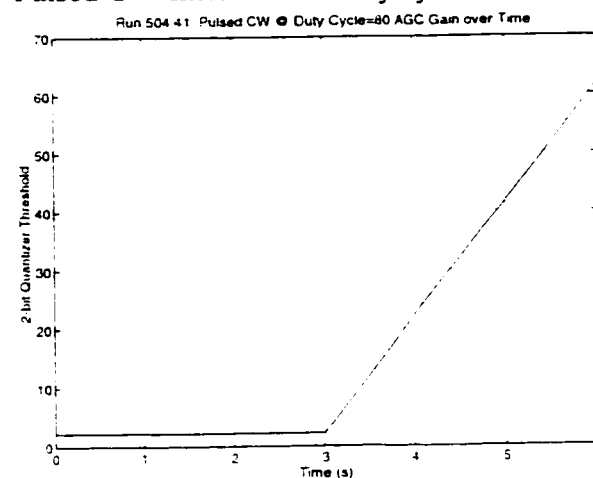


Figure 504.41.e: AGC, 80% Pulsed CW Interference duty cycle

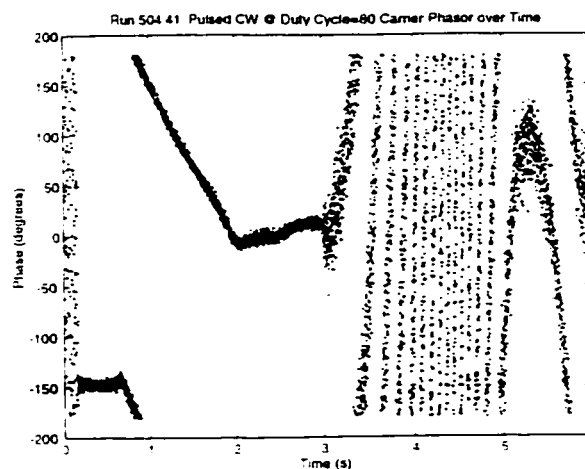


Figure 504.41.b: Carrier Phase, 80% Pulsed CW Interference duty cycle

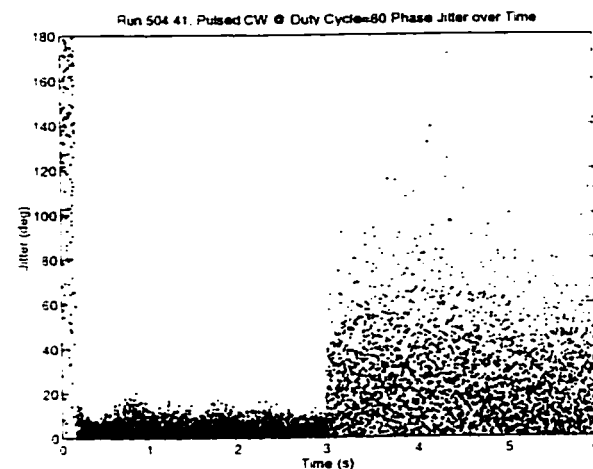


Figure 504.41.d: Carrier Phase Jitter, 80% Pulsed CW Interference duty cycle

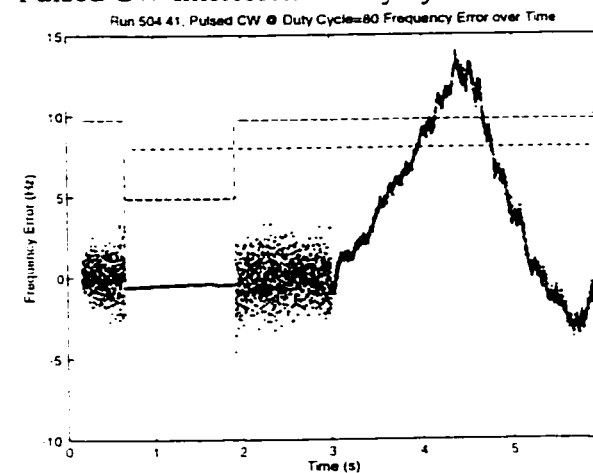


Figure 504.41.f: Frequency Error, 80% Pulsed CW Interference duty cycle

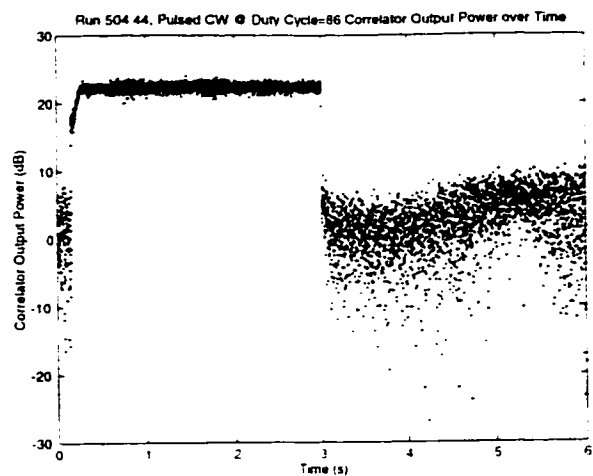


Figure 504.44.a: Correlator Output Power, 86% Pulsed CW Interference duty cycle

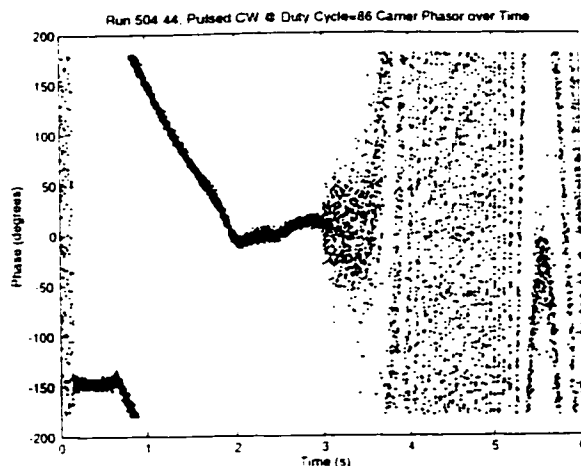


Figure 504.44.b: Carrier Phase, 86% Pulsed CW Interference duty cycle

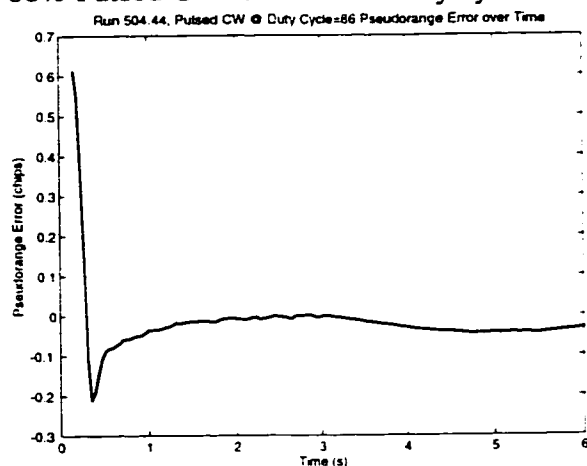


Figure 504.44.c: Pseudorange Error, 86% Pulsed CW Interference duty cycle

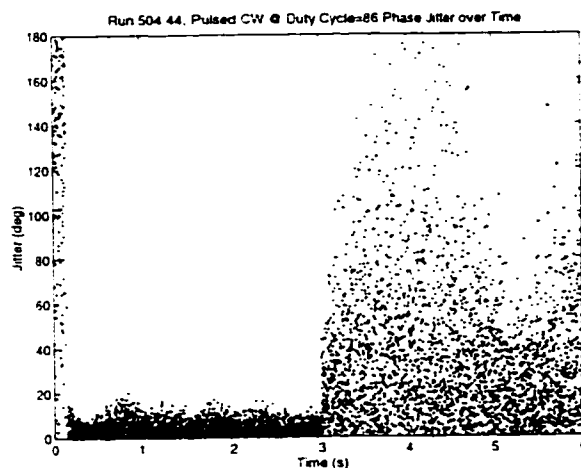


Figure 504.44.d: Carrier Phase Jitter, 86% Pulsed CW Interference duty cycle

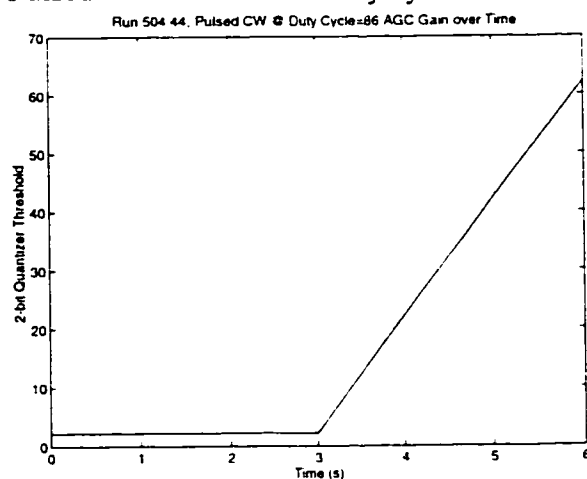


Figure 504.44.e: AGC, 86% Pulsed CW Interference duty cycle

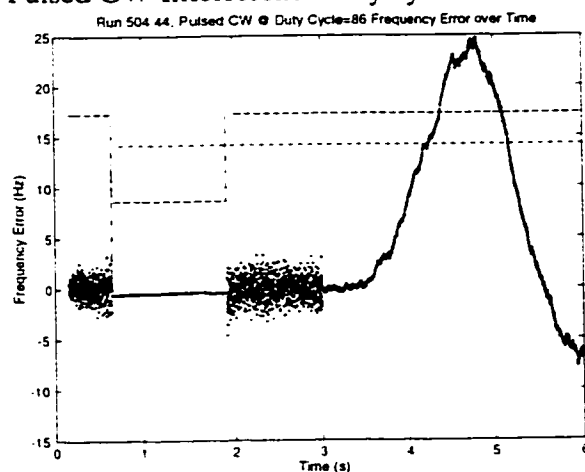


Figure 504.44.f: Frequency Error, 86% Pulsed CW Interference duty cycle

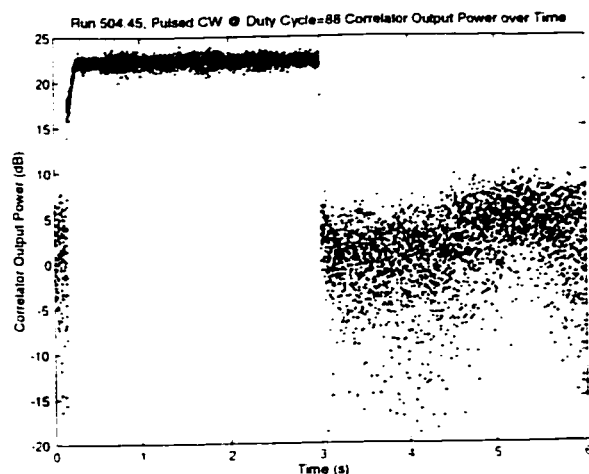


Figure 504.45.a: Correlator Output Power, 88% Pulsed CW Interference duty cycle

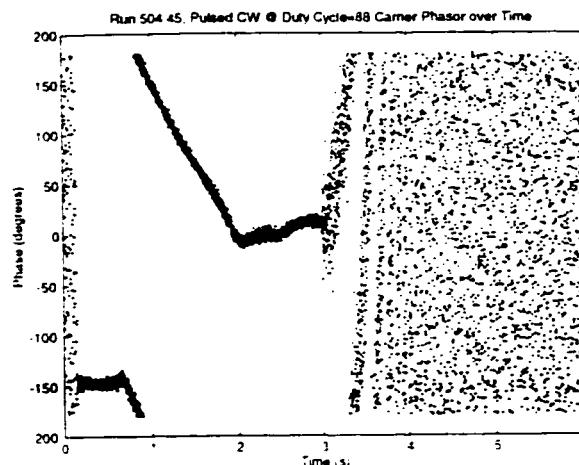


Figure 504.45.b: Carrier Phase, 88% Pulsed CW Interference duty cycle

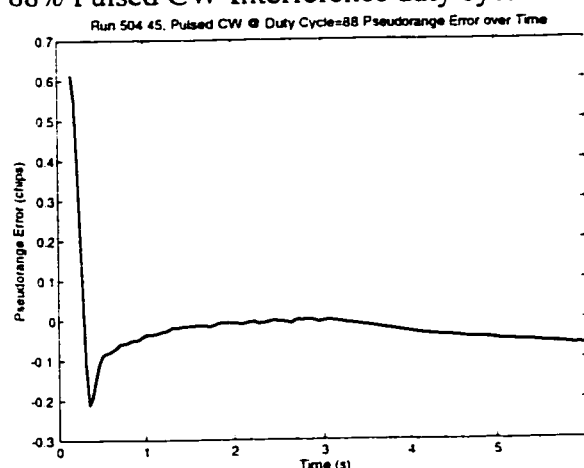


Figure 504.45.c: Pseudorange Error, 88% Pulsed CW Interference duty cycle

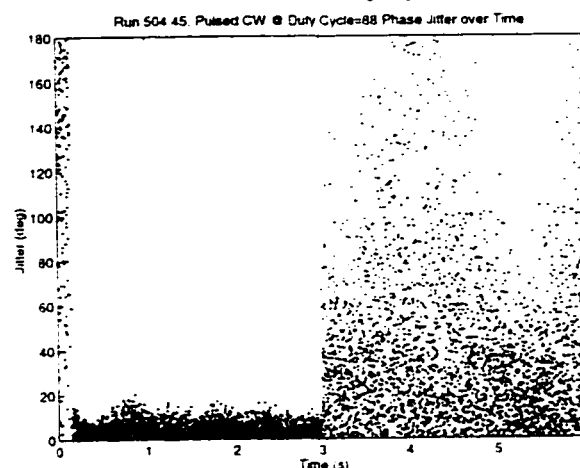


Figure 504.45.d: Carrier Phase Jitter, 88% Pulsed CW Interference duty cycle

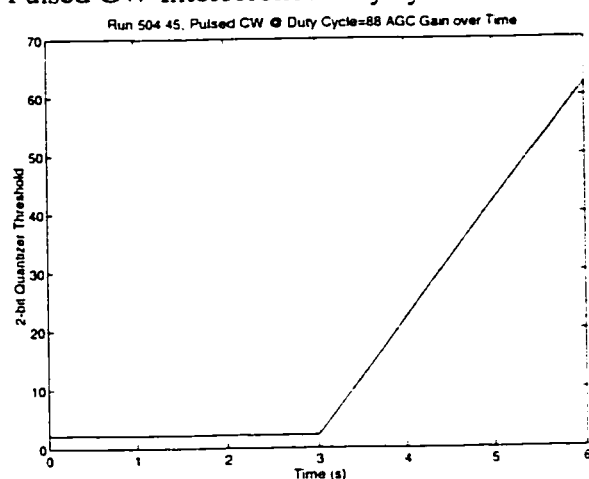


Figure 504.45.e: AGC, 88% Pulsed CW Interference duty cycle

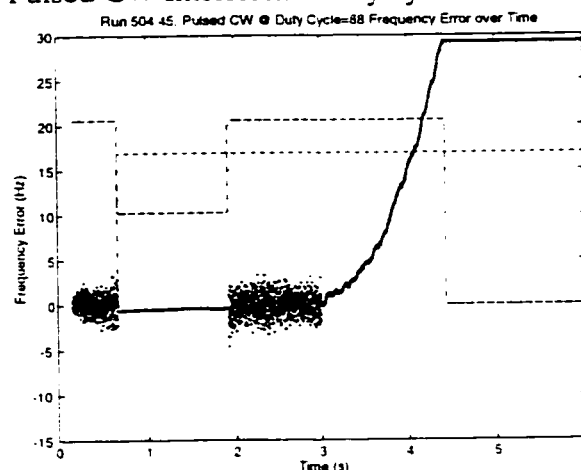


Figure 504.45.f: Frequency Error, 88% Pulsed CW Interference duty cycle

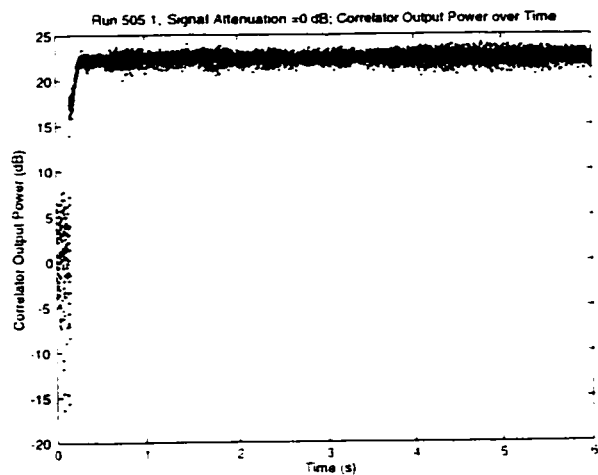


Figure 505.1.a: Correlator Output Power, 0dB Signal Attenuation

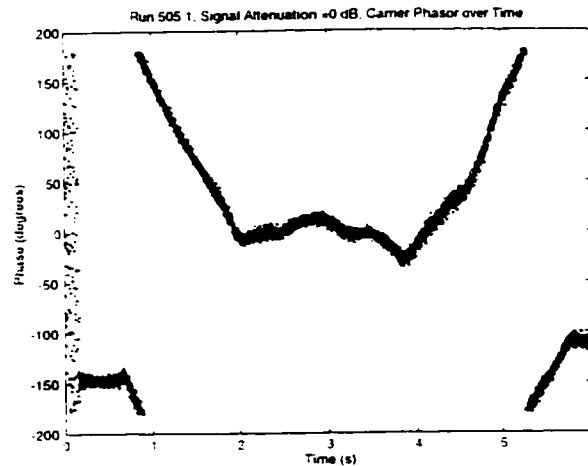


Figure 505.1.b: Carrier Phase, 0dB Signal Attenuation

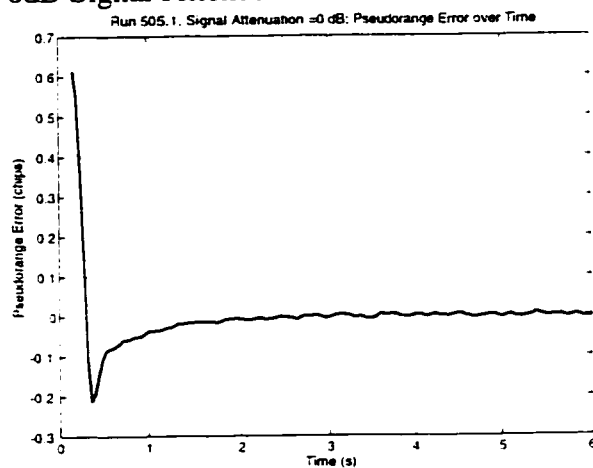


Figure 505.1.c: Pseudorange Error, 0dB Signal Attenuation

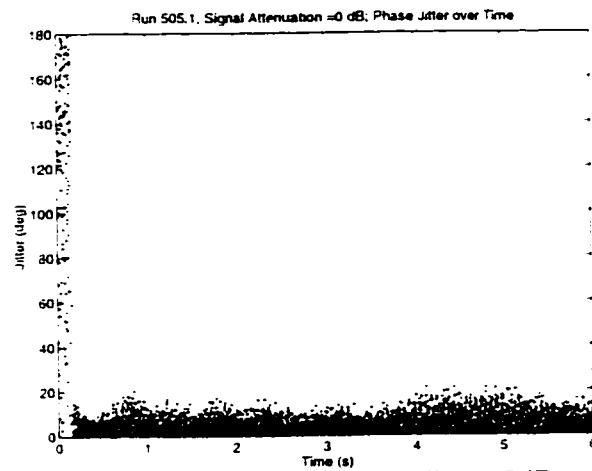


Figure 505.1.d: Carrier Phase Jitter, 0dB Signal Attenuation

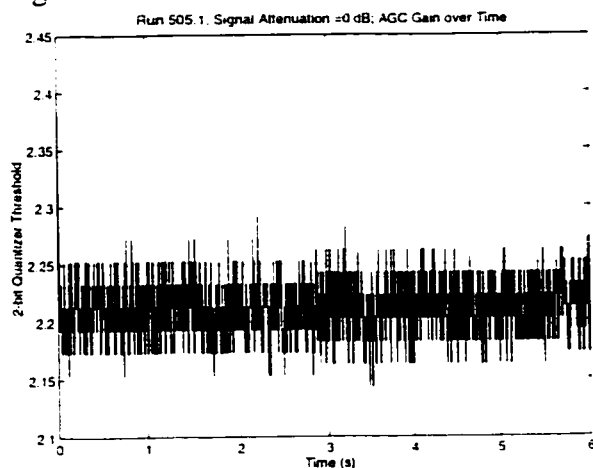


Figure 505.1.e: AGC, 0dB Signal Attenuation

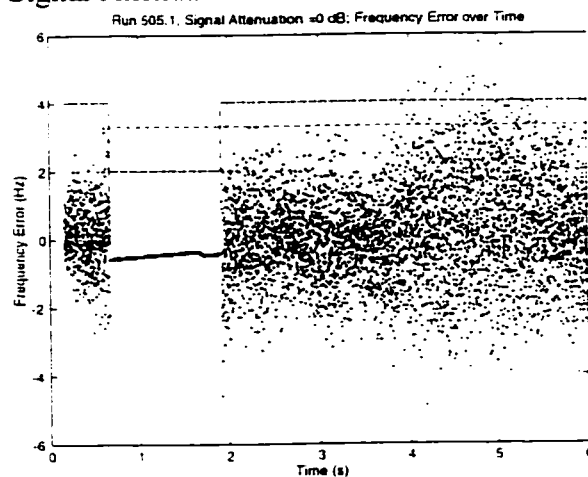


Figure 505.1.f: Frequency Error, 0dB Signal Attenuation

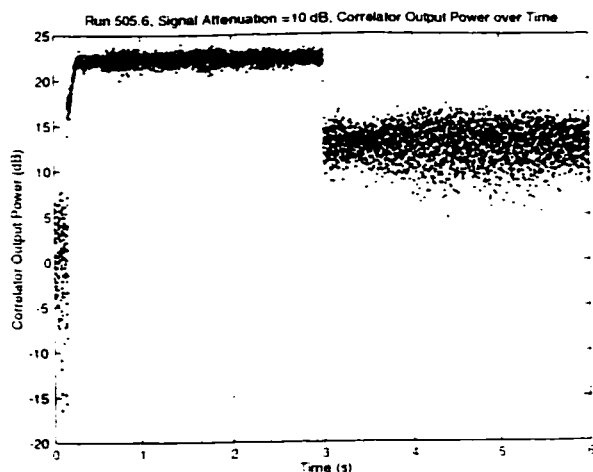


Figure 505.6.a: Correlator Output Power, 10dB Signal Attenuation

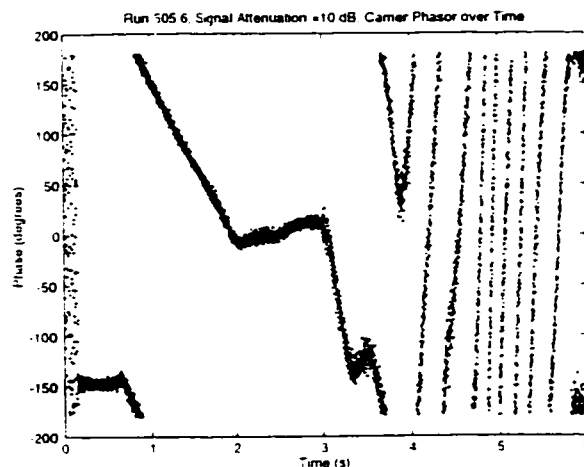


Figure 505.6.b: Carrier Phase, 10dB Signal Attenuation

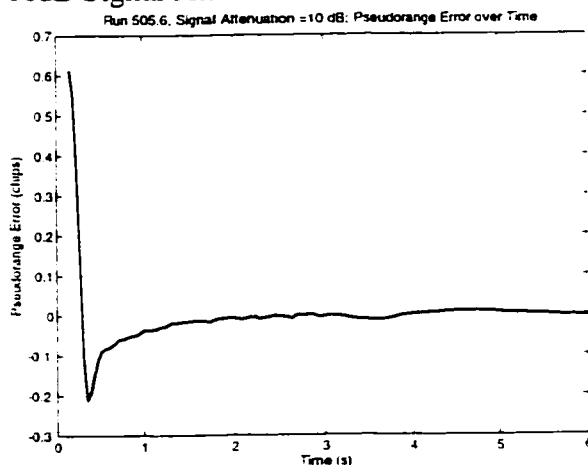


Figure 505.6.c: Pseudorange Error, 10dB Signal Attenuation

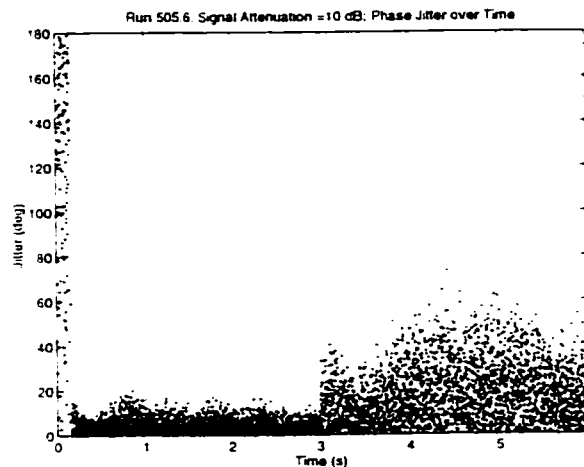


Figure 505.6.d: Carrier Phase Jitter, 10dB Signal Attenuation

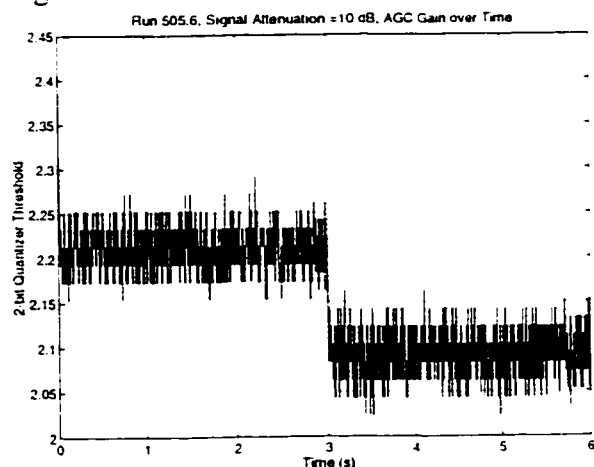


Figure 505.6.e: AGC, 10dB Signal Attenuation

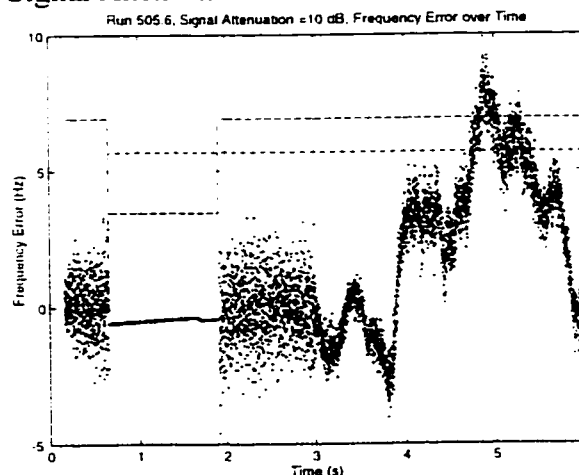


Figure 505.6.f: Frequency Error, 10dB Signal Attenuation

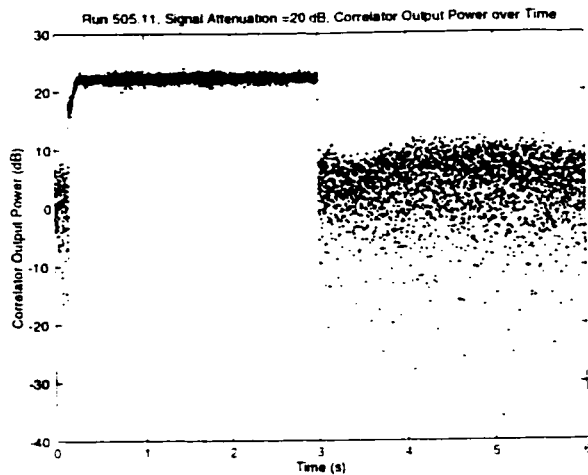


Figure 505.11.a: Correlator Output Power, 20dB Signal Attenuation

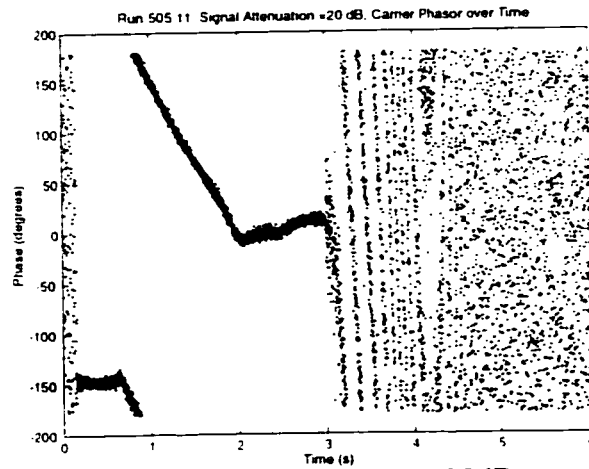


Figure 505.11.b: Carrier Phase, 20dB Signal Attenuation

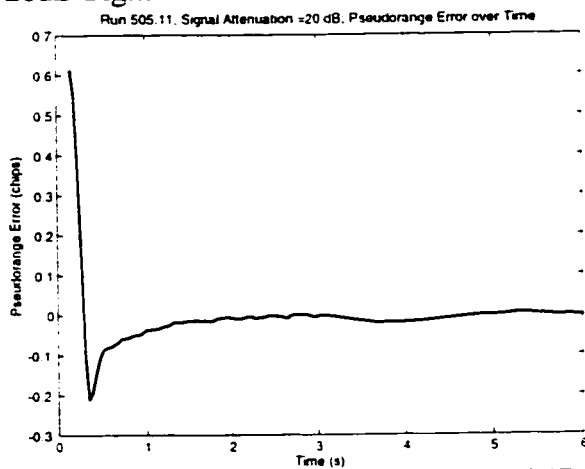


Figure 505.11.c: Pseudorange Error, 20dB Signal Attenuation

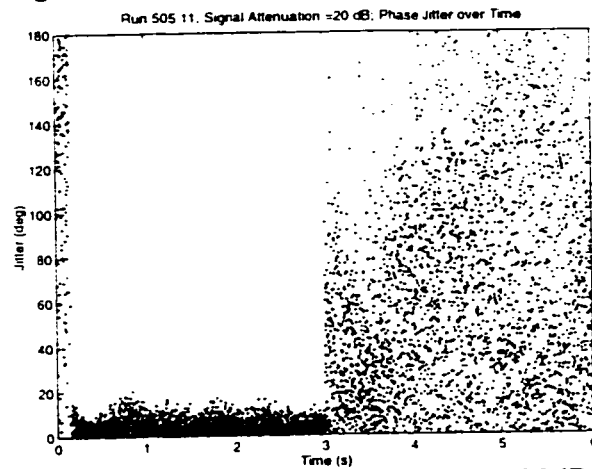


Figure 505.11.d: Carrier Phase Jitter, 20dB Signal Attenuation

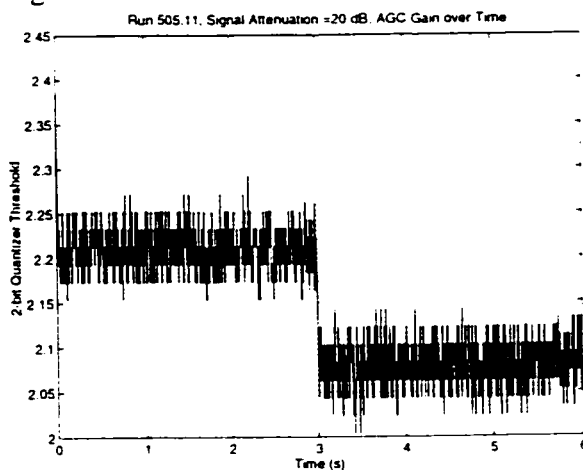


Figure 505.11.e: AGC, 20dB Signal Attenuation

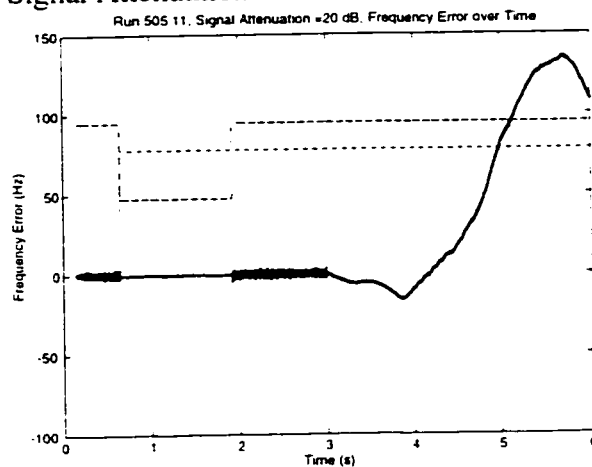


Figure 505.11.f: Frequency Error, 20dB Signal Attenuation

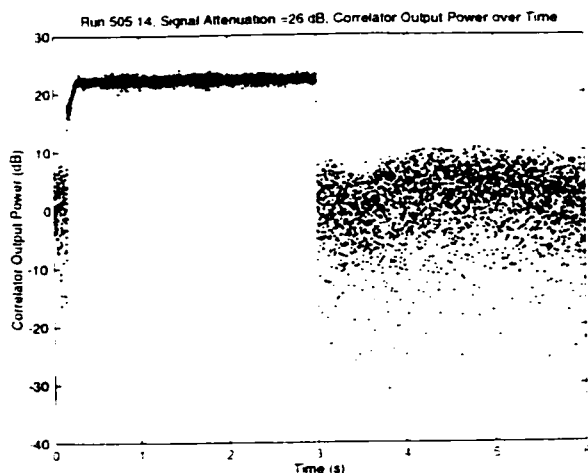


Figure 505.14.a: Correlator Output Power, 26dB Signal Attenuation

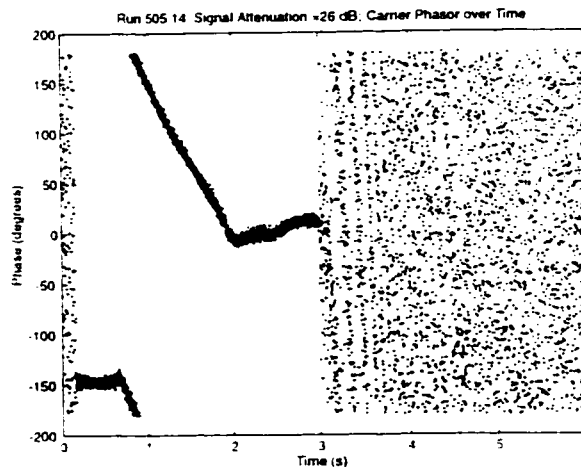


Figure 505.14.b: Carrier Phase, 26dB Signal Attenuation

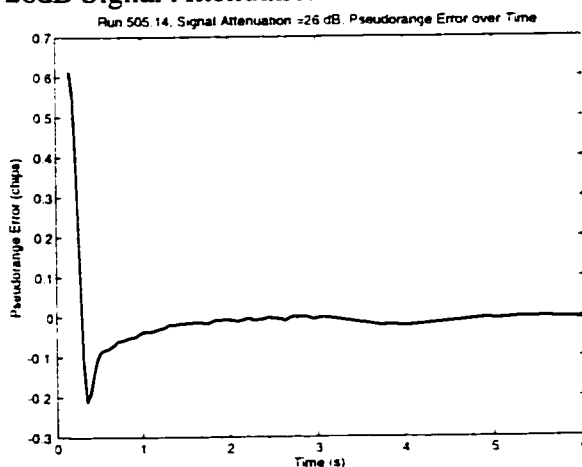


Figure 505.14.c: Pseudorange Error, 26dB Signal Attenuation

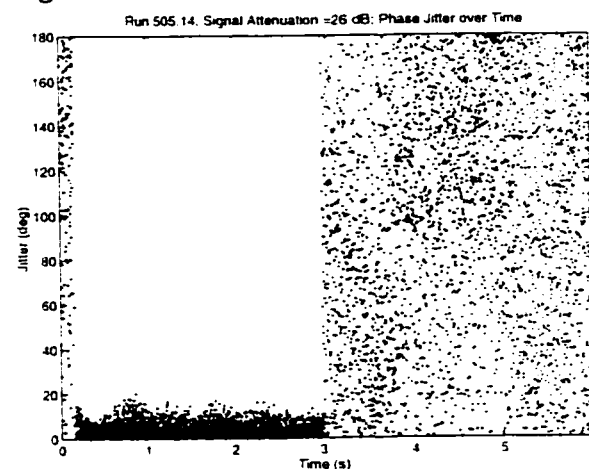


Figure 505.14.d: Carrier Phase Jitter, 26dB Signal Attenuation

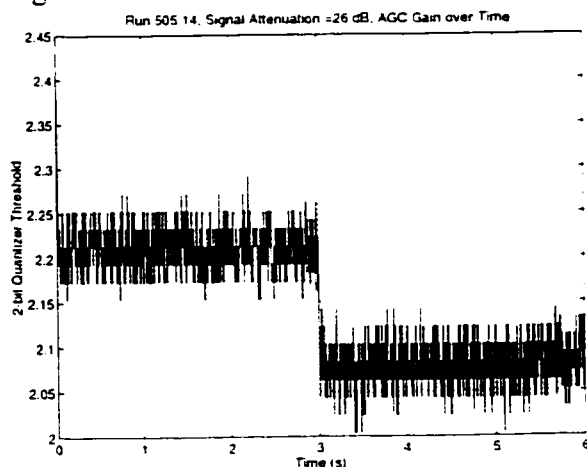


Figure 505.14.e: AGC, 26dB Signal Attenuation

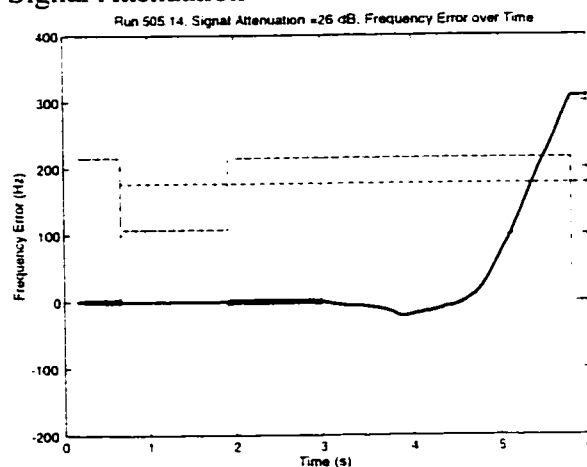


Figure 505.14.f: Frequency Error, 26dB Signal Attenuation

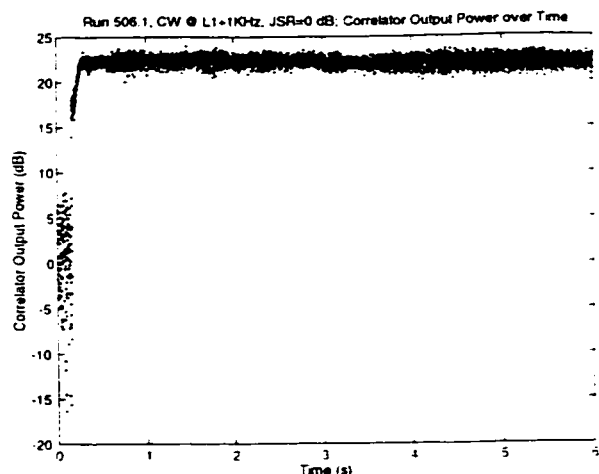


Figure 506.1.a: Correlator Output Power, 0dB CW @ L1+1KHz

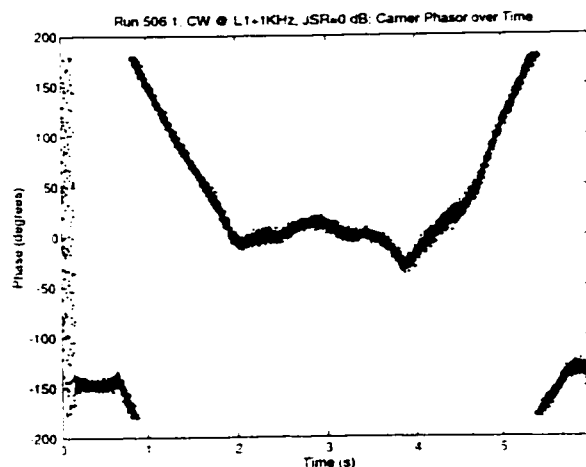


Figure 506.1.b: Carrier Phase, 0dB CW @ L1+1KHz

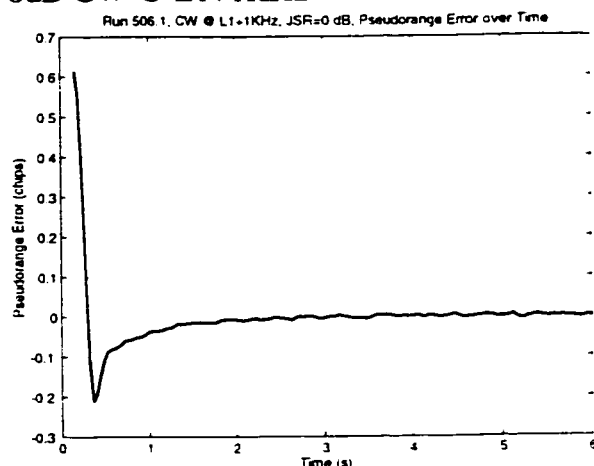


Figure 506.1.c: Pseudorange Error, 0dB CW @ L1+1KHz

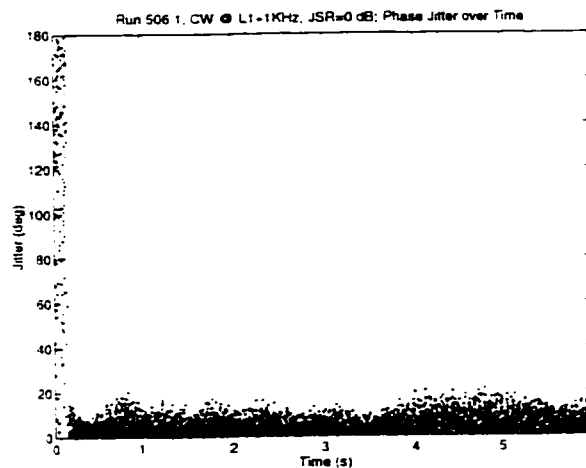


Figure 506.1.d: Carrier Phase Jitter, 0dB CW @ L1+1KHz

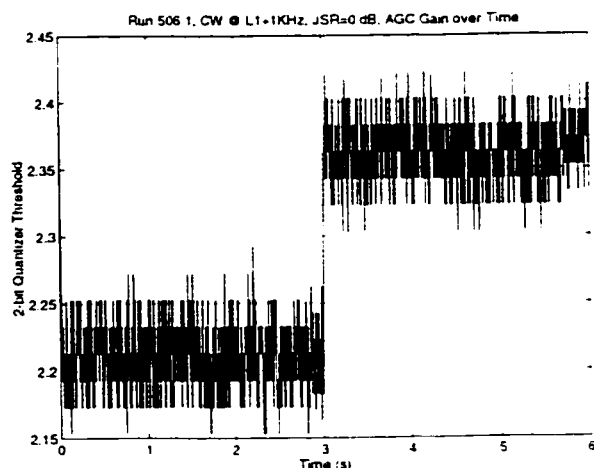


Figure 506.1.e: AGC, 0dB CW @ L1+1KHz

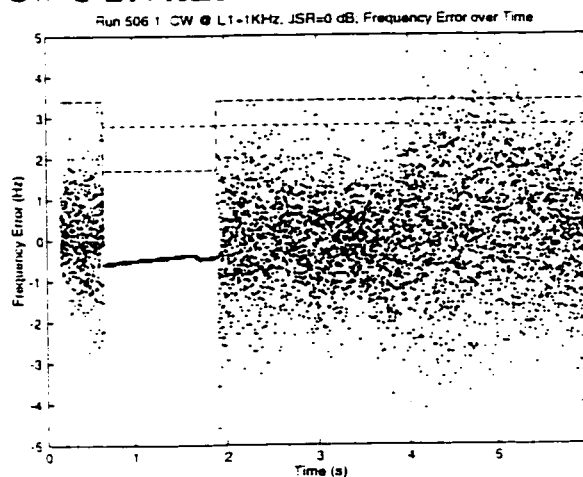


Figure 506.1.f: Frequency Error, 0dB CW @ L1+1KHz

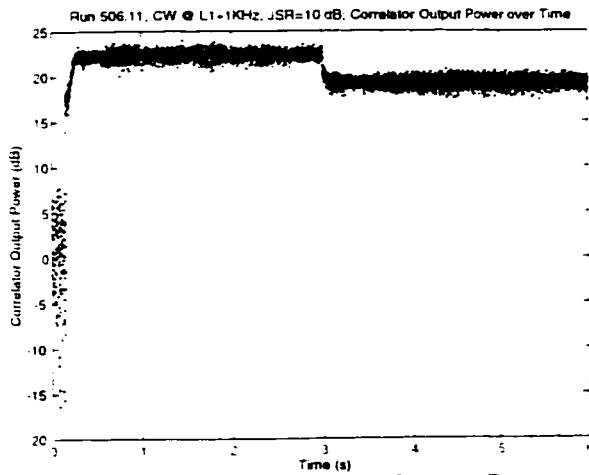


Figure 506.11.a: Correlator Output Power, 10dB CW @ L1+1KHz

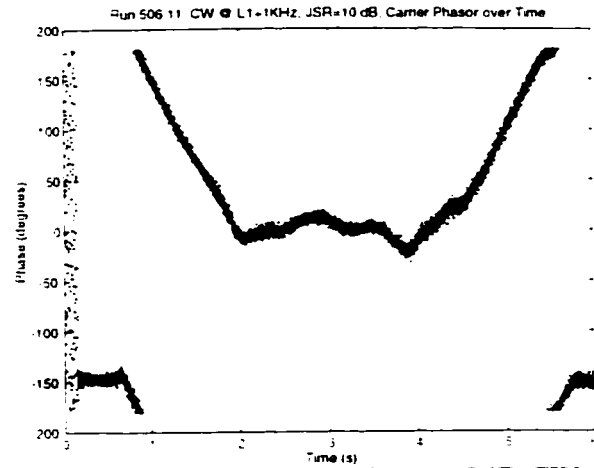


Figure 506.11.b: Carrier Phase, 10dB CW @ L1+1KHz

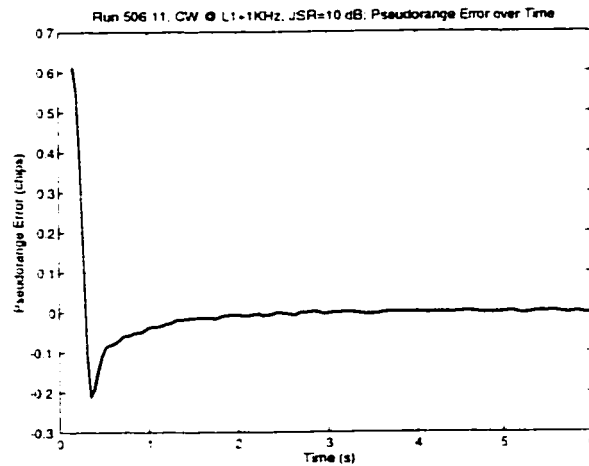


Figure 506.11.c: Pseudorange Error, 10dB CW @ L1+1KHz

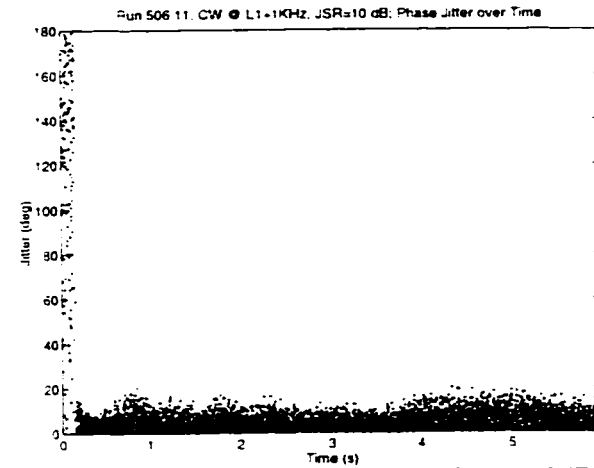


Figure 506.11.d: Carrier Phase Jitter, 10dB CW @ L1+1KHz

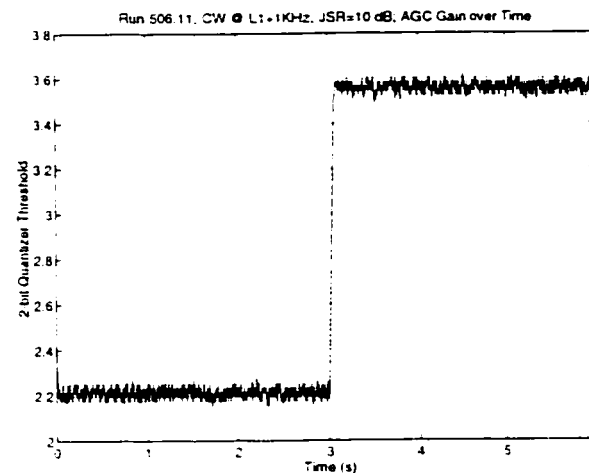


Figure 506.11.e: AGC, 10dB CW @ L1+1KHz

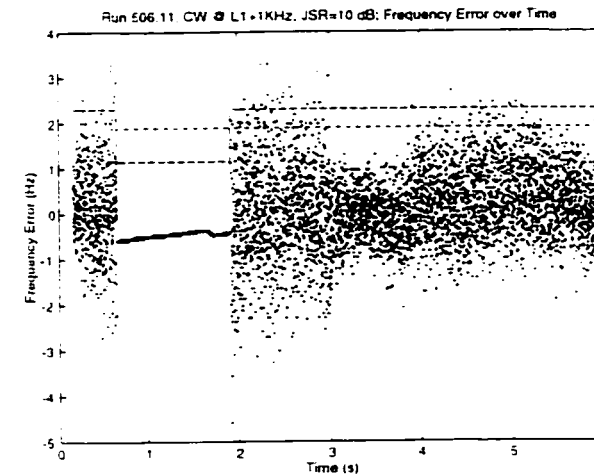


Figure 506.11.f: Frequency Error, 10dB CW @ L1+1KHz

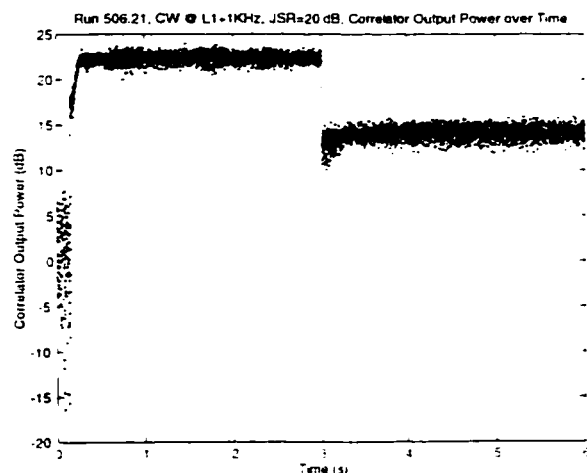


Figure 506.21.a: Correlator Output Power, 20dB CW @ L1+1KHz

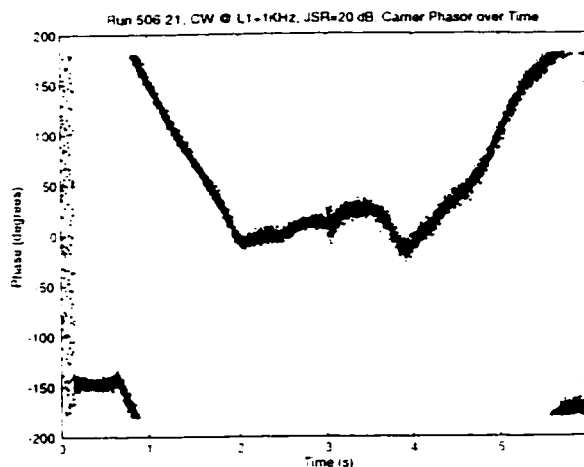


Figure 506.21.b: Carrier Phase, 20dB CW @ L1+1KHz

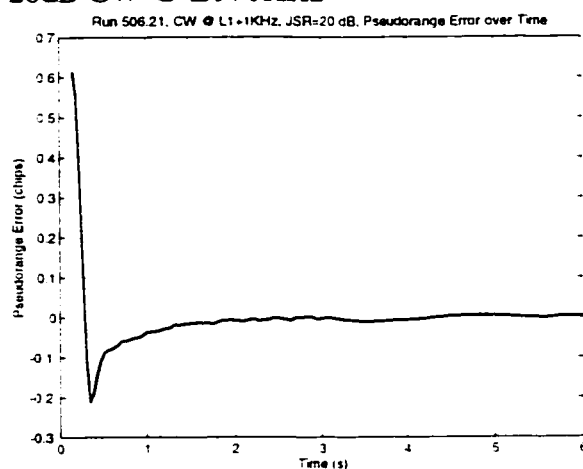


Figure 506.21.c: Pseudorange Error, 20dB CW @ L1+1KHz

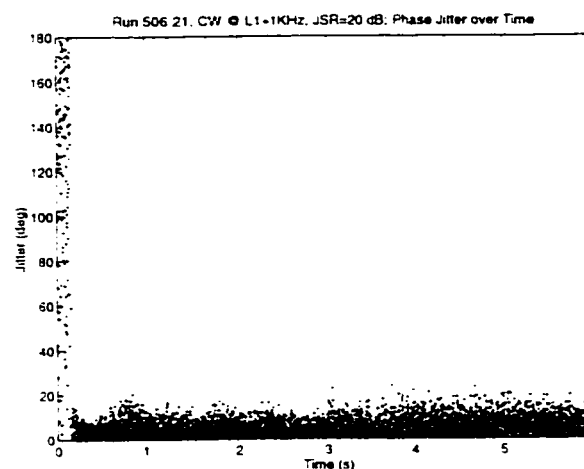


Figure 506.21.d: Carrier Phase Jitter, 20dB CW @ L1+1KHz

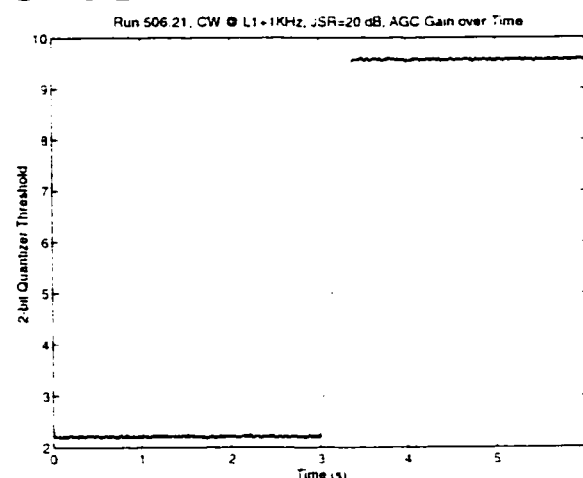


Figure 506.21.e: AGC, 20dB CW @ L1+1KHz

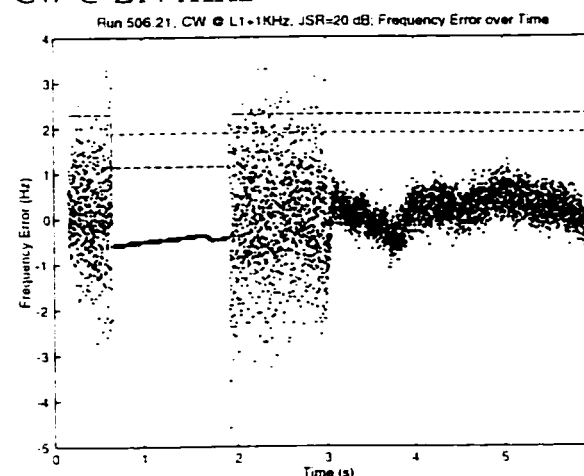


Figure 506.21.f: Frequency Error, 20dB CW @ L1+1KHz

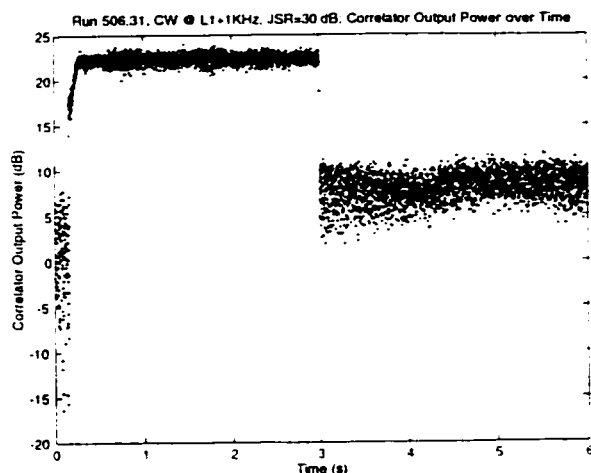


Figure 506.31.a: Correlator Output Power, 30dB CW @ L1+1KHz

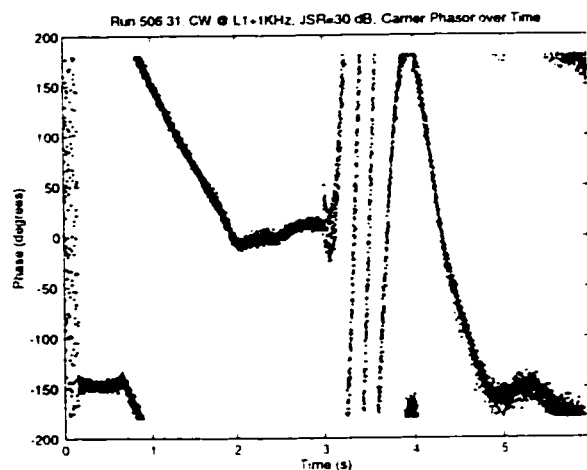


Figure 506.31.b: Carrier Phase, 30dB CW @ L1+1KHz

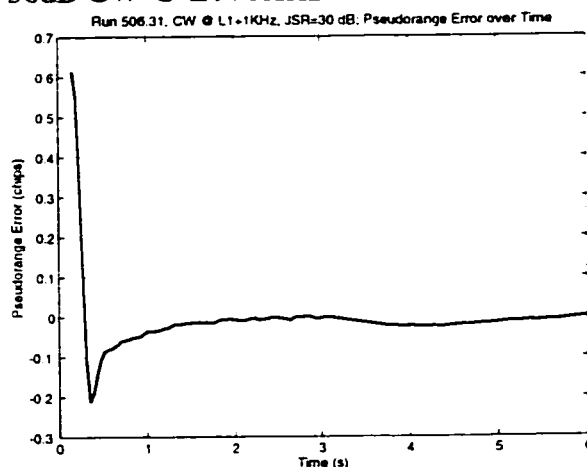


Figure 506.31.c: Pseudorange Error, 30dB CW @ L1+1KHz

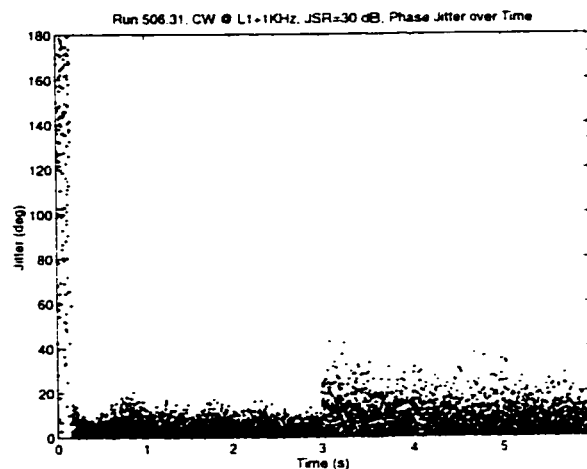


Figure 506.31.d: Carrier Phase Jitter, 30dB CW @ L1+1KHz

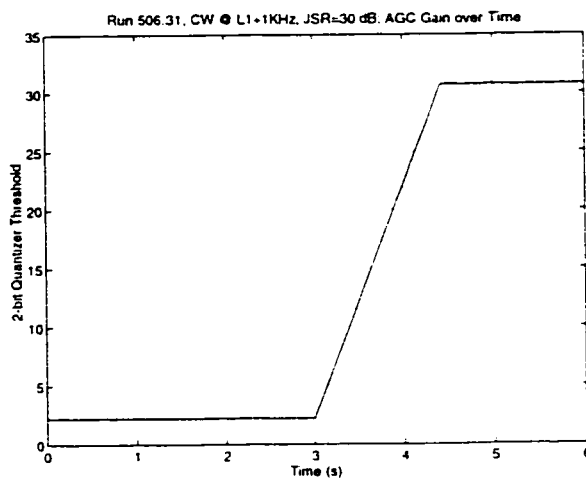


Figure 506.31.e: AGC, 30dB CW @ L1+1KHz

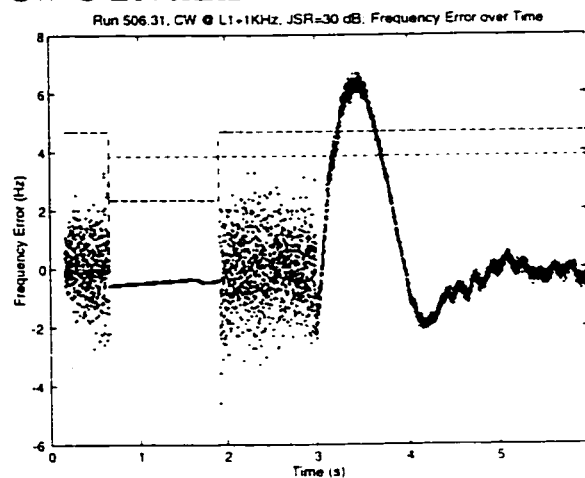


Figure 506.31.f: Frequency Error, 30dB CW @ L1+1KHz

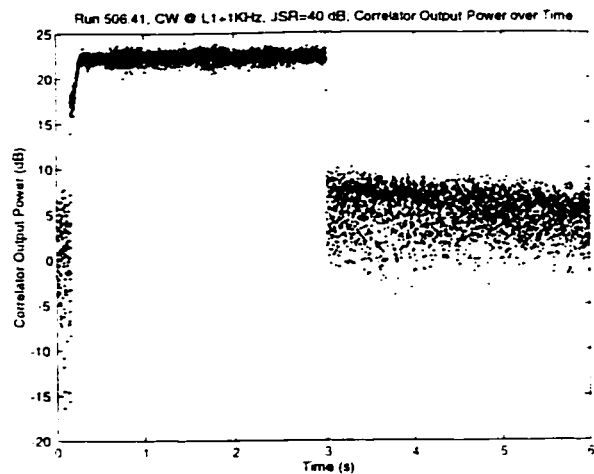


Figure 506.41.a: Correlator Output Power, 40dB CW @ L1+1KHz

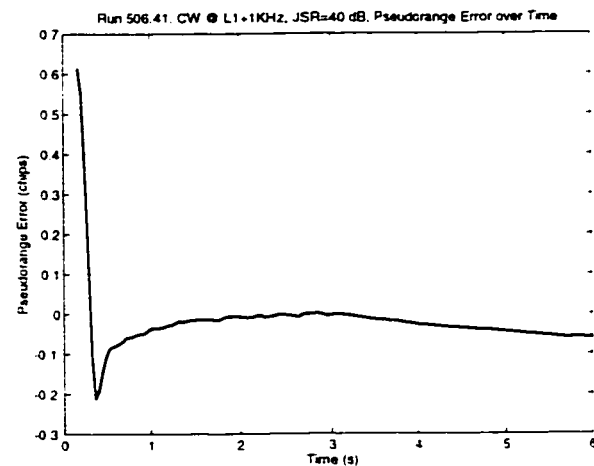


Figure 506.41.c: Pseudorange Error, 40dB CW @ L1+1KHz

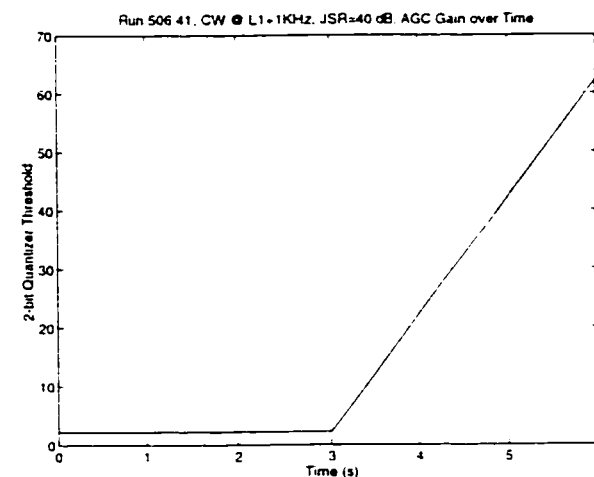


Figure 506.41.e: AGC, 40dB CW @ L1+1KHz

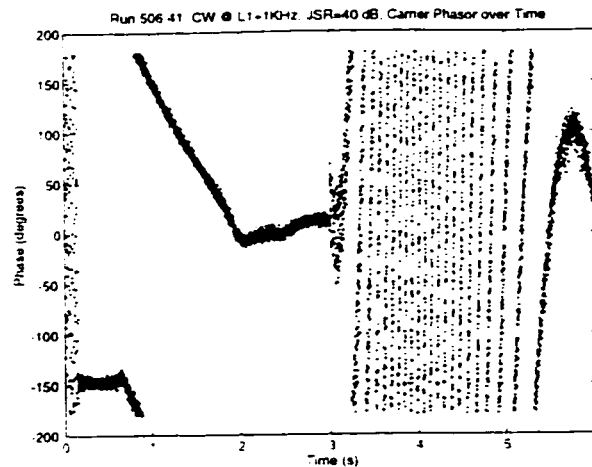


Figure 506.41.b: Carrier Phase, 40dB CW @ L1+1KHz

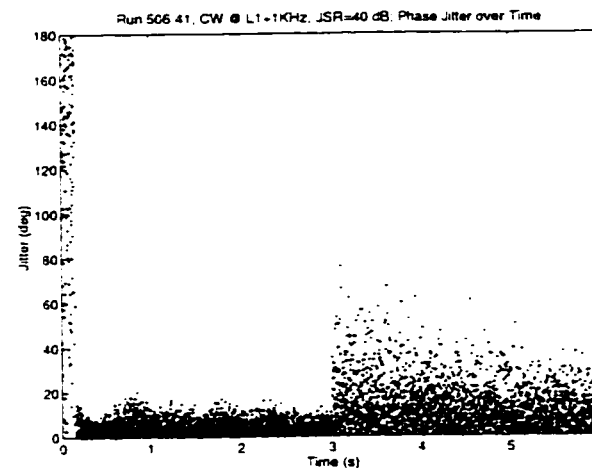


Figure 506.41.d: Carrier Phase Jitter, 40dB CW @ L1+1KHz

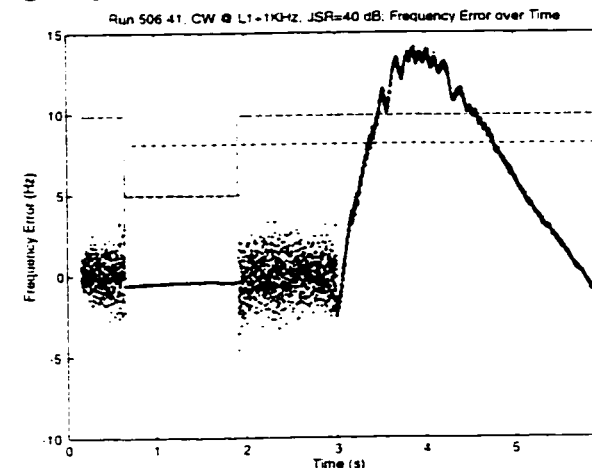


Figure 506.41.f: Frequency Error, 40dB CW @ L1+1KHz

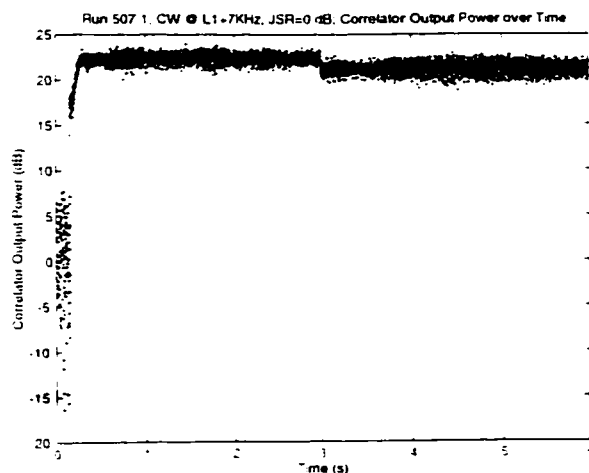


Figure 507.1.a: Correlator Output Power, 0dB CW @ L1+7KHz

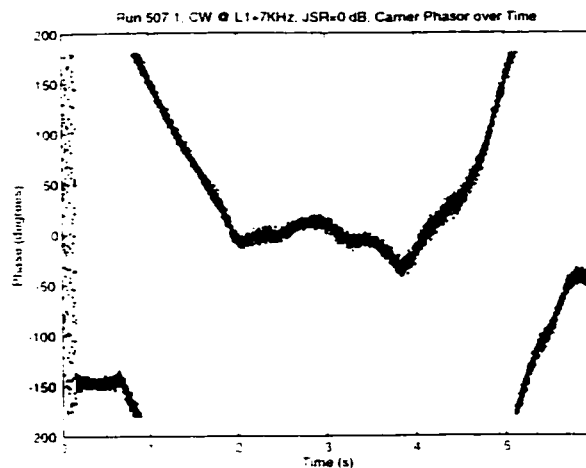


Figure 507.1.b: Carrier Phase, 0dB CW @ L1+7KHz

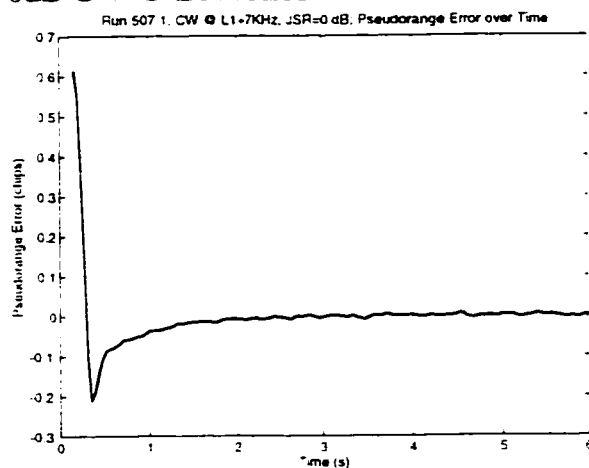


Figure 507.1.c: Pseudorange Error, 0dB CW @ L1+7KHz

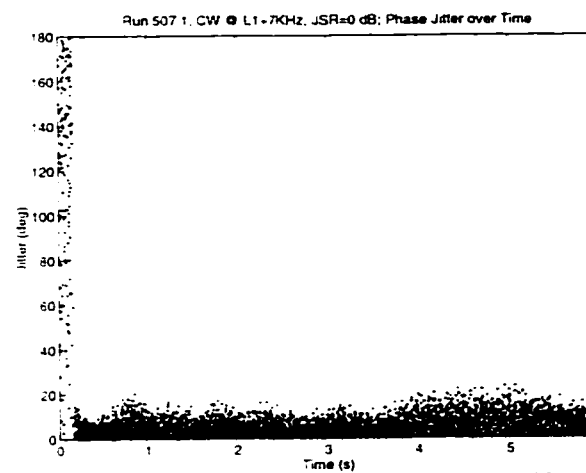


Figure 507.1.d: Carrier Phase Jitter, 0dB CW @ L1+7KHz

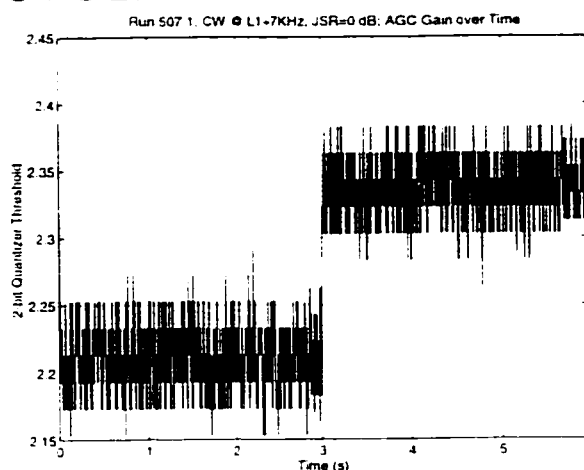


Figure 507.1.e: AGC, 0dB CW @ L1+7KHz

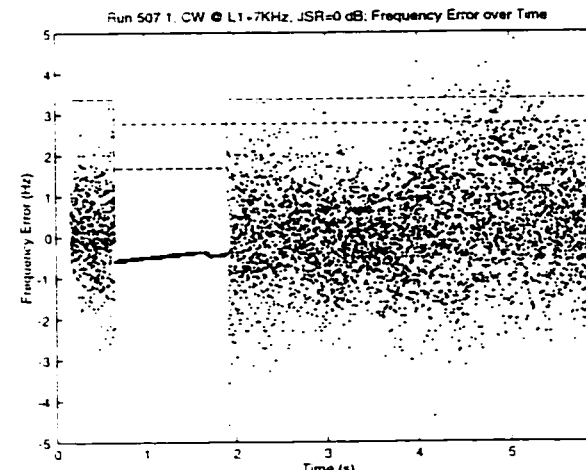


Figure 507.1.f: Frequency Error, 0dB CW @ L1+7KHz

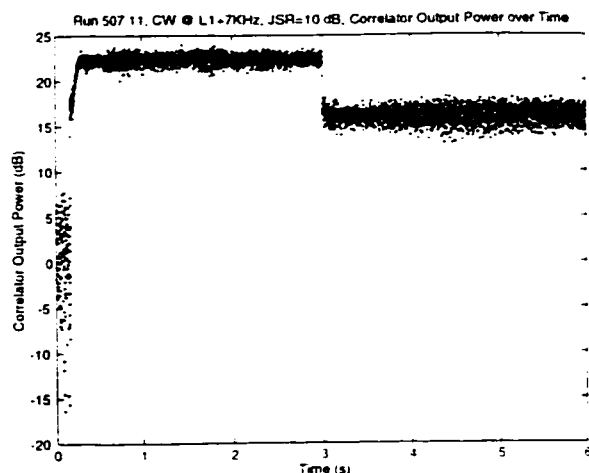


Figure 507.11.a: Correlator Output Power, 10dB CW @ L1+7KHz

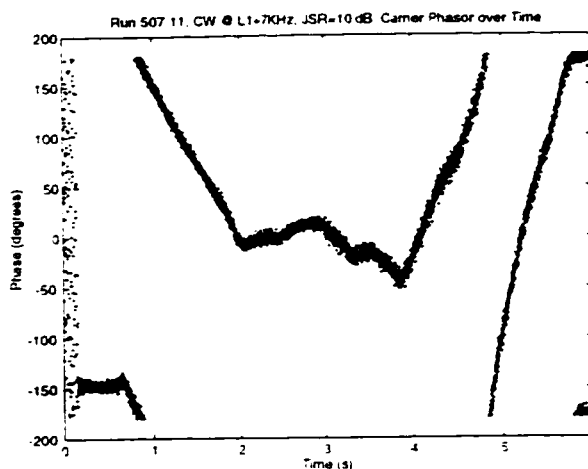


Figure 507.11.b: Carrier Phase, 10dB CW @ L1+7KHz

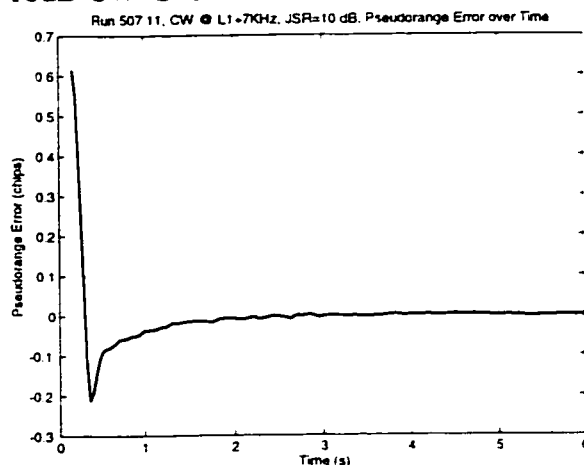


Figure 507.11.c: Pseudorange Error, 10dB CW @ L1+7KHz

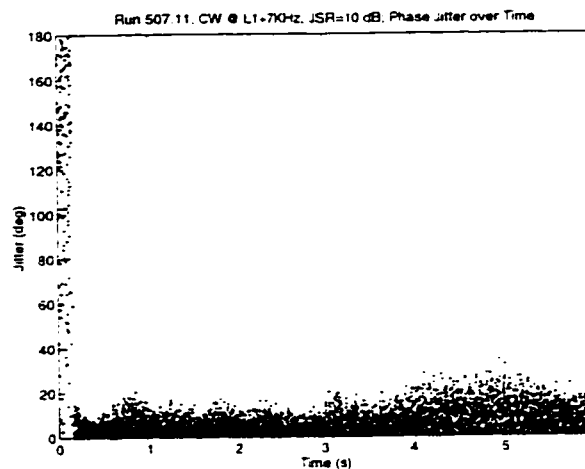


Figure 507.11.d: Carrier Phase Jitter, 10dB CW @ L1+7KHz

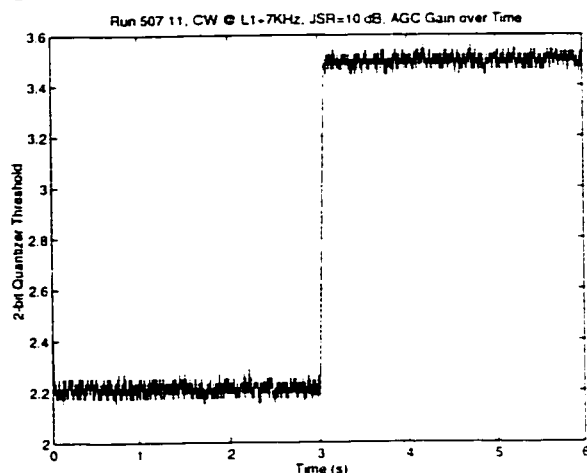


Figure 507.11.e: AGC, 10dB CW @ L1+7KHz

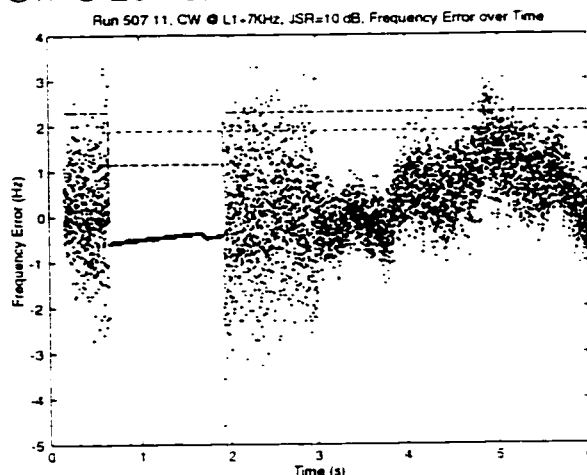


Figure 507.11.f: Frequency Error, 10dB CW @ L1+7KHz

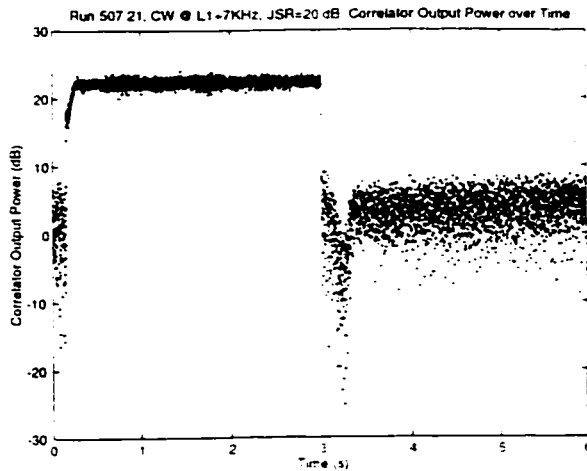


Figure 507.21.a: Correlator Output Power, 20dB CW @ L1+7KHz

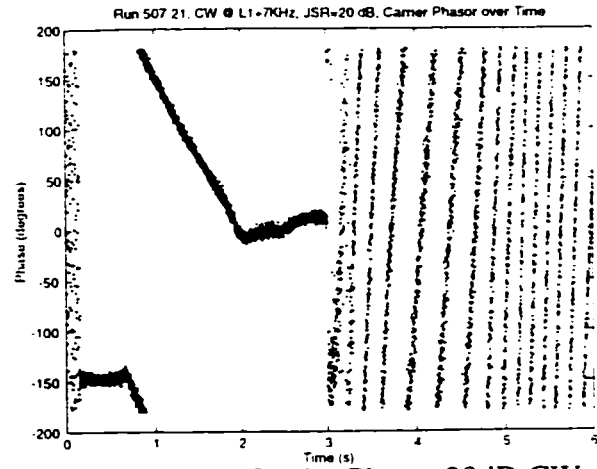


Figure 507.21.b: Carrier Phase, 20dB CW @ L1+7KHz

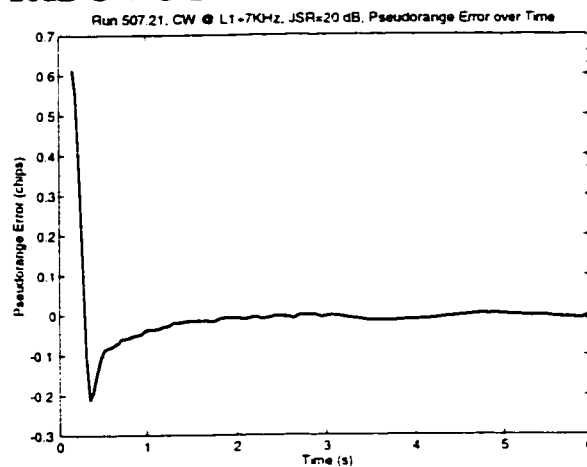


Figure 507.21.c: Pseudorange Error, 20dB CW @ L1+7KHz

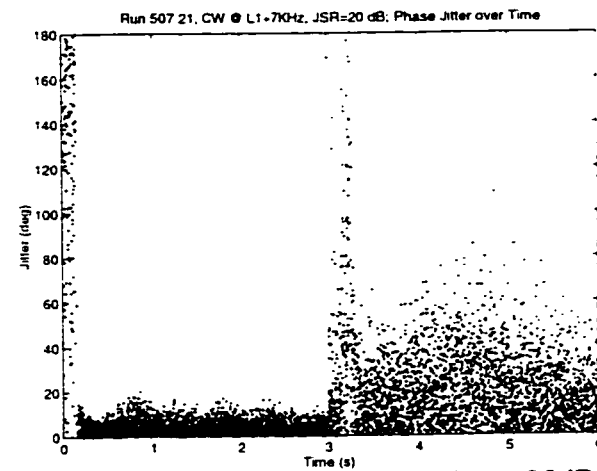


Figure 507.21.d: Carrier Phase Jitter, 20dB CW @ L1+7KHz

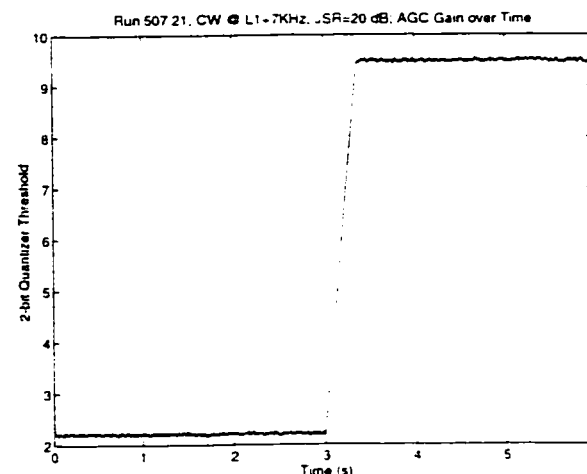


Figure 507.21.e: AGC, 20dB CW @ L1+7KHz

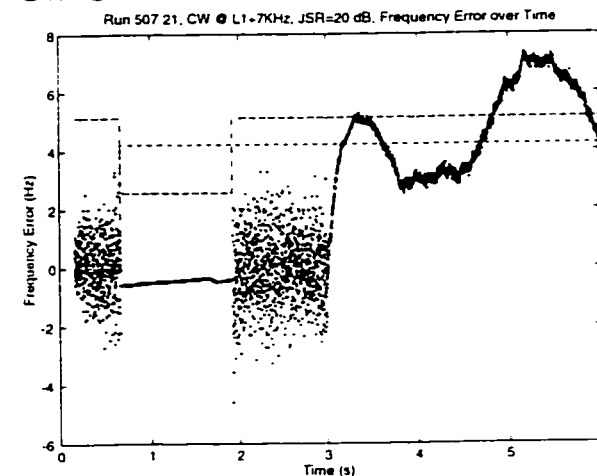


Figure 507.21.f: Frequency Error, 20dB CW @ L1+7KHz

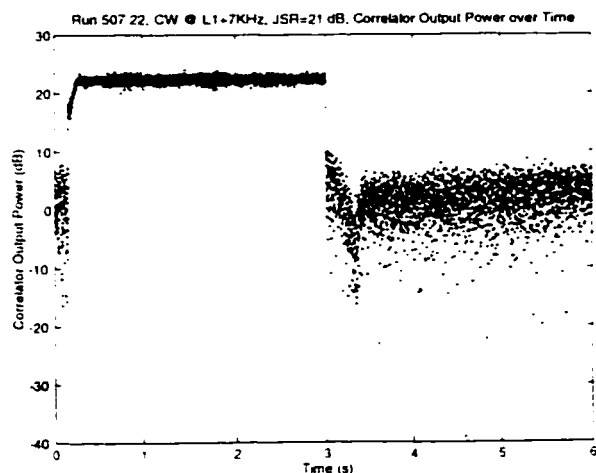


Figure 507.22.a: Correlator Output Power, 21dB CW @ L1+7KHz

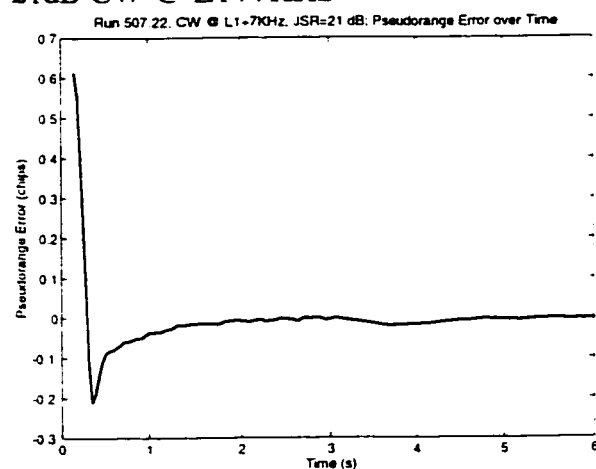


Figure 507.22.c: Pseudorange Error, 21dB CW @ L1+7KHz

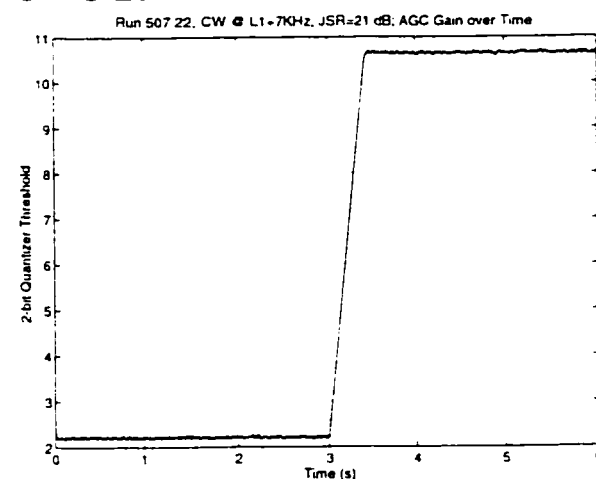


Figure 507.22.e: AGC, 21dB CW @ L1+7KHz

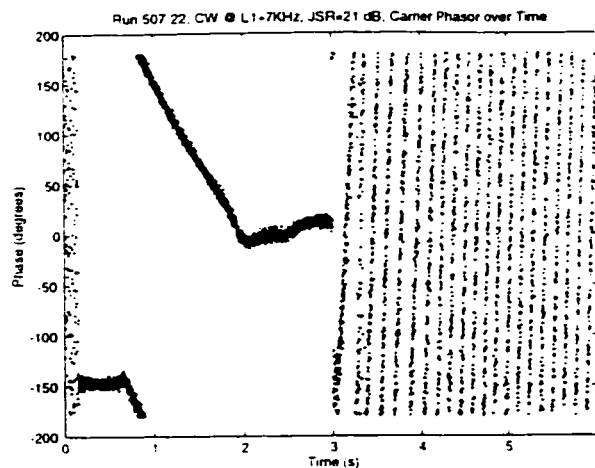


Figure 507.22.b: Carrier Phase, 21dB CW @ L1+7KHz

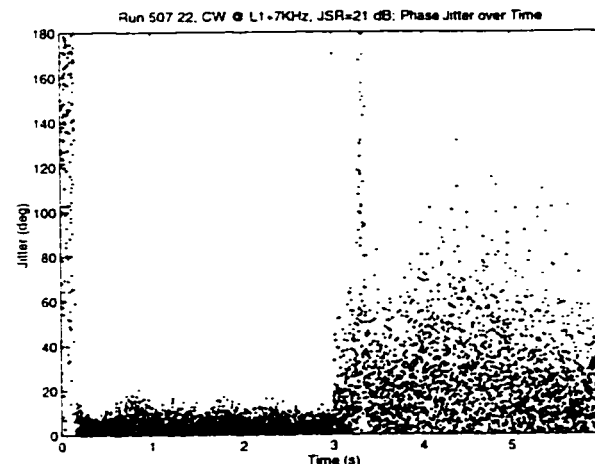


Figure 507.22.d: Carrier Phase Jitter, 21dB CW @ L1+7KHz

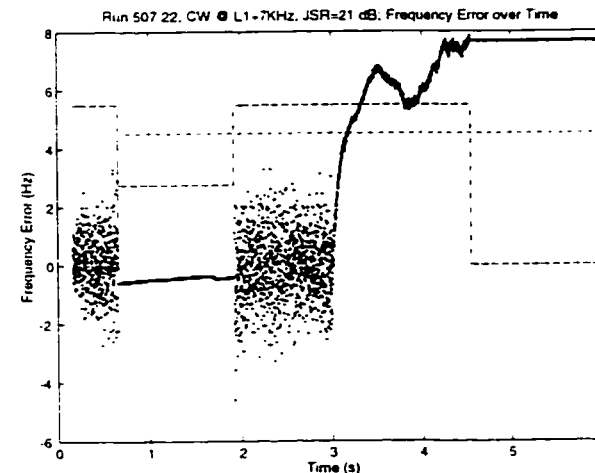


Figure 507.22.f: Frequency Error, 21dB CW @ L1+7KHz

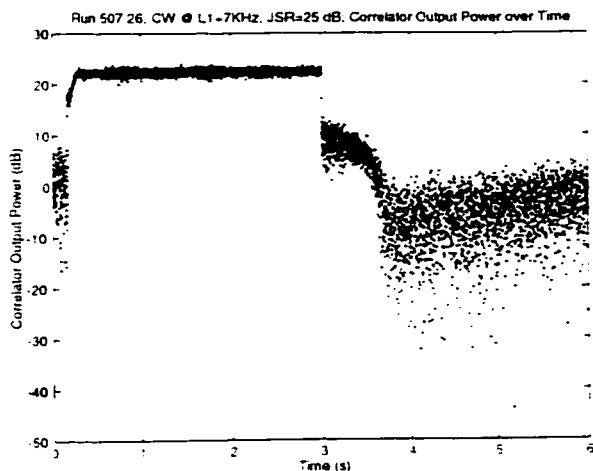


Figure 507.26.a: Correlator Output Power, 25dB CW @ L1+7KHz

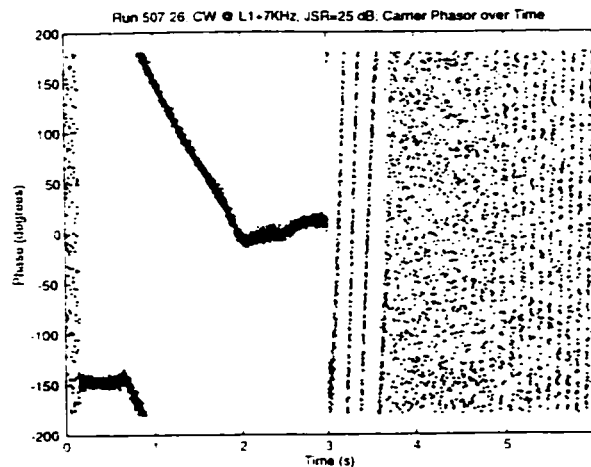


Figure 507.26.b: Carrier Phase, 25dB CW @ L1+7KHz

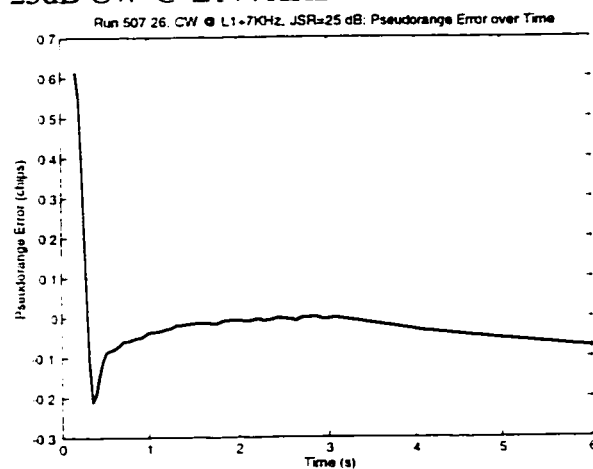


Figure 507.26.c: Pseudorange Error, 25dB CW @ L1+7KHz

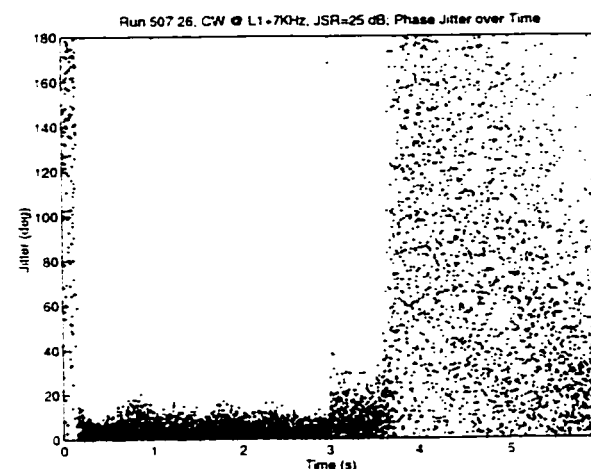


Figure 507.26.d: Carrier Phase Jitter, 25dB CW @ L1+7KHz

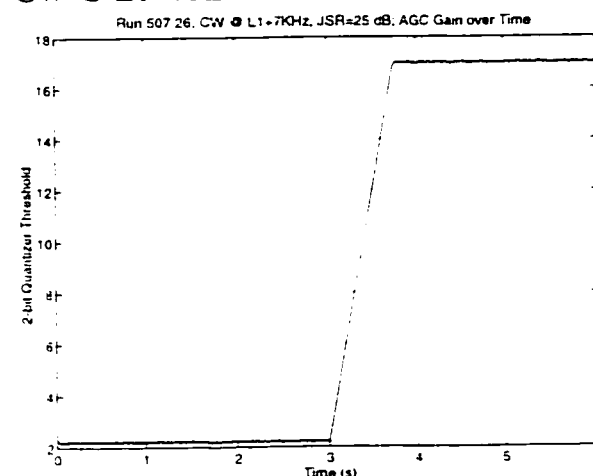


Figure 507.26.e: AGC, 25dB CW @ L1+7KHz

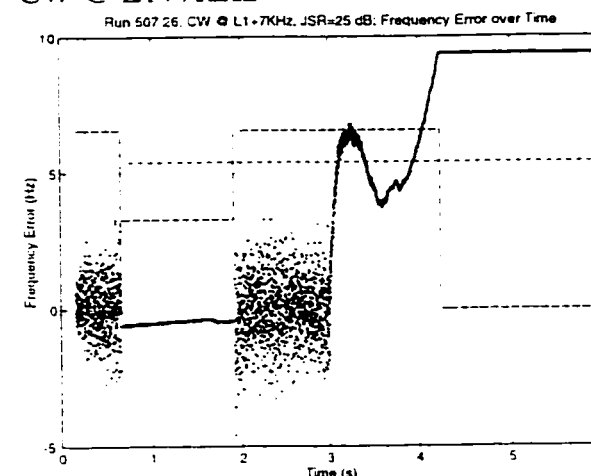


Figure 507.26.f: Frequency Error, 25dB CW @ L1+7KHz

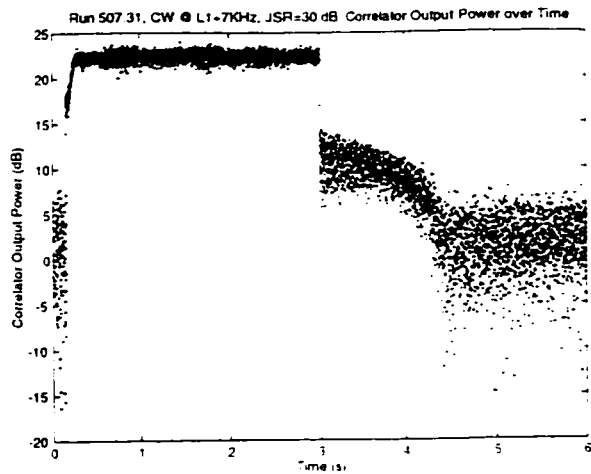


Figure 507.31.a: Correlator Output Power, 30dB CW @ L1+7KHz

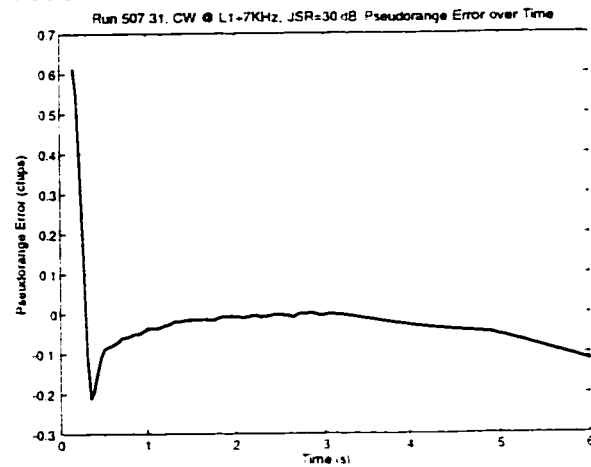


Figure 507.31.c: Pseudorange Error, 30dB CW @ L1+7KHz

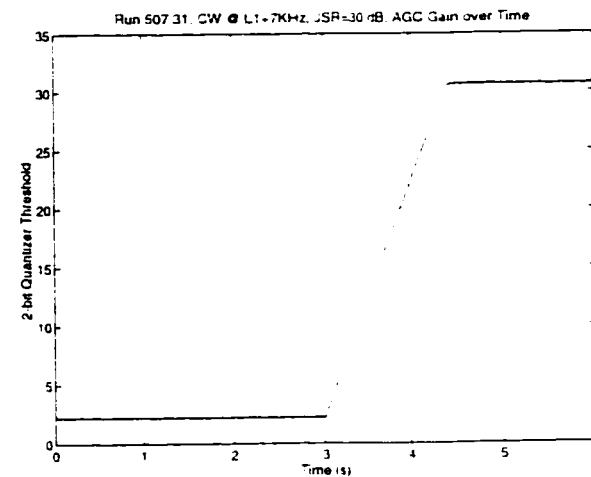


Figure 507.31.e: AGC, 30dB CW @ L1+7KHz

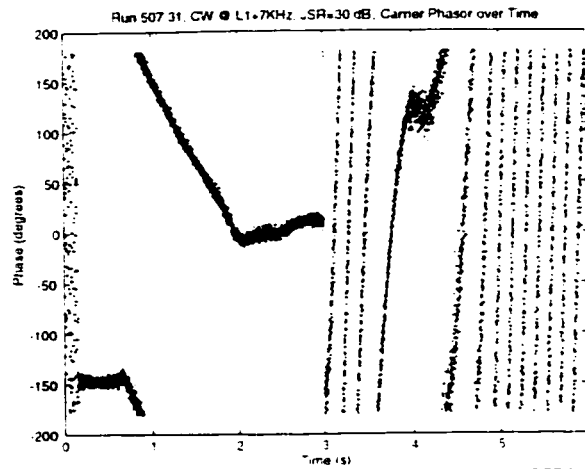


Figure 507.31.b: Carrier Phase, 30dB CW @ L1+7KHz

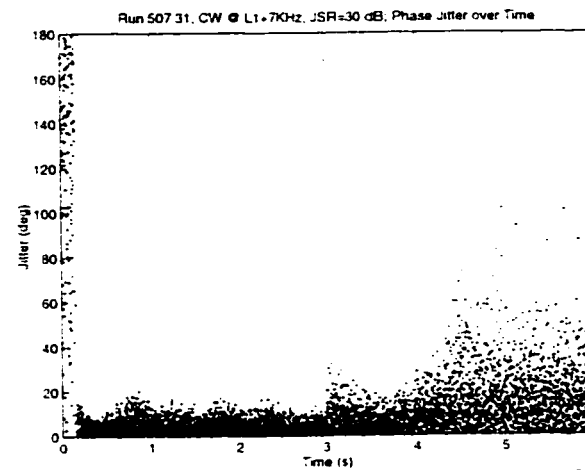


Figure 507.31.d: Carrier Phase Jitter, 30dB CW @ L1+7KHz

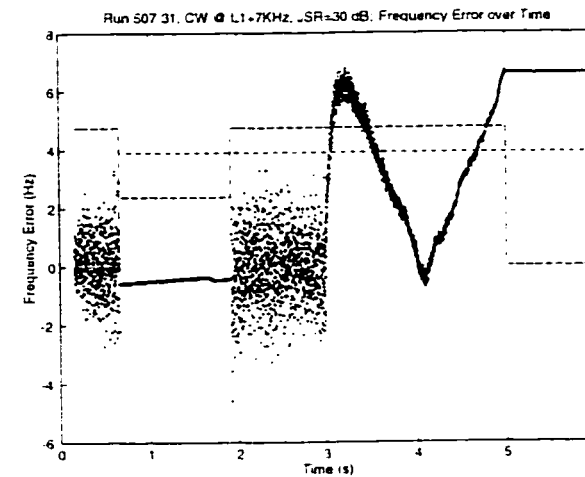


Figure 507.31.f: Frequency Error, 30dB CW @ L1+7KHz

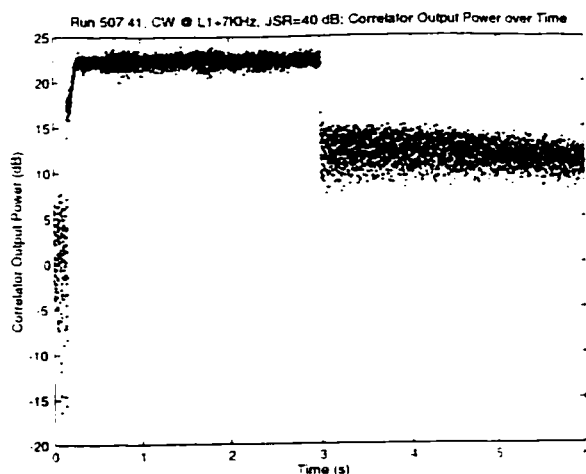


Figure 507.41.a: Correlator Output Power. 40dB CW @ L1+7KHz

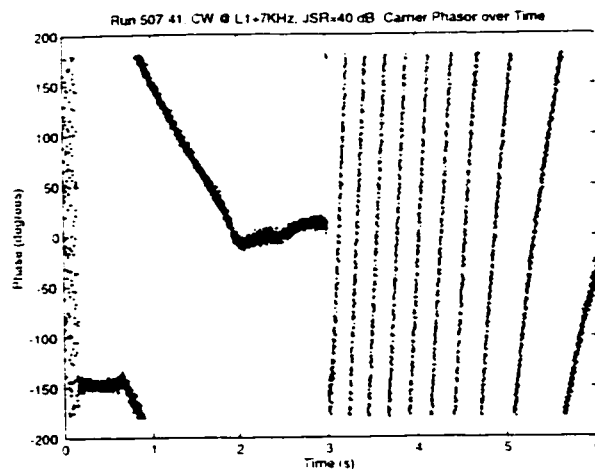


Figure 507.41.b: Carrier Phase. 40dB CW @ L1+7KHz

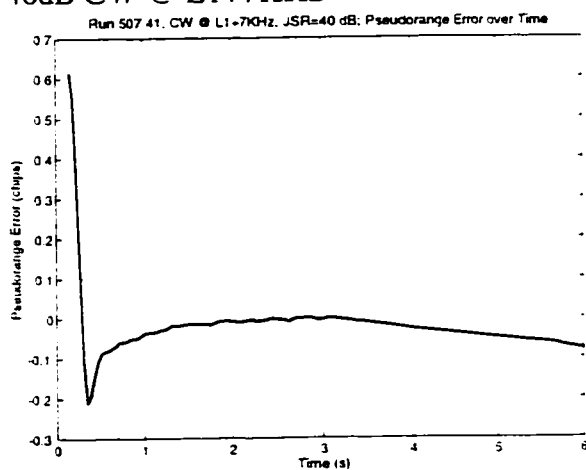


Figure 507.41.c: Pseudorange Error. 40dB CW @ L1+7KHz

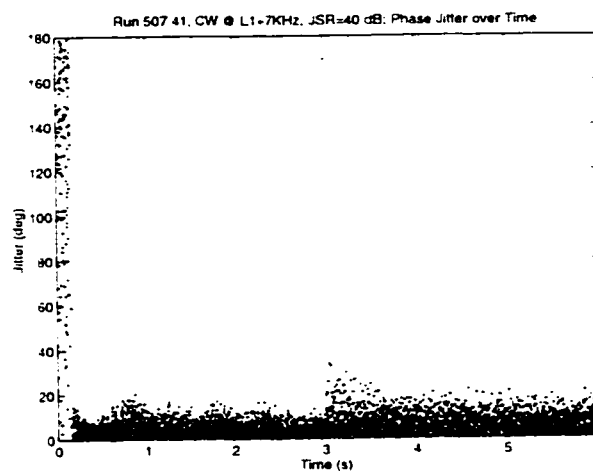


Figure 507.41.d: Carrier Phase Jitter. 40dB CW @ L1+7KHz

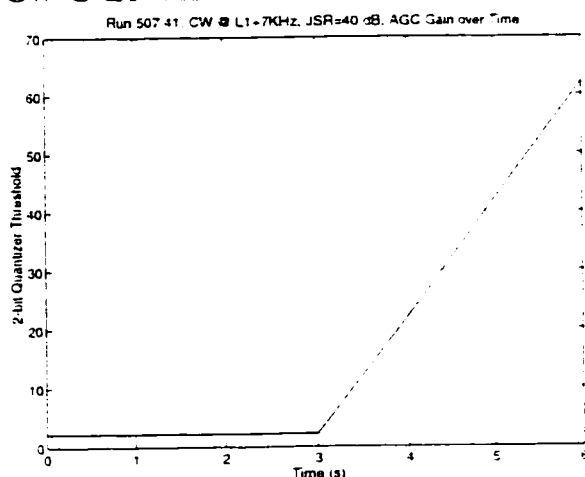


Figure 507.41.e: AGC, 40dB CW @ L1+7KHz

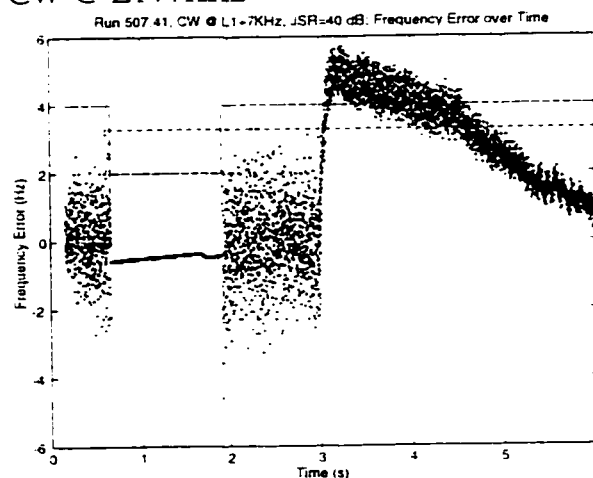


Figure 507.41.f: Frequency Error. 40dB CW @ L1+7KHz

Appendix B

Bench Test Results

This section contains time histories of bench tests performed on a GEC Plessey GPS receiver, as described in section 4.4. For each test, there are 11 sets of plots, labeled as:

Figure XYZ.a ... j

where XYZ is a three digit unique identifier for each test. XYZ=401, 402, ...409 for AWGN tests, and XYZ=421, 422, ...429 for CW interference tests.

The suffixes a through j are plots described as in the table below:

Plot Suffix	Description
a	COP vs. Time
b	COP vs. Applied Interference Power Setting
c	COP- σ vs. Time
d	COP- σ vs. Applied Interference Power Setting
e	Carrier Phase Jitter vs. Time
f	Carrier Phase Jitter vs. Applied Interference Power Setting
g	Horizontal Position Error vs. Time
h	Vertical Position Error vs. Time
i	Estimated Pseudorange Error vs. Time
j	Number of Tracked Satellites vs. Time
k	Applied Interference Power Setting vs. Time

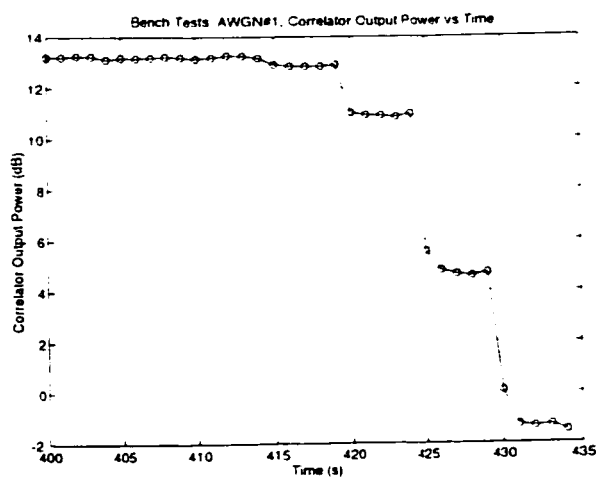


Figure 401.a: AWGN Bench Test:
Correlator Output Power vs Time

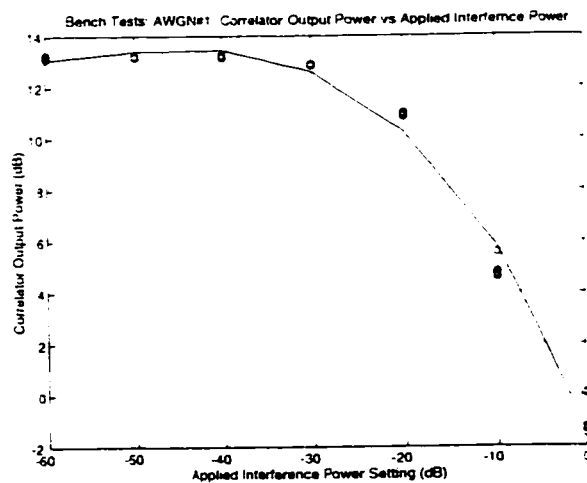


Figure 401.b: AWGN Bench Test:
Correlator Output Power vs Applied
Interference

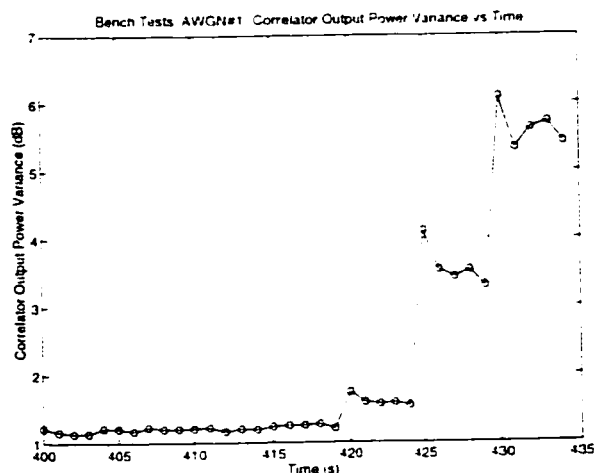


Figure 401.c: AWGN Bench Test:
Correlator Output Power Variance vs Time

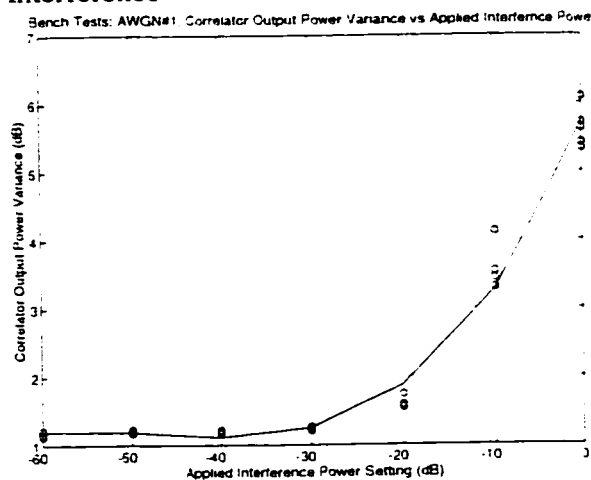


Figure 401.d: AWGN Bench Test:
Correlator Output Power Variance vs
Applied Interference

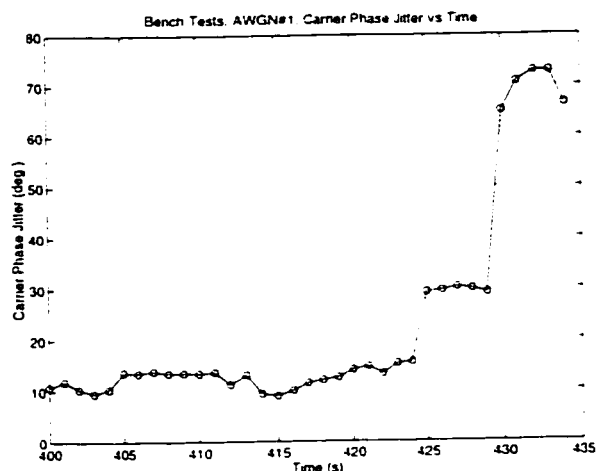


Figure 401.e: AWGN Bench Test: Carrier
Phase Jitter vs Time

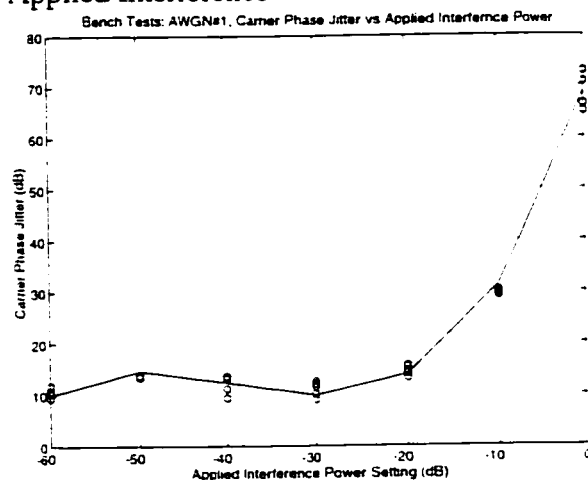


Figure 401.f: AWGN Bench Test: Carrier
Phase Jitter vs Applied Interference

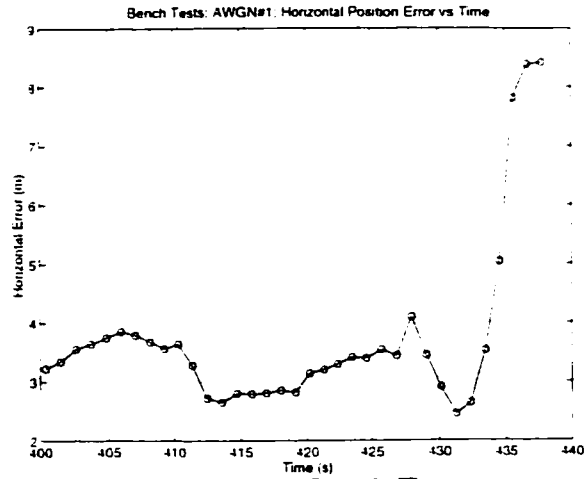


Figure 401.g: AWGN Bench Test: Horizontal Position Error vs Time

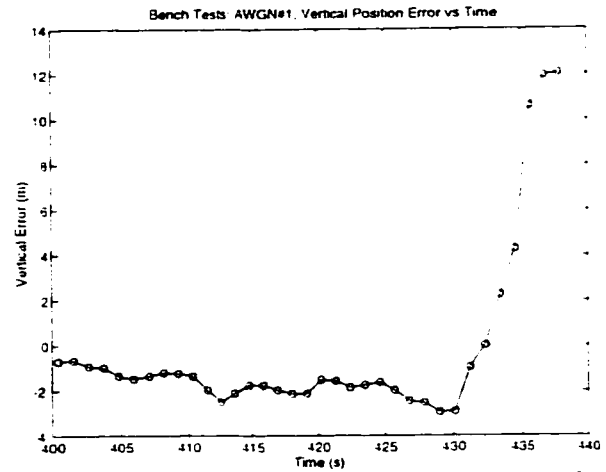


Figure 401.h: AWGN Bench Test: Vertical Position Error vs Time

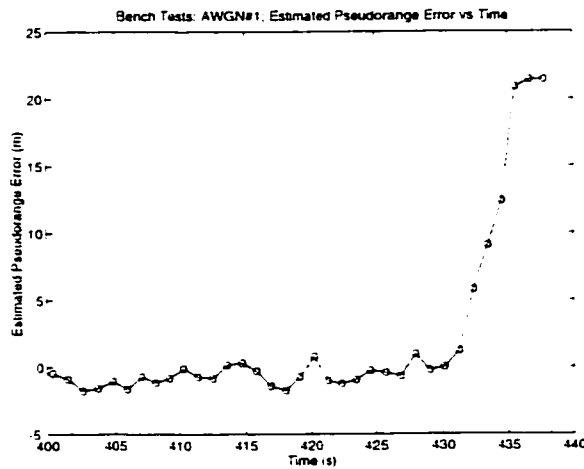


Figure 401.i: AWGN Bench Test: Estimate of Pseudorange Error vs Time

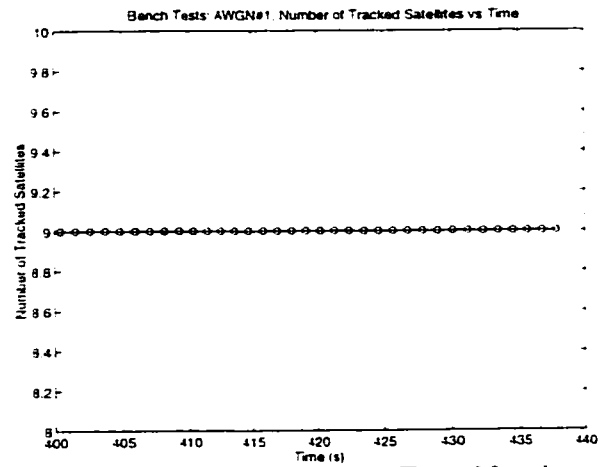


Figure 401.j: AWGN Bench Test: Number of Tracked Satellites vs Time

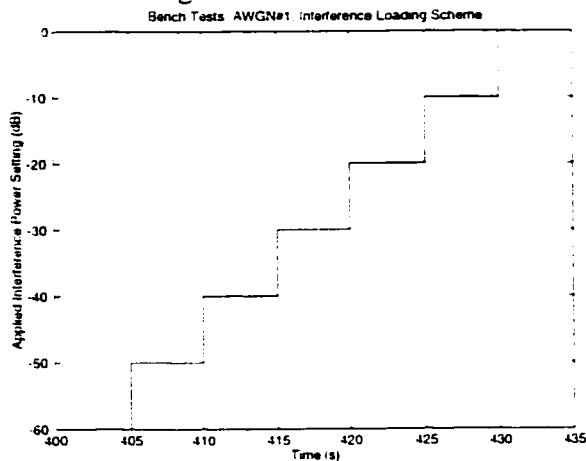


Figure 401.k: AWGN Bench Test: Interference Load Settings vs Time

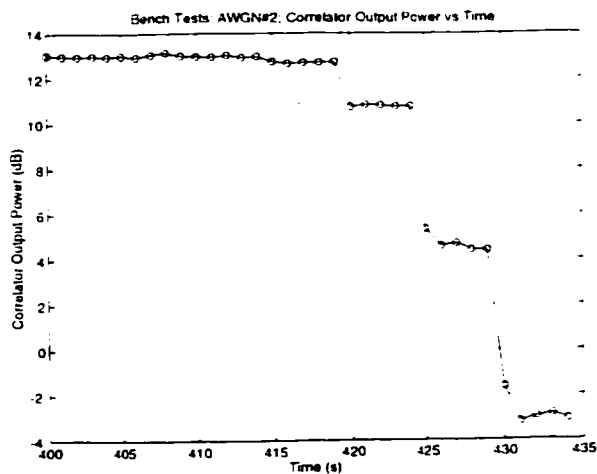


Figure 402.a: AWGN Bench Test:
Correlator Output Power vs Time

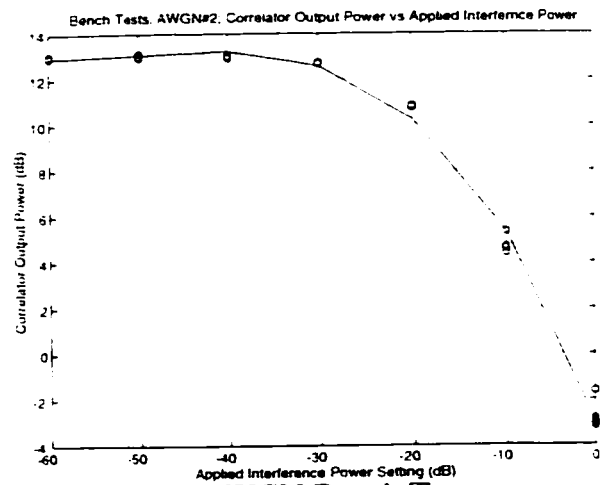


Figure 402.b: AWGN Bench Test:
Correlator Output Power vs Applied
Interference

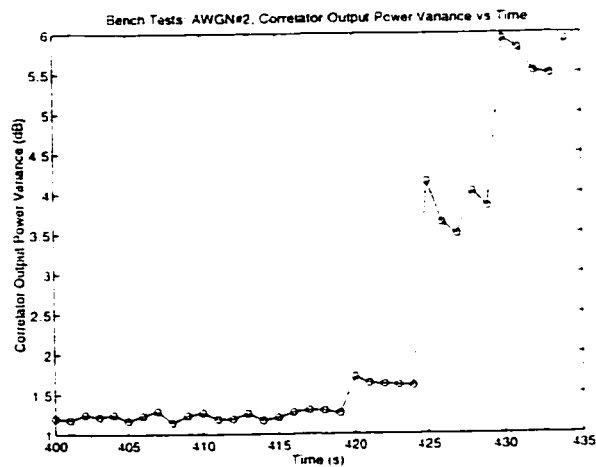


Figure 402.c: AWGN Bench Test:
Correlator Output Power Variance vs Time

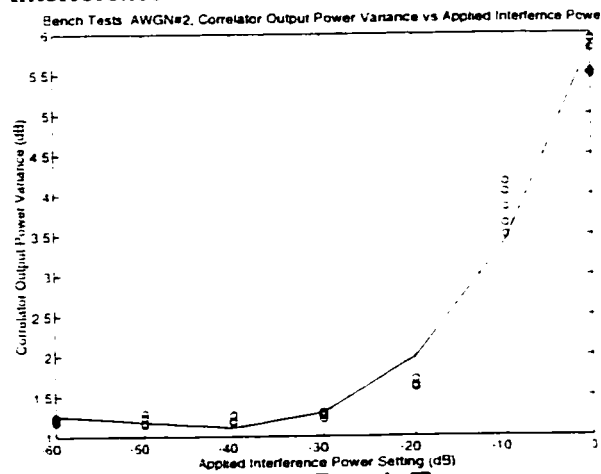


Figure 402.d: AWGN Bench Test:
Correlator Output Power Variance vs
Applied Interference

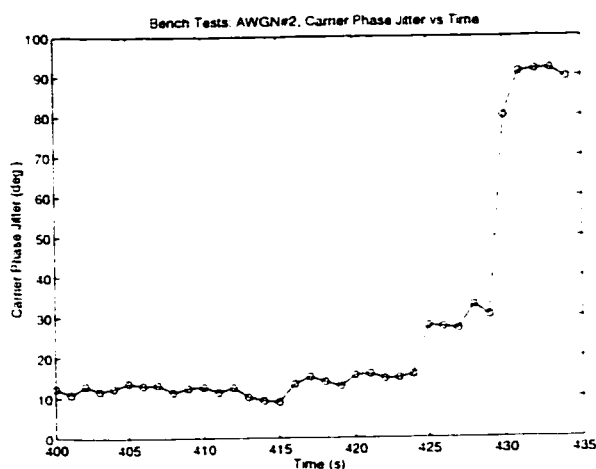


Figure 402.e: AWGN Bench Test: Carrier
Phase Jitter vs Time

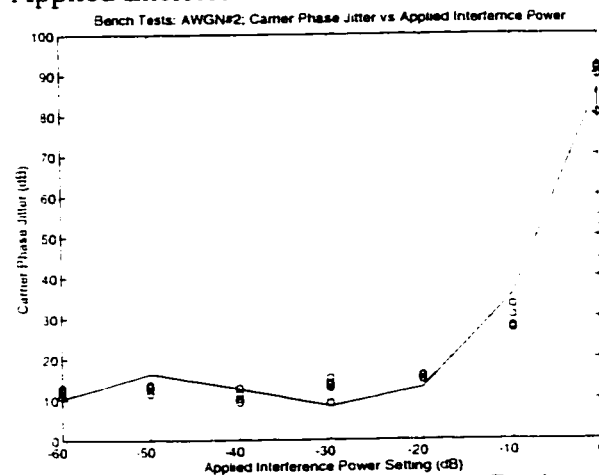


Figure 402.f: AWGN Bench Test: Carrier
Phase Jitter vs Applied Interference

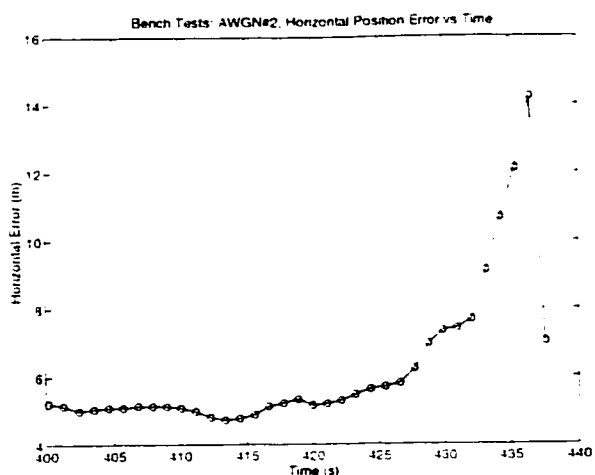


Figure 402.g: AWGN Bench Test: Horizontal Position Error vs Time

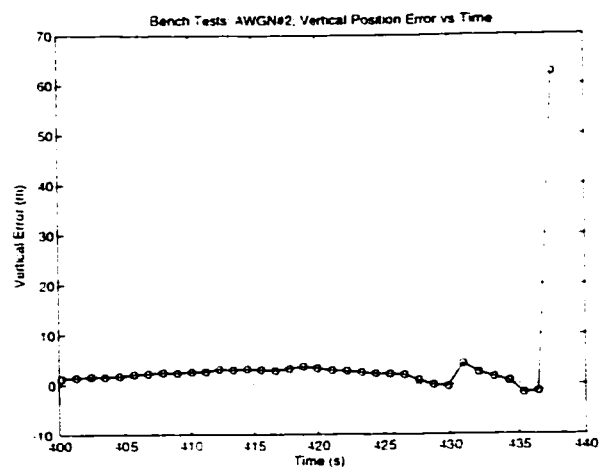


Figure 402.h: AWGN Bench Test: Vertical Position Error vs Time

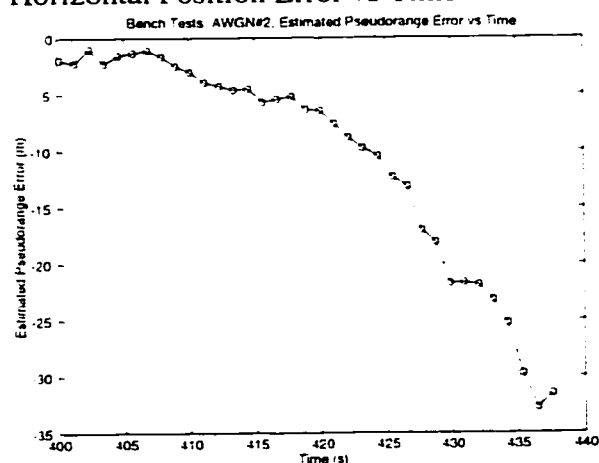


Figure 402.i: AWGN Bench Test: Estimate of Pseudorange Error vs Time

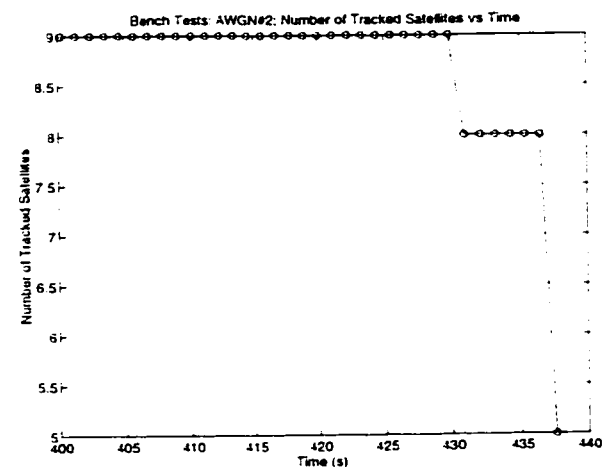


Figure 402.j: AWGN Bench Test: Number of Tracked Satellites vs Time

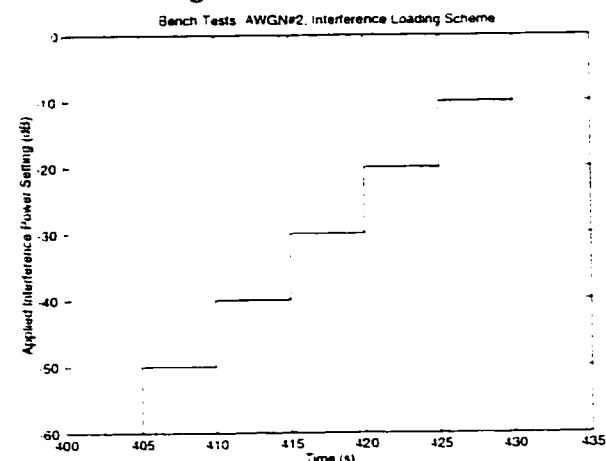


Figure 402.k: AWGN Bench Test: Interference Load Settings vs Time

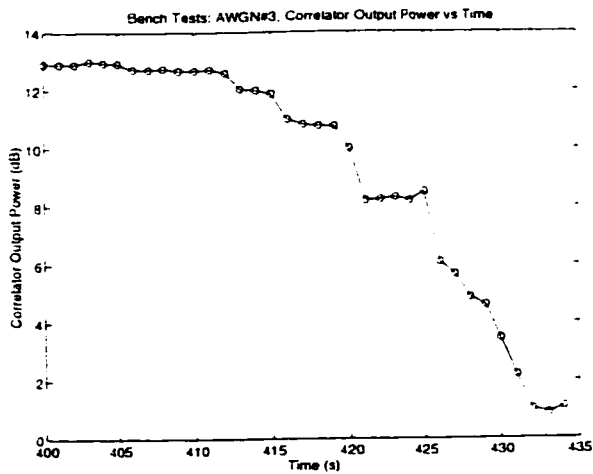


Figure 403.a: AWGN Bench Test:
Correlator Output Power vs Time

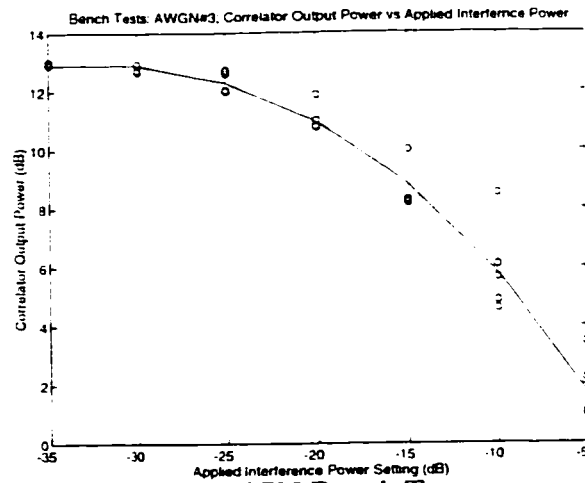


Figure 403.b: AWGN Bench Test:
Correlator Output Power vs Applied
Interference

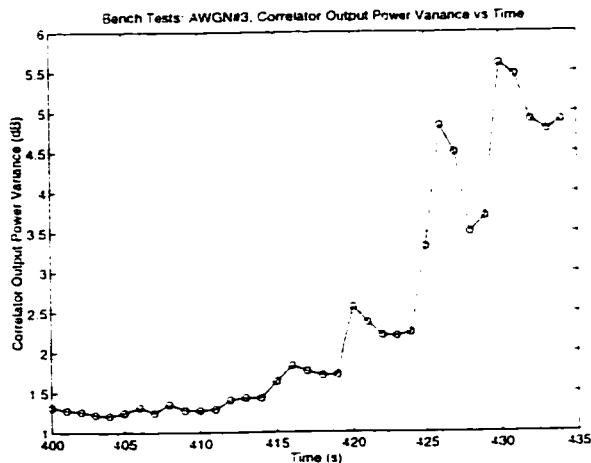


Figure 403.c: AWGN Bench Test:
Correlator Output Power Variance vs Time

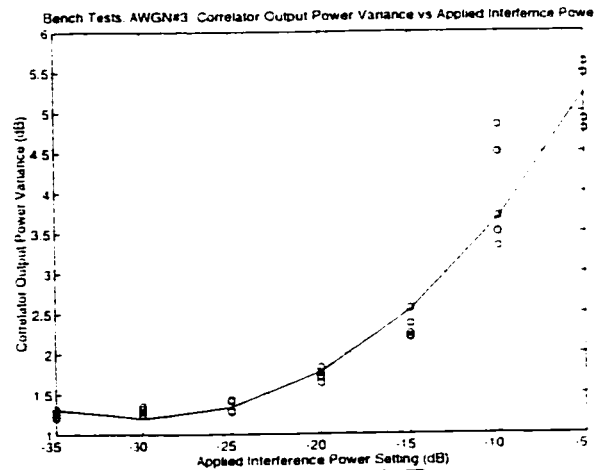


Figure 403.d: AWGN Bench Test:
Correlator Output Power Variance vs
Applied Interference

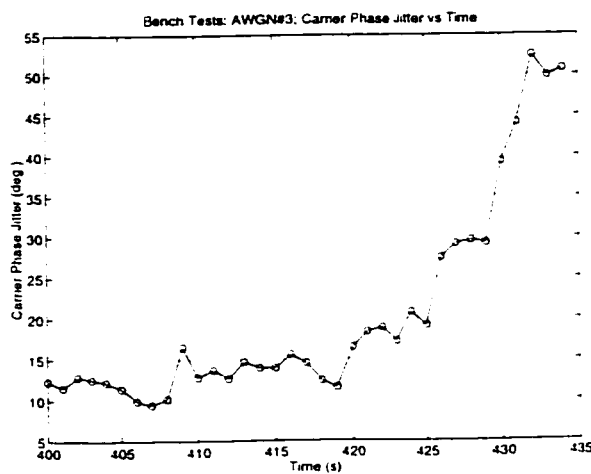


Figure 403.e: AWGN Bench Test: Carrier
Phase Jitter vs Time

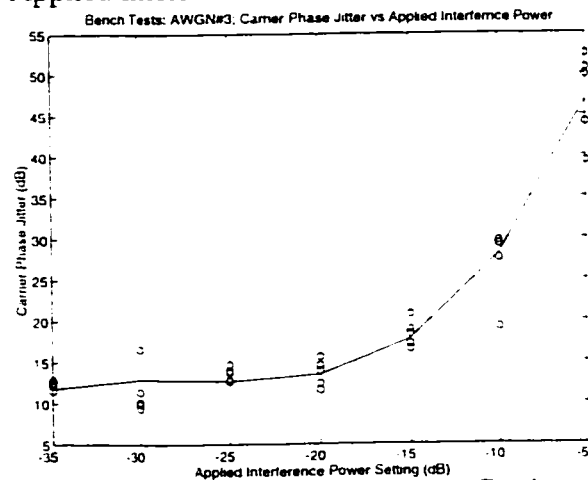


Figure 403.f: AWGN Bench Test: Carrier
Phase Jitter vs Applied Interference

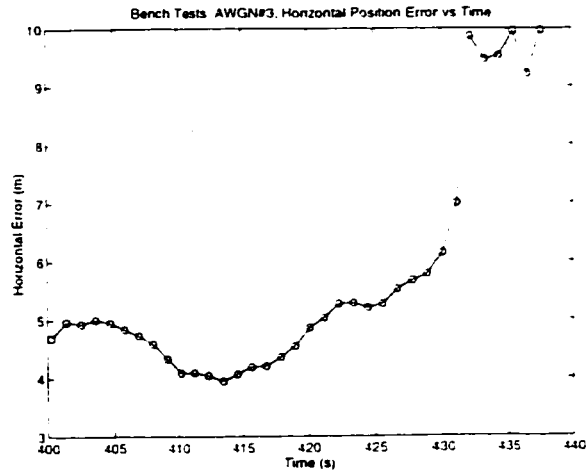


Figure 403.g: AWGN Bench Test: Horizontal Position Error vs Time

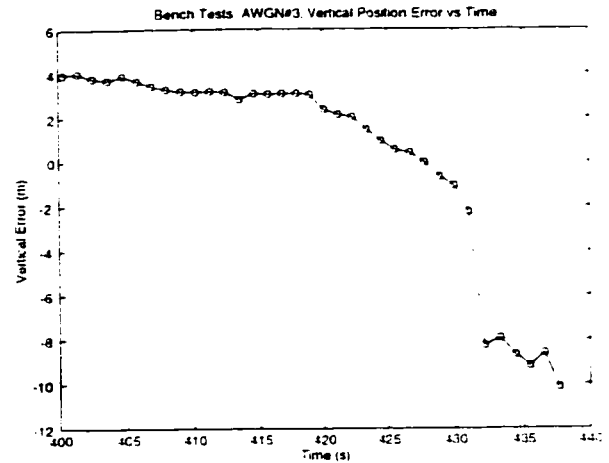


Figure 403.h: AWGN Bench Test: Vertical Position Error vs Time

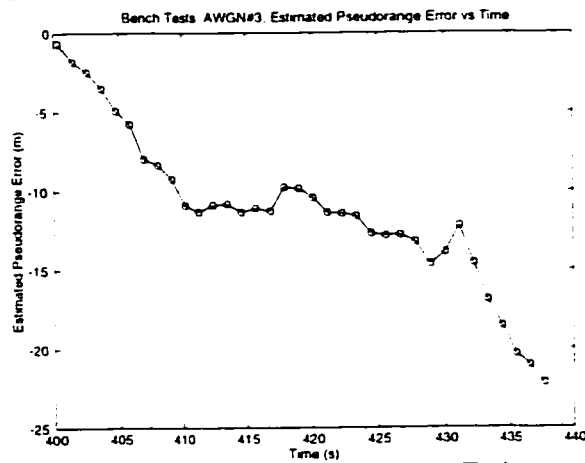


Figure 403.i: AWGN Bench Test: Estimate of Pseudorange Error vs Time

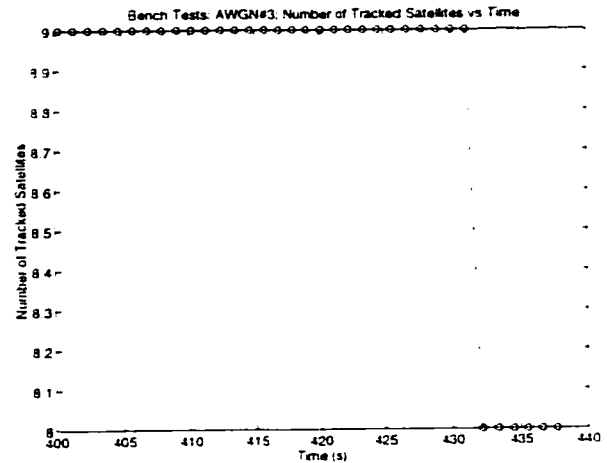


Figure 403.j: AWGN Bench Test: Number of Tracked Satellites vs Time

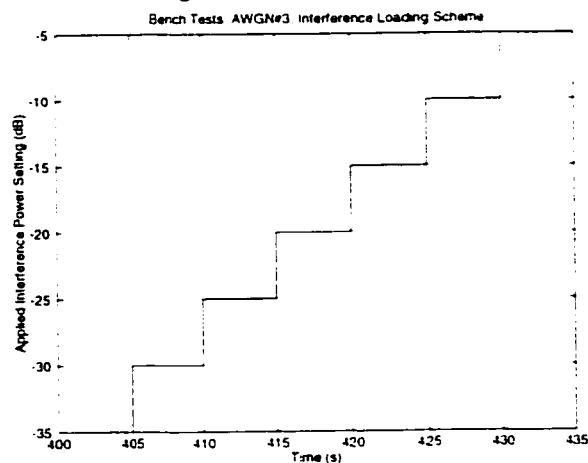


Figure 403.k: AWGN Bench Test: Interference Load Settings vs Time

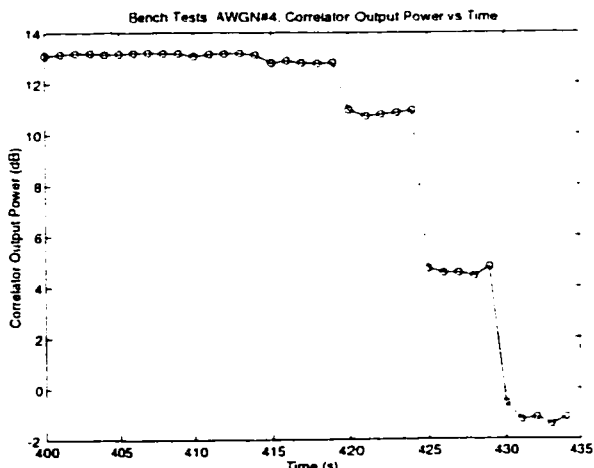


Figure 404.a: AWGN Bench Test:
Correlator Output Power vs Time

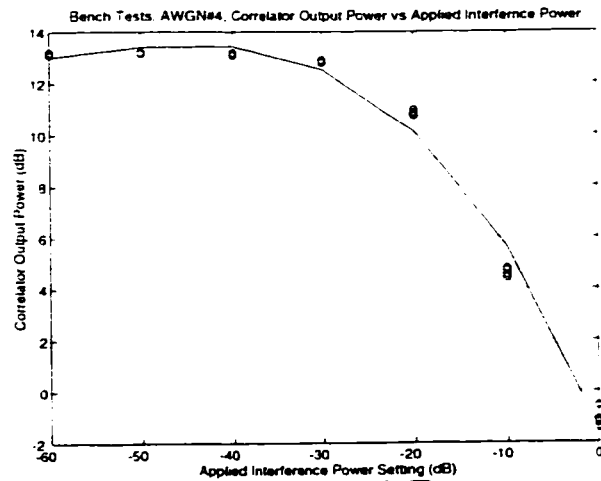


Figure 404.b: AWGN Bench Test:
Correlator Output Power vs Applied
Interference

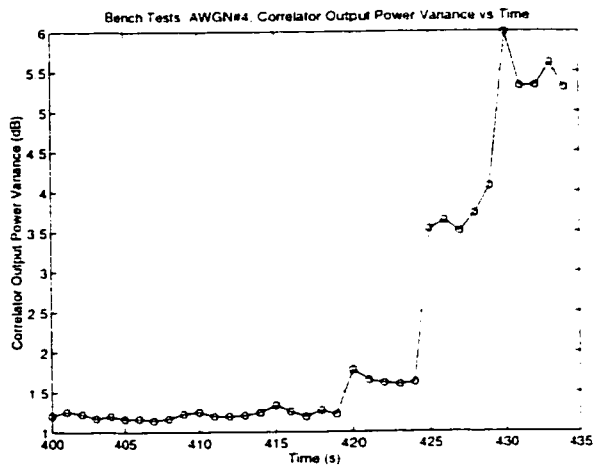


Figure 404.c: AWGN Bench Test:
Correlator Output Power Variance vs Time

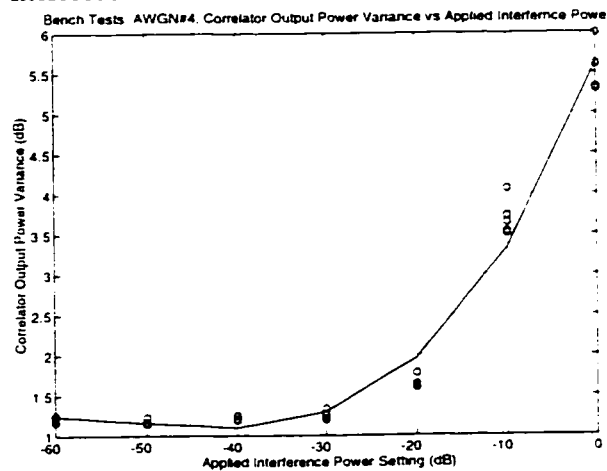


Figure 404.d: AWGN Bench Test:
Correlator Output Power Variance vs
Applied Interference

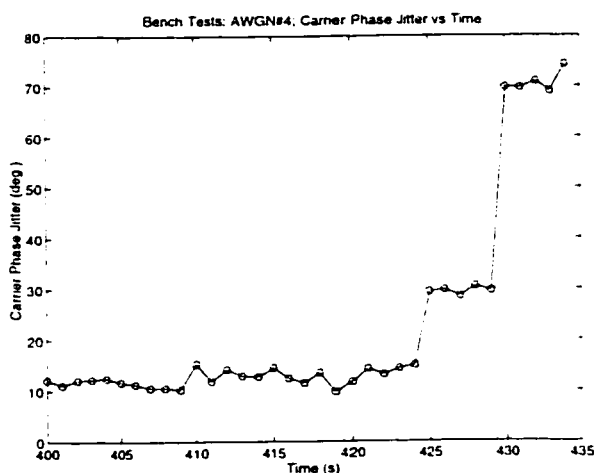


Figure 404.e: AWGN Bench Test: Carrier
Phase Jitter vs Time

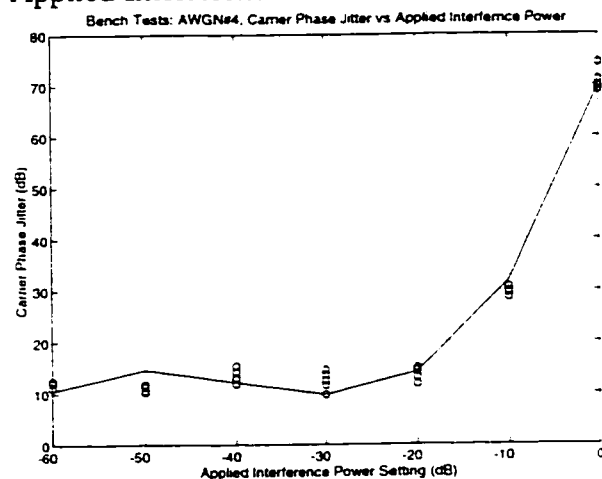


Figure 404.f: AWGN Bench Test: Carrier
Phase Jitter vs Applied Interference

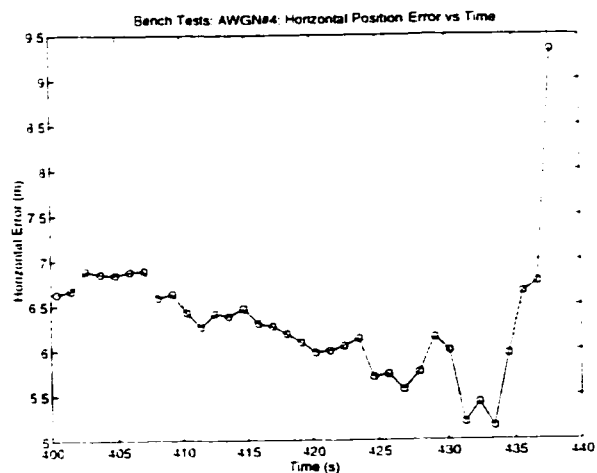


Figure 404.g: AWGN Bench Test: Horizontal Position Error vs Time

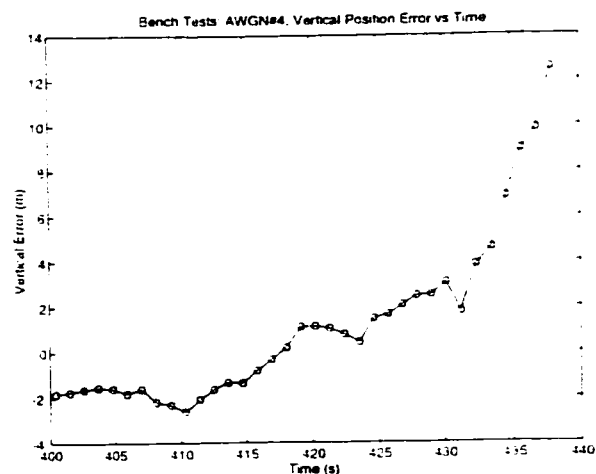


Figure 404.h: AWGN Bench Test: Vertical Position Error vs Time

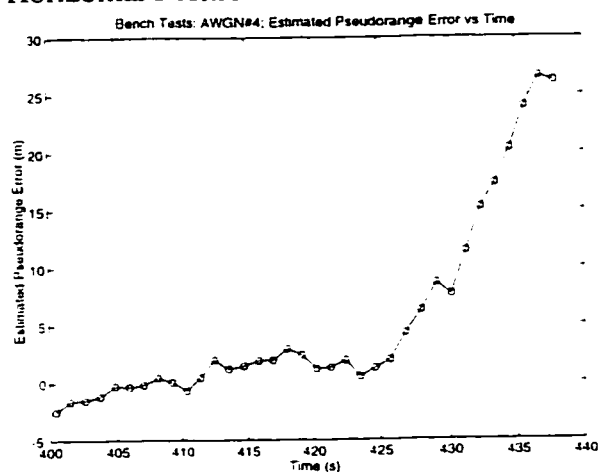


Figure 404.i: AWGN Bench Test: Estimate of Pseudorange Error vs Time

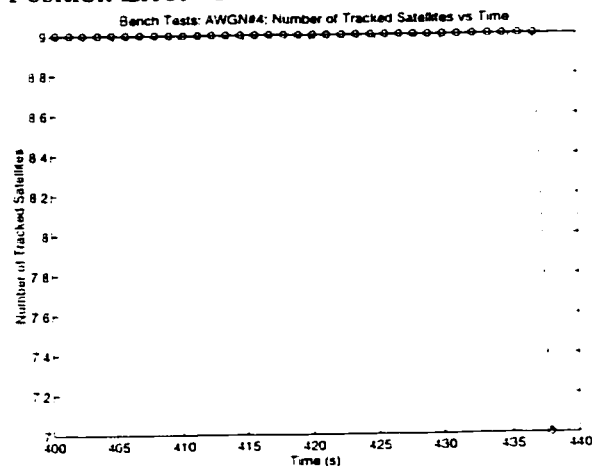


Figure 404.j: AWGN Bench Test: Number of Tracked Satellites vs Time

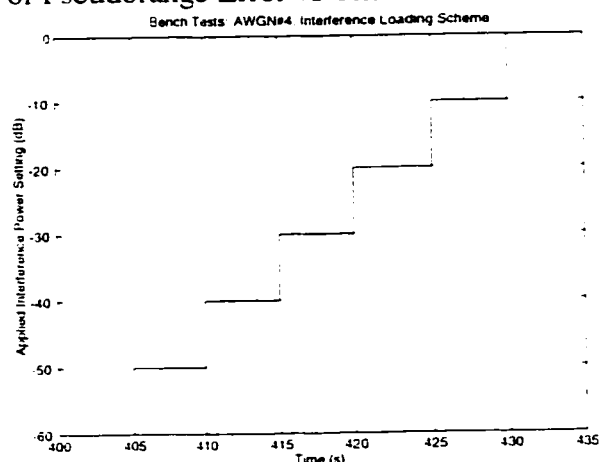


Figure 404.k: AWGN Bench Test: Interference Load Settings vs Time

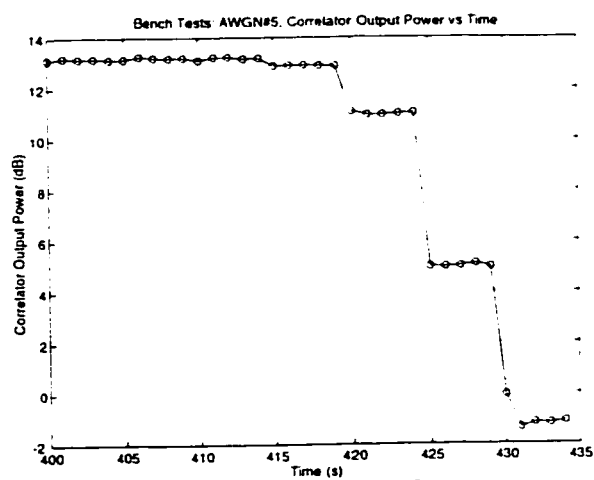


Figure 405.a: AWGN Bench Test:
Correlator Output Power vs Time

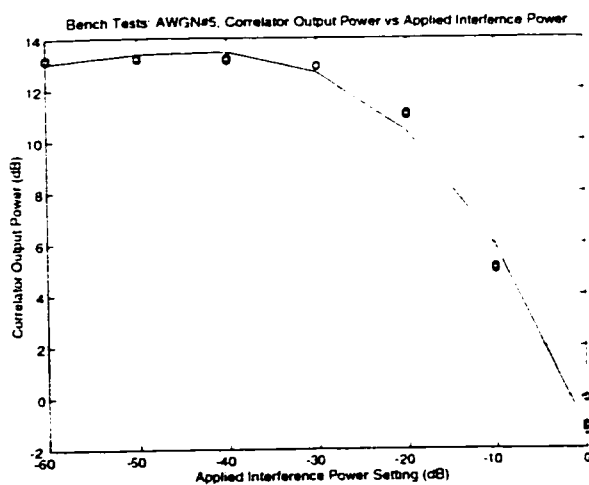


Figure 405.b: AWGN Bench Test:
Correlator Output Power vs Applied Interference

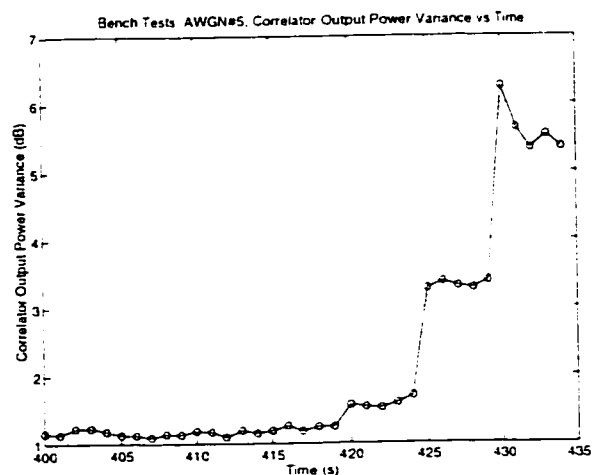


Figure 405.c: AWGN Bench Test:
Correlator Output Power Variance vs Time

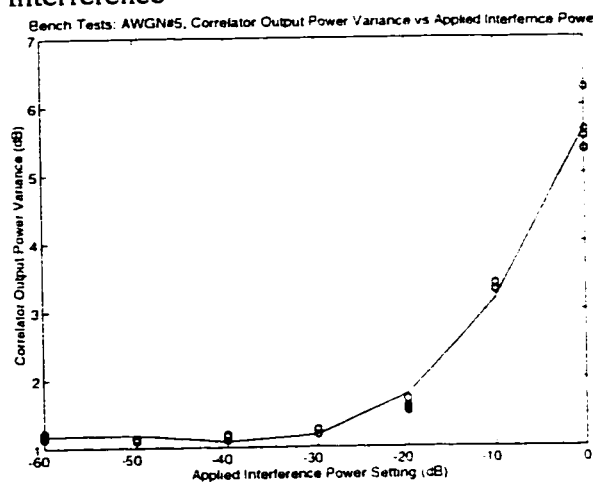


Figure 405.d: AWGN Bench Test:
Correlator Output Power Variance vs
Applied Interference

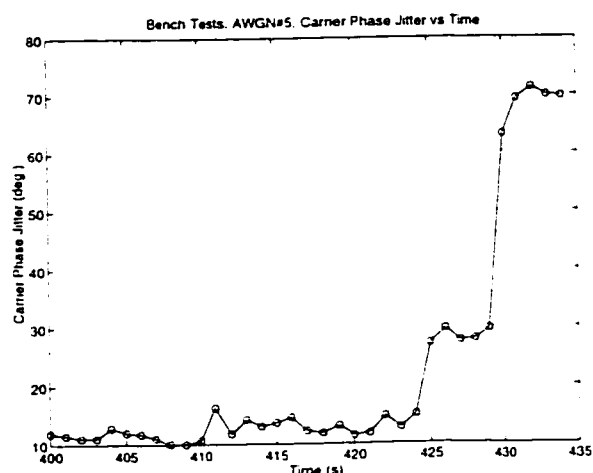


Figure 405.e: AWGN Bench Test: Carrier
Phase Jitter vs Time

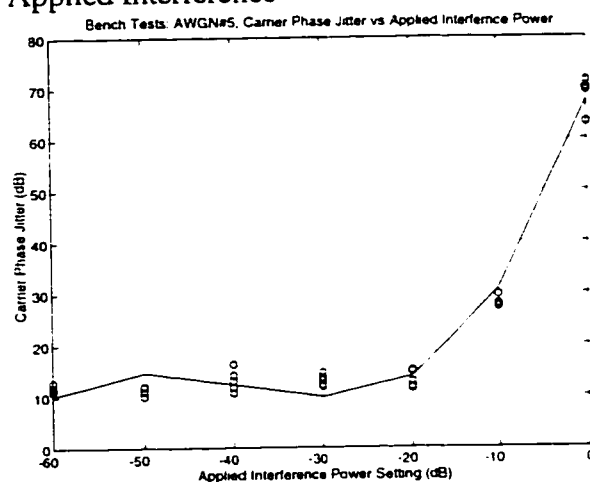


Figure 405.f: AWGN Bench Test: Carrier
Phase Jitter vs Applied Interference

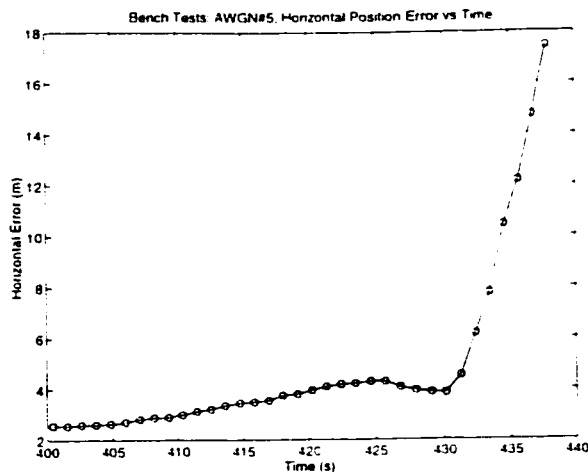


Figure 405.g: AWGN Bench Test: Horizontal Position Error vs Time

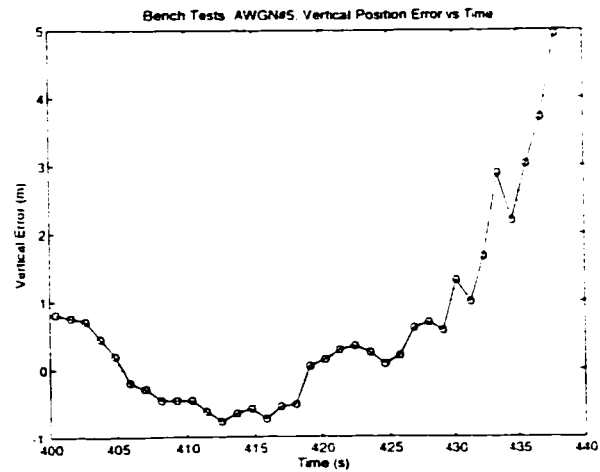


Figure 405.h: AWGN Bench Test: Vertical Position Error vs Time

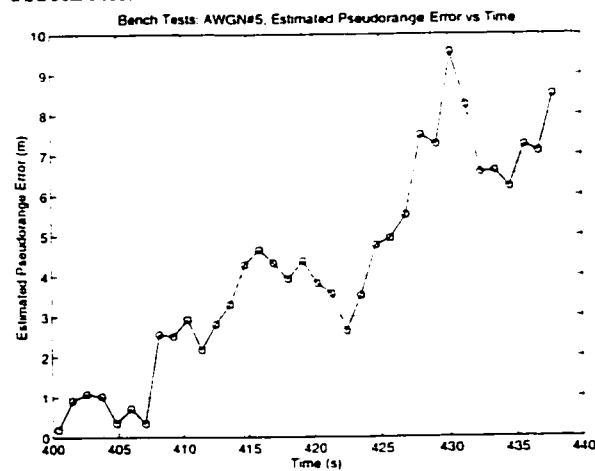


Figure 405.i: AWGN Bench Test: Estimate of Pseudorange Error vs Time

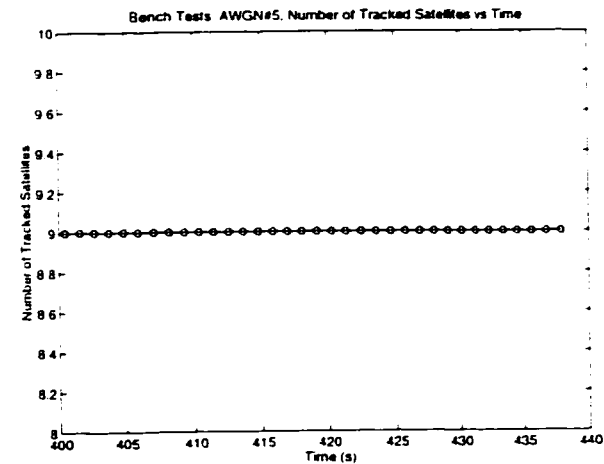


Figure 405.j: AWGN Bench Test: Number of Tracked Satellites vs Time

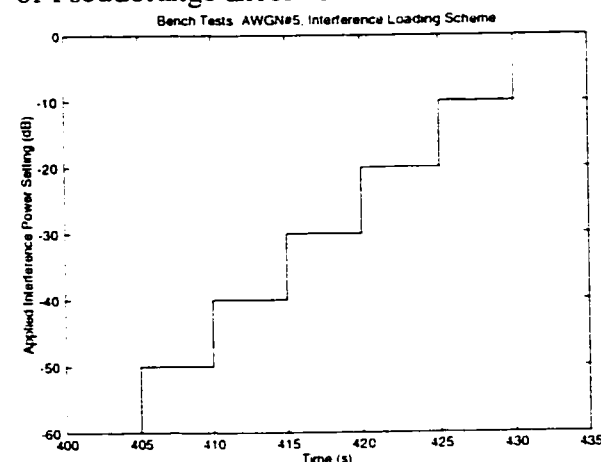


Figure 405.k: AWGN Bench Test: Interference Load Settings vs Time

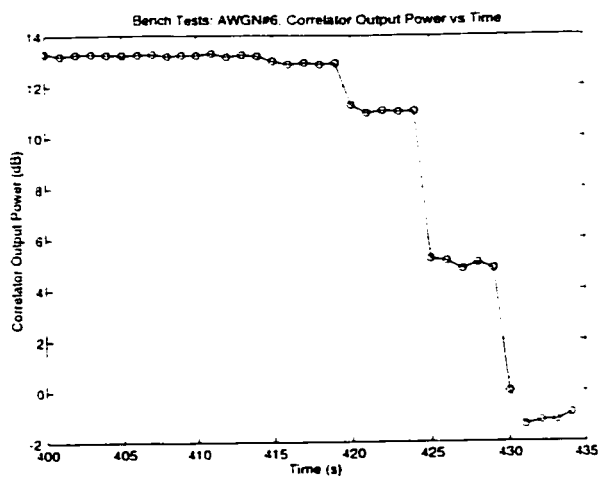


Figure 406.a: AWGN Bench Test: Correlator Output Power vs Time

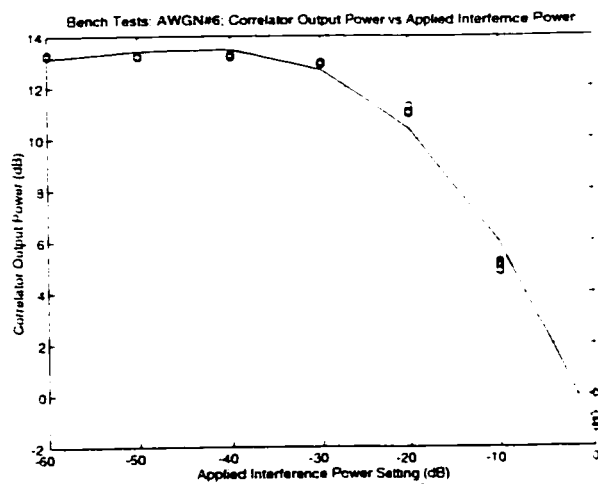


Figure 406.b: AWGN Bench Test: Correlator Output Power vs Applied Interference

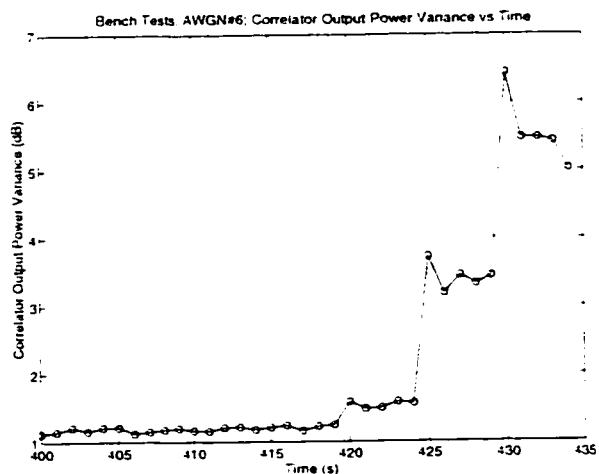


Figure 406.c: AWGN Bench Test: Correlator Output Power Variance vs Time

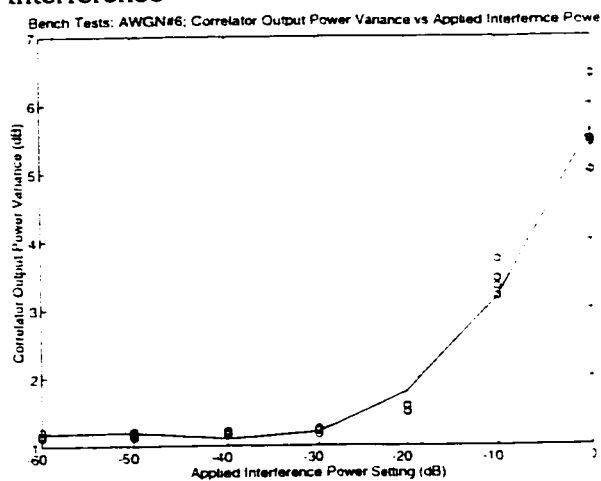


Figure 406.d: AWGN Bench Test: Correlator Output Power Variance vs Applied Interference

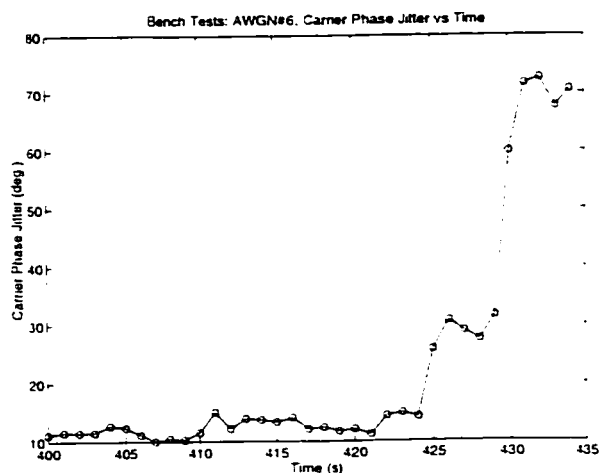


Figure 406.e: AWGN Bench Test: Carrier Phase Jitter vs Time

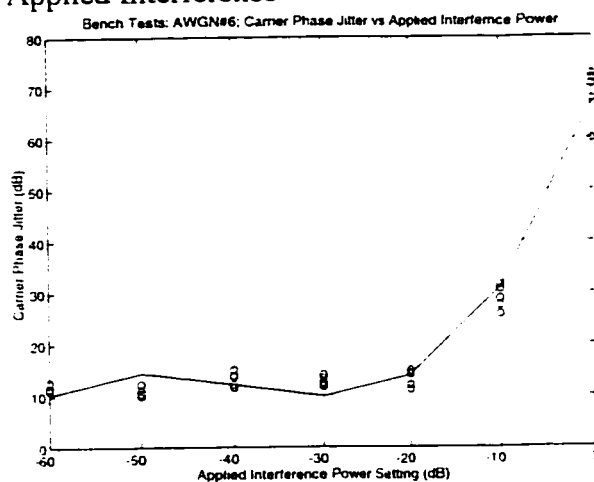


Figure 406.f: AWGN Bench Test: Carrier Phase Jitter vs Applied Interference

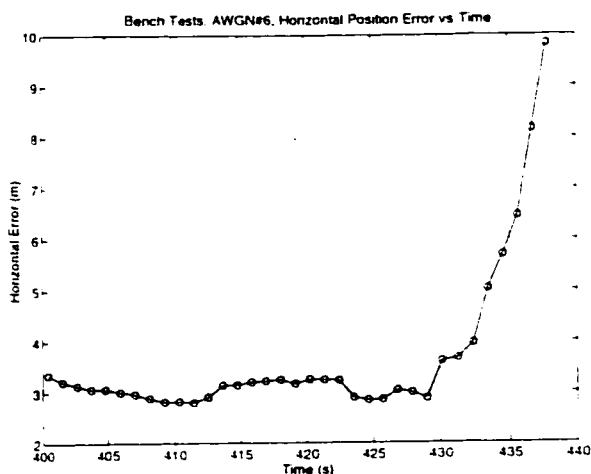


Figure 406.g: AWGN Bench Test: Horizontal Position Error vs Time

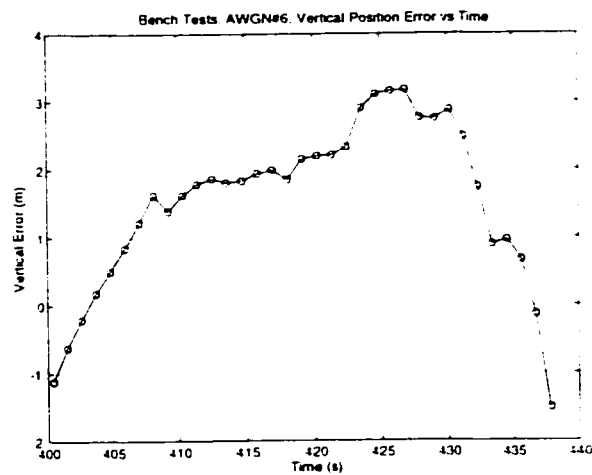


Figure 406.h: AWGN Bench Test: Vertical Position Error vs Time

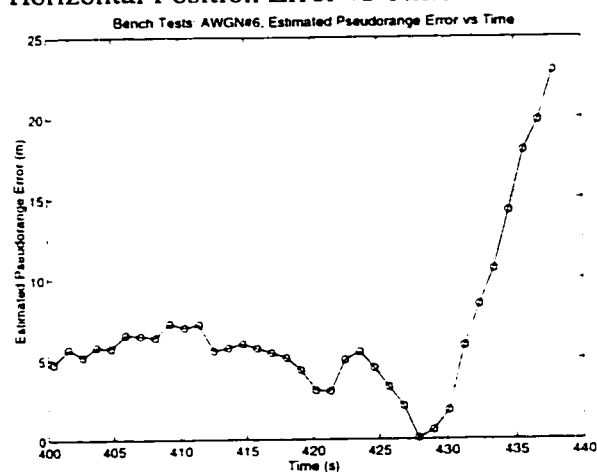


Figure 406.i: AWGN Bench Test: Estimate of Pseudorange Error vs Time

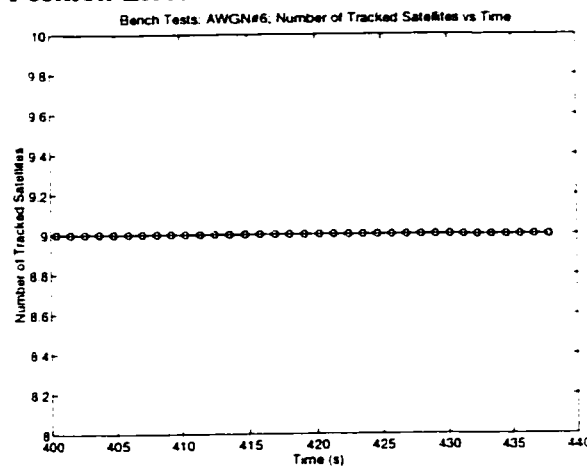


Figure 406.j: AWGN Bench Test: Number of Tracked Satellites vs Time

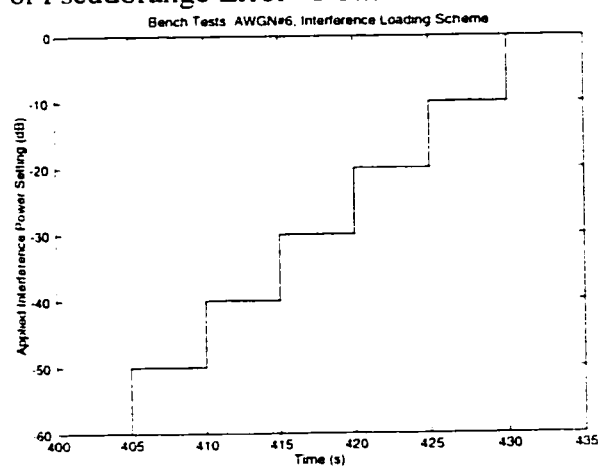


Figure 406.k: AWGN Bench Test: Interference Load Settings vs Time

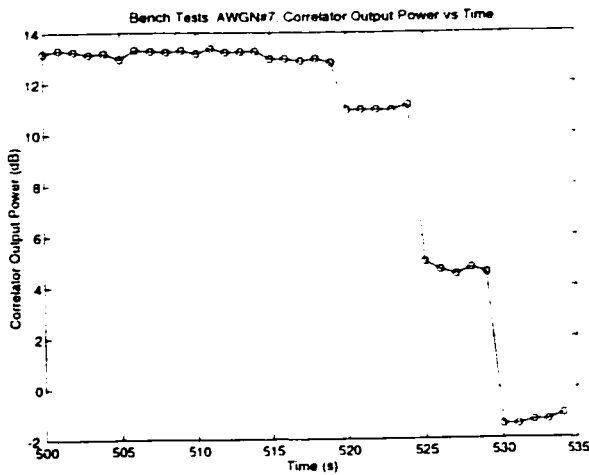


Figure 407.a: AWGN Bench Test:
Correlator Output Power vs Time

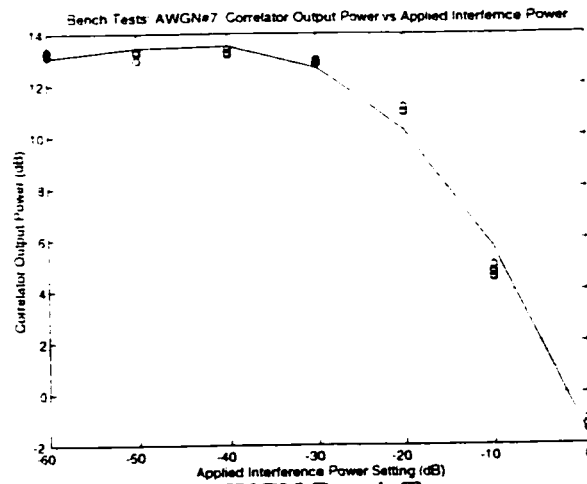


Figure 407.b: AWGN Bench Test:
Correlator Output Power vs Applied
Interference

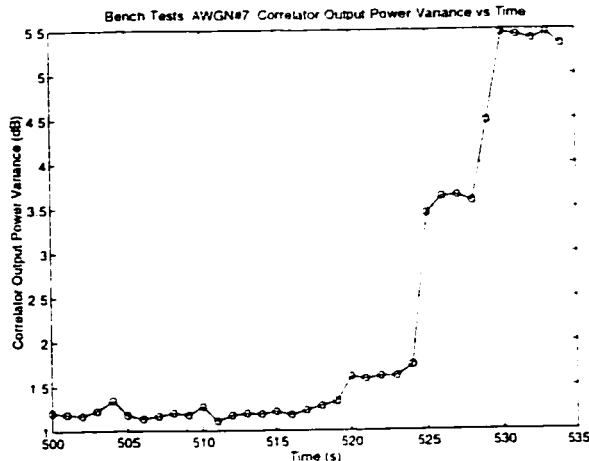


Figure 407.c: AWGN Bench Test:
Correlator Output Power Variance vs Time

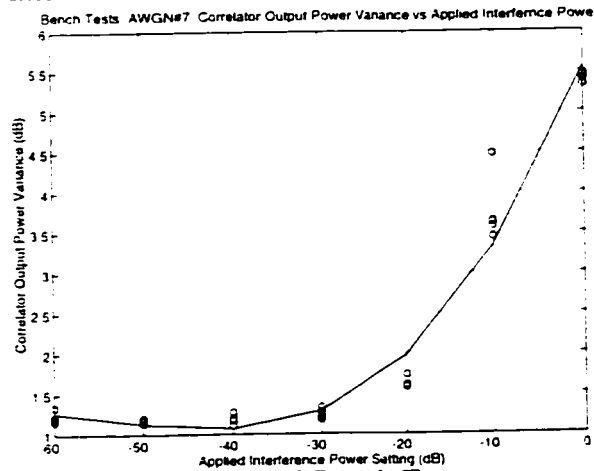


Figure 407.d: AWGN Bench Test:
Correlator Output Power Variance vs
Applied Interference

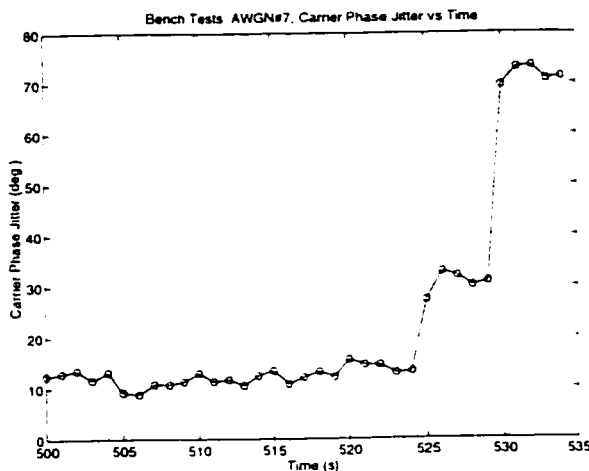


Figure 407.e: AWGN Bench Test: Carrier
Phase Jitter vs Time

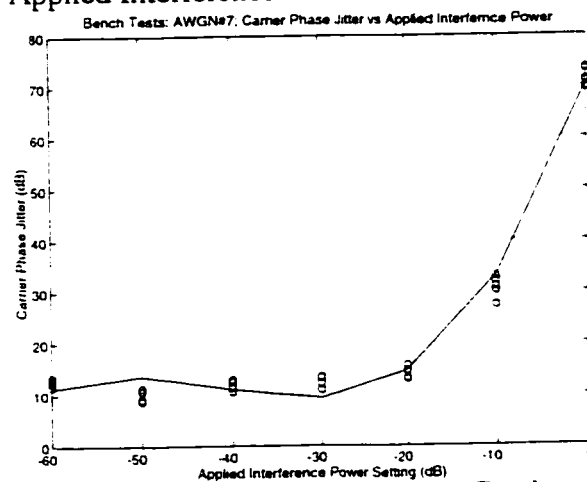


Figure 407.f: AWGN Bench Test: Carrier
Phase Jitter vs Applied Interference

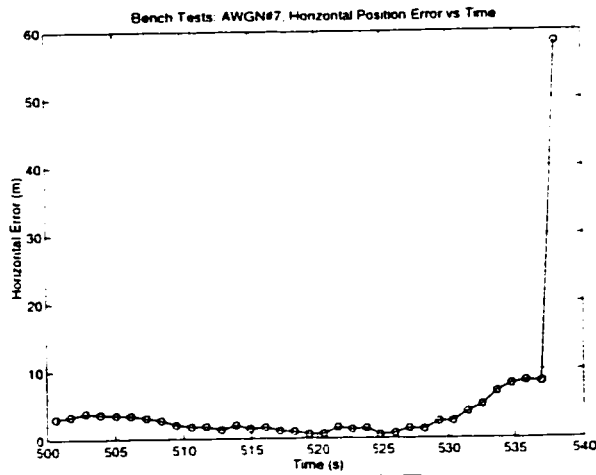


Figure 407.g: AWGN Bench Test: Horizontal Position Error vs Time

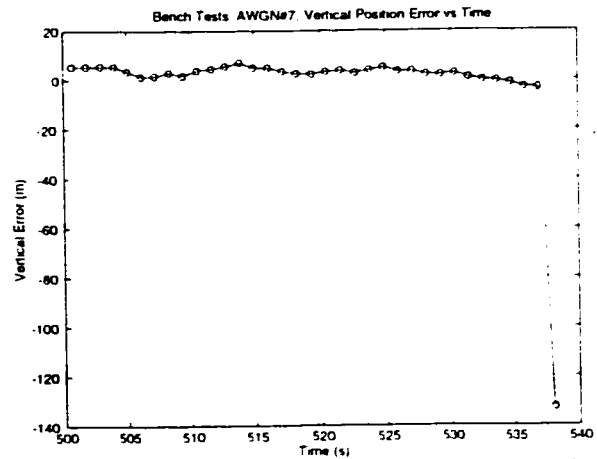


Figure 407.h: AWGN Bench Test: Vertical Position Error vs Time

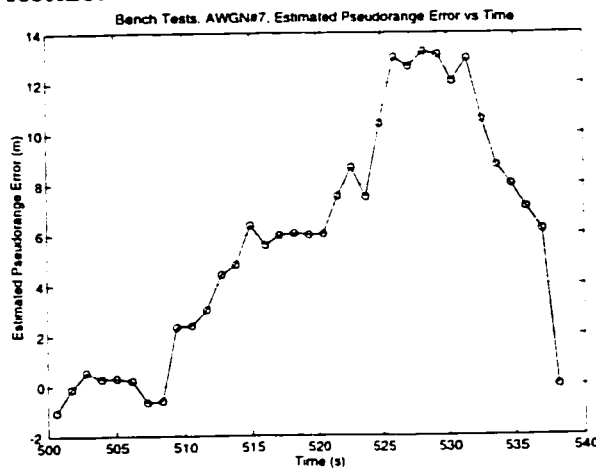


Figure 407.i: AWGN Bench Test: Estimate of Pseudorange Error vs Time

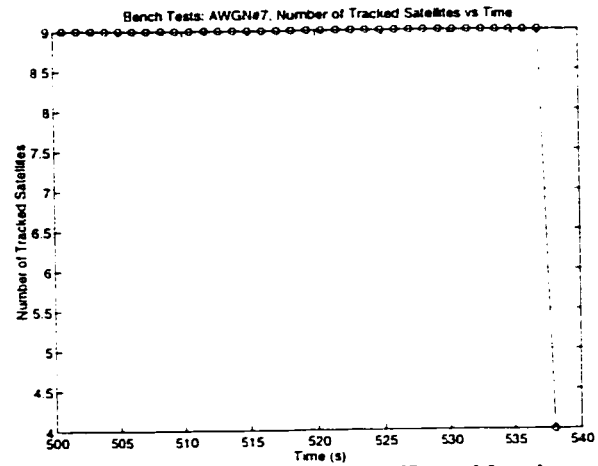


Figure 407.j: AWGN Bench Test: Number of Tracked Satellites vs Time

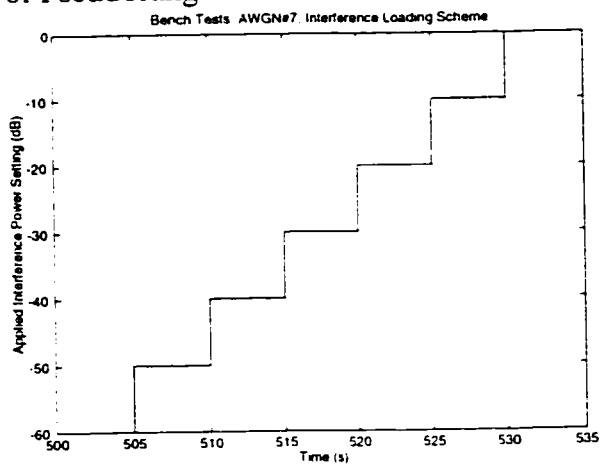


Figure 407.k: AWGN Bench Test: Interference Load Settings vs Time

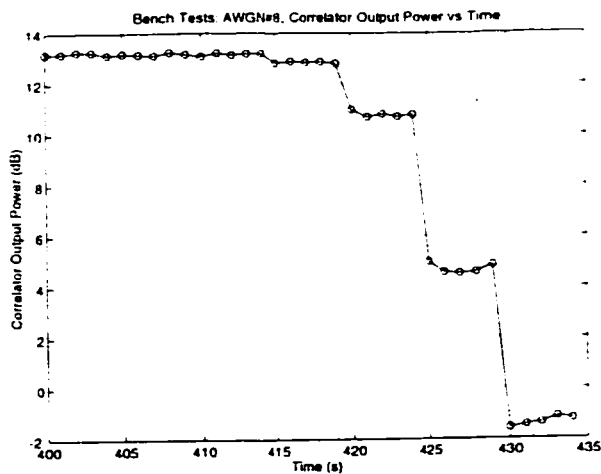


Figure 408.a: AWGN Bench Test:
Correlator Output Power vs Time

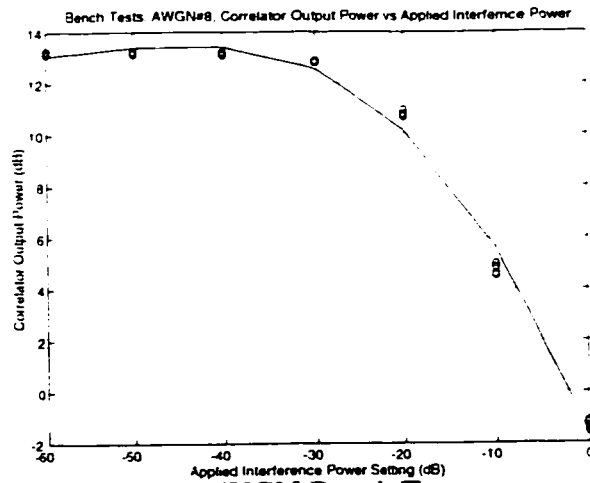


Figure 408.b: AWGN Bench Test:
Correlator Output Power vs Applied
Interference

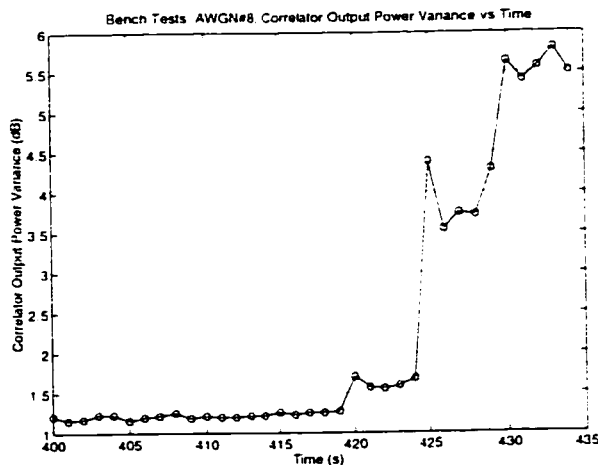


Figure 408.c: AWGN Bench Test:
Correlator Output Power Variance vs Time

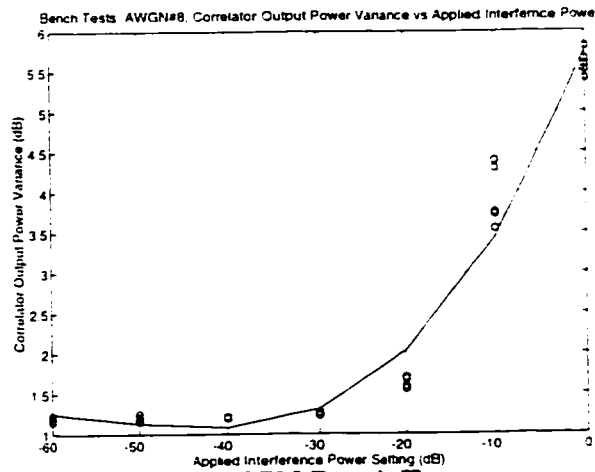


Figure 408.d: AWGN Bench Test:
Correlator Output Power Variance vs
Applied Interference

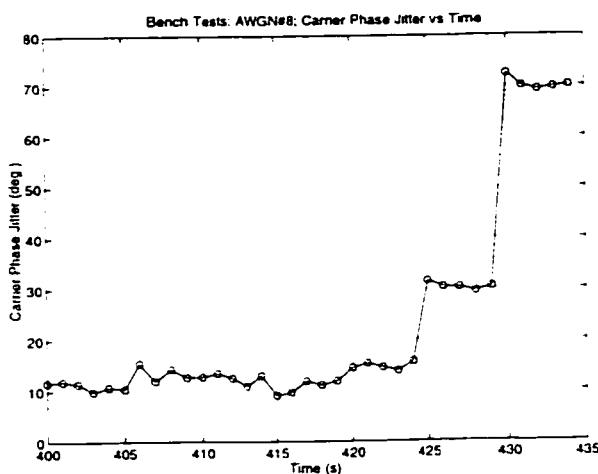


Figure 408.e: AWGN Bench Test: Carrier
Phase Jitter vs Time

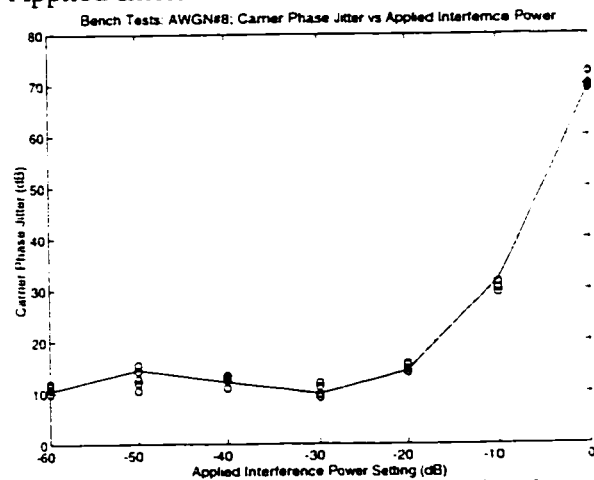


Figure 408.f: AWGN Bench Test: Carrier
Phase Jitter vs Applied Interference

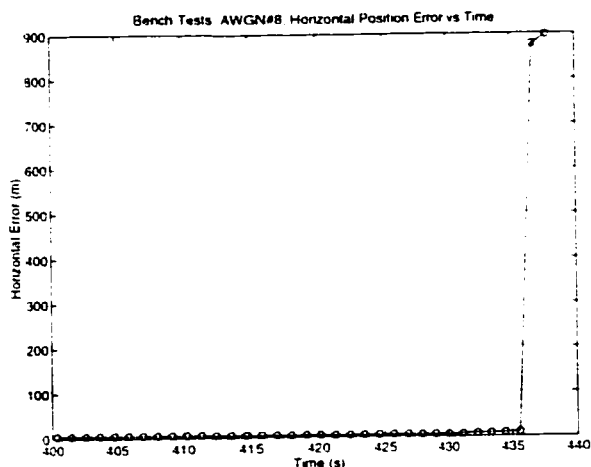


Figure 408.g: AWGN Bench Test: Horizontal Position Error vs Time

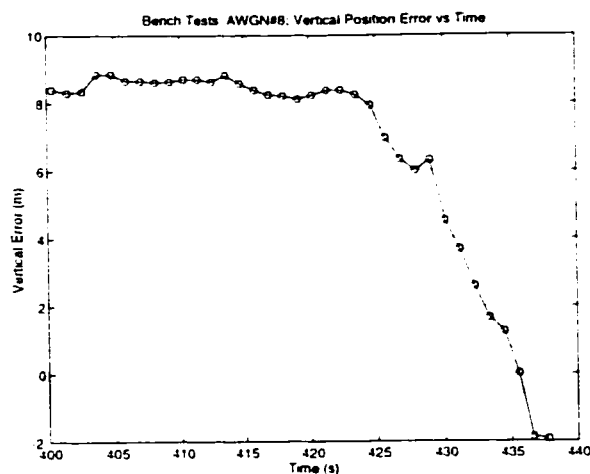


Figure 408.h: AWGN Bench Test: Vertical Position Error vs Time

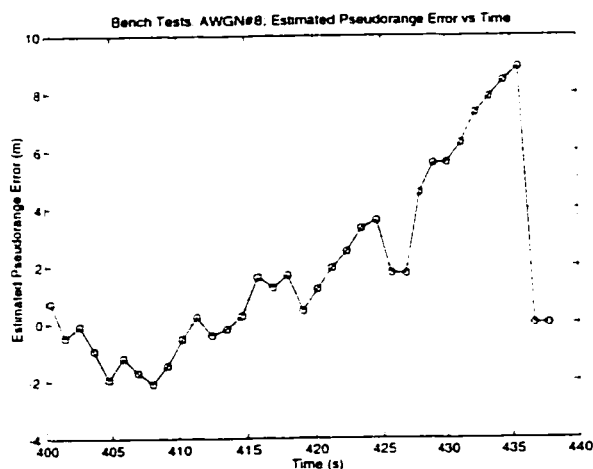


Figure 408.i: AWGN Bench Test: Estimate of Pseudorange Error vs Time

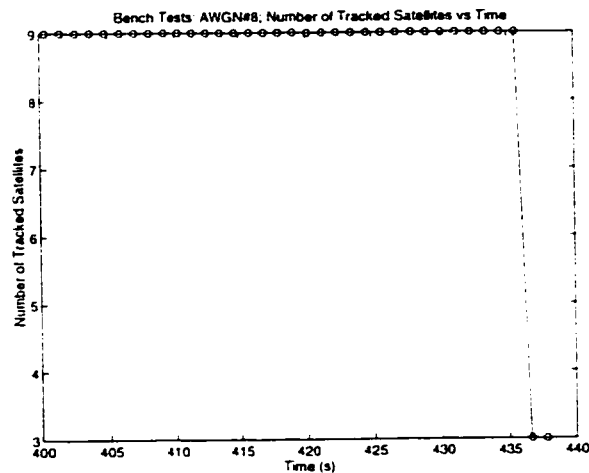


Figure 408.j: AWGN Bench Test: Number of Tracked Satellites vs Time

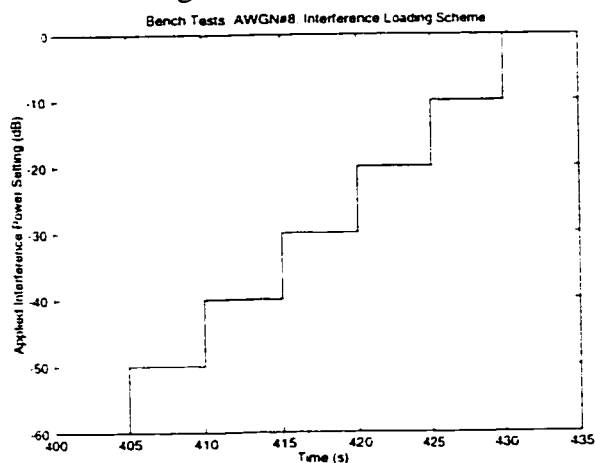


Figure 408.k: AWGN Bench Test: Interference Load Settings vs Time

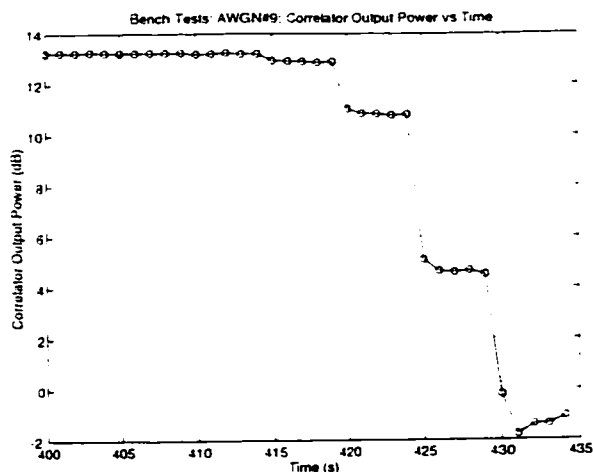


Figure 409.a: AWGN Bench Test:
Correlator Output Power vs Time

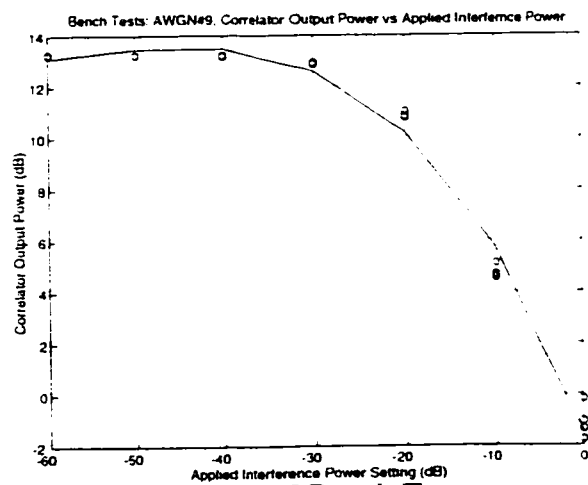


Figure 409.b: AWGN Bench Test:
Correlator Output Power vs Applied
Interference

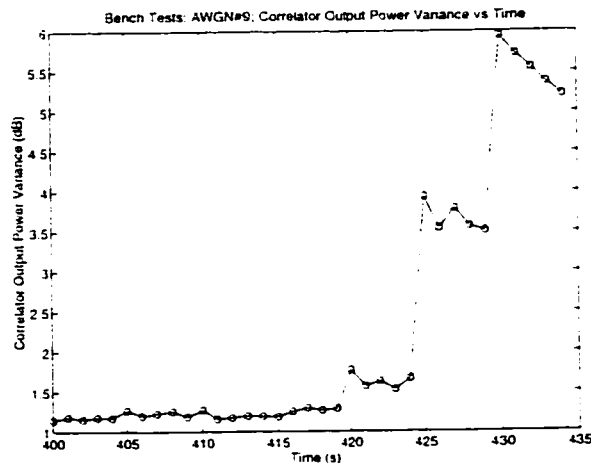


Figure 409.c: AWGN Bench Test:
Correlator Output Power Variance vs Time

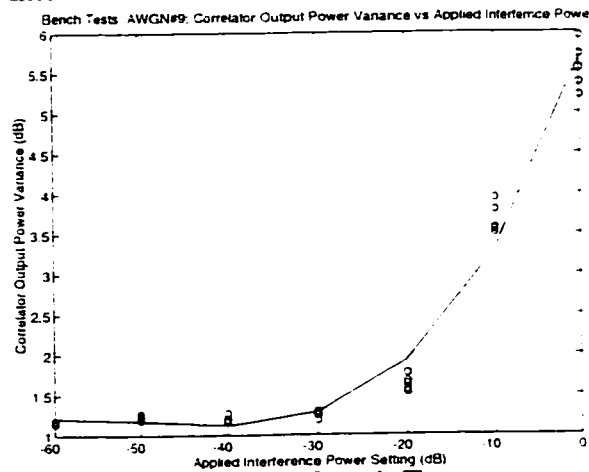


Figure 409.d: AWGN Bench Test:
Correlator Output Power Variance vs
Applied Interference

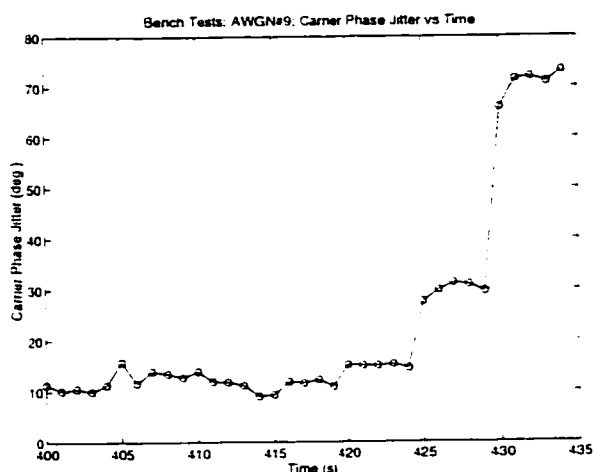


Figure 409.e: AWGN Bench Test: Carrier
Phase Jitter vs Time

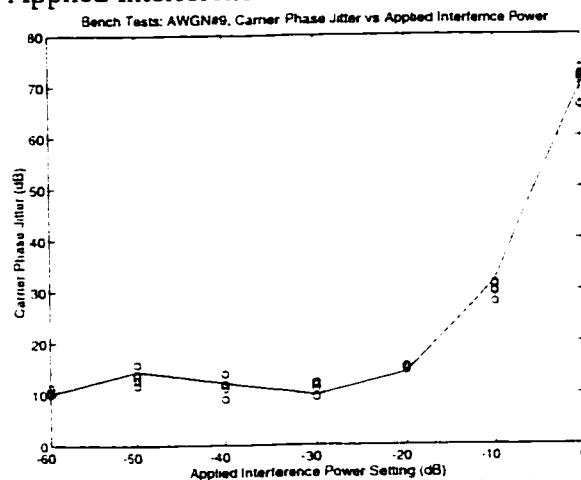


Figure 409.f: AWGN Bench Test: Carrier
Phase Jitter vs Applied Interference

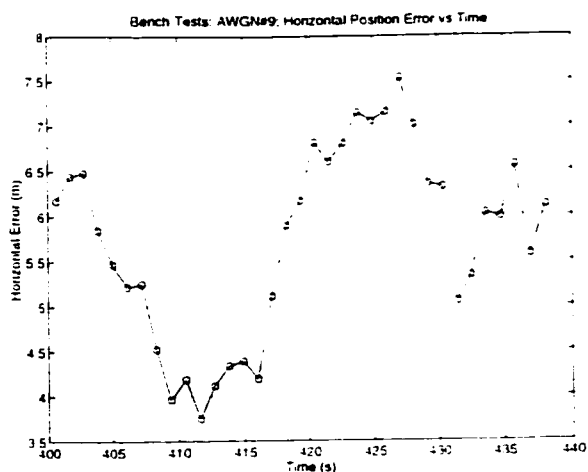


Figure 409.g: AWGN Bench Test: Horizontal Position Error vs Time

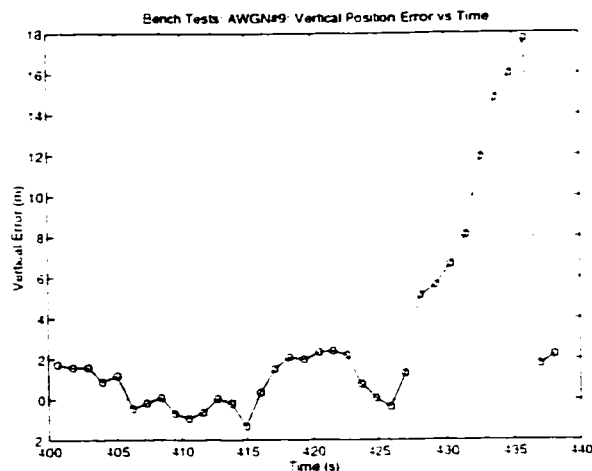


Figure 409.h: AWGN Bench Test: Vertical Position Error vs Time

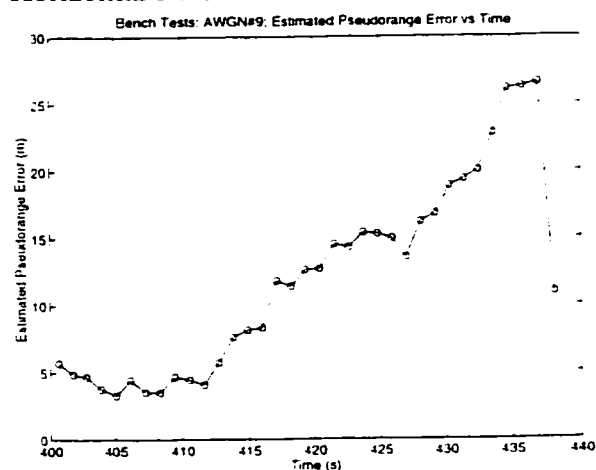


Figure 409.i: AWGN Bench Test: Estimate of Pseudorange Error vs Time

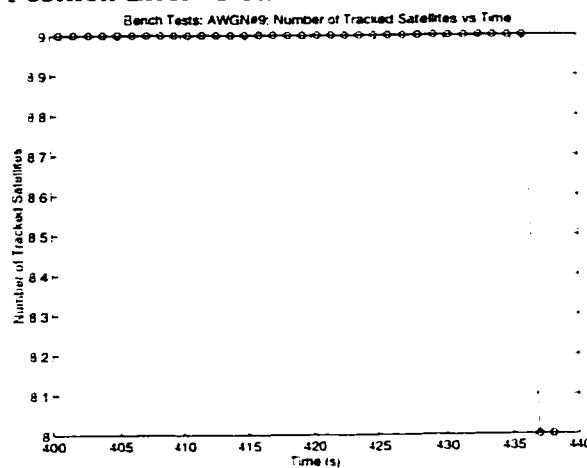


Figure 409.j: AWGN Bench Test: Number of Tracked Satellites vs Time

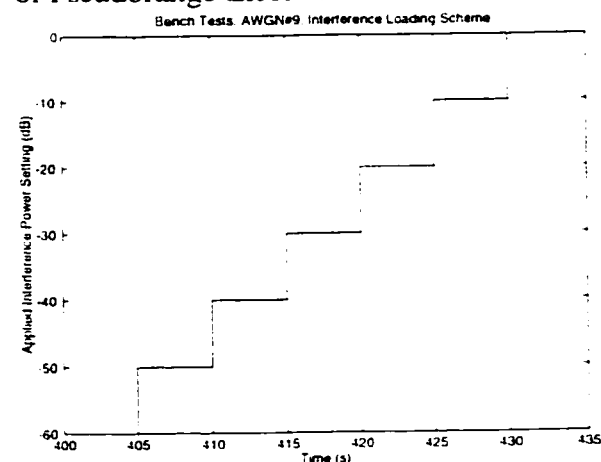


Figure 409.k: AWGN Bench Test: Interference Load Settings vs Time

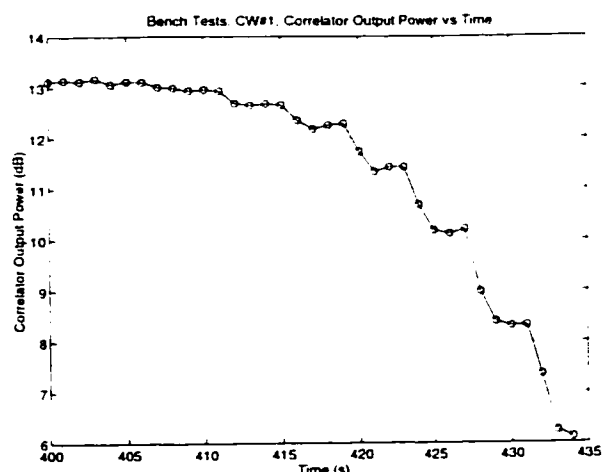


Figure 421.a: CW Interference Bench Test: Correlator Output Power vs Time

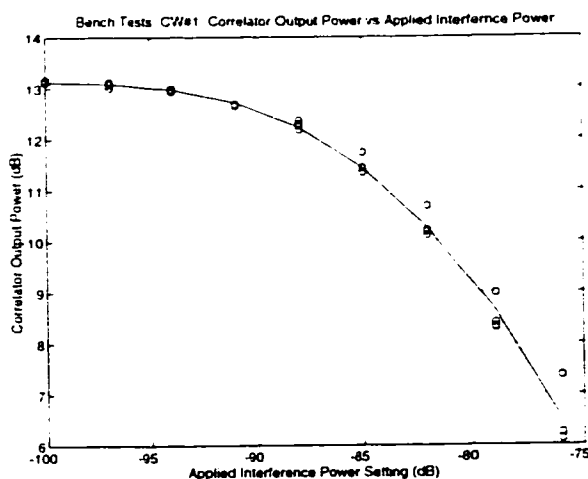


Figure 421.b: CW Interference Bench Test: Correlator Output Power vs Applied Interference

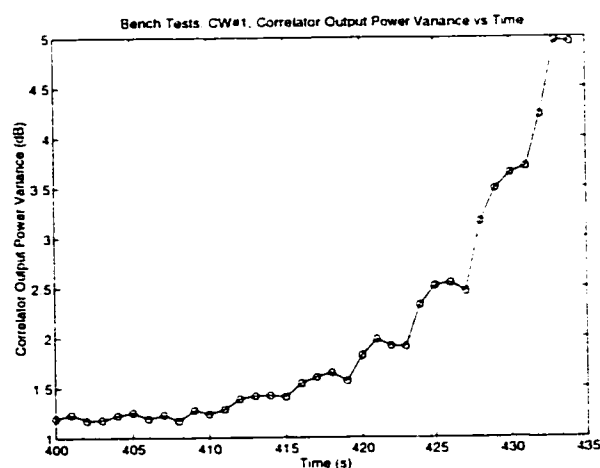


Figure 421.c: CW Interference Bench Test: Correlator Output Power Variance vs Time

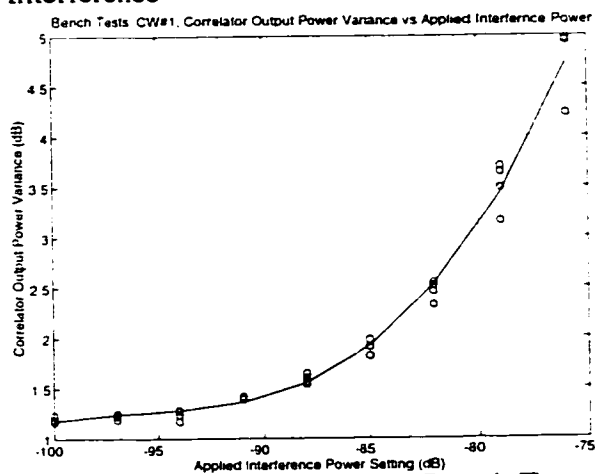


Figure 421.d: CW Interference Bench Test: Correlator Output Power Variance vs Applied Interference

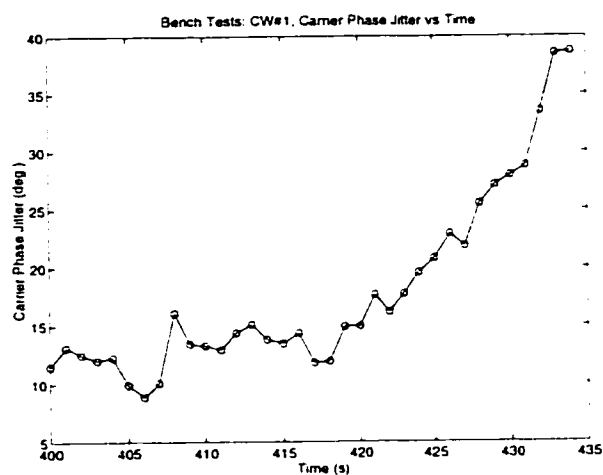


Figure 421.e: CW Interference Bench Test: Carrier Phase Jitter vs Time

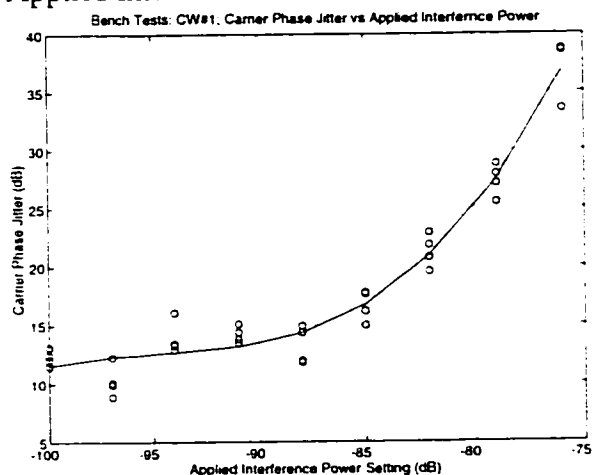


Figure 421.f: CW Interference Bench Test: Carrier Phase Jitter vs Applied Interference

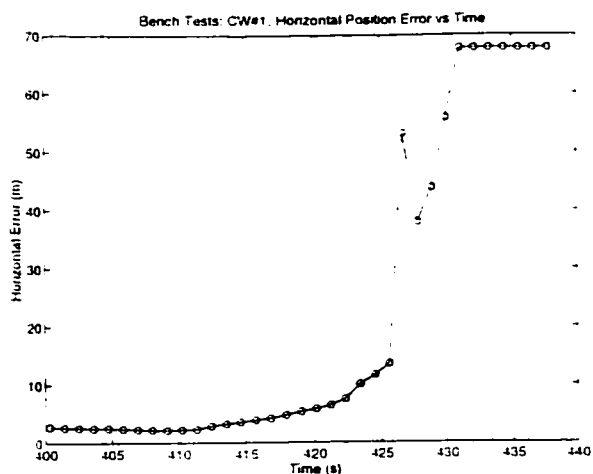


Figure 421.g: CW Interference Bench Test: Horizontal Position Error vs Time

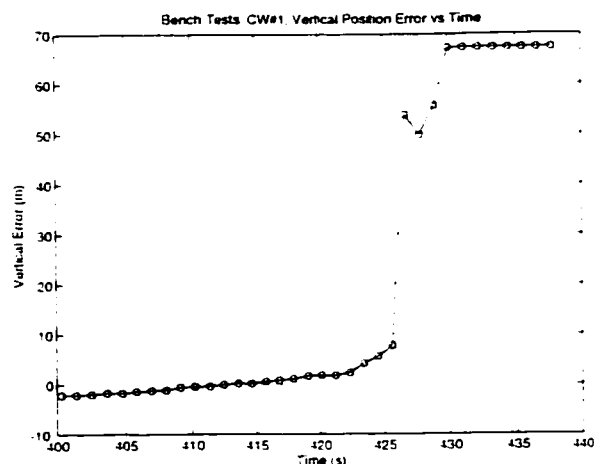


Figure 421.h: CW Interference Bench Test: Vertical Position Error vs Time

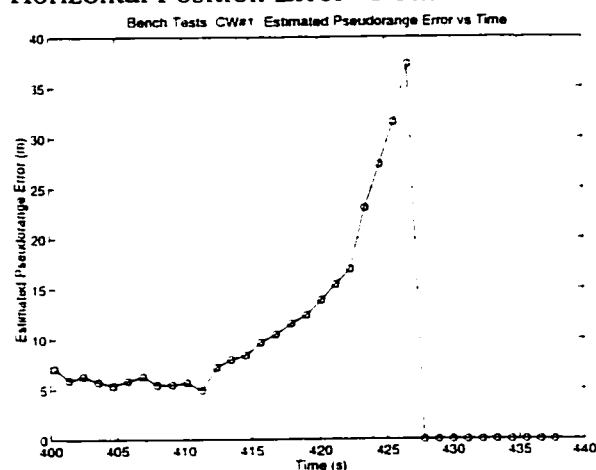


Figure 421.i: CW Interference Bench Test: Estimate of Pseudorange Error vs Time

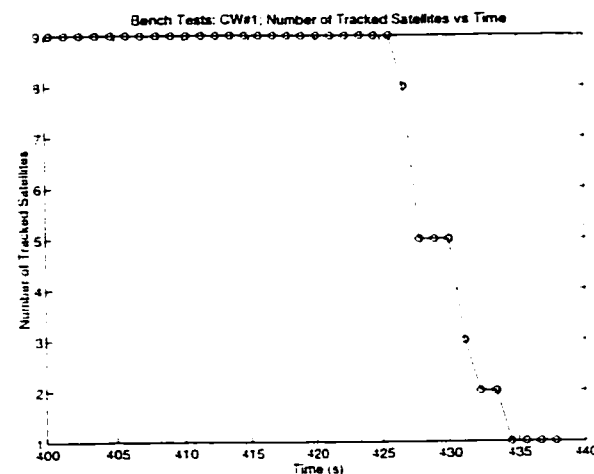


Figure 421.j: CW Interference Bench Test: Number of Tracked Satellites vs Time

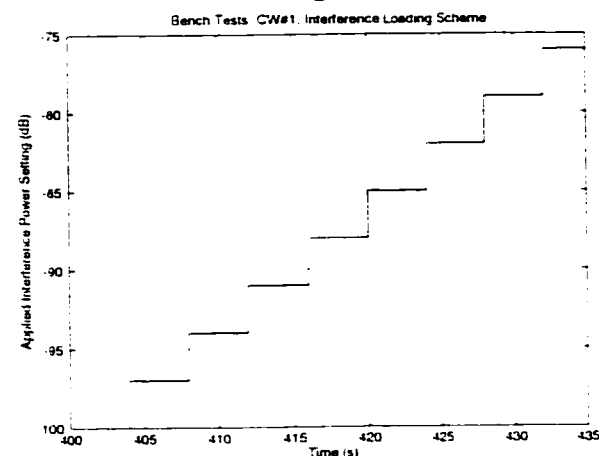


Figure 421.k: CW Interference Bench Test: Interference Load Settings vs Time

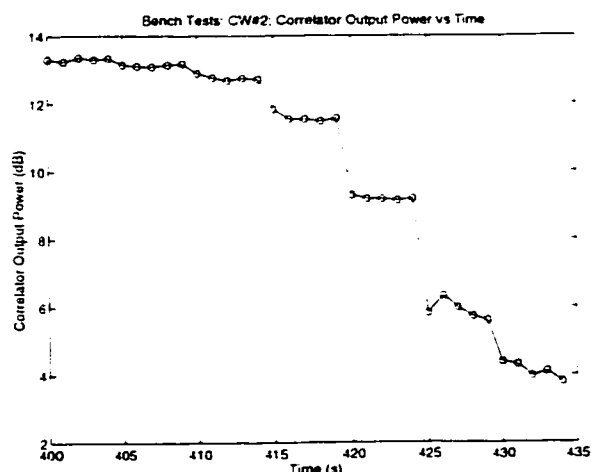


Figure 422.a: CW Interference Bench Test: Correlator Output Power vs Time

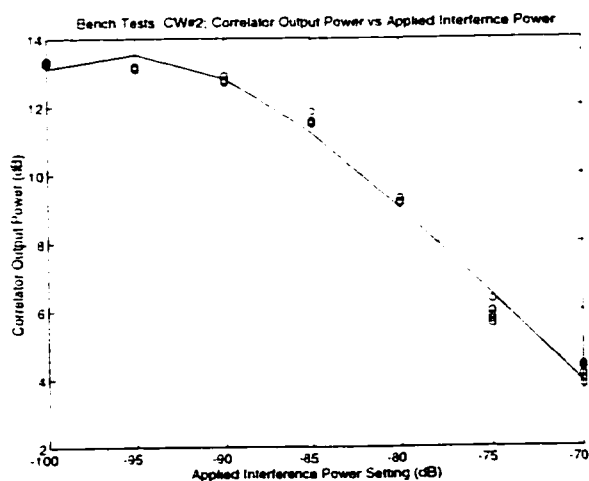


Figure 422.b: CW Interference Bench Test: Correlator Output Power vs Applied Interference

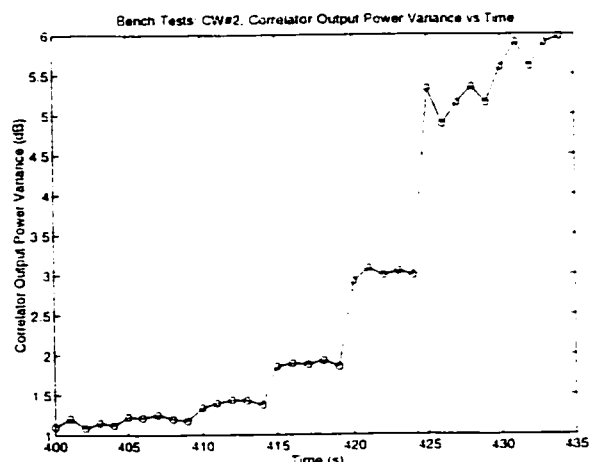


Figure 422.c: CW Interference Bench Test: Correlator Output Power Variance vs Time

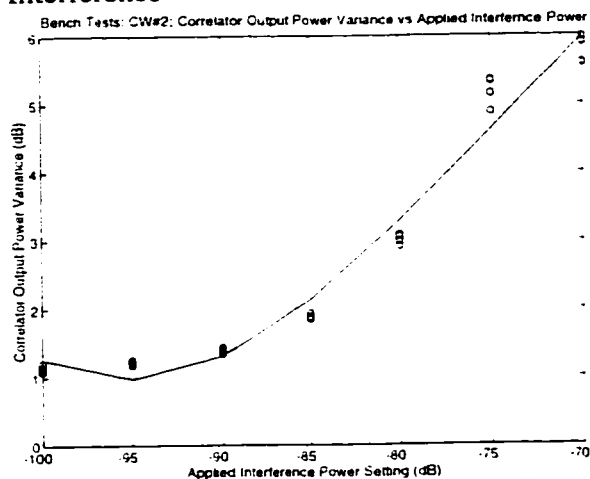


Figure 422.d: CW Interference Bench Test: Correlator Output Power Variance vs Applied Interference

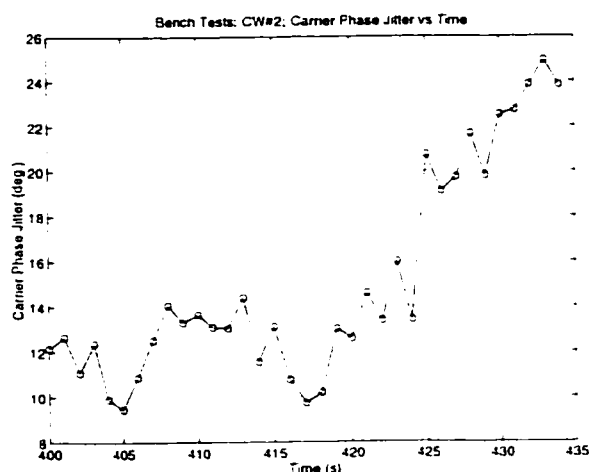


Figure 422.e: CW Interference Bench Test: Carrier Phase Jitter vs Time

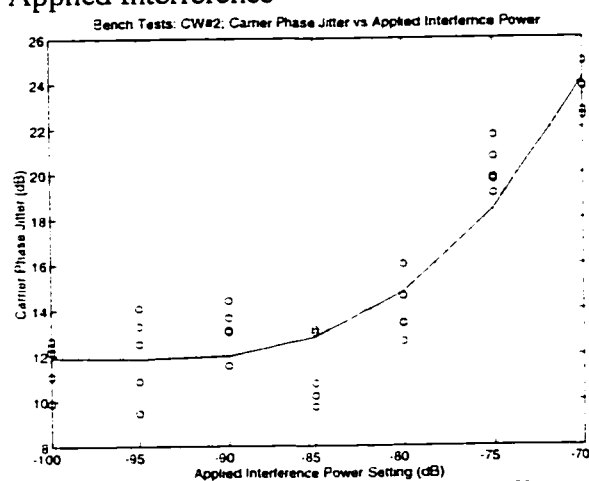


Figure 422.f: CW Interference Bench Test: Carrier Phase Jitter vs Applied Interference

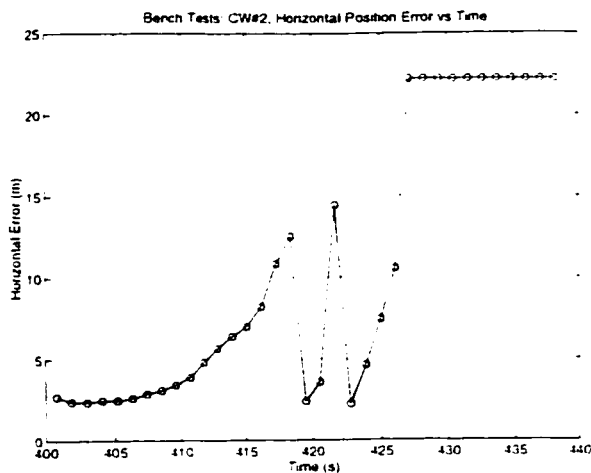


Figure 422.g: CW Interference Bench Test: Horizontal Position Error vs Time

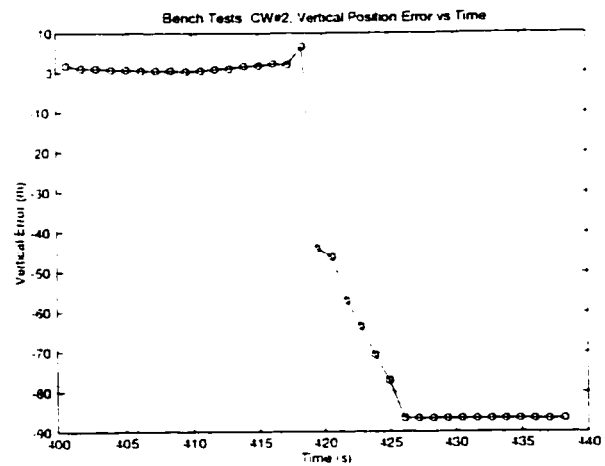


Figure 422.h: CW Interference Bench Test: Vertical Position Error vs Time

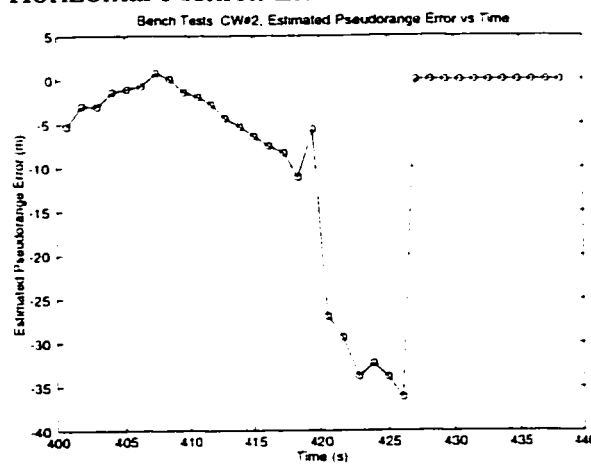


Figure 422.i: CW Interference Bench Test: Estimate of Pseudorange Error vs Time

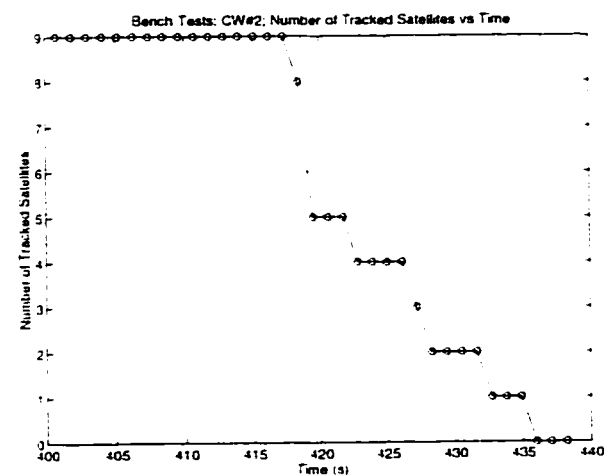


Figure 422.j: CW Interference Bench Test: Number of Tracked Satellites vs Time

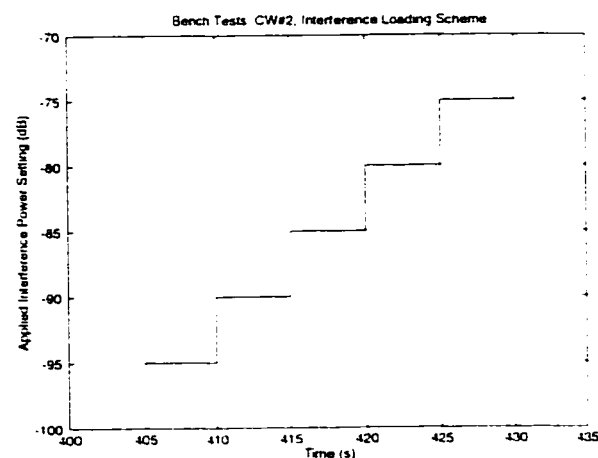


Figure 422.k: CW Interference Bench Test: Interference Load Settings vs Time

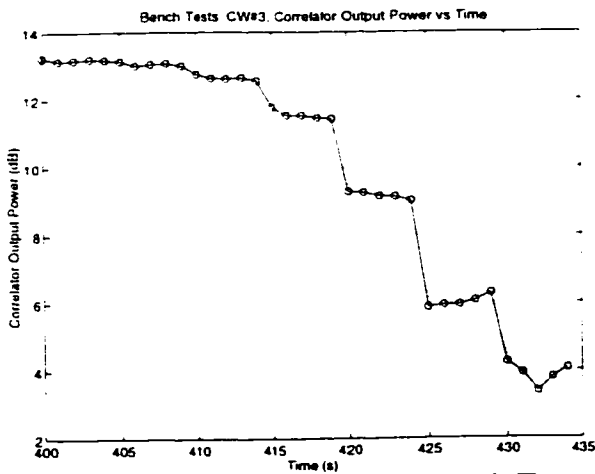


Figure 423.a: CW Interference Bench Test: Correlator Output Power vs Time

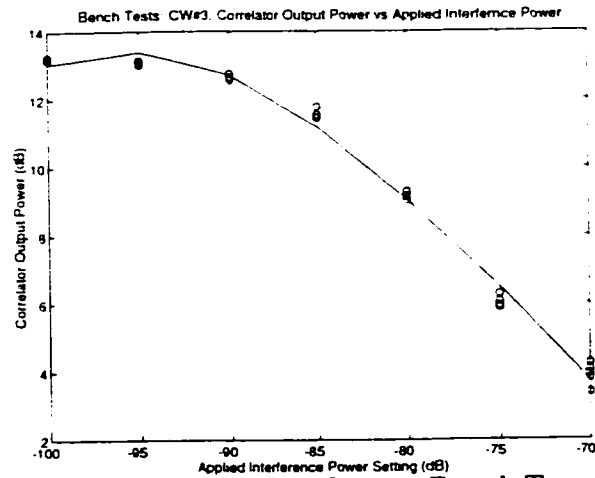


Figure 423.b: CW Interference Bench Test: Correlator Output Power vs Applied Interference

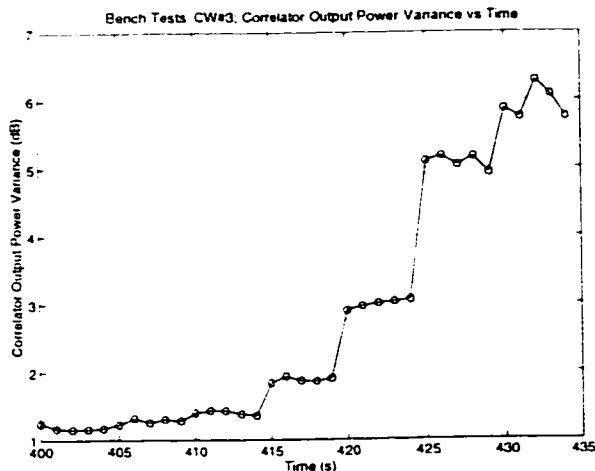


Figure 423.c: CW Interference Bench Test: Correlator Output Power Variance vs Time

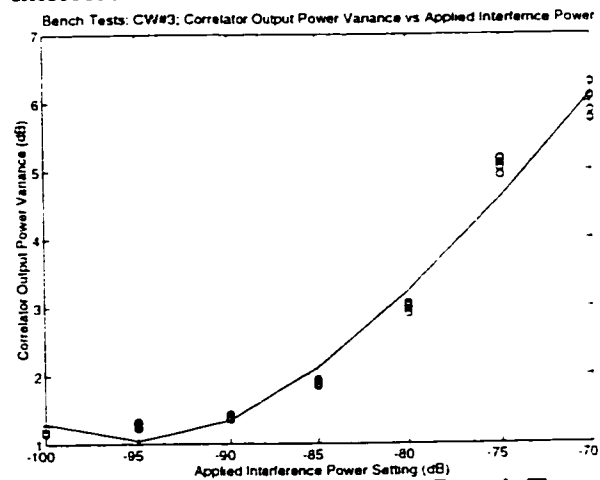


Figure 423.d: CW Interference Bench Test: Correlator Output Power Variance vs Applied Interference

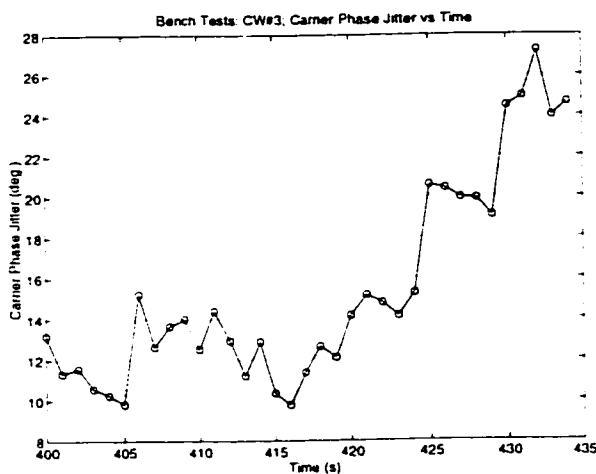


Figure 423.e: CW Interference Bench Test: Carrier Phase Jitter vs Time

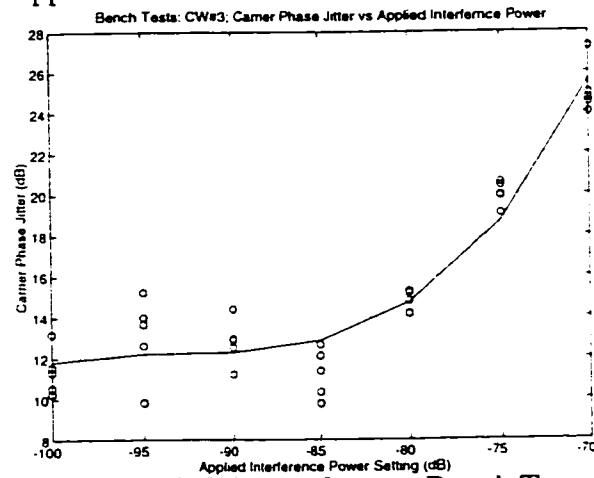


Figure 423.f: CW Interference Bench Test: Carrier Phase Jitter vs Applied Interference

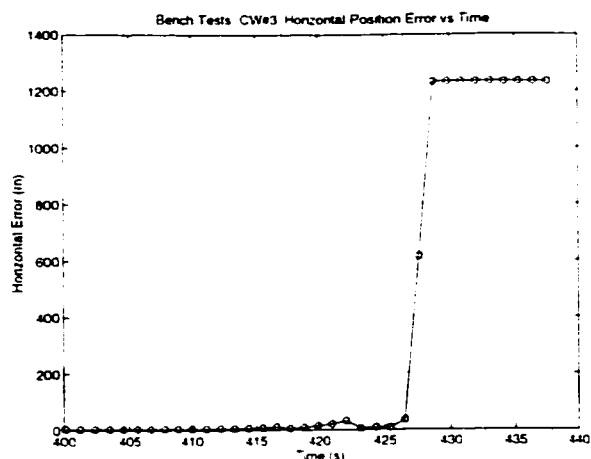


Figure 423.g: CW Interference Bench Test: Horizontal Position Error vs Time

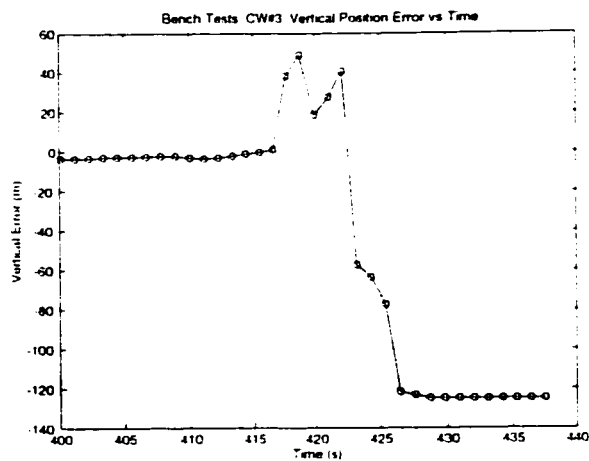


Figure 423.h: CW Interference Bench Test: Vertical Position Error vs Time

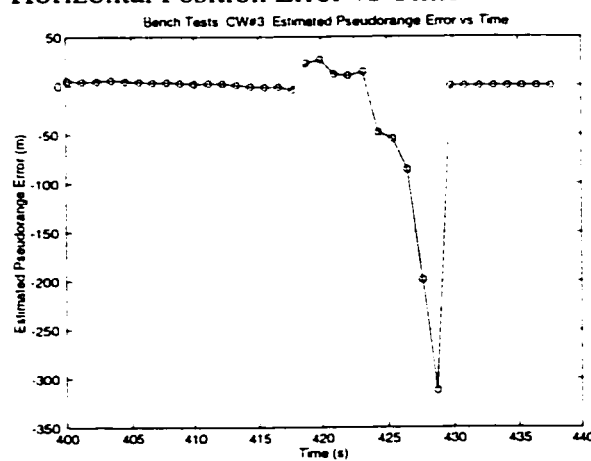


Figure 423.i: CW Interference Bench Test: Estimate of Pseudorange Error vs Time

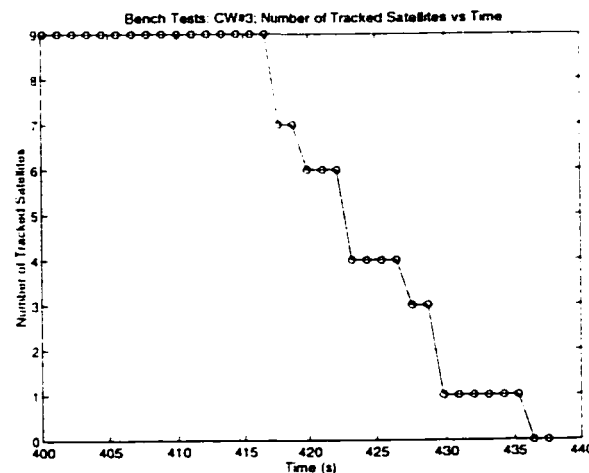


Figure 423.j: CW Interference Bench Test: Number of Tracked Satellites vs Time

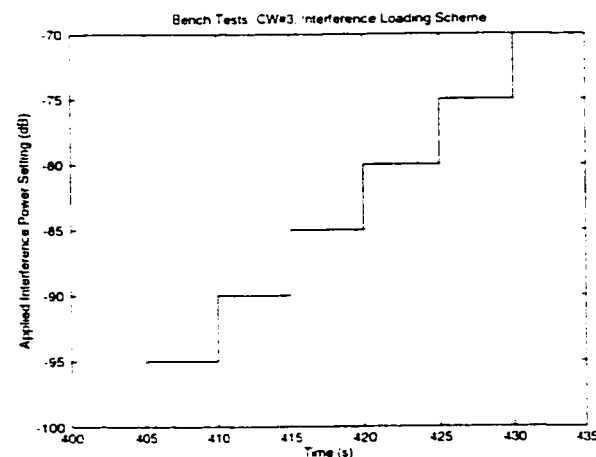


Figure 423.k: CW Interference Bench Test: Interference Load Settings vs Time

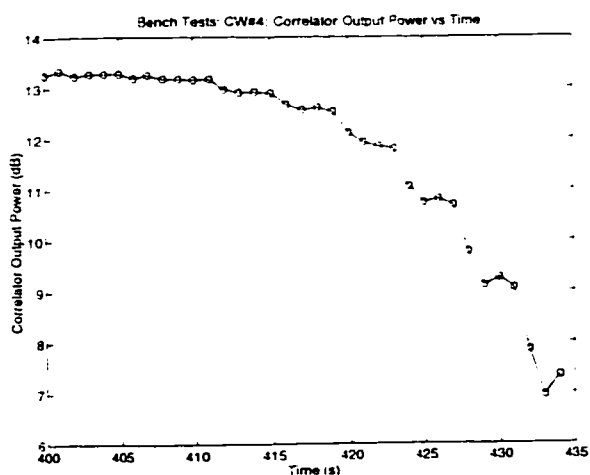


Figure 424.a: CW Interference Bench Test: Correlator Output Power vs Time

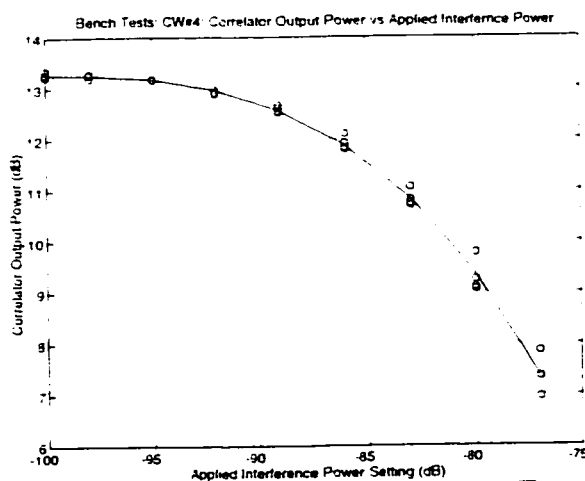


Figure 424.b: CW Interference Bench Test: Correlator Output Power vs Applied Interference

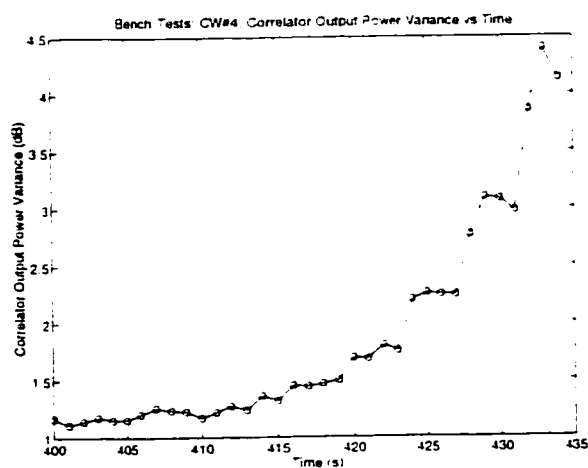


Figure 424.c: CW Interference Bench Test: Correlator Output Power Variance vs Time

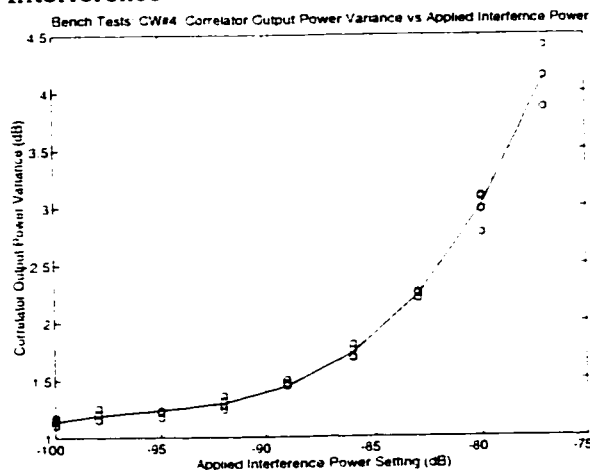


Figure 424.d: CW Interference Bench Test: Correlator Output Power Variance vs Applied Interference

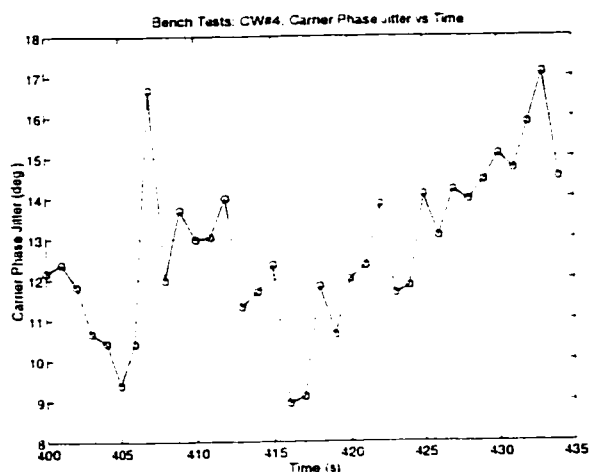


Figure 424.e: CW Interference Bench Test: Carrier Phase Jitter vs Time

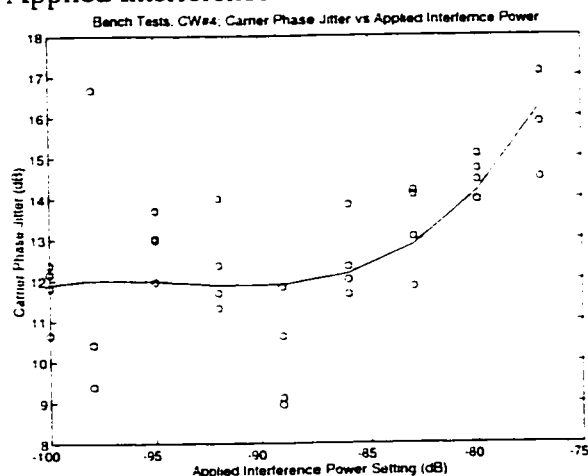


Figure 424.f: CW Interference Bench Test: Carrier Phase Jitter vs Applied Interference

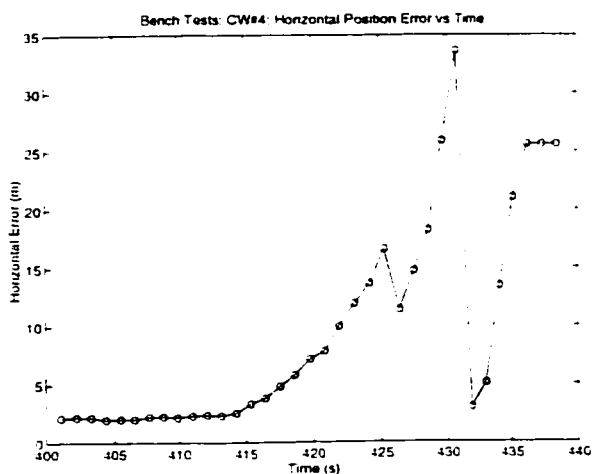


Figure 424.g: CW Interference Bench Test: Horizontal Position Error vs Time

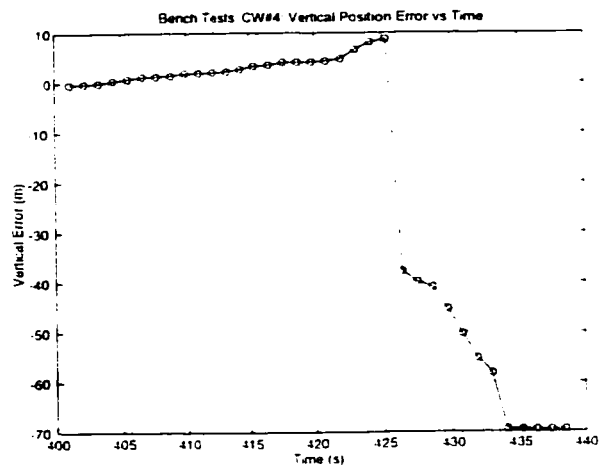


Figure 424.h: CW Interference Bench Test: Vertical Position Error vs Time

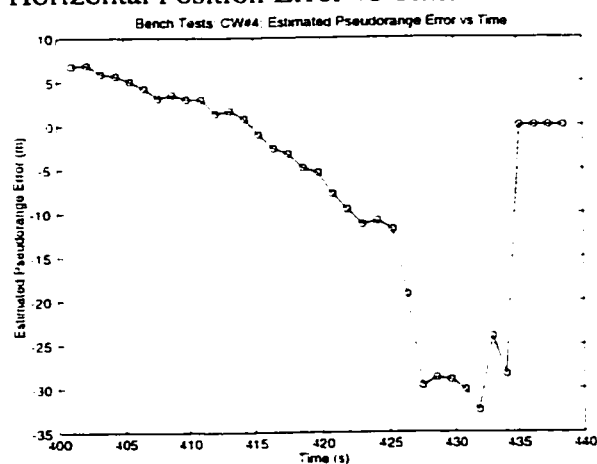


Figure 424.i: CW Interference Bench Test: Estimate of Pseudorange Error vs Time

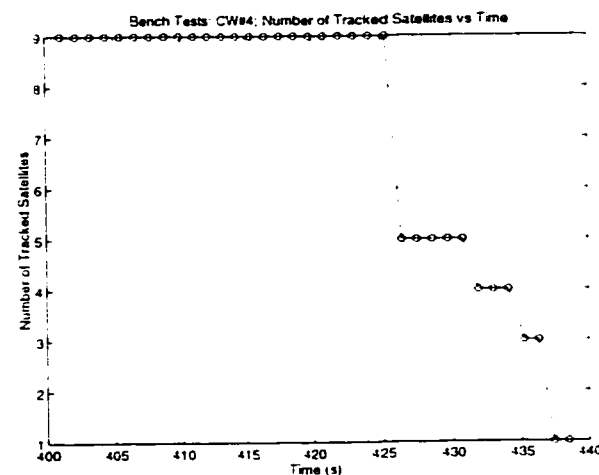


Figure 424.j: CW Interference Bench Test: Number of Tracked Satellites vs Time

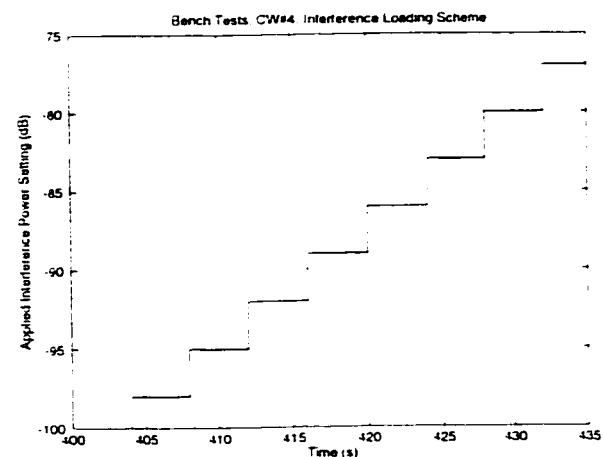


Figure 424.k: CW Interference Bench Test: Interference Load Settings vs Time

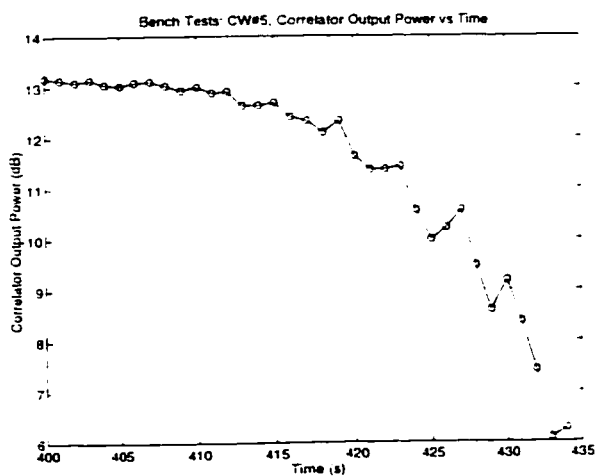


Figure 425.a: CW Interference Bench Test: Correlator Output Power vs Time

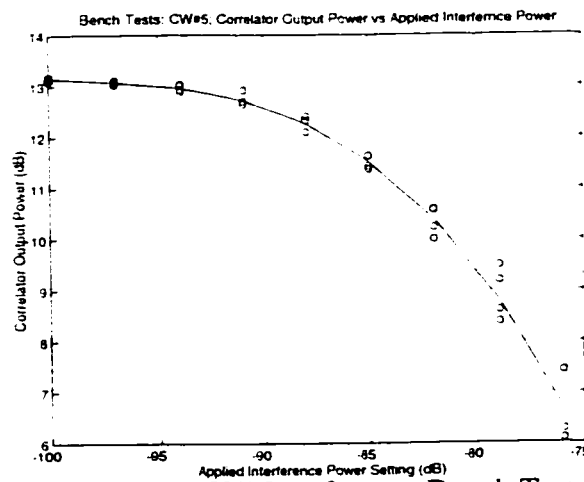


Figure 425.b: CW Interference Bench Test: Correlator Output Power vs Applied Interference

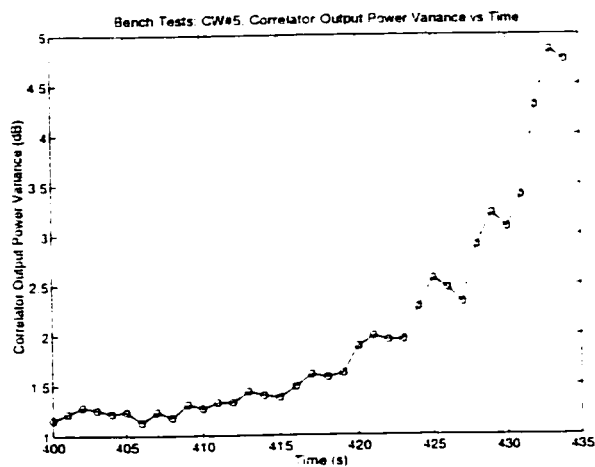


Figure 425.c: CW Interference Bench Test: Correlator Output Power Variance vs Time

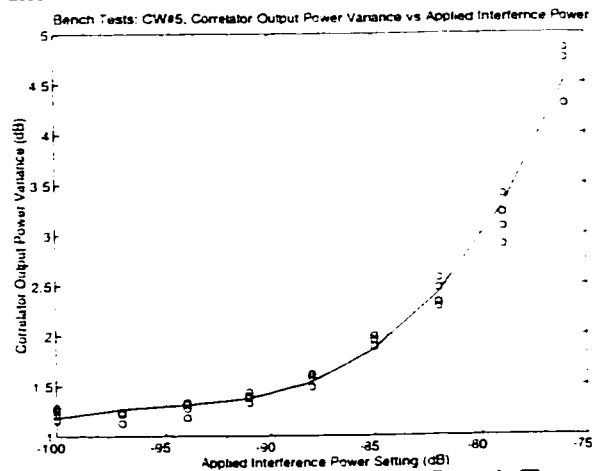


Figure 425.d: CW Interference Bench Test: Correlator Output Power Variance vs Applied Interference

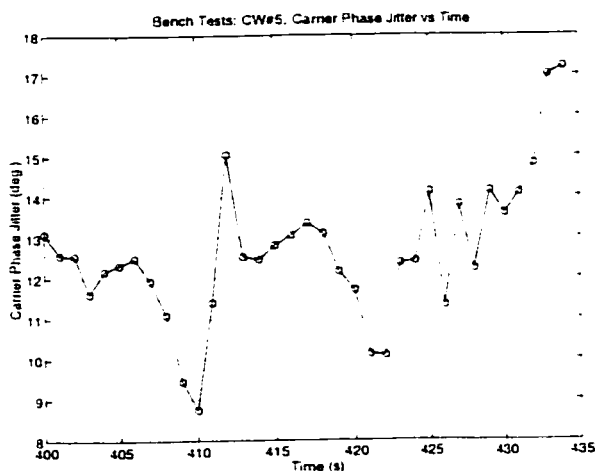


Figure 425.e: CW Interference Bench Test: Carrier Phase Jitter vs Time

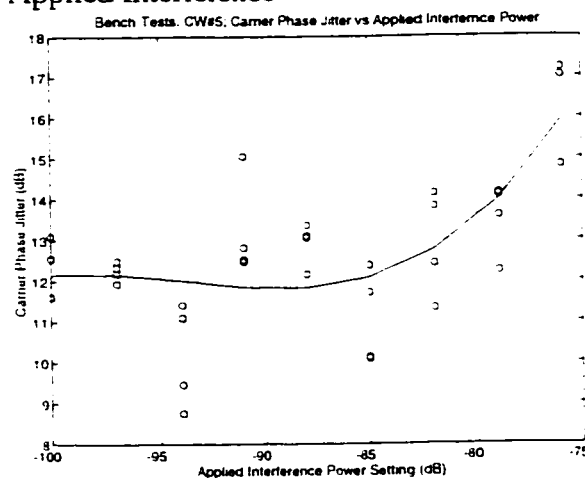


Figure 425.f: CW Interference Bench Test: Carrier Phase Jitter vs Applied Interference

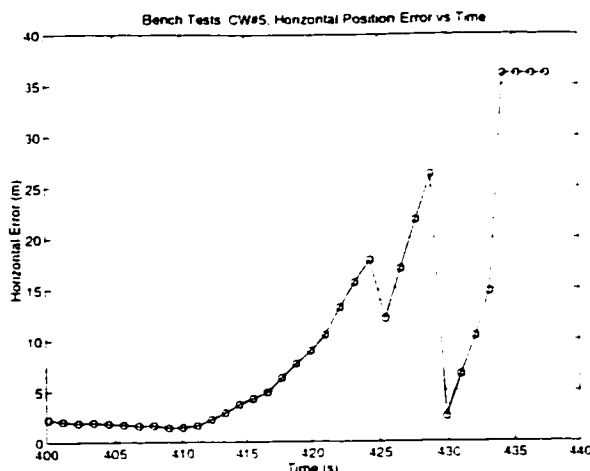


Figure 425.g: CW Interference Bench Test: Horizontal Position Error vs Time

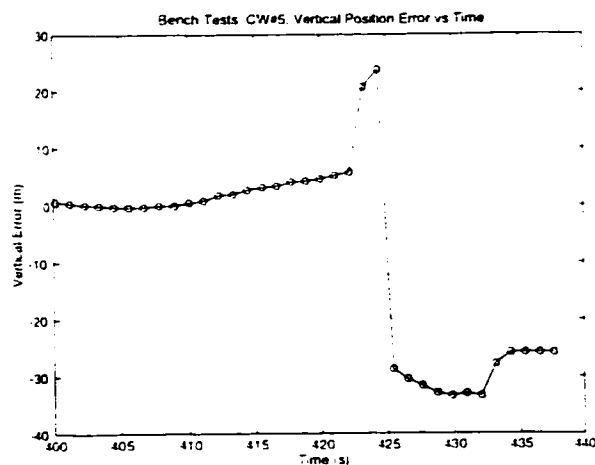


Figure 425.h: CW Interference Bench Test: Vertical Position Error vs Time

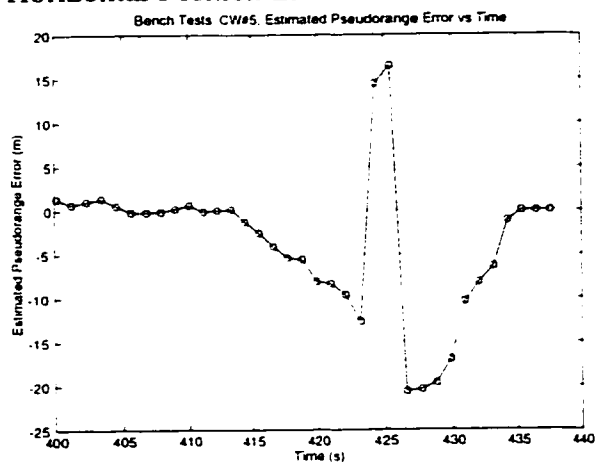


Figure 425.i: CW Interference Bench Test: Estimate of Pseudorange Error vs Time

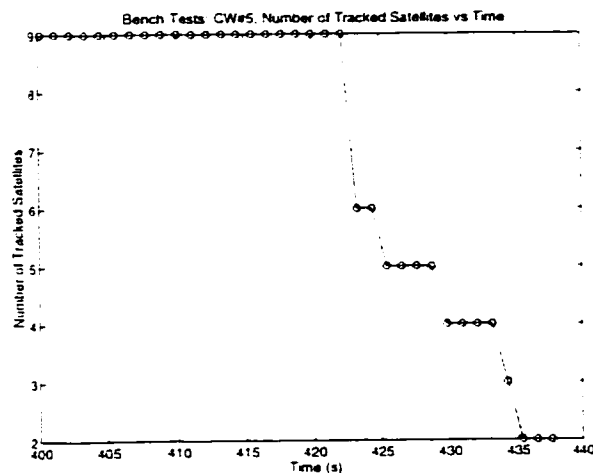


Figure 425.j: CW Interference Bench Test: Number of Tracked Satellites vs Time

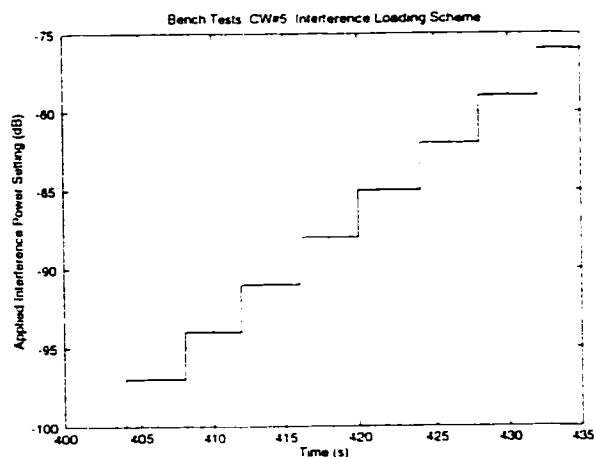


Figure 425.k: CW Interference Bench Test: Interference Load Settings vs Time

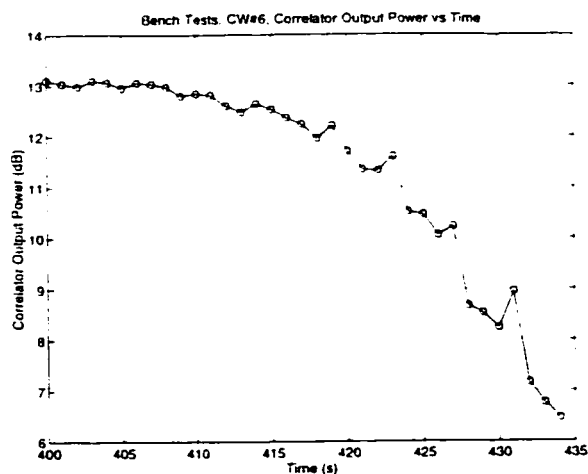


Figure 426.a: CW Interference Bench Test: Correlator Output Power vs Time

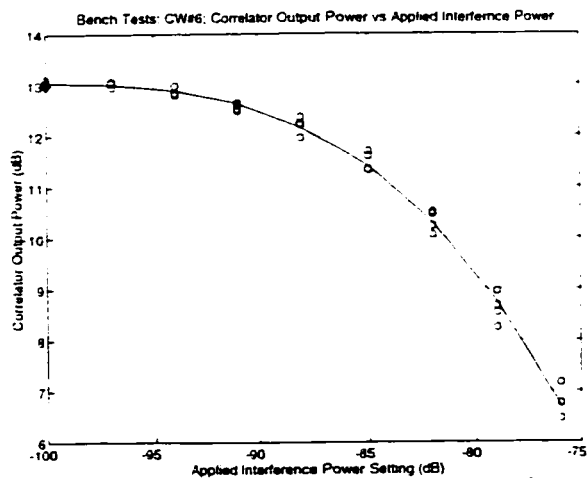


Figure 426.b: CW Interference Bench Test: Correlator Output Power vs Applied Interference

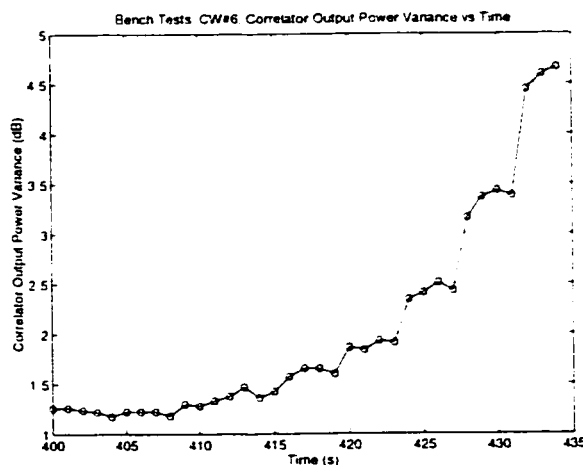


Figure 426.c: CW Interference Bench Test: Correlator Output Power Variance vs Time

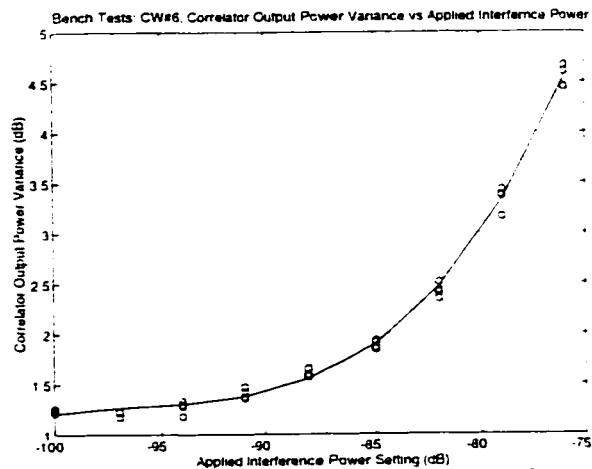


Figure 426.d: CW Interference Bench Test: Correlator Output Power Variance vs Applied Interference

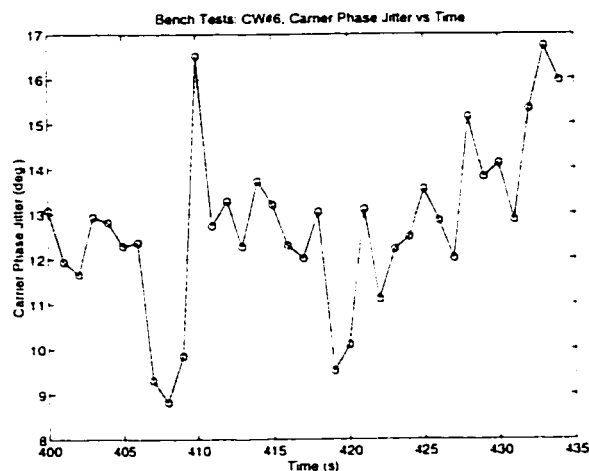


Figure 426.e: CW Interference Bench Test: Carrier Phase Jitter vs Time

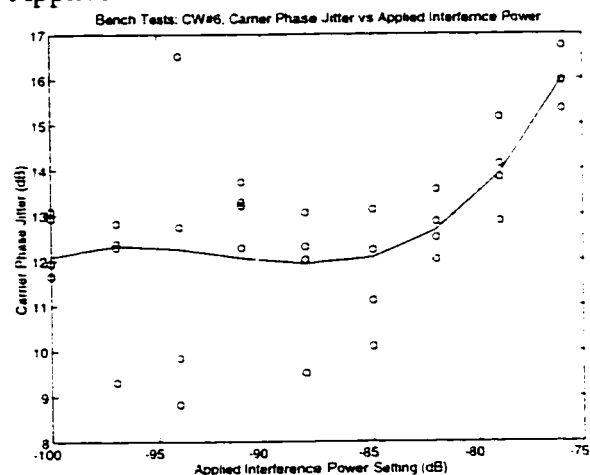


Figure 426.f: CW Interference Bench Test: Carrier Phase Jitter vs Applied Interference

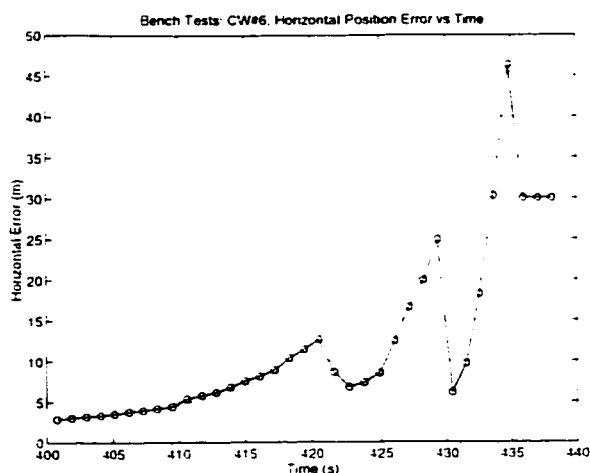


Figure 426.g: CW Interference Bench Test: Horizontal Position Error vs Time

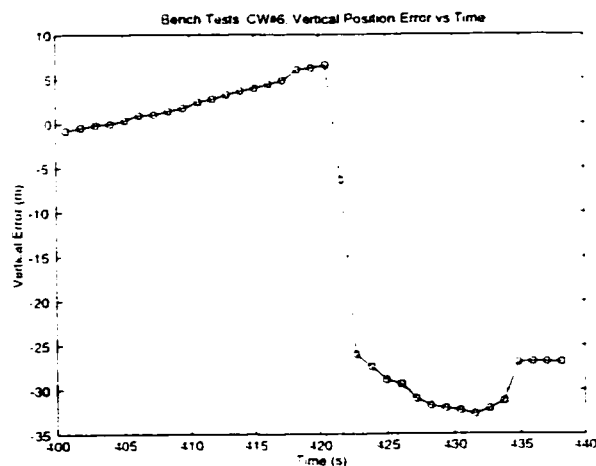


Figure 426.h: CW Interference Bench Test: Vertical Position Error vs Time

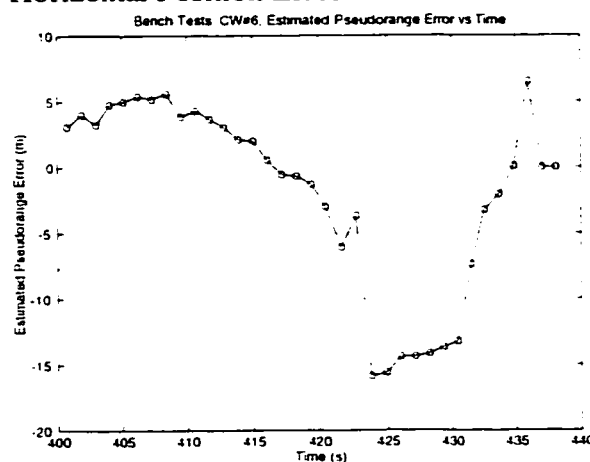


Figure 426.i: CW Interference Bench Test: Estimate of Pseudorange Error vs Time

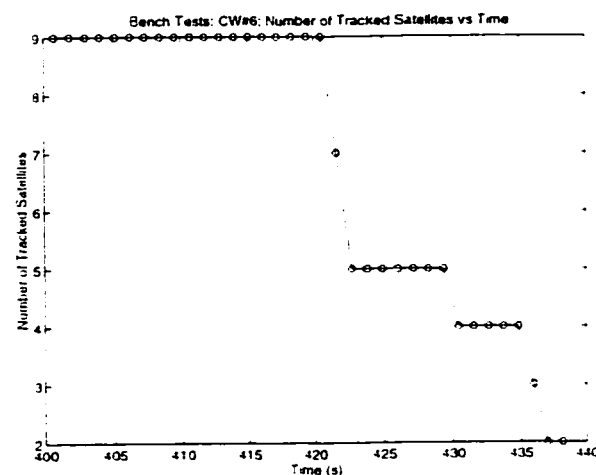


Figure 426.j: CW Interference Bench Test: Number of Tracked Satellites vs Time

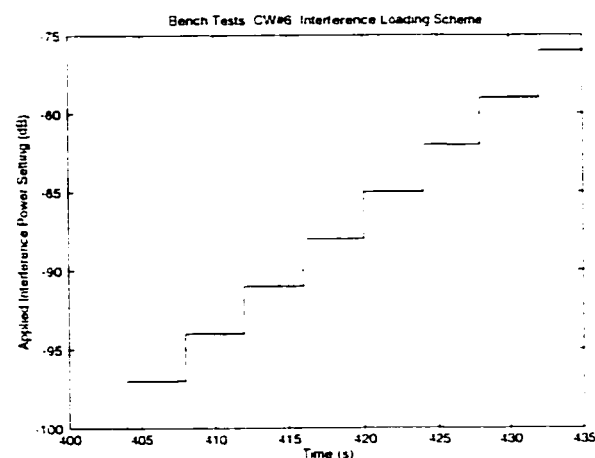


Figure 426.k: CW Interference Bench Test: Interference Load Settings vs Time

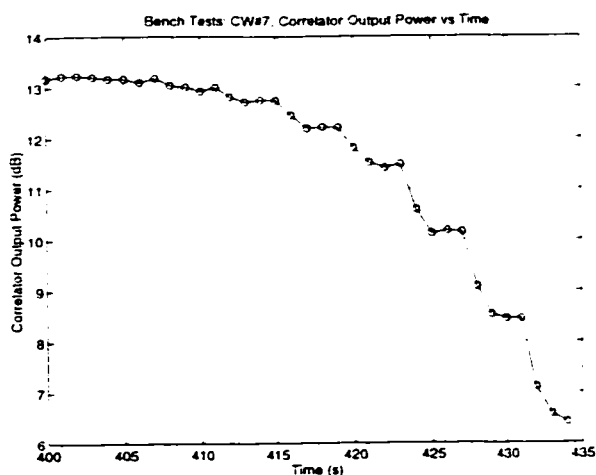


Figure 427.a: CW Interference Bench Test: Correlator Output Power vs Time

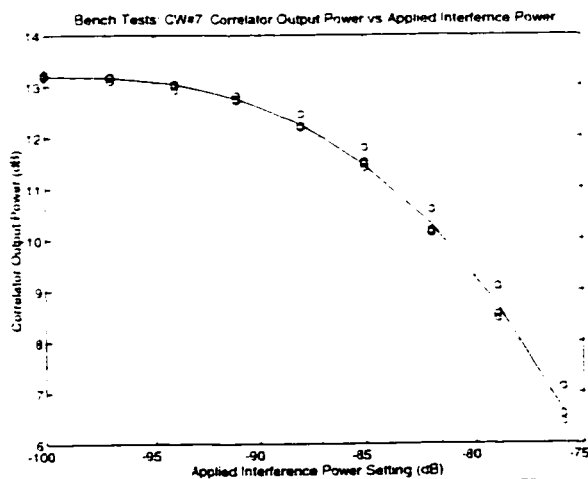


Figure 427.b: CW Interference Bench Test: Correlator Output Power vs Applied Interference

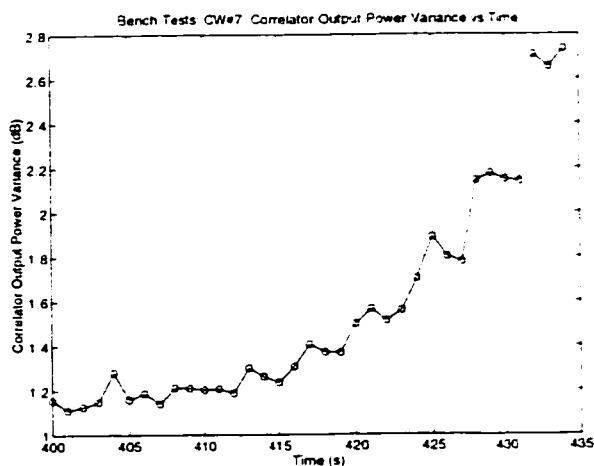


Figure 427.c: CW Interference Bench Test: Correlator Output Power Variance vs Time

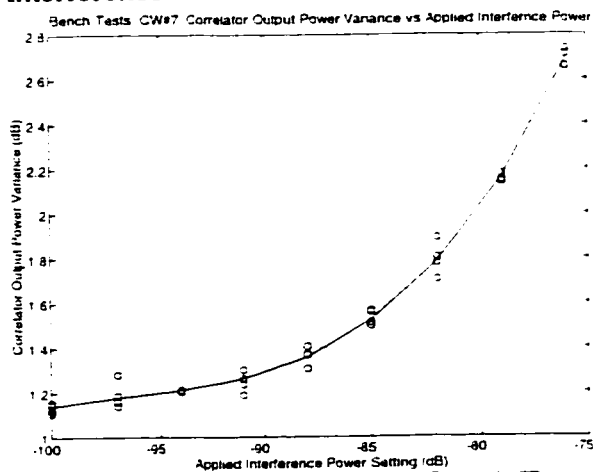


Figure 427.d: CW Interference Bench Test: Correlator Output Power Variance vs Applied Interference

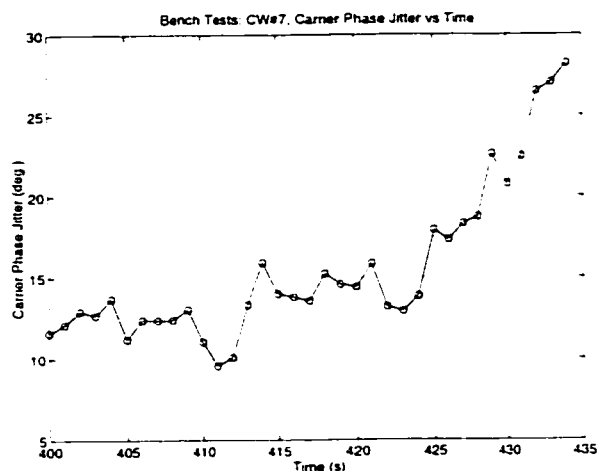


Figure 427.e: CW Interference Bench Test: Carrier Phase Jitter vs Time

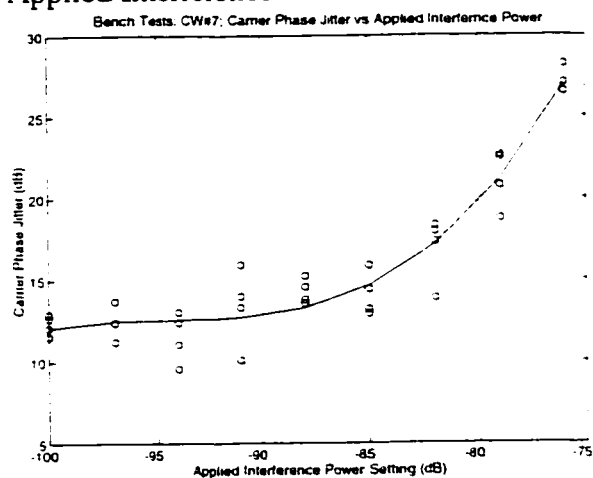


Figure 427.f: CW Interference Bench Test: Carrier Phase Jitter vs Applied Interference

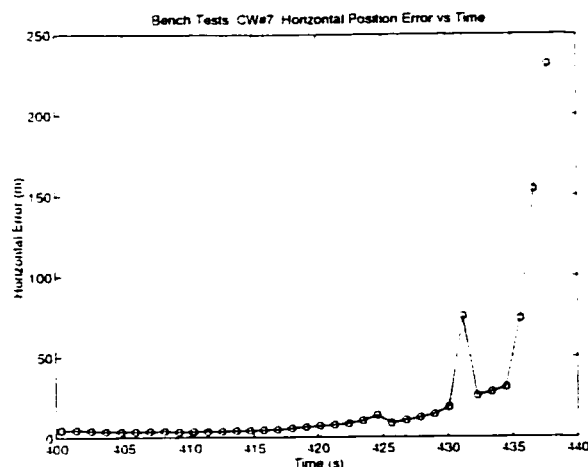


Figure 427.g: CW Interference Bench Test: Horizontal Position Error vs Time

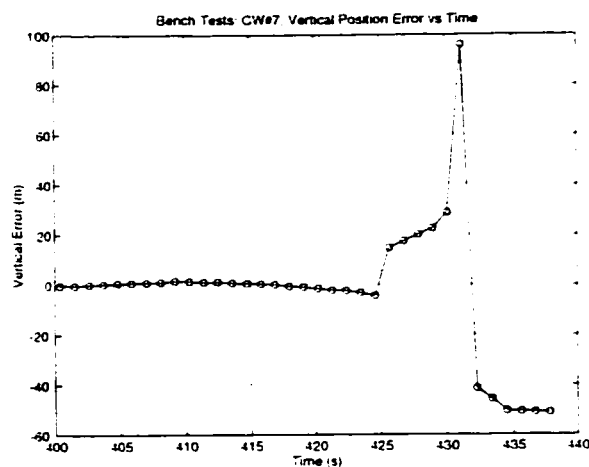


Figure 427.h: CW Interference Bench Test: Vertical Position Error vs Time

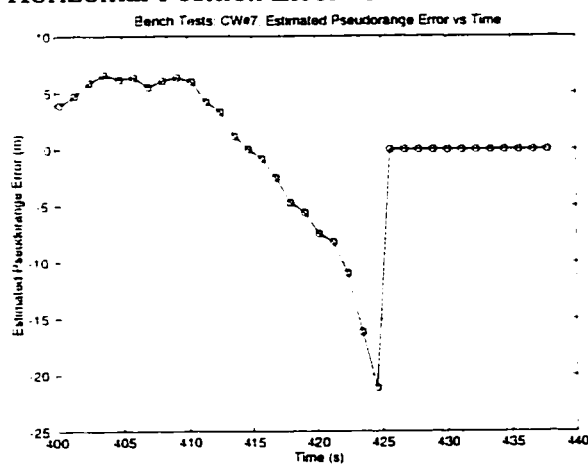


Figure 427.i: CW Interference Bench Test: Estimate of Pseudorange Error vs Time

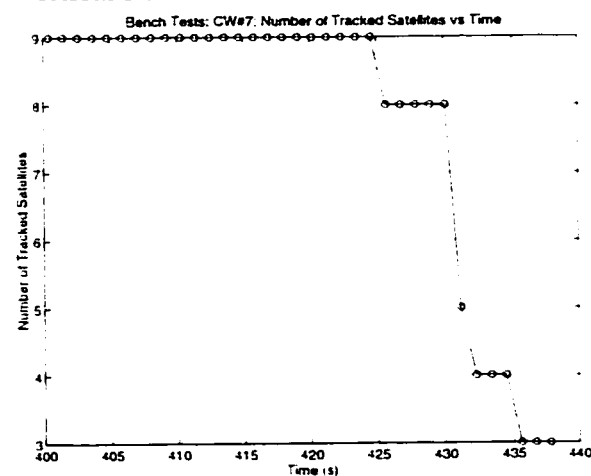


Figure 427.j: CW Interference Bench Test: Number of Tracked Satellites vs Time

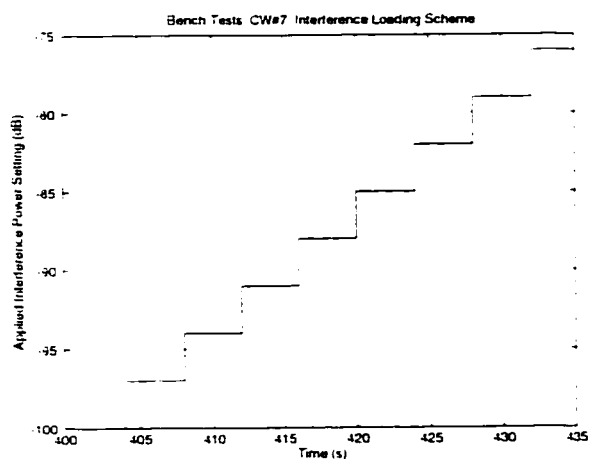


Figure 427.k: CW Interference Bench Test: Interference Load Settings vs Time

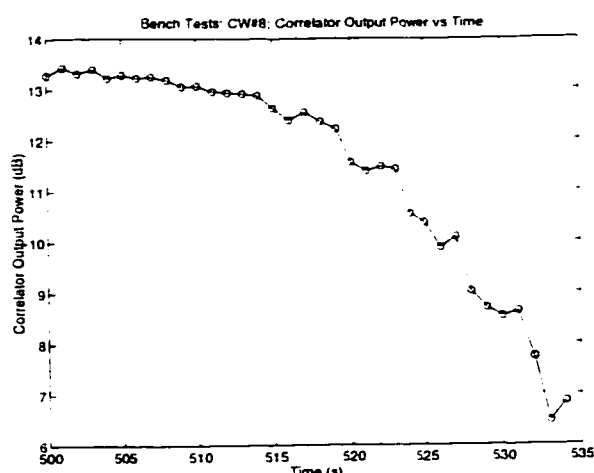


Figure 428.a: CW Interference Bench Test: Correlator Output Power vs Time

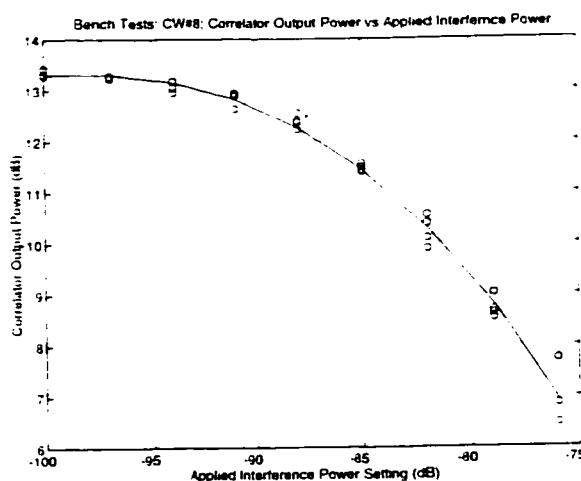


Figure 428.b: CW Interference Bench Test: Correlator Output Power vs Applied Interference

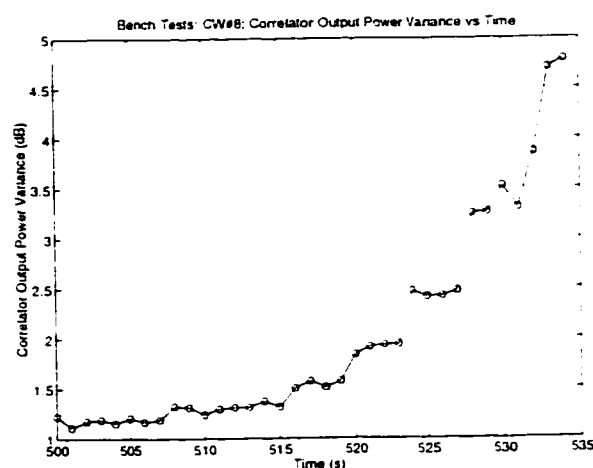


Figure 428.c: CW Interference Bench Test: Correlator Output Power Variance vs Time

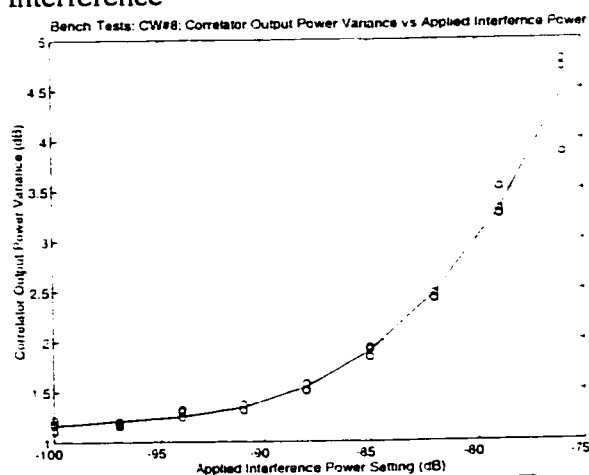


Figure 428.d: CW Interference Bench Test: Correlator Output Power Variance vs Applied Interference

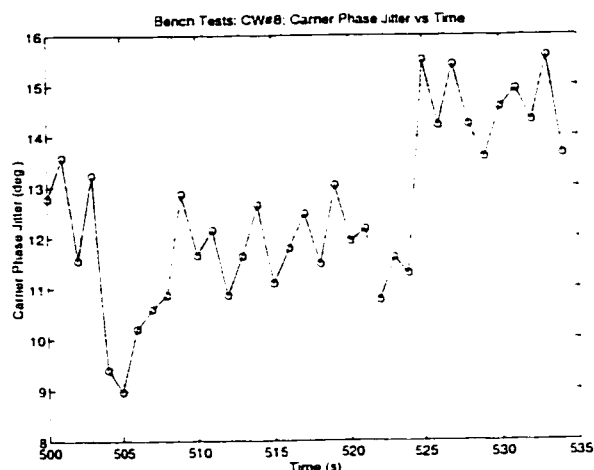


Figure 428.e: CW Interference Bench Test: Carrier Phase Jitter vs Time

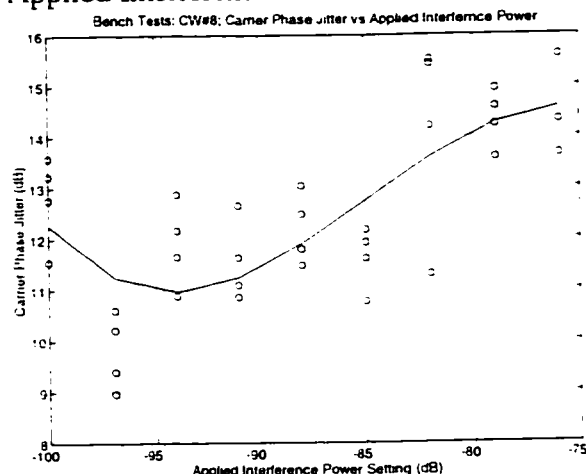


Figure 428.f: CW Interference Bench Test: Carrier Phase Jitter vs Applied Interference

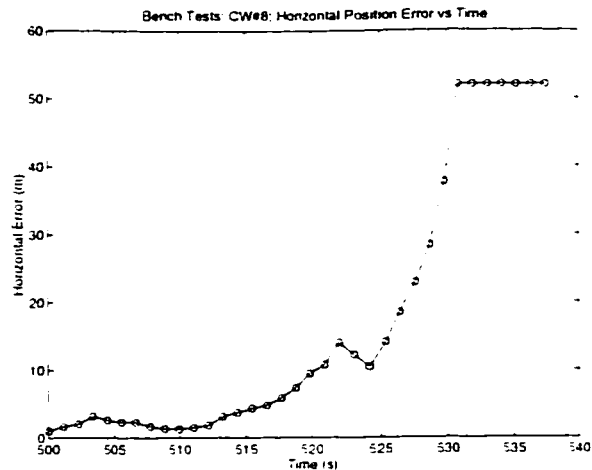


Figure 428.g: CW Interference Bench Test: Horizontal Position Error vs Time

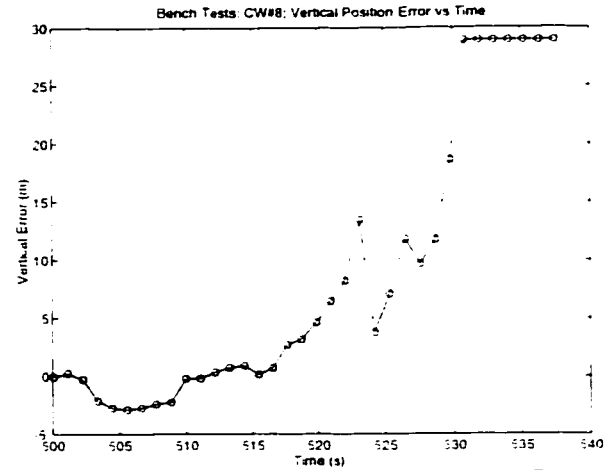


Figure 428.h: CW Interference Bench Test: Vertical Position Error vs Time

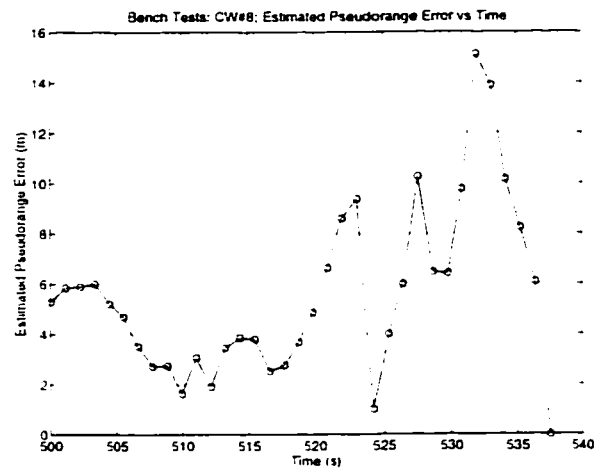


Figure 428.i: CW Interference Bench Test: Estimate of Pseudorange Error vs Time

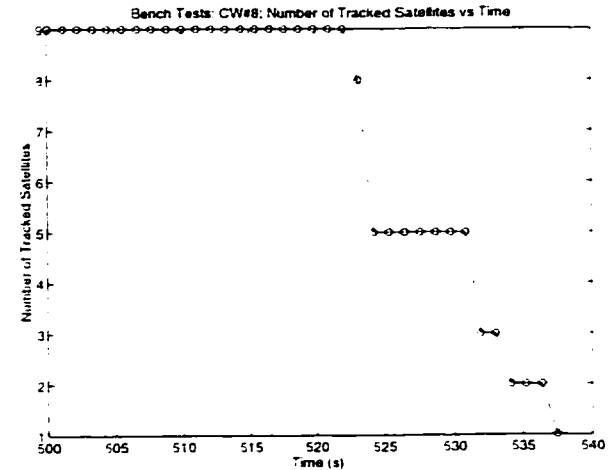


Figure 428.j: CW Interference Bench Test: Number of Tracked Satellites vs Time

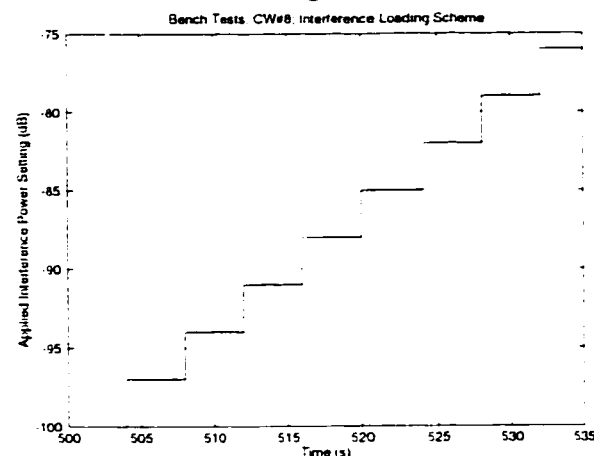


Figure 428.k: CW Interference Bench Test: Interference Load Settings vs Time

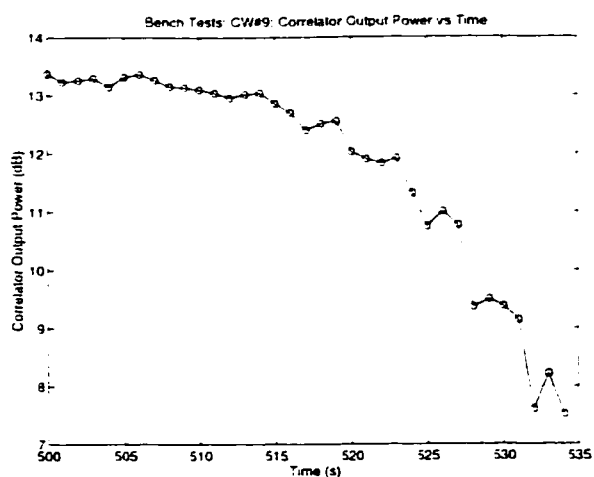


Figure 429.a: CW Interference Bench Test: Correlator Output Power vs Time

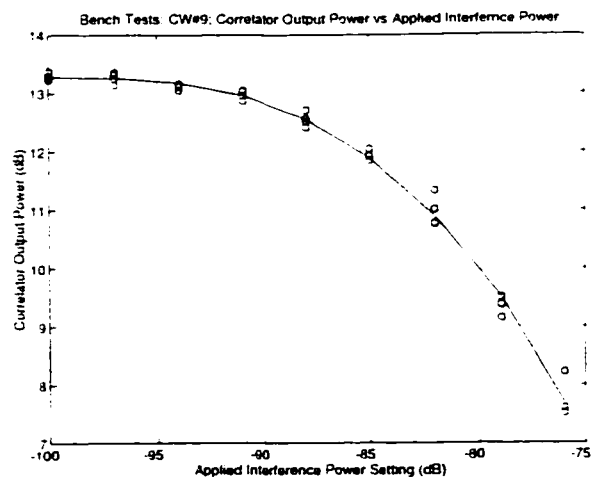


Figure 429.b: CW Interference Bench Test: Correlator Output Power vs Applied Interference

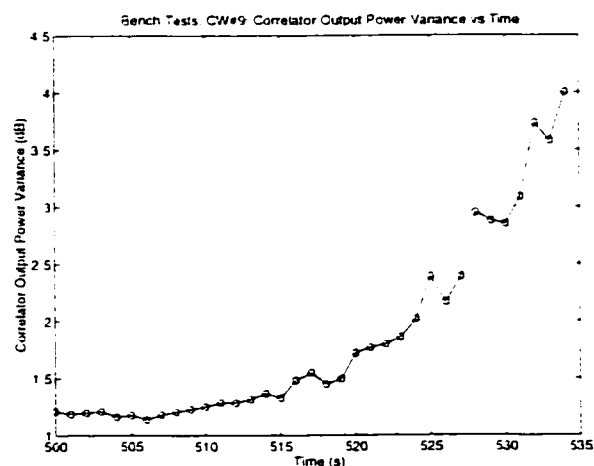


Figure 429.c: CW Interference Bench Test: Correlator Output Power Variance vs Time

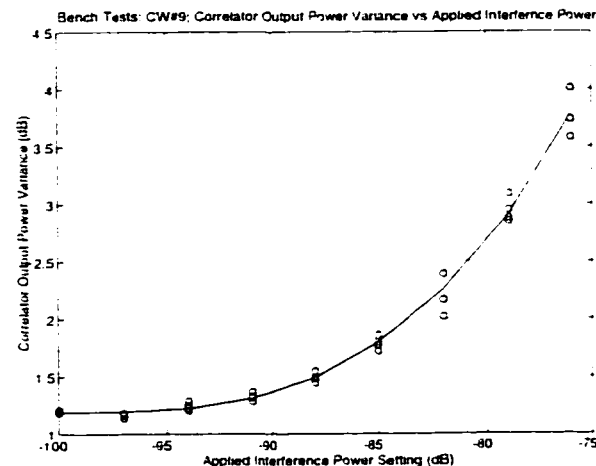


Figure 429.d: CW Interference Bench Test: Correlator Output Power Variance vs Applied Interference

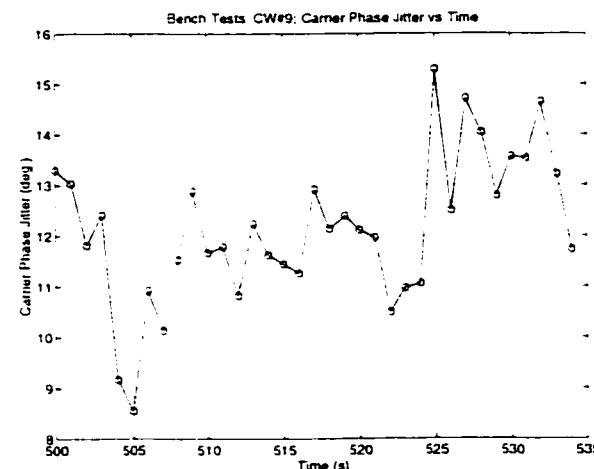


Figure 429.e: CW Interference Bench Test: Carrier Phase Jitter vs Time

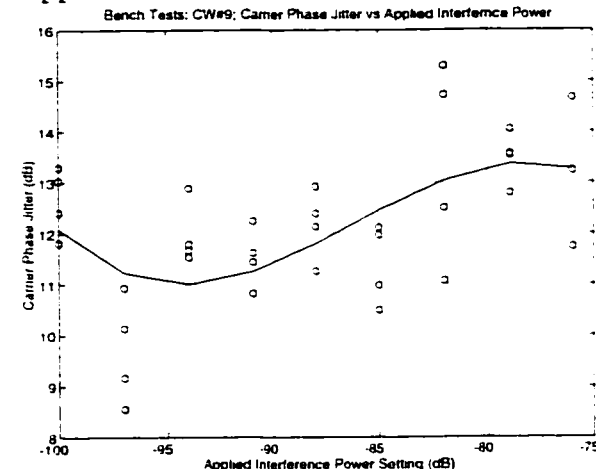


Figure 429.f: CW Interference Bench Test: Carrier Phase Jitter vs Applied Interference

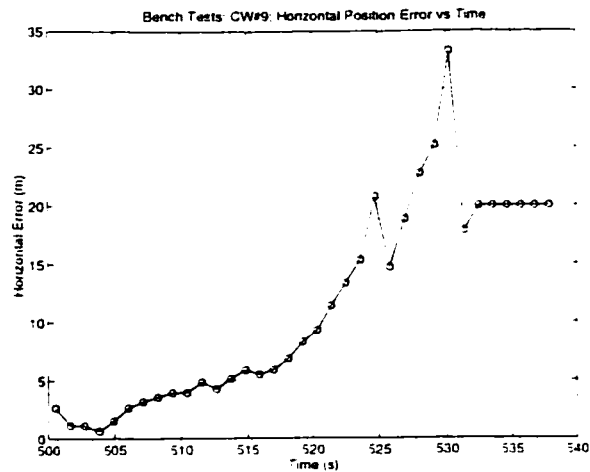


Figure 429.g: CW Interference Bench Test: Horizontal Position Error vs Time

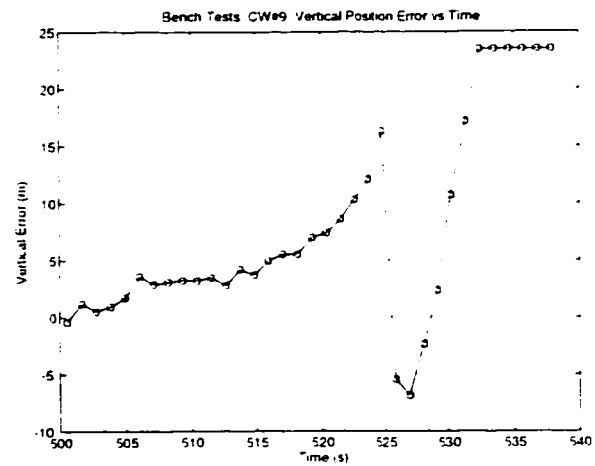


Figure 429.h: CW Interference Bench Test: Vertical Position Error vs Time

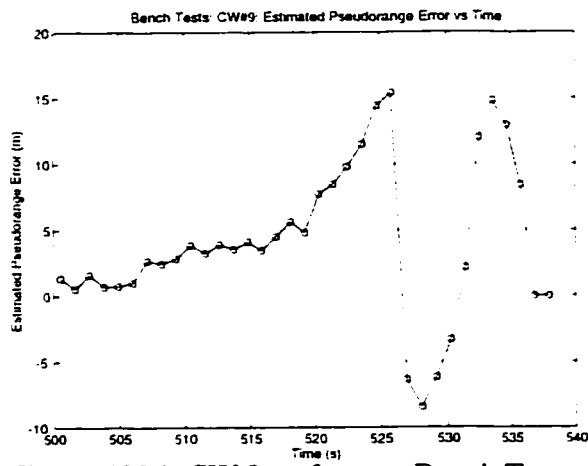


Figure 429.i: CW Interference Bench Test: Estimate of Pseudorange Error vs Time

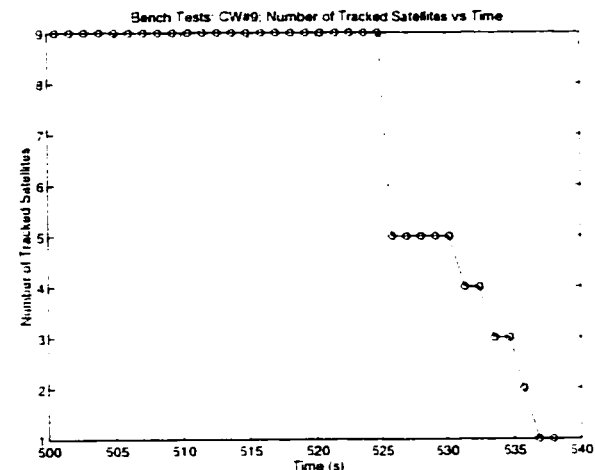


Figure 429.j: CW Interference Bench Test: Number of Tracked Satellites vs Time

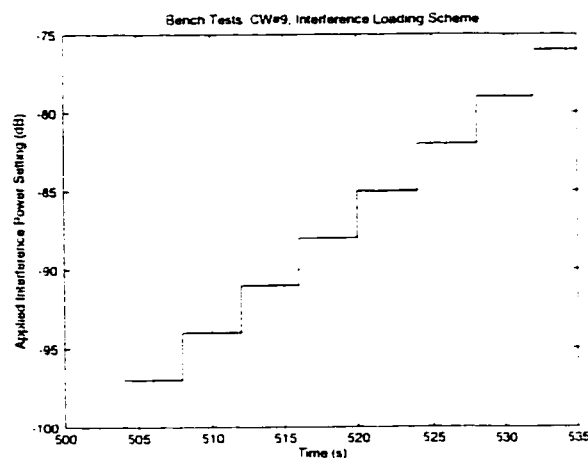


Figure 429.k: CW Interference Bench Test: Interference Load Settings vs Time

Appendix C

Simulation Software Code

This section contains the software code for the GPS simulation used in this study. The entire software was written in C programming language. The following are the main modules of the software:

GPS_SIM.C:

Contains the main() function, and core loop, including the code and carrier tracking loops, software correlators, A-D digitizer and AGC loop. This module calls all other modules, including the initialization routines and noise generators. This module writes output to a number of files: the first, "run_info.dat", contains a comprehensive summary of the results of all runs performed in a batch. This file is a tab delimited ASCII text file. Column headers are written to another file, "run_infoHdr.out". Another set of output files contain detailed results of each run. These files are named using the format "f_gXXcYYpZZdAAaBB.dat", where XX = AWGN power level, in dB, for the specific run:

YY = CW interference power level, in dB, for the specific run;

ZZ = pulse duty cycle, in %, for the specific run;

AA = Doppler offset of applied CW interference, in Hz;

BB = Attenuation of tracked GPS signal, in dB;

GOLDCODE.C:

Generates signals for GPS satellites, including code and carrier. Reads in almanac from the ASCII text file "almanac.in", and initializes all parameters related to the GPS signal.

INIT.C:

Performs initialization routines on code startup. initializes all global variables, setting receiver defaults. Reads in the receiver configuration input file "rcv_spec.in" and configures the receiver and noise environment as specified in the input file. Initializes all GPS signal and receiver channel parameters, including initial code and carrier phases and signal strengths for all satellites in view. To add pseudolites to the simulation, simply modify one of the broadcast satellite signals to have the desired Gold code, carrier phase and signal amplitude. This module writes out details of the receiver configuration to the file "rcv_spec.out".

GAUSSIAN.C

Generates white Gaussian numbers used for creating receiver thermal noise and any applied AWGN interference.

R250.C

Subroutine to gaussian.c, for generation of Gaussian numbers;

RANDLCG.C

Subroutine to gaussian.c, for generation of Gaussian numbers;

```

FILE:      GPS_SIM.C.C
AUTHOR:    Awele Ndili
START DATE: June 1, 1996

```

```

/*
 * GPS_Sim.c - GPS Receiver Simulation
 * by Awele Ndili
 * June 1, 1996
 *
 */

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

#include "defines.h"
#include "structs.h"
#include "funcprot.h"
#include "globals.h"

#define dcoQuantize(x) ((x<-0.7071)?-2:((x<0.0)?-1:((x<0.7071)?1:2)))
#define QNTZPRINT
fprintf(fp_QntzLvls,"%12.6f\t%12.6f\t%12.6f\n",AdativeQuantizerLevel1,Ad
ativeQuantizerLevel2,AdativeQuantizerLevel3)
//=====
//FILE *fp_PRerr;
FILE *fp_IF4Analog;
FILE *fp_Summary;
FILE *fp_SubRunInfo;
FILE *fp_RunSchedule;
double if3FilterBuffer;
int includeSVAttenuationTMP;

//_____

int main ()
{
    long i;
    char fname[40];
    long dTime;          //clock_t dTime;
    double sqrt2;        // 970502

    sqrt2 = sqrt(2);

    OPEN_FILE(fp_Summary,"run_infoHdr.out","w");          // Open
Output summary File
    // OPEN_FILE(fp_IF4Analog,"if4_analog.dat","w");      // Open IF4
Output File

    /* Run Summary File: run_infoc.dat

```

```

1    Vis. SVs
2    AWGNSR_dB
3    QTZ-
4    QTZ+
5    MaxPreAcqIQ2
6    MaxIQ2
7    RunTime
8    CPUTime
9    InitCodeOffset
10   FinalCodeOffset
11   InitFreqOffset
12   FinalFreqOffset
13   FinalCorlk
14   FinalCarfrlk
15   CWISR_dB
16   RunId
17   DutyCycle
18   CWDopplerOffset
19   SVAttenuation
*/

    fprintf(fp_Summary, "Vis. SVs\tAWGNSR_dB\tQTZ-
\tQTZ+\tMaxPreAcqIQ2\tMaxIQ2\tRunTime\tCPUTime");
    fprintf(fp_Summary, "\tInitCodeOffset\tFinalCodeOffset\tInitFreqOff
set\tFinalFreqOffset");
    fprintf(fp_Summary, "\tFinalCorlk\tFinalCarfrlk\tCWISR_dB\tRunId\tD
utyCycle\tCWDopplerOffset\tSVAttenuation\n");
    fclose(fp_Summary);
    OPEN_FILE(fp_Summary, "run_info.dat", "w");
    fclose(fp_Summary);

    initialize();
    printoutGlobals();
    bg_awgn_std_dev =
gpsSignal[0].carrierAmplitude/sqrt2*sqrt(pow(10.0,BG_AWGNSR_dB/10.0));
    OPEN_FILE(fp_RunSchedule, "run_spec.in", "r");    // Open Runs
Schedule File

    // Read Varying Parameters and apply run
    while(fscanf(fp_RunSchedule, "%f\t%f\t%f\t%f\t%f", &AWGNSR_dB, &CWISR
_dB, &pulseDutyCycle, &CWInterferenceDopplerOffset, &SV_ATTNUATN_dB) != EOF
&& AWGNSR_dB != -1){
        sprintf(fname, "f_g%dc%dp%dd%da%d.dat", includeAWGN ?
(int)AWGNSR_dB: 99, includeCWInterference ?
(int)CWISR_dB: 99, includePULSEInterference ? (int) (pulseDutyCycle*100):
99, (int) CWInterferenceDopplerOffset, includeSVAttenuation ? (int)
SV_ATTNUATN_dB : 99);
        awgn_std_dev =
gpsSignal[0].carrierAmplitude/sqrt2*sqrt(pow(10.0,AWGNSR_dB/10.0));
        // the 1.10149 factor takes into account ampl. of tracked SV
signal
        cw_interference_ampl = gpsSignal[0].carrierAmplitude *
sqrt(pow(10.0,CWISR_dB/10.0));
        sv_attenuation_ratio = pow(10.0,SV_ATTNUATN_dB/10.0);

```

```

        if (includePULSEInterference){
            pulseWidth = pulseDutyCycle*accumulationTime;
            pulseStartWindow = accumulationTime - pulseWidth;
        }
        printf("%s\n", fname);

        OPEN_FILE(fp_SubRunInfo, fname, "w");          // Open Run
summary File

        initialize();

        maxPreAcquistionIQ2=maxIQ2=0;
        for (i=0; i<QTZHISTORY; i++){
            stableQTZlevelP[i] = 0;
            stableQTZlevelM[i] = 0;
        }
        dTime = 0; //clock();
        includeSVAttenuationTMP = includeSVAttenuation;
        RunTimeLoop();

        fclose(fp_SubRunInfo);

        for (i=1; i<QTZHISTORY; i++){
            stableQTZlevelP[0] += stableQTZlevelP[i];
            stableQTZlevelM[0] += stableQTZlevelM[i];
        }

        OPEN_FILE(fp_Summary, "run_info.dat", "a");    // Reopens
Output summary File

        fprintf(fp_Summary, "%d\t%10.4f\t%12.6g\t%12.6g\t%1u\t%1u\t%12.6g\t%1d\t%1d\t%12.6g\t%12.6g\t%12.6g\t%1d\t%1d\t%1d\t%1d\t%12.6g\t%10.4f\t%10.4f\n",
            NumberOfSimulatedSVs, AWGNSR_dB,

            stableQTZlevelP[0]/QTZHISTORY, stableQTZlevelM[0]/QTZHISTORY,

            maxPreAcquistionIQ2, maxIQ2, totalRunTime, (long) (0/*clock()*/-dTime)/60,

            IF3_INIT_CAOFFSET-DCO_INIT_CAOFFSET,
            gpsSignal[0].codePhase-
            (CH[0].dcoCodePhaseE+CH[0].dcoCodePhaseL)/2.0,
            Init_CarrFreqErr, gpsSignal[0].carrierFrequency-
            CH[0].dcoCarrierFreq, CH[0].corlk, CH[0].carfrlk, (int)CWISR_dB, run_id,

            pulseDutyCycle, CWInterferenceDopplerOffset, SV_ATTNUATN_dB);

        fclose(fp_Summary);
    }

    BEEP(3);
    return 1;
}

```

```

void RunTimeLoop(void)
{
    int i;
    while (realTime < totalRunTime)
    {
        doOneAccum();
        CarrierTrackingLoop();

        i = CH[0].iCURACCUM;    // Temp

        ProcAccumPhase();

        if (screenPrinting)

            printf("Lock=%d\tdFreq=%8fHz\tdCode=%f\tsNR=%5.2fdB\tCLId-
SNR=%5.2fdB\n",CH[0].corlk,gpsSignal[0].carrierFrequency-
CH[0].dcoCarrierFreq,gpsSignal[0].codePhase-
(CH[0].dcoCodePhaseE+CH[0].dcoCodePhaseL)/2.0,10*log10(CH[0].ACCUM[i].I2
_Plus_Q2/NOISE_FLOOR_FLOAT),10*log10(CH[0].CdLI/NOISE_FLOOR_FLOAT));

        fprintf(fp_SubRunInfo,"%12.6g\t%d\t%d\t%u\t%12.6f\t%12.6f\t%12.6f\t
%12.6f\t%12.6f\t%8f\t%d\t%d\t%d\t%d\t%lu\t%d\n",
            realTime,CH[0].corlk,CH[0].carfrlk,CH[0].coasting,

            AdativeQuantizerLevel1,AdativeQuantizerLevel2,AdativeQuantizerLeve
13,
                ((long ) fmod(gpsSignal[0].codePhase, 1023))-((long)
fmod(CH[0].dcoCodePhaseE,1023)),
                gpsSignal[0].codePhase-
(CH[0].dcoCodePhaseE+CH[0].dcoCodePhaseL)/2.0,
                gpsSignal[0].codeRate-CH[0].dcoCodeRate,
                gpsSignal[0].carrierFrequency-CH[0].dcoCarrierFreq,

            CH[0].ACCUM[i].IP,CH[0].ACCUM[i].ID,CH[0].ACCUM[i].QP,CH[0].ACCUM[
i].QD,
                CH[0].ACCUM[i].I2_Plus_Q2,whichFLL);

        /* Run Detail File: f_gXcXpXdXaX.dat (X = Gaussion, CW, Pulse,
Doppler, Attenuate values)
1    realTime
2    CH[0].corlk
3    CH[0].carfrlk
4    CH[0].coasting
5    AdativeQuantizerLevel1
6    AdativeQuantizerLevel2
7    AdativeQuantizerLevel3
8    ((long ) fmod(gpsSignal[0].codePhase, 1023))-((long)
fmod(CH[0].dcoCodePhaseE,1023))
9    gpsSignal[0].codePhase-
(CH[0].dcoCodePhaseE+CH[0].dcoCodePhaseL)/2.0
10   gpsSignal[0].codeRate-CH[0].dcoCodeRate
11   gpsSignal[0].carrierFrequency-CH[0].dcoCarrierFreq

```

```

12     CH[0].ACCUM[i].IP
13     CH[0].ACCUM[i].ID
14     CH[0].ACCUM[i].QP
15     CH[0].ACCUM[i].QD
16     CH[0].ACCUM[i].I2_Plus_Q2
17     whichFLL
    */

    accumCnt++;

}

return;

}

//=====

void doOneAccum(void)
{
    chanstruc *chPtr;
    accumstruc *A;
    SV_Signal *aSig;
    int i,j,k,subSampIdx;
    double endTime;

    double if4SigAnalog;
    double dcoSigAnalogIP;
    double dcoSigAnalogQP;
    double dcoSigAnalogID;
    double dcoSigAnalogQD;
    double pulseStartTime;
    double pulseEndTime;
    double filterTmpBuffer;

    int if4SigDig;
    int dcoSigDigIP;
    int dcoSigDigQP;
    int dcoSigDigID;
    int dcoSigDigQD;

    int if3CodeIdx;
    int dcoCodeIdxE;
    int dcoCodeIdxL;

    double noiseCarrierPhase;

    // Empty out the current accumulation buffers on each channel
    for (i=0; i<MAXCHANNELS; i++)
    {
        A = &CH[i].ACCUM[CH[i].iNXTACCUM];
        A->IP = 0;
        A->QP = 0;
        A->ID = 0;
        A->QD = 0;
    }
}

```



```

endTime = realTime + accumulationTime;

if (includePULSEInterference){
    if (pulsePositionIsRandom){
        pulseStartTime = realTime +
pulseStartWindow*((double)rand() * RAND_MAX);
    }
    else{
        pulseStartTime = realTime;
    }
    pulseEndTime = pulseStartTime + pulseWidth;
}

while ( realTime < endTime )
{
    for(subSampIdx=0;subSampIdx<4;subSampIdx++){
        *** SubSampling
        Starts here ***
        // Get Real Sig Values (before Quantization): if4 =
cos(2*pi*fc*Time)
        if4SigAnalog=0.0;
        for (i=0; i<MaxAlmanacSize; i++)
        {
            aSig = &gpsSignal[i];
            if (aSig->elevation> SV_Elevation_Mask && aSig->prn >
0)
            {
                if3CodeIdx      = (int ) fmod(aSig->codePhase,
1023); // Obtain CA code indices
                // Get Real Sig Values (before Quantization):
if4 = cos(2*pi*fc*Time)
                if(goldCode[aSig->prn-1][if3CodeIdx])
                    if4SigAnalog  += aSig-
>carrierAmplitude*(-cos(aSig->carrierPhase));
                else
                    if4SigAnalog  += aSig-
>carrierAmplitude*cos(aSig->carrierPhase );

                if(i==0) // SV Being Tracked
                    noiseCarrierPhase = aSig->carrierPhase;

                // Update incoming signal Code and Carrier
phases
                aSig->carrierPhase += aSig-
>twoPIfcSubSamplePeriod; //->twoPIfcSamplePeriod; // in radians
                aSig->codePhase += aSig-
>codeRateSubSamplePeriod; //->codeRateSamplePeriod; // in Chips
            } // Shift two lines up, to update unused satellites
        }

        if (includeSVAttenuationTMP)
            { // 971016 ANN - SV Signal attenuation

```

```

        if (realTime>delayB4SVAttenuation)
        {
            gpsSignal[0].carrierAmplitude /=
sqrt(sv_attenuation_ratio); // apply once, then ...
            includeSVAttenuationTMP = 0; // turn off this
check
        }

        if (includeBGAWGN)
        { // 970504 ANN - Background AWGN
            box_muller2();
            if4SigAnalog += (Ni*cos(noiseCarrierPhase) +
Nq*sin(noiseCarrierPhase));
        }

        if ((!includePULSEInterference) || (includePULSEInterference
&& realTime>pulseStartTime && realTime<pulseEndTime)){

            if (includeAWGN && realTime>delayB4AWGN)
            { // Generate and Add Gaussian noise
                box_muller();
                if4SigAnalog += (Ni*cos(noiseCarrierPhase) +
Nq*sin(noiseCarrierPhase));
            }

            if (includeCWInterference &&
realTime>delayB4CWInterference)
            { // Generate and Add CW
                if4SigAnalog +=
cw_interference_ampl*cos(cwInterference.carrierPhase);
//cwInterference.carrierAmplitude * ...
            }

            cwInterference.carrierPhase +=
cwInterference.twoPIfcSubSamplePeriod; // 970928 Update cw
interference carrier phase

            // Implementation of Butterworth filter, na=nb ANN 970523
            // y(n) = b(0)x(n) + b(1)x(n-1) + b(2)x(n-3) + ... +
b(nb)x(n-nb)
            // - a(1)y(n-1) - a(2)y(n-2) - .... -
a(na)y(n-na)

            inputX[filterIdx] = if4SigAnalog; // Store x(n)
            filterTmpBuffer = dFilterB[0]*if4SigAnalog; // Accumulate
y(n)

            k = filterIdx+FILTERBUFFERSIZE;

            for (i=1;i<filterSizePl;i++)
            {
                j = (k-i)%FILTERBUFFERSIZE;

```

```

        filterTmpBuffer += (dFilterB[i]*inputX[j] -
dFilterA[i]*outputY[j]);
    }

    outputY[filterIdx] = filterTmpBuffer;    // Store y(n)
    filterIdx = (filterIdx+1) % FILTERBUFFERSIZE;

    //fprintf(fp_IF4Analog,"%g\t%g\t%g\n",realTime,if4SigAnalog,filter
TmpBuffer);
    } // for(subSampIdx=0;subSampIdx<4;subSampIdx++) *** SubSampling
Ends here ***

    // Quantize if3 Signal
    if4SigDig = if4Quantize( filterTmpBuffer); //if4SigAnalog
);

    // Next, for each channel, perform the respective
correlation
    for (i=0; i<MAXCHANNELS; i++)
    {
        chPtr = &CH[i];
        A = &chPtr->ACCUM[chPtr->iNXTACCUM];

        // Obtain CA code indices for dco sigs (=discretize
time for CA codes)
        dcoCodeIdxE = (int ) fmod(chPtr->
>dcoCodePhaseE,1023);
        dcoCodeIdxL = (int ) fmod(chPtr->
>dcoCodePhaseL,1023);

        if(goldCode[0][dcoCodeIdxE]){
            dcoSigAnalogIP = -cos(chPtr->dcoCarrierPhaseL);
            dcoSigAnalogQP = sin(chPtr->dcoCarrierPhaseL);
        }
        else {
            dcoSigAnalogIP = cos(chPtr->dcoCarrierPhaseL);
            dcoSigAnalogQP = -sin(chPtr->dcoCarrierPhaseL);
        }

        if(goldCode[0][dcoCodeIdxL]){
            dcoSigAnalogID = -cos(chPtr->dcoCarrierPhaseE);
            dcoSigAnalogQD = sin(chPtr->dcoCarrierPhaseE);
        }
        else {
            dcoSigAnalogID = cos(chPtr->dcoCarrierPhaseE);
            dcoSigAnalogQD = -sin(chPtr->dcoCarrierPhaseE);
        }

        // Quantize dco Signals
        dcoSigDigIP = dcoQuantize( dcoSigAnalogIP );    //
inphase, prompt
        dcoSigDigQP = dcoQuantize( dcoSigAnalogQP );    //
quadrature, prompt

```

```

        dcoSigDigID = dcoQuantize( dcoSigAnalogID );    //
inphase, dithered (or early)
        dcoSigDigQD = dcoQuantize( dcoSigAnalogQD );    //
quadrature, dithered

        A->IP += dcoSigDigIP*if4SigDig;
        A->QP += dcoSigDigQP*if4SigDig;
        A->ID += dcoSigDigID*if4SigDig;
        A->QD += dcoSigDigQD*if4SigDig;

        // Update dco Code and Carrier phases
        chPtr->dcoCarrierPhaseE += chPtr-
>twoPiFrSamplePeriod;
        chPtr->dcoCarrierPhaseL += chPtr-
>twoPiFrSamplePeriod;
        chPtr->dcoCodePhaseE += chPtr-
>dcoCodeRateSamplePeriod; // in Chips
        chPtr->dcoCodePhaseL += chPtr-
>dcoCodeRateSamplePeriod; // in Chips
    }

    // Update realtime
    realTime += SamplePeriod;
}

/**
// Update Channel and Accum Structure
chPtr->IM1 = A->I_Prompt;
chPtr->QM1 = A->Q_Prompt;
A->I_Prompt = A->IP + A->ID;
A->Q_Prompt = A->QP + A->QD;
A->I2_plus_Q2 = ((A->IP*A->IP - A->QP*A->QP)>>2);
*/

/*
I_PromptM1 = I_Prompt;
Q_PromptM1 = Q_Prompt;
I_Prompt = I_AccumP + I_AccumD;
Q_Prompt = Q_AccumP + Q_AccumD;
I2_plus_Q2 = (I_Prompt*I_Prompt+Q_Prompt*Q_Prompt)/4;
*/

#ifdef NOTONAMAC
if (kbhit()){
    char c = getch();

    switch(c) {
        case '.': // More Freq
            if3CarrierFreq+=100;
            twoPiFcSamplePeriod = twoPi * if3CarrierFreq * SamplePeriod;
            break;
        case ',': // less freq
            if3CarrierFreq-=100;
            twoPiFcSamplePeriod = twoPi * if3CarrierFreq * SamplePeriod;

```

```

        break;
    case ')': // More Freq
        if3CodeRate++;
        if3CodeRateSamplePeriod = if3CodeRate * SamplePeriod;
        break;
    case '(': // less freq
        if3CodeRate--;
        if3CodeRateSamplePeriod = if3CodeRate * SamplePeriod;
        break;
    case 'q':
        exit(0);
        break;
    default:
        while (!kbhit)
            continue;
        getch();
    }
}
#endif
return;
}

//=====
==

void CarrierTrackingLoop(void)
{
    chanstruc *chPtr;
    accumstruc *A;
    int i;
    // unsigned epoch_check; /* Contents of EPOCH_CHECK register */
    long carrdco_update; /* Update to the carrier DCO */
    long deltaI, deltaQ; /* Change in I and Q over last ms */
    // long Inew, Qnew; /* I and Q rotated by carrier phase error */
    /*
    // long delta_phase_error; /* Phase error of Inew and Qnew */
    long phase_change; /* Change in carrier phase over last ms */
    long delta_carrier_dot; /* Change in carrier frequency dot over last
ms */
    long delta_carrier; /* Change in carrier frequency over last
ms */
    long discriminator; /* Coarse measurement of carrier phase
change */
    // long cos_phase_error; /* Cosine of the I and Q phase error */
    // long sin_phase_error; /* Sine of the I and Q phase error */

    for (i=0; i<MAXCHANNELS; i++)
    {
        chPtr = &CH[i];
        A = &chPtr->ACCUM[chPtr->iNXTACCUM];

        A->I_Prompt = (long)(A->IP + A->ID);
        A->Q_Prompt = (long)(A->QP + A->QD);
    }
}

```

```

    /// A->I2_Plus_Q2 = (unsigned long)((A->IP*A->IP + A->QP*A-
>QP)>>2);

    whichFLL = 0;
    if(chPtr->corlk && chPtr->coasting==0)
    {
        phase_change = iSAR(A->Q_Prompt*chPtr->IM1 - A-
>I_Prompt*chPtr->QM1 + 16,5);

        if(chPtr->avg_phase_change<3758L)
        { // Perform a discretised Jaffe-Rechtin 2nd Order FLL
            /* Get the change in the carrier frequency
derivative over the
            last sample period. */

            whichFLL = 2;
            delta_carrier_dot = (phase_change<<2);

            /* Get the change in the carrier frequency over
the last sample
            period scaled by 2^13. */

            delta_carrier = iSAR(chPtr->wdot_c + 512L,10) +
(phase_change<<1);

            /* Save the carrier frequency rate for next
time. */

            chPtr->wdot_c += delta_carrier_dot;

            /* Get the carrier DCO update. delta_carrier is
still scaled by
            2^13 but only shift by 11 places because the
scale factor
            for the carrier dco unit to radians/s is
approximately 2^-2. */

            carrdco_update = iSAR(delta_carrier+1024,9);

            /* Update the carrier DCO for this channel. */

            chPtr->CARRDCO += carrdco_update;

            /* Now do the data demodulation loop */

            /* Rotate I and Q by the current phase error */

            /// cos_phase_error = CosApprox(chPtr->phase_error);
            /// sin_phase_error = SinApprox(chPtr->phase_error);

            /// Inew = A->I_Prompt*cos_phase_error + A-
>Q_Prompt*sin_phase_error;
            /// Qnew = A->Q_Prompt*cos_phase_error - A-
>I_Prompt*sin_phase_error;

```

```

        /// A->IDataBit = Inew>>15;

        /// delta_phase_error = (Inew>=0)? Qnew:-Qnew;

        /* Update the phase error, scaled by 2^13 */

        /// chPtr->phase_error +=
iSAR(delta_phase_error+16384L,15);
    }
    else // Use the 4-quadrant discriminator
    {
        whichFLL = 1;

        deltaI = A->I_Prompt - chPtr->IM1;
        deltaQ = A->Q_Prompt - chPtr->QM1;

        if (labs(A->I_Prompt) > labs(A->Q_Prompt))
            discriminator = (A->I_Prompt > 0L) ?
deltaQ: -deltaQ;
        else
            discriminator = (A->Q_Prompt > 0L) ? -
deltaI: deltaI;

        carrdco_update = iSAR(discriminator+128, 8);
        chPtr->CARRDCO += carrdco_update;

        chPtr->avg_phase_change += iSAR(carrdco_update -
chPtr->avg_phase_change + 512, 10);
    }

    chPtr->dcoCarrierFreq = (double)((chPtr-
>CARRDCO+8)>>4)/(1<<27)*if4SampleRate;
    chPtr->twoPiFrSamplePeriod = twoPi * chPtr-
>dcoCarrierFreq * SamplePeriod; // radians (b/w Samples)
}

    chPtr->IM1 = A->I_Prompt;
    chPtr->QM1 = A->Q_Prompt;

    if(++chPtr->iNXTACCUM>=NACCUM)
        chPtr->iNXTACCUM=0;
    //Not yet needed ++chPtr->AccumPending;
}
}

/*****
void ProcAccumPhase(void)
Process accumulation data
*****/

```

```

void ProcAccumPhase(void)
{
    int i;
    for (i=0; i<MAXCHANNELS; i++)
    {
        ProcAccum(&CH[i]);
    }
}

/*****
****
* Function: void ProcAccum(chanstruc *CHPTR)
*
* Processes the accumulation data for (i) data demodulation, (ii)
bitsync,
* (iii) code and carrier lock monitoring and (iv) code tracking loop
updates.
*
* Input: *CHPTR - parameter block for the channel in question.
*
* Output: None.
*
* Return Value: None.
****
*****/

void ProcAccum(chanstruc *CHPTR)
{
    accumstruc *A;          /* Pointer to the Accum Buffer. */

    A = &CHPTR->ACCUM[CHPTR->iCURACCUM];

    /* Maintain a running sum of the I arm over the past 20ms (1 data
bit is
    20ms long). To maintain a sliding window of length 20ms (20
accumula-
    tions), each time a new I arm sum is added the I arm sum 20
readings
    before needs to be subtracted. */

    ///    CHPTR->IPSUM20 += A->IDataBit
    ///    - CHPTR->ACCUM[(CHPTR->iCURACCUM+NACCUM-20)%NACCUM].IDataBit;

    /* Only do further processing if the channel is not idle (SV!=0). */

    if(CHPTR->SV!=0)
    {
        /* Update the code and carrier lock indicators. */

        UpdateLockIndicators(CHPTR,A);

        /* Check is coast mode should be entered or aborted. */

```



```

        CoastMode(CHPTR);

        /* Only attempt to bitsync and monitor the locks if we're
not        coasting and we have code and carrier lock. */

        if(CHPTR->coasting==0 && CHPTR->corlk!=0 && CHPTR->
>carfrlk!=0)
        ///      BitSync(CHPTR,A->_lms_epoch);
        ///

        /* Only attempt to monitor the locks if we're not coasting.
*/

        if(CHPTR->coasting==0)
        {
            CodeLockMonitor(CHPTR,A->I2_Plus_Q2);
            CarrierLockMonitor(CHPTR);
        }

        CodeTrackingLoop(CHPTR,A);

        /* If the lms epoch counter is zero then a new data bit is
ready for        processing. */

        if(A->_lms_epoch==0)
        ///      ProcessDataBit(CHPTR, (CHPTR->IPSUM20>=0),A->
>_20ms_epoch);
        }

        if(--CHPTR->icURACCUM >= NACCUM)        /* Check if the buffer is
full. */
            CHPTR->icURACCUM=0;
        /*for later    CHPTR->AccumPending--;        /* An accumulation set has
been processed. */
    }

/*****
* Function: void UpdateLockIndicators( chanstruc  *CHPTR, accumstruc
*A)
*
* Updates the code and carrier lock indicators.
*
* Input: *CHPTR - parameter block for the channel in question.
*        *A - accumulation data for the channel in question.
*
* Output: *CHPTR - updates to the code and carrier lock indicators.
*        *A - update of the signal power.
*
* Return Value: None.
*****/

```

```

void UpdateLockIndicators( chanstruc *CHPTR, accumstruc *A)
{
    long i2;                /* The inphase correlation
squared. */
    long q2;                /* The quadrature correlation
squared. */
    long iim1_plus_qqm1;    /* i(k)*i(k-1) + q(k)*q(k-
1). */

    i2 = A->I_Prompt*A->I_Prompt;
    q2 = A->Q_Prompt*A->Q_Prompt;
    A->I2_Plus_Q2 = (i2 + q2)>>2;    /* Power in the signal */
    iim1_plus_qqm1 = (A->I_Prompt*CHPTR->IM1 + A->Q_Prompt*CHPTR->
QM1)>>2;

    CHPTR->CdLI += iSAR(A->I2_Plus_Q2-CHPTR->CdLI+128,8);
    CHPTR->CrfrLI += iSAR(iim1_plus_qqm1 - CHPTR->CrfrLI+2048,12);

    // Keep Track of Max PreAcquisition and Max Sig. Values (for
offline analyses)
    if (A->I2_Plus_Q2>maxIQ2) maxIQ2=A->I2_Plus_Q2;
    if (accumCnt<preAcqAccumCnt && A->I2_Plus_Q2>maxPreAcquistionIQ2)
maxPreAcquistionIQ2 = A->I2_Plus_Q2;
}

/*****
****
* Function: void CoastMode( chanstruc *CHPTR)
*
* If coast mode is active then determines if (i) signal is back or (ii)
* coasting should continue or (iii) loss of signal. If coast mode is not
* active then determines if coast mode should be entered.
*
* Input: *CHPTR - parameter block for the channel in question.
*
* Output: *CHPTR - update to the coast time-out counter.
*
* Return Value: None.
*****/
void CoastMode( chanstruc *CHPTR)
{
    /* If the channel is already in coast mode then determine if it
should continue or abandon (due to a timeout or the signal
re-acquired). */

    if(CHPTR->coasting)
    {
        /* Compare the code and carrier lock indicators against
their
threshold values. If both are above the threshold then
quit
coast mode. */

```

```

        if((CHPTR->CdLI>=Ta) && (CHPTR->CrfrLI>=Ta))
            CHPTR->coasting=0;      /* Quit coast mode */
        else
        {
            /* Still don't have the signal back so decrement the
coast
            counter. If the counter has reached zero then a
timeout occurs
            and code/carrier lock and frame sync are negated. */

            CHPTR->coasting--;
            if(CHPTR->coasting==0)
                /*
Timeout. */
                {
                    CHPTR->carfrlk = CHPTR->bitlk = 0;
                    CHPTR->FrameSync = _FALSE;
                    CHPTR->FrameSyncAndTIC = _FALSE;
                    CHPTR->LostLockDuringLastTIC = _TRUE;
                    CHPTR->LostCodeLockDuringLastTIC = _TRUE;
                    CHPTR->LostCarrierLockDuringLastTIC = _TRUE;
                }
        }
    }
    else    // Not coasting
    {
        /* Coast mode is entered if whilst in carrier lock and bit lock
        either the code or carrier lock indicators drop below the
        thresholds. */

        if(CHPTR->carfrlk /** && CHPTR->bitlk */ && ((CHPTR->
>CrfrLI<Tl)|| (CHPTR->CdLI<Tl)))
            CHPTR->coasting = Coast;      /* Determines the timeout
period. */
    }
}

/*****
* Function: void CodeLockMonitor( chanstruc *CHPTR, unsigned long I2pQ2)
*
* Update the code lock monitor to check if the signal is present or if
it
* has been lost.
*
* Input: *CHPTR - parameter block for the channel in question.
*         I2pQ2 - signal power.
*
* Output: *CHPTR - code lock monitor and initial parameters for code
loop.
*
* Return Value: None.
*****/
void CodeLockMonitor( chanstruc *CHPTR, unsigned long I2pQ2)

```

```

{
    if(I2pQ2 > Ta)
    {
        /* Instantaneous signal power is above the acquisition
threshold.

        Is this the initial acquisition? */

        if(CHPTR->corlk==0)
        {
            /* Yes - initialize all the following. */
            CHPTR->carfrlk = CHPTR->bitlk = 0;
            CHPTR->LostLockDuringLastTIC = _TRUE;
            CHPTR->LostCodeLockDuringLastTIC = _TRUE;
            CHPTR->CdLI = Ta; /* 200000L;
            CHPTR->EML = 0L;
            CHPTR->NEML = 0;

            // ANN
            CHPTR->CODEDCO = (unsigned
long)(2*DCO_CA_Code_Rate*32.0/4SampleRate*(1L<<26));
            CHPTR->dcoCodeRate = (double)((CHPTR-
>CODEDCO+16L)>>5)/(1L<<27)*4SampleRate; /* sets code dco on not-
searching freq
            CHPTR->dcoCodeRateSamplePeriod = CHPTR-
>dcoCodeRate*SamplePeriod; // chips (b/w Samples)

            /// CHPTR->CARRDCO = CHPTR->pCARRDCO + CarrDoppFromClk -
CHPTR->CarrDoppBin;

            /* 770 is the scaling to go from carrier to code
frequency and is
            equal to 1575.42E6/1.023E6/2. The factor of 2
accounts for the
            difference in scaling between the 2 constants
            CARR_DCO_ZERO_DOPPLER and CODE_DCO_ZERO_DOPPLER. */

            /// CHPTR->CODEDCO = CODE_DCO_ZERO_DOPPLER - (long)(CHPTR-
>CARRDCO-CARR_DCO_ZERO_DOPPLER+385L)/770;

            CHPTR->wdot_c = 0;
            CHPTR->CrfrLI = 0;
            CHPTR->corlk=1;
        } // end if corlk==0 (initial acquisition)
    } // endif I2pQ2>Ta

    if(CHPTR->CdLI < T1)
    {
        /* Signal power is below the signal loss threshold. Lose
locks

        on code, carrier, and also lose bit and frame sync. */
        CHPTR->corlk = CHPTR->carfrlk = CHPTR->bitlk = 0;
        CHPTR->FrameSync = _FALSE;
        CHPTR->FrameSyncAndTIC = _FALSE;
        CHPTR->LostLockDuringLastTIC = _TRUE;
    }
}

```

```

        CHPTR->LostCodeLockDuringLastTIC = _TRUE;
        ///    CHPTR->CARRDCO = CHPTR->pCARRDCO + CarrDoppFromClk -
CHPTR->CarrDoppBin;
        ///    CHPTR->CODEDCO = CHPTR->pCODEDCO + CodeSrchIncr +
CodeDoppFromClk + CHPTR->CodeDoppBin;
        CHPTR->CODEDCO = (unsigned
long)(2*DCO_CA_Code_SRate*32.0/if4SampleRate*(1L<<26));
        CHPTR->dcoCodeRate = (double)((CHPTR-
>CODEDCO+16L)>>5)/(1L<<27)*if4SampleRate;    // sets code dco on not-
searching freq
        CHPTR->dcoCodeRateSamplePeriod = CHPTR-
>dcoCodeRate*SamplePeriod; // chips (b/w Samples)
    }
}

/*****
****
* Function: void CarrierLockMonitor( chanstruc *CHPTR)
*
* Update the carrier lock monitor to check if the signal is present or
if it
* has been lost.
*
* Input: *CHPTR - parameter block for the channel in question.
*
* Output: *CHPTR - carrier lock monitor and initial parameter for
carrier
*
*          loop.
*
* Return Value: None.
*****/
void CarrierLockMonitor( chanstruc *CHPTR)
{
    if(CHPTR->carfrlk==0)    /* No carrier frequency
lock. */
    {
        if(CHPTR->CrfrLI > Ta)    /* Got carrier frequency
lock. */
        {
            CHPTR->carfrlk=1;
            CHPTR->phase_error = 0;
            CHPTR->avg_phase_change = 12860L;
        }
    }
    else
    {
        if(CHPTR->CrfrLI <= Tl)    /* Lost carrier frequency
lock. */
        {
            /* We have just lost carrier lock.  Indicate carrier
lock and
            also set the carrier loss-of-lock flag.*/

```

```

        CHPTR->carfrlk = 0;
        CHPTR->LostLockDuringLastTIC = _TRUE;
        CHPTR->LostCarrierLockDuringLastTIC = _TRUE;
        CHPTR->phase_error = 0;
        CHPTR->avg_phase_change = 12860L;
    }
}

/*****
****
* Function: void CodeTrackingLoop( chanstruc *CHPTR, accumstruc *A)
*
* Updates the code tracking loop.
*
* Input: *CHPTR - parameter block for the channel in question.
*        *A - accumulation data for the channel in question.
*
* Output: *CHPTR - updates to the code tracking parameters.
*
* Return Value: None.
*****/
void CodeTrackingLoop( chanstruc *CHPTR, accumstruc *A)
{
    long id2;                /* Inphase dither
squared. */
    long qd2;                /* Quadrature dither
squared. */
    long ip2;                /* Inphase prompt
squared. */
    long qp2;                /* Quadrature prompt
squared. */
    long codedco_update;     /* Update to the code DCO tracking
loop. */

    /* Get the squares of the prompt and dither correlations from the
inphase a quadrature arms. */

    ip2 = ISquare(A->IP);
    qp2 = ISquare(A->QP);
    id2 = ISquare(A->ID);
    qd2 = ISquare(A->QD);

    /* Get the running sum early minus late. This is the measurement
of
the code phase error. */

    CHPTR->EML += iSAR(id2+qd2,2) - iSAR(ip2+qp2,2);

    /* If channel actually has code lock then tracking can occur. */
    if(CHPTR->corlk)

```

```

{
    CHPTR->NEML++;
counter. */
    /* Increment the EML

    /* Only update the code tracking if EML has been accumulated
over
    EMLREADINGS readings. */

    if(CHPTR->NEML>=EMLREADINGS)
    {
        if(CHPTR->carfrlk && carrierSmoothingOn)
/* Carrier aiding can be used. */
        {
            * 770 is the scaling to go from carrier to code
frequency
            and is equal to 1575.42E6/1.023E6/2. The
factor of 2
            accounts for the difference in scaling
between the 2
            constants CARR_DCO_ZERO_DOPPLER and
            CODE_DCO_ZERO_DOPPLER. */

            CHPTR->CODEDCO = CODE_DCO_ZERO_DOPPLER -
(long)(CHPTR->CARRDCO-CARR_DCO_ZERO_DOPPLER+385L)/770;

            /* This filter is equivalent to:
            *
            * codedco(i) = codedco(i-1)
            *           + (phase_error(i) -
phase_error(i-1))/2^16
            *           + phase_error(i)/2^21
            */

            codedco_update = iSAR(CHPTR->EML+16,5) - CHPTR-
>EML - CHPTR->EMLOLD;
            CHPTR->CODEDCO +=
iSAR(codedco_update+32768L,16);
        }
        else
            /* No carrier
aiding. */
            {
                /* This filter is equivalent to:
                *
                * codedco(i) = codedco(i-1)
                *           + (phase_error(i) -
phase_error(i-1))/2^16
                *           + phase_error(i)/2^20
                */

                codedco_update = iSAR(CHPTR->EML+8,4) + CHPTR-
>EML - CHPTR->EMLOLD;
                CHPTR->CODEDCO +=
iSAR(codedco_update+32768L,16);
            }

```

```

/* Save the old early minus late value for the next
iteration
of the tracking loop. */
CHPTR->EMLOLD = CHPTR->EML;

/* Reset the early minus late running total and
accumulations
counter. */

CHPTR->EML = 0L;
CHPTR->NEML = 0;
} //Endif(CHPTR->NEML>=EMLREADINGS)
}/* EndIf code lock. */

/* Send out the new code DCO increment to the correlator after
scaling the loop gain. See the GPS Builder manual for an
explanation
and derivation of these values. */

// ANN
CHPTR->dcoCodeRate = (double)((CHPTR-
>CODEDCO+16L)>>5)/(1L<<27)*if4SampleRate; // sets code dco on not-
searching freq
CHPTR->dcoCodeRateSamplePeriod = CHPTR->dcoCodeRate*SamplePeriod;
// chips (b/w Samples)
}

//=====================================================

int if4Quantize (double dAnalogSig)
{
// Adaptive 2-bit quantizer

int dLevel;

if (dAnalogSig < AdativeQuantizerLevel1) {
dLevel = -3;
QuantizerTrack[0]++;
}
else {
if (dAnalogSig < AdativeQuantizerLevel2) {
dLevel = -1;
QuantizerTrack[1]++;
}
else {
if (dAnalogSig < AdativeQuantizerLevel3) {
dLevel = 1;
QuantizerTrack[2]++;
}
else {
dLevel = 3;
}
}
}
}

```



```

        QuantizerTrack[3]++;
    }
}

if (++QuantizerCnt >= QuantizerEvaluationCount) {
    unsigned long qCnt12 = QuantizerTrack[0] + QuantizerTrack[1];
    unsigned long qCnt34 = QuantizerTrack[2] + QuantizerTrack[3];
    double dSum =
QuantizerTrack[0]+QuantizerTrack[1]+QuantizerTrack[2]+QuantizerTrack[3];
    double bin1 = QuantizerTrack[0]/dSum;
    double bin4 = QuantizerTrack[3]/dSum;

    if (qCnt12 > qCnt34) {
        AdativeQuantizerLevel2-= qAdjustment;
    }
    else if (qCnt12 < qCnt34) {
        AdativeQuantizerLevel2+= qAdjustment;
    }

    if (bin1 > 0.15) {
        AdativeQuantizerLevel1-= qAdjustment;
    }
    else if (bin1 < 0.15) {
        AdativeQuantizerLevel1+= qAdjustment;
    }

    if (bin4 > 0.15) {
        AdativeQuantizerLevel3+= qAdjustment;
    }
    else if (bin4 < 0.15) {
        AdativeQuantizerLevel3-= qAdjustment;
    }

    QuantizerCnt = 0;

QuantizerTrack[0]=QuantizerTrack[1]=QuantizerTrack[2]=QuantizerTrack[3]=
0;
    //QNTZPRINT;

    // Keep running average of QTZ levels for later;
    {
        long qIdx;
        qIdx = accumCnt % QTZHISTORY;
        stableQTZlevelP[qIdx] = AdativeQuantizerLevel1;
        stableQTZlevelM[qIdx] = AdativeQuantizerLevel3;
    }

}

return dLevel;
}
//=====

```

```

FILE:      GOLDCODE.C
AUTHOR:    Awele Ndili
START DATE: June 1, 1996

```

```

/*
 * GoldCode.c - Generate Gold Codes
 * by Awele Ndili
 * June 1, 1996
 */

#include <stdio.h>
#include <math.h>

#include "defines.h"
#include "structs.h"
#include "funcprot.h"
#include "externs.h"

// Choose the PN2 offset on the G2 register to produce desired Gold
codes
// See Vol 1 page 80 - Spilker's chapter in GPS Theory & Applics
// PRN Signals, #1 thru' 32
int goldCodeOffsets[] = {    1018,
                             1017,
                             1016,
                             1015,
                             1006,
                             1005,
                             884,
                             883,
                             882,
                             772,
                             771,
                             769,
                             768,
                             767,
                             766,
                             765,
                             554,
                             553,
                             552,
                             551,
                             550,
                             549,
                             514,
                             511,
                             510,
                             509,
                             508,
                             507,

```

```

164,
163,
162,
161};

//=====

void generateSVsignals ( void )
{
    readAlmanac();
    generateCAcodes();
    updateGPSSignal(0.0);
}

//=====

void generateCAcodes ( void )
{
    int i,j;
    char XG1[2047];
    char XG2[2047];
    //FILE *fp; fp=fopen("PRNCodes.dat","w");
    // Generate the Maximal Length PRN Codes XG1 XG2
    generatePNcodes( XG1, XG2 );

    // Generate Gold Code from the Maximal Length PRN Codes
    for ( i = 0; i < 1023; i++ ) {
        for (j=0; j<MAXGOLDCODECOUNT; j++) {
            goldCode[j][i]=(int)(0x01 & (XG1[i] -
XG2[i+goldCodeOffsets[j]])); // ? -1 : 1;
        }
        //fprintf(fp,"%d\t%d\t%d\n",goldCode[0][i],goldCode[1][i],goldCode[2][i]
    );
    }
}

//=====

void generatePNcodes( char *prn1, char *prn2 )
{
    // Generate Maximal Length PRN Codes

    int    G1 = 0x0FFF;      /* PN Shift register 1 (G1) */
    int    G2 = 0x0FFF;      /* PN Shift register 2 (G2) */
    int    i;

    for ( i = 0; i < 1023; i++ ) {

        // #ifdef PLUSMINUS_ONE
        //     prn1[i]=((G1 & 0x01)==1? -1:1);
        //     prn2[i]=((G2 & 0x01)==1? -1:1);
        // #else
        prn1[i]=(G1 & 0x01); //Generate zeros and ones for these PRNs

```

```

    prn2[i]=(G2 & 0x01);
// #endif

    prn1[1023+i] = prn1[i]; /* Double length for ease in subsequent use
*/
    prn2[1023+i] = prn2[i];

    G1 <= 1;    // Shift registers
    G2 <= 1;
    G1 |= (0x01 & ((G1>>3)+(G1>>10)));
    G2 |= (0x01 & ((G2>>2)+(G2>>3)+(G2>>6)+(G2>>8)+(G2>>9)+(G2>>10)));
}
}

//=====

void readAlmanac( void )
{
    FILE *fp_alma;
    float tmpfloat;
    int i;

    // almanac(:,1) = PRN
    // almanac(:,2) = Time of almanac (toa)
    // almanac(:,3) = sqrt(a) (a = semi major axis)
    // almanac(:,4) = e (eccentricity)
    // almanac(:,5) = i (inclination)
    // almanac(:,6) = OMEGA - longitude of ascending node (referenced
to start of week)
    // almanac(:,7) = OMEGAdot
    // almanac(:,8) = omega - argument of perigee
    // almanac(:,9) = Mo (intial mean anomaly at toa)

    if((fp_alma=fopen("almanac.in","r"))==NULL){
        fprintf(stderr,"%cCould not open almanac file 'almanac.in'
for input\n",7);
    }
    else {
        for (i=0;i<SVS_IN_ALMANAC;i++){

            if(fscanf(fp_alma,"%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f",&tmpfloat,

            &almanac[i].toa,&almanac[i].sqrta,&almanac[i].ecc,&almanac[i].incl
in,

            &almanac[i].OMEGA,&almanac[i].OMEGAdot,&almanac[i].omega,&almanac[
i].Mo) == 9)
            {
                almanac[i].prn = (int) tmpfloat;
            }
            else
            {
                almanac[i].prn = 0;
            }
        }
    }
}

```

```

        //
        printf("%d\t%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\n", almanac[i].prn,
        //
        almanac[i].toa, almanac[i].sqrta, almanac[i].ecc, almanac[i].inclin,
        //
        almanac[i].OMEGA, almanac[i].OMEGAdot, almanac[i].omega, almanac[i].M
o);

        }
        fclose(fp_alma);
    }
}

=====

void updateGPSSignal( double timeOffsetFromToa )
{
    // timeOffsetFromToa = currentTime - Time of issue of Almanac
    // This routine runs through the almanac and computes, for the
    specified time offset,
    // the specifics of each SV signal:
    // Elevation;
    // Azimuth;
    // Signal Strength
    // Code & Carrier doppler;

    int i;
    NumberOfSimulatedSVs = 0;

    for (i=0; i<MaxAlmanacSize; i++)
    {
        computeSVparameters(timeOffsetFromToa, &almanac[i],
&gpsSignal[i], 0);
        if (gpsSignal[i].prn > 0 && gpsSignal[i].elevation>
SV_Elevation_Mask) NumberOfSimulatedSVs++;
    }
    // printf("Number of Simulated Satellites =
%d\n",NumberOfSimulatedSVs);
}

=====

void computeSVparameters( double timeOffsetFromToa, almstruc *dAlmanac,
SV_Signal *dGPSSignal, int recurseFlag)
{
    double Mo_now; // Mean anomaly at this point, in radians;
    double Ea_now; // Eccentric anomaly now;
    double f_true; // True anomaly
    double arg_of_lat; // argument of latitude
    double dRadius; // Distance of SV from ref. origin
    double lambda; // longitude of ascending node
    double x1, y1, z1;
    double x2, y2, z2;
    double dInclin;
    double relativeVelocity;

```

```

SV_Signal aGPSSignal;

dGPSSignal->prn = dAlmanac->prn;          . Set the prns
if (dAlmanac->prn > 0) // Contains info
{
    // First solve Kepler's equation iteratively to compute E ->
The Eccentric anomaly
    Mo_now = ((double) dAlmanac->Mo*DTR) + timeOffsetFromToa *
sqrt(miu)/(((double)dAlmanac->sqrta)*((double)dAlmanac-
>sqrta)*((double)dAlmanac->sqrta));
    Ea_now = kepler(Mo_now, (double) dAlmanac->ecc);
    f_true = atan2(sqrt(1.0-dAlmanac->ecc*dAlmanac-
>ecc)*sin(Ea_now), (cos(Ea_now)-dAlmanac->ecc));

    arg_of_lat = dAlmanac->omega*DTR + f_true;
    dRadius = dAlmanac->sqrta*dAlmanac->sqrta*(1.0-dAlmanac-
>ecc*cos(Ea_now));
    lambda = (dAlmanac->OMEGA + dAlmanac-
>OMEGAdot*timeOffsetFromToa)*DTR - omega_earth*(dAlmanac-
>toa+timeOffsetFromToa);
    dInclin = dAlmanac->inclin*DTR;
    // Will now rotate through five (four) ref. frames:
    //R3: 1. rotate (around z) : x on orbit plane through arg.
of latitude to ascending node (on equatorial plane)
    //R1: 2. rotate (around x) : z through angle of inclination
to earth's axis of rotation
    //R3: 3/4 rotate (around z) : x through longitude of
ascending node to Greenwich (long. 0), and (4) onto user's longitude
    //R2: 5. rotate (around y) : x from equatorial plane through
users latitude to user's position.
    // At the end of these rotations, the satellite's local
azimuth & elevation can be easily computed.
    x1=dRadius;
    y1=0.0;
    z1=0.0;

    //-----Rotation 1:-----
    //R3: 1. rotate (around z) : x on orbit plane through arg.
of latitude to ascending node (on equatorial plane)
    x2 = x1*cos(arg_of_lat); // - y1*sin(arg_of_lat);
    y2 = x1*sin(arg_of_lat); // + y1*cos(arg_of_lat);
    z2=z1;
    // Reset
    x1=x2; y1=y2; //z1=z2;

    //-----Rotation 2:-----
    //R1: 2. rotate (around x) : z through angle of inclination
to earth's axis of rotation
    x2 = x1;
    y2 = y1*cos(dInclin); // - z1*sin(dInclin);
    z2 = y1*sin(dInclin); // + z1*cos(dInclin);
    // Reset
    z1=z2; y1=y2; //x1=x2;

```

```

//-----Rotation 3/4:-----
//R3: 3/4rotate (around z) : x through longitude of
ascending node to Greenwich (long. 0), and (4) onto user's longitude
lambda -= user_longitude*DTR; // add in location of user
x2 = x1*cos(lambda) - y1*sin(lambda);
y2 = x1*sin(lambda) + y1*cos(lambda);
z2 = z1;
// Reset
x1=x2; y1=y2; //z1=z2;

//-----Rotation 5:-----
//R2: 5. rotate (around y) : x from equatorial plane through
users latitude to user's position.
lambda = user_latitude*DTR; // set location of user
(latitude)
x2 = x1*cos(lambda) + z1*sin(lambda);
y2 = y1;
z2 = -x1*sin(lambda) + z1*cos(lambda);

dGPSSignal->x = x2;
dGPSSignal->y = y2;
dGPSSignal->z = z2;

if (recurseFlag == 0) // Don't need to proceed beyond this
point if called recursively
{
    // Now, set the user's position in this coord. system
    x1 = radius_earth + user_height;
    y1 = 0.0;
    z1 = 0.0;

    // Now compute the azimuth and elevations
    dGPSSignal->elevation = atan((x2-
x1)/sqrt(y2*y2+z2*z2))/DTR;
    dGPSSignal->azimuth = atan2(y2,z2)/DTR;

    // Carrier Amplitude is a function of elevation
    dGPSSignal->carrierAmplitude =
getSigStrengthfromElev(dGPSSignal->elevation);

    // Compute doppler frequency
    // Method is to find position of SV 1 second from now,
and use the difference for velocity
    computeSVparameters(timeOffsetFromToa+1.0, dAlmanac,
&aGPSSignal, 1);
    relativeVelocity = (aGPSSignal.x-dGPSSignal-
>x)*sin(dGPSSignal->elevation*DTR) +
((aGPSSignal.y-dGPSSignal-
>y)*sin(dGPSSignal->azimuth*DTR) +
(aGPSSignal.z-dGPSSignal-
>z)*cos(dGPSSignal->azimuth*DTR)) * cos(dGPSSignal->elevation*DTR);

    dGPSSignal->carrierFrequency = SPEED_OF_LIGHT * L1 /
(SPEED_OF_LIGHT + relativeVelocity);

```

```

        dGPSSignal->codeRate = dGPSSignal-
>carrierFrequency/L1_TO_CA;
        dGPSSignal->carrierFrequency -= L1_TO_IF3; // Should
this line come b4 above line? no.
        dGPSSignal->twoPifcSamplePeriod = twoPi * dGPSSignal-
>carrierFrequency * SamplePeriod;
        dGPSSignal->codeRateSamplePeriod = dGPSSignal-
>codeRate * SamplePeriod;

        //      printf("PRN %d: Elev %g Azim %g Amplitude %g Doppler
%g\n",dGPSSignal->prn,dGPSSignal->elevation,dGPSSignal-
>azimuth,dGPSSignal->carrierAmplitude,dGPSSignal-
>carrierFrequency+L1_TO_IF3-L1);
    }
    } //end if (dAlmanac->prn > 0)
}

//=====
==

double kepler(double Mo_now, double ecc)
{
    // Iteratively solve Kepler's equation: Mo = Eo - e.Sin(Eo)
    double Ea, Eb; // Eccentric anomaly now;

    Ea = Mo_now;
    while(1)
    {
        Eb = Mo_now + ecc*sin(Ea);
        if (fabs(Eb-Ea)<1e-12) break;
        Ea = Eb;
    }
    return Eb;
}

//=====

double getSigStrengthfromElev(double SV_elevation)
{
    // Set's satellite signal strength from its elevation
    // curve-fit to observed signal performance
    // Ref.1 : LARC persentation, Oct. 96, by Per Enge & Awele Ndili
    // Ref.2 page 87 of Vol. I of GPS Theory & Applics; Chap. 3 by
J.J. Spilker: Signal Structure and Theoretical Performance
    // Elevation in degrees

    double carrierPower; // in dBW
    double maxCarrierPower;
    double powerRatio;

    if (SV_elevation>40.0)
        carrierPower=50.0;
    else
        if (SV_elevation>10)

```



```

        carrierPower= 43.0 + 0.2333*(SV_elevation-10.0);
//43+(50-43)/(40-10)*(SV_elevation-10);
    else
        carrierPower = 35.0 + 0.8*SV_elevation;    //35+(43-
35)/(10)*SV_elevation;

```

```

    carrierPower -= 200.0;
    maxCarrierPower = -150; // 50 - 200;
    powerRatio = carrierPower - maxCarrierPower;    // now in dB
    powerRatio = pow(10.0, powerRatio/10.0);

```

```

    if (SV_elevation<-5)
        powerRatio = 0.0;

```

```

    return sqrt(2*powerRatio);    // return signal amplitude ???
sqrt(2*powerRatio?)
}

```

```

//=====

```

```

//=====

```

```

FILE:      INIT.C
AUTHOR:    Awele Ndili
START DATE: November 23, 1996

```

```

/*
 * init.c - GPS Receiver Simulation
 * by Awele Ndili
 * November 23, 1996
 *
 */

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

#include "init.h"
#include "defines.h"
#include "structs.h"
#include "funcprot.h"
#include "externs.h"

// _____

void initialize(void)
{
    setDefaults();
    readinGlobals();
    generateSVsignals (); // Generate CA Code & initialize
    initializeVar();
    r250_init( 1 ); // Initialize gaussian random number
generator
}
// _____

void setDefaults(void)
{
    //----Incoming Signal Properties
    // IF3_Carrier_Freq = 100.0+(35.42-1400.0/45)*1.0e6; // Hz
    // IF3_CA_Code_Rate = 1.023e6; // cycles per second
    // DelayB4NoiseInclude = 0.000; // Delay before
inclusion of Noise (seconds)
    IF3_INIT_CAOFFSET = 14; // Initial CA Code
Offset
    SV_Elevation_Mask = -5.0; // Min. SV satellite
elevation to simulate
    MaxAlmanacSize = SVS_IN_ALMANAC; // Maximum number of SVs to
include in almanac, default=25

    includeAWGN = 1; // Include AWGN
    includeCWInterference = 1; // Include CW interference

```

```

includeBGAWGN = 0; // Include Background
AWGN
includeSVAttenuation = 0; // Include SV attenuation
includePULSEInterference = 0; // Include Pulsed interference
delayB4AWGN = 0.0; // Delay before addition
of AWGN
delayB4CWInterference = 0.0; // Delay before addition of CW
delayB4SVAttenuation = 0.0; // Delay before SV signal
attenuation
BG_AWGN_SNR_dB = 0; // Background AWGN level, in
dB
pulsePositionIsRandom = 1; // Pulsing Scheme is random

//----Receiver Properties
IF4_Sample_Rate = (40.0/7.0e-6); // Samples per
second
// MAXCHANNELS = 1; // Number of
channels this receiver has

//----Correlator Properties
INTEGRATIONTIME = 0.001; // Sample n Dump
time (seconds)
EarlyLateCorrSpace = 0.5; // Chip
Width (Note: 0.5=>.25 each side, for GEC Plessey)
carrierSmoothingOn = 0; // Carrier
smoothing of code DLL off - default

//----Acquisition Properties
DetectionThreshold = 1690000L; // Signal
acquisition threshold
LockLossThreshold = 360000L; // Signal loss-of-
lock threshold
DCO_CA_Code_SRate = 1.02325e6; // dco ca code
rate during search (cycles per second)
DCO_CA_Code_Rate = 1.0230004e6; // dco ca code
rate after lock (cycles per second)
DCO_INIT_CAOFFSET = 0; // Initial dco ca
code offset
FreqSearchBinWidth = 300.0; // Hz
Init_CarrFreqErr = 100.0; // Hz; Initial Carrier
Freq. error in DCO
DCOMaxDoppler = 6000; // Hz

//----Tracking Loop Properties
COASTING_PERIOD = 20.0; //Coasting period,
in seconds
EMLREADINGS = 40; // Number of EML
readings to integrate for code tracking

//----Adaptive Quantizer Properties
ADAPTIVEQUANTIZERPERIOD = 0.0005; // Time to reevaluate
quantizer levels, in secs (GEC Spec=0.5ms)

```

```

        A_QUANTIZER_INCREMENT    =    0.01; // Adaptive Quantizer
adjustment increments

        //----Generic Constants
        RUNTIME    =    0.100;                // Total run
time (seconds)
        run_id = 0;                // Unique id for
this run
        user_latitude = 37.7;
        user_longitude = -122.0;
        user_height = 0.0;
        screenPrinting = 1;                // Screen printing
defaults to true

}

// _____

void readinGlobals(void)
{
    // Readin Runtime parameters
    int i;
    FILE *fp_rcv;
    char key[120], token[120], value[120];

    // Open up receiver specification file to configure this receiver
    if( (fp_rcv=fopen("rcv_spec.in","r"))==NULL){
        printf("Could not open Receiver Specification File
'rcv_spec.in' for input\n");
        BEEP(1);
        exit(1);
    }

    while(i=fscanf(fp_rcv,"%s",key),i!=0 && i!=EOF)
    {
        if (!strcmp(key,"#define"))
            { // A Match occurred, therefore process this token
                if(fscanf(fp_rcv,"%s %s",token,value)==2)
                    processToken(get_tokenId(token),value);
            }
        else
            { // No Match so skip to end of line
                while(getc(fp_rcv)!=10);
            }
    }
    fclose(fp_rcv);

    return;
}

// _____

void initializeVar( void )
{

```

```

// Initialize Runtime parameters
int i;
double tmpDouble;
chanstruc *chPtr;

twoPi = 2.0*PI;
// Set values based on defaults or overridden values
Ta = DetectionThreshold;
Tl = LockLossThreshold;

dcoCAOffset      = DCO_INIT_CAOFFSET;
totalRunTime = RUNTIME;           // seconds
accumulationTime = INTEGRATIONTIME;
Coast = COASTING_PERIOD / INTEGRATIONTIME;           in epochs
if4SampleRate = IF4_Sample_Rate; // Hz
SamplePeriod = 1.0/if4SampleRate; // Seconds
SubSamplePeriod = SamplePeriod/4.0; // Seconds (sample 4x faster
for upstream filtering)

for (i=0; i<MaxAlmanacSize; i++){ // Initialize phases for SV
signals in almanac
    gpsSignal[i].codePhase = IF3_INIT_CAOFFSET + i*2;
    gpsSignal[i].carrierPhase = 0.0 + i*4;
    gpsSignal[i].twoPIfcSamplePeriod = twoPi *
gpsSignal[i].carrierFrequency * SamplePeriod;
    gpsSignal[i].codeRateSamplePeriod = gpsSignal[i].codeRate *
SamplePeriod;
    gpsSignal[i].twoPIfcSubSamplePeriod = twoPi *
gpsSignal[i].carrierFrequency * SubSamplePeriod;
    gpsSignal[i].codeRateSubSamplePeriod = gpsSignal[i].codeRate
* SubSamplePeriod;
}

// 970928
cwInterference.carrierFrequency = gpsSignal[0].carrierFrequency +
CWInterferenceDopplerOffset;
cwInterference.carrierPhase = 0.0;           // Initialize carrier
phase
cwInterference.twoPIfcSamplePeriod = twoPi *
cwInterference.carrierFrequency * SamplePeriod;
cwInterference.twoPIfcSubSamplePeriod = twoPi *
cwInterference.carrierFrequency * SubSamplePeriod;
cwInterference.carrierAmplitude = cw_interference_ampl;

// dcoDwellTimePerFreqBin = if3CodeRate[0]/(DCO_CA_Code_SRate-
if3CodeRate[0]);
// dcoDwellTime = 0L;

corrSpacing = EarlyLateCorrSpace / DCO_CA_Code_SRate; // Fraction
* Chip Width (secs)

DCOInitFreq      = gpsSignal[0].carrierFrequency -
Init_CarrFreqErr; // (35.42-1400.0/45)*1.0e6; // Hz

```

```

//-- Initialize Channel Structure
for (i=0; i<MAXCHANNELS; i++)
{
    chPtr = &CH[i];

    chPtr->SV = i+1; // Non-zero value
    chPtr->IM1 = 0L;
    chPtr->QM1 = 0L;
    chPtr->EML = 0L;
    chPtr->EMLOLD = 0L;
    chPtr->NEML = 0;
    chPtr->CdLI = 0L;
    chPtr->CrfrLI = 0L;
    chPtr->coasting = 0;
    chPtr->corlk = 0;
    chPtr->carfrlk = 0;
    chPtr->iNXTACCUM = 0;
    chPtr->iCURACCUM = 0;

    chPtr->CARRDCO = (unsigned
long) (DCOInitFreq*16.0/iff4SampleRate*(1L<<27));
    chPtr->dcoCarrierFreq = (double)((chPtr-
>CARRDCO+8)>>4)/(1L<<27)*iff4SampleRate; // Starting Points
    chPtr->twoPiFrSamplePeriod = twoPi * chPtr->dcoCarrierFreq *
SamplePeriod; // radians (b/w Samples)
    chPtr->twoPiFrCorrSpacing = twoPi * chPtr->dcoCarrierFreq *
corrSpacing; // radians (between Early/Late corrs

    chPtr->CODEDCO = (unsigned
long) (2*DCO_CA_Code_SRate*32.0/iff4SampleRate*(1L<<26));
    chPtr->dcoCodeRate = (double)((chPtr-
>CODEDCO+16L)>>5)/(1L<<27)*iff4SampleRate; // Starts code dco on
search freq
    chPtr->dcoCodeRateSamplePeriod = chPtr-
>dcoCodeRate*SamplePeriod; // chips (b/w Samples)
    chPtr->dcoCodeRateCorrSpacing = chPtr-
>dcoCodeRate*corrSpacing; // chips (between Early/Late corrs

    chPtr->dcoCarrierPhaseE = 0.0 - chPtr-
>twoPiFrCorrSpacing/2.0;
    chPtr->dcoCarrierPhaseL = chPtr->dcoCarrierPhaseE + chPtr-
>twoPiFrCorrSpacing;
    chPtr->dcoCodePhaseE = dcoCAOffset - chPtr-
>dcoCodeRateCorrSpacing/2.0;
    chPtr->dcoCodePhaseL = chPtr->dcoCodePhaseE + chPtr-
>dcoCodeRateCorrSpacing;

}

dcoMaxFreq = DCOInitFreq + DCOMaxDoppler;
dcoMinFreq = DCOInitFreq - DCOMaxDoppler;

```

```

    realTime = 0.0;

    // .891*(0.83-0.17*EXP(1-C4))*sqrt(Num.SVs)
    tmpDouble = 0.891*(0.83+0.17*exp(1-NumberOfSimulatedSVs)) *
sqrt(NumberOfSimulatedSVs); // Empirical formula derives by Awele
961114
    if (!includePULSEInterference){ // 970512
        if (includeBGAWGN && tmpDouble<bg_awgn_std_dev)
            tmpDouble = bg_awgn_std_dev; // Empirical formula
derived by Awele 961114

        if (includeAWGN && delayB4AWGN<0.1 &&
tmpDouble<awgn_std_dev)
            tmpDouble = awgn_std_dev; // Empirical formula
derived by Awele 961114

        if (includeCWInterference && delayB4CWInterference<0.1 &&
tmpDouble<cw_interference_ampl/sqrt(2))
            tmpDouble = cw_interference_ampl/sqrt(2); // Empirical
formula derived by Awele 970511
    }

    AdativeQuantizerLevel1 = -tmpDouble;
    AdativeQuantizerLevel2 = 0.0;
    AdativeQuantizerLevel3 = tmpDouble;
    //QNTZPRINT;
    QuantizerTrack[0]=QuantizerTrack[1]=QuantizerTrack[2]=QuantizerTra
ck[3]=0;
    QuantizerEvaluationCount = ADAPTIVEQUANTIZERPERIOD *
if4SampleRate; // Samples
    QuantizerCnt = 0;
    qAdjustment = A_QUANTIZER_INCREMENT;

    // Information Tracking Variables
    maxPreAcquisitionIQ2 = 0.0;
    maxIQ2 = 0.0;
    accumCnt = 0;
    preAcqAccumCnt = (IF3_INIT_CAOFFSET-DCO_INIT_CAOFFSET-
2)/(DCO_CA_Code_SRate-DCO_CA_Code_Rate)/INTEGRATIONTIME;

    // Initialize if4 butterworth filter
    filterIdx = 0;
    filterSizeP1 = ORDEROFFILTER+1;
    for (i=0; i<FILTERBUFFERSIZE; i++)
        inputX[i]=outputY[i]=0.0;
}

//
void printoutGlobals(void)
{
    FILE *fp_Output, *fp_RunSchedule;
    int i=0;
    OPEN_FILE2(fp_Output,"rcv_spec.out","w"); // Open Output File

```

```

fprintf(fp_Output, "PARAMETERS FOR RUN ID: %ld\n", run_id);
fprintf(fp_Output, "=====\n");

fprintf(fp_Output, "\nRUN PARAMETERS:\n");
fprintf(fp_Output, "\tTotal Run Time = \t%g seconds\n", RUNTIME);

fprintf(fp_Output, "\nGPS SV SIGNAL PARAMETERS:\n");
fprintf(fp_Output, "\tInitial CA Code Offset = \t%d
chips\n", IF3_INIT_CAOFFSET);
fprintf(fp_Output, "\tSV_Elevation_Mask = \t%g
degrees\n", SV_Elevation_Mask);
fprintf(fp_Output, "\tAlmanac Size = \t%d
satellites\n", MaxAlmanacSize);
fprintf(fp_Output, "\tNumber of Visible Satellites =
\t%d\n", NumberOfSimulatedSVs);

fprintf(fp_Output, "\nNOISE/INTERFERENCE PARAMETERS:\n");
fprintf(fp_Output, "\tBackground AWGN %s included\n", includeBGAWGN
? "is" : "is NOT");
if(includeBGAWGN) fprintf(fp_Output, "\tBackground AWGNSR = \t%g
dB\n", BG_AWGNSR_dB);
fprintf(fp_Output, "\tAWGN %s included\n", includeAWGN ? "is" : "is
NOT");
if(includeAWGN) fprintf(fp_Output, "\tDelay before AWGN = \t%g
seconds\n", delayB4AWGN);
fprintf(fp_Output, "\tCW Int. %s included\n", includeCWInterference
? "is" : "is NOT");
if(includeCWInterference) fprintf(fp_Output, "\tDelay before CW
Interference = \t%g seconds\n", delayB4CWInterference);
fprintf(fp_Output, "\tPulsed Interference %s
included\n", includePULSEInterference ? "is" : "is NOT");
if(includePULSEInterference){
    fprintf(fp_Output, "\tPulsing Scheme =
\t%s\n", pulsePositionIsRandom?"Random":"Fixed");
    //fprintf(fp_Output, "\tPulse Duty Cycle = \t%6.2f
percent\n", pulseDutyCycle*100.0);
}
fprintf(fp_Output, "\tSV Signal attenuation %s
included\n", includeSVAttenuation ? "is" : "is NOT");
if(includeSVAttenuation) fprintf(fp_Output, "\tDelay before SV
Signal Attenuation = \t%g seconds\n", delayB4SVAttenuation);

fprintf(fp_Output, "\nGPS RECEIVER PARAMETERS:\n");
fprintf(fp_Output, "\tNumber of Channels = \t%d\n", MAXCHANNELS);
fprintf(fp_Output, "\tIF4_Sample_Rate = \t%g samples per
second\n", IF4_Sample_Rate);
fprintf(fp_Output, "\tINTEGRATIONTIME = \t%g
seconds\n", INTEGRATIONTIME);
fprintf(fp_Output, "\tEarly - Late Corr. Spacing = \t%g
chips\n", EarlyLateCorrSpace);

fprintf(fp_Output, "\nSEARCH & ACQUISITION PARAMETERS:\n");

```



```

        fprintf(fp_Output, "\tDetectionThreshold = \t%d units (SNR = %g
dB)\n", DetectionThreshold, 10.0*log10(DetectionThreshold/97142.0));
        fprintf(fp_Output, "\tLockLossThreshold = \t%d units (SNR = %g
dB)\n", LockLossThreshold, 10.0*log10(LockLossThreshold/97142.0));
//      FreqSearchBinWidth      = 300.0;          // Hz
//      DCOMaxDoppler           = 6000;          // Hz

        fprintf(fp_Output, "\tDCO_CA_Code_SearchRate = \t%g
Hz\n", DCO_CA_Code_SRate);
        fprintf(fp_Output, "\tDCO_CA_Code_SettledRate = \t%g
Hz\n", DCO_CA_Code_Rate);
        fprintf(fp_Output, "\tInitial DCO CA Code Offset = \t%d
chips\n", DCO_INIT_CAOFFSET);
        fprintf(fp_Output, "\tInitial CarrierDCO Frequency Error = \t%g
Hz\n", Init_CarrFreqErr);

        fprintf(fp_Output, "\nTRACKING LOOP PARAMETERS:\n");
        fprintf(fp_Output, "\tCOASTING_PERIOD = \t%g
seconds\n", COASTING_PERIOD);
        fprintf(fp_Output, "\tCode Loop Bandwidth = \t%g
Hz\n", 1000.0/EMLREADINGS);
        fprintf(fp_Output, "\tCarrier Smoothing %s on\n", carrierSmoothingOn
? "is" : "is NOT");

        fprintf(fp_Output, "\nADAPTIVE QUANTIZER PARAMETERS:\n");
        fprintf(fp_Output, "\tADAPTIVEQUANTIZERPERIOD = \t%g
seconds\n", ADAPTIVEQUANTIZERPERIOD);
        fprintf(fp_Output, "\tA_QUANTIZER_INCREMENT =
\t%g\n", A_QUANTIZER_INCREMENT);

        fprintf(fp_Output, "\nUSER POSITION:\n");
        fprintf(fp_Output, "\tUser Latitude = \t%g
degrees\n", user_latitude);
        fprintf(fp_Output, "\tUser Longitude = \t%g
degrees\n", user_longitude);
        fprintf(fp_Output, "\tUser Height = \t%g meters\n", user_height);

        fprintf(fp_Output, "\nRUN SCHEDULE:\n");
        fprintf(fp_Output, "\tS/N\tAWGN\tCWISR\tDutyCycle\tCW Doppler
Offset\tSV-Attenuation\t(Values in dB, dB, fraction, Hz, dB)\n");

        OPEN_FILE2(fp_RunSchedule, "run_spec.in", "r");    // Open Runs
Schedule File
        while(fscanf(fp_RunSchedule, "%f\t%f\t%f\t%f\t%f", &AWGNSR_dB, &CWISR
_dB, &pulseDutyCycle, &CWInterferenceDopplerOffset, &SV_ATTNUATN_dB) != EOF
&& AWGNSR_dB != -1)
            fprintf(fp_Output, "\t%d\t%g\t%g\t%g\t%g\t%g\n", ++i, AWGNSR_dB, CWISR
_dB, pulseDutyCycle, CWInterferenceDopplerOffset, SV_ATTNUATN_dB);
        fclose(fp_RunSchedule);
        fclose(fp_Output);
        return;
}

//

```

```

int get_tokenId(char *token)
{ // Returns the associated token Id
    //if (!strcmp(token, "DelayB4NoiseInclude")) return
    TOKEN_DelayB4NoiseInclude;
    //if (!strcmp(token, "IF3_CA_Code_Rate")) return
    TOKEN_IF3_CA_Code_Rate;
    if (!strcmp(token, "IF3_INIT_CAOFFSET")) return
    TOKEN_IF3_INIT_CAOFFSET;
    if (!strcmp(token, "IF4_Sample_Rate")) return
    TOKEN_IF4_Sample_Rate;
    //if (!strcmp(token, "MAXCHANNELS")) return TOKEN_MAXCHANNELS;
    if (!strcmp(token, "INTEGRATIONTIME")) return
    TOKEN_INTEGRATIONTIME;
    if (!strcmp(token, "EarlyLateCorrSpace")) return
    TOKEN_EarlyLateCorrSpace;
    if (!strcmp(token, "DetectionThreshold")) return
    TOKEN_DetectionThreshold;
    if (!strcmp(token, "LockLossThreshold")) return
    TOKEN_LockLossThreshold;
    if (!strcmp(token, "DCO_CA_Code_SRate")) return
    TOKEN_DCO_CA_Code_SRate;
    if (!strcmp(token, "DCO_CA_Code_Rate")) return
    TOKEN_DCO_CA_Code_Rate;
    if (!strcmp(token, "DCO_INIT_CAOFFSET")) return
    TOKEN_DCO_INIT_CAOFFSET;
    if (!strcmp(token, "FreqSearchBinWidth")) return
    TOKEN_FreqSearchBinWidth;
    //if (!strcmp(token, "DCOInitFreq")) return TOKEN_DCOInitFreq;
    if (!strcmp(token, "DCOMaxDoppler")) return TOKEN_DCOMaxDoppler;
    if (!strcmp(token, "COASTING_PERIOD")) return
    TOKEN_COASTING_PERIOD;
    if (!strcmp(token, "EMLREADINGS")) return TOKEN_EMLREADINGS;
    if (!strcmp(token, "ADAPTIVEQUANTIZERPERIOD")) return
    TOKEN_ADAPTIVEQUANTIZERPERIOD;
    if (!strcmp(token, "A_QUANTIZER_INCREMENT")) return
    TOKEN_A_QUANTIZER_INCREMENT;
    if (!strcmp(token, "RUNTIME")) return TOKEN_RUNTIME;
    if (!strcmp(token, "Init_CarrFreqErr")) return TOKEN_INITFREQERR;
    if (!strcmp(token, "SV_Elevation_Mask")) return TOKEN_SV_ELEV_MASK;

    if (!strcmp(token, "INCLUDE_AWGN")) return TOKEN_INCLUDE_AWGN;
    if (!strcmp(token, "DELAY_B4_AWGN")) return TOKEN_DELAY_B4_AWGN;
    if (!strcmp(token, "INCLUDE_CW")) return TOKEN_INCLUDE_CW;
    if (!strcmp(token, "DELAY_B4_CW")) return TOKEN_DELAY_B4_CW;
    if (!strcmp(token, "INCLUDE_PULSE")) return TOKEN_INCLUDE_PULSE;
    if (!strcmp(token, "INCLUDE_BG_AWGN")) return
    TOKEN_INCLUDE_BG_AWGN;
    if (!strcmp(token, "BG_AWGNSR_DB")) return TOKEN_BG_AWGNSR_DB;
    if (!strcmp(token, "RUNID")) return TOKEN_RUNID;
    if (!strcmp(token, "MAX_ALMANAC_SIZE")) return
    TOKEN_MAX_ALMANAC_SIZE;
    //if (!strcmp(token, "PULSE_DUTYCYCLE")) return
    TOKEN_PULSE_DUTYCYCLE;

```

```

        if (!strcmp(token, "PULSING_SCHEME")) return TOKEN_PULSING_SCHEME;
        if (!strcmp(token, "INCLUDE_SVATTENUATION")) return
TOKEN_INCLUDE_SVATTENUATION;
        if (!strcmp(token, "DELAY_B4_SVATTENUATION")) return
TOKEN_DELAY_B4_SVATTENUATION;
        if (!strcmp(token, "PRINT2SCREEN")) return TOKEN_PRINT2SCREEN;
        if (!strcmp(token, "CARRIERSMOOTHINGON")) return
TOKEN_CARRIERSMOOTHINGON;

        return TOKEN_NULL;
}

```

```

void processToken( int dToken, char *value)
{ // Process each Initialization variable read in.
    switch(dToken)
    {
        // case( TOKEN_DelayB4NoiseInclude):
        //     sscanf(value, "%lf", &DelayB4NoiseInclude);
        //     fprintf(fp_Output, "DelayB4NoiseInclude = %g (from
%s)\n", DelayB4NoiseInclude, value);
        //     break;
        // case( TOKEN_IF3_CA_Code_Rate):
        //     sscanf(value, "%lf", &IF3_CA_Code_Rate);
        //     fprintf(fp_Output, "IF3_CA_Code_Rate = %g (from
%s)\n", IF3_CA_Code_Rate, value);
        //     break;
        // case( TOKEN_IF3_INIT_CAOFFSET):
        //     sscanf(value, "%d", &IF3_INIT_CAOFFSET);
        //     fprintf(fp_Output, "IF3_INIT_CAOFFSET = %d (from
%s)\n", IF3_INIT_CAOFFSET, value);
        //     break;
        // case( TOKEN_IF4_Sample_Rate):
        //     sscanf(value, "%lf", &IF4_Sample_Rate);
        //     fprintf(fp_Output, "IF4_Sample_Rate = %g (from
%s)\n", IF4_Sample_Rate, value);
        //     break;
        // case( TOKEN_MAXCHANNELS):
        //     sscanf(value, "%d", &MAXCHANNELS);
        //     fprintf(fp_Output, "MAXCHANNELS = %d (from
%s)\n", MAXCHANNELS, value);
        //     break;
        // case( TOKEN_INTEGRATIONTIME):
        //     sscanf(value, "%lf", &INTEGRATIONTIME);
        //     fprintf(fp_Output, "INTEGRATIONTIME = %g (from
%s)\n", INTEGRATIONTIME, value);
        //     break;
        // case( TOKEN_EarlyLateCorrSpace):
        //     sscanf(value, "%lf", &EarlyLateCorrSpace);
        //     fprintf(fp_Output, "EarlyLateCorrSpace = %g (from
%s)\n", EarlyLateCorrSpace, value);
        //     break;
        // case( TOKEN_DetectionThreshold):

```

```

        sscanf(value, "%lu", &DetectionThreshold);
        // fprintf(fp_Output, "DetectionThreshold = %lu (from
%s)\n", DetectionThreshold, value);
        break;
        case( TOKEN_LockLossThreshold):
        sscanf(value, "%lu", &LockLossThreshold);
        // fprintf(fp_Output, "LockLossThreshold = %lu (from
%s)\n", LockLossThreshold, value);
        break;
        case( TOKEN_DCO_CA_Code_SRate):
        sscanf(value, "%lf", &DCO_CA_Code_SRate);
        // fprintf(fp_Output, "DCO_CA_Code_SRate = %g (from
%s)\n", DCO_CA_Code_SRate, value);
        break;
        case( TOKEN_DCO_CA_Code_Rate):
        sscanf(value, "%lf", &DCO_CA_Code_Rate);
        // fprintf(fp_Output, "DCO_CA_Code_Rate = %g (from
%s)\n", DCO_CA_Code_Rate, value);
        break;
        case( TOKEN_DCO_INIT_CAOFFSET):
        sscanf(value, "%d", &DCO_INIT_CAOFFSET);
        // fprintf(fp_Output, "DCO_INIT_CAOFFSET = %d (from
%s)\n", DCO_INIT_CAOFFSET, value);
        break;
        case( TOKEN_FreqSearchBinWidth):
        sscanf(value, "%lf", &FreqSearchBinWidth);
        // fprintf(fp_Output, "FreqSearchBinWidth = %g (from
%s)\n", FreqSearchBinWidth, value);
        break;
        case( TOKEN_DCOInitFreq):
        sscanf(value, "%lf", &DCOInitFreq);
        // fprintf(fp_Output, "DCOInitFreq = %g (from
%s)\n", DCOInitFreq, value);
        break;
        case( TOKEN_DCOMaxDoppler):
        sscanf(value, "%lf", &DCOMaxDoppler);
        // fprintf(fp_Output, "DCOMaxDoppler = %g (from
%s)\n", DCOMaxDoppler, value);
        break;
        case( TOKEN_COASTING_PERIOD):
        sscanf(value, "%lf", &COASTING_PERIOD);
        // fprintf(fp_Output, "COASTING_PERIOD = %g (from
%s)\n", COASTING_PERIOD, value);
        break;
        case( TOKEN_EMLREADINGS):
        sscanf(value, "%d", &EMLREADINGS);
        // fprintf(fp_Output, "EMLREADINGS = %d (from
%s)\n", EMLREADINGS, value);
        break;
        case( TOKEN_ADAPTIVEQUANTIZERPERIOD):
        sscanf(value, "%lf", &ADAPTIVEQUANTIZERPERIOD);
        // fprintf(fp_Output, "ADAPTIVEQUANTIZERPERIOD = %g (from
%s)\n", ADAPTIVEQUANTIZERPERIOD, value);
        break;

```

```

        case( TOKEN_A_QUANTIZER_INCREMENT):
            sscanf(value, "%lf", &A_QUANTIZER_INCREMENT);
            // fprintf(fp_Output, "A_QUANTIZER_INCREMENT = %g (from
%s)\n", A_QUANTIZER_INCREMENT, value);
            break;
        case( TOKEN_RUNTIME):
            sscanf(value, "%lf", &RUNTIME);
            // fprintf(fp_Output, "RUNTIME = %g (from
%s)\n", RUNTIME, value);
            break;
        case( TOKEN_INITFREQERR):
            sscanf(value, "%lf", &Init_CarrFreqErr);
            // fprintf(fp_Output, "INITFREQERR = %g (from
%s)\n", Init_CarrFreqErr, value);
            break;
        case( TOKEN_SV_ELEV_MASK):
            sscanf(value, "%lf", &SV_Elevation_Mask);
            // fprintf(fp_Output, "SV_ELEV_MAST = %g (from
%s)\n", SV_Elevation_Mask, value);
            break;
        case( TOKEN_INCLUDE_AWGN):
            includeAWGN = (value[0]=='Y' || value[0]=='y') ? 1:0;
            // fprintf(fp_Output, "INCLUDE_AWGN = %d (from
%s)\n", includeAWGN, value);
            break;
        case( TOKEN_INCLUDE_CW):
            includeCWInterference = (value[0]=='Y' || value[0]=='y') ?
1:0;
            // fprintf(fp_Output, "INCLUDE_CW = %d (from
%s)\n", includeCWInterference, value);
            break;
        case( TOKEN_DELAY_B4_CW):
            sscanf(value, "%lf", &delayB4CWInterference);
            // fprintf(fp_Output, "DELAY_B4_CW = %g (from
%s)\n", delayB4CWInterference, value);
            break;
        case( TOKEN_INCLUDE_PULSE):
            includePULSEInterference = (value[0]=='Y' || value[0]=='y')
? 1:0;
            // fprintf(fp_Output, "INCLUDE_PULSE = %d (from
%s)\n", includePULSEInterference, value);
            break;
        case( TOKEN_DELAY_B4_AWGN):
            sscanf(value, "%lf", &delayB4AWGN);
            // fprintf(fp_Output, "DELAY_B4_AWGN = %g (from
%s)\n", delayB4AWGN, value);
            break;
        case( TOKEN_INCLUDE_BG_AWGN):
            includeBGAWGN = (value[0]=='Y' || value[0]=='y') ? 1:0;
            // fprintf(fp_Output, "INCLUDE_BG_AWGN = %d (from
%s)\n", includeBGAWGN, value);
            break;
        case( TOKEN_BG_AWGNSR_DB):
            sscanf(value, "%f", &BG_AWGNSR_db);

```

```

        // printf("BG_AWGNSR_DB = %g (from
%s)\n", BG_AWGNSR_dB, value);
        break;
        case( TOKEN_RUNID):
            sscanf(value, "%ld", &run_id);
            // fprintf(fp_Output, "RUNID = %ld (from
%s)\n", run_id, value);
            break;
        case( TOKEN_MAX_ALMANAC_SIZE):
            sscanf(value, "%d", &MaxAlmanacSize);
            // printf("MAX_ALMANAC_SIZE = %d (from
%s)\n", MaxAlmanacSize, value);
            break;
        //case( TOKEN_PULSE_DUTYCYCLE):
        //sscanf(value, "%f", &pulseDutyCycle);
        // printf("PULSE_DUTYCYCLE = %f (from
%s)\n", pulseDutyCycle, value);
        break;
        case( TOKEN_PULSING_SCHEME):
            pulsePositionIsRandom = (value[0]=='R' || value[0]=='r') ?
1:0;
            // printf("PULSING_SCHEME = %s (from
%s)\n", pulsePositionIsRandom?"Random":"Fixed", value);
            break;
        case( TOKEN_INCLUDE_SVATTENUATION):
            includeSVAttenuation = (value[0]=='Y' || value[0]=='y') ?
1:0;
            // fprintf(fp_Output, "INCLUDE_SVATTENUATION = %d (from
%s)\n", includeSVAttenuation, value);
            break;
        case( TOKEN_DELAY_B4_SVATTENUATION):
            sscanf(value, "%lf", &delayB4SVAttenuation);
            // fprintf(fp_Output, "DELAY_B4_SVATTENUATION = %g (from
%s)\n", delayB4SVAttenuation, value);
            break;
        case( TOKEN_PRINT2SCREEN):
            screenPrinting = (value[0]=='Y' || value[0]=='y') ? 1:0;
            // fprintf(fp_Output, "PRINT2SCREEN = %d (from
%s)\n", screenPrinting, value);
            break;
        case( TOKEN_CARRIERSMOOTHINGON):
            carrierSmoothingOn = (value[0]=='Y' || value[0]=='y') ? 1:0;
            // fprintf(fp_Output, "CARRIERSMOOTHINGON = %d (from
%s)\n", carrierSmoothingOn, value);
            break;

        default:
            break;
    }
}
//

```

```

FILE:      GAUSSIAN.C
AUTHOR:    Everett F. Carter Jr
DATE:      1994
MODIFIED:  Awele Ndili
DATE:      January 17, 1997

```

```

/* gaussian.c
 * Jan 17, 1997 - Awele Ndili (Modifications, Adaptations)
 * Code to generate Gaussian Random Variable
 */

```

```

/* boxmuller.c          Implements the Polar form of the Box-Muller
                        Transformation

```

```

                        (c) Copyright 1994, Everett F. Carter Jr.
                        Permission is granted by the author to use
                        this software for any application provided this
                        copyright notice is preserved.

```

```

#include <math.h>
#include "defines.h"
#include "structs.h"
#include "funcprot.h"
#include "externs.h"

```

```

void box_muller(void) /* normal random variate generator */
{
    // Mean = 0, Std Dev = awgn_std_dev
    // Returns two independent (float)

    Random Variables: Ni, Nq
    float x1, x2, w;
    do {
        x1 = 2.0 * fr250() - 1.0;
        x2 = 2.0 * fr250() - 1.0;
        w = x1 * x1 + x2 * x2;
    } while ( w >= 1.0 );

    w = sqrt( (-2.0 * log( w ) ) / w ) * awgn_std_dev; // Standard
    Dev. = awgn_std_dev
    Ni = x1 * w;
    Nq = x2 * w;
}

```

```

void box_muller2(void) /* normal random variate generator */
{
    // Mean = 0, Std Dev = awgn_std_dev
    // Returns two independent (float)

    Random Variables: Ni, Nq
    float x1, x2, w;
    do {
        x1 = 2.0 * fr250() - 1.0;
        x2 = 2.0 * fr250() - 1.0;
        w = x1 * x1 + x2 * x2;
    }

```

```
    } while ( w >= 1.0 );  
  
    w = sqrt( (-2.0 * log( w ) ) / w ) * bg_awgn_std_dev; // Standard  
Dev. = bg_awgn_std_dev  
    Ni = x1 * w;  
    Nq = x2 * w;  
}
```

```

FILE:      R250.C
AUTHOR:    Kirkpatrick, S., and E. Stoll
DATE:      1981
MODIFIED:   Awele Ndili
DATE:      January 20, 1997

```

```

/* r250.c   the r250 uniform random number algorithm

           Kirkpatrick, S., and E. Stoll, 1981; "A Very Fast
           Shift-Register Sequence Random Number Generator",
           Journal of Computational Physics, V.40

           also:

           see W.L. Maier, DDJ May 1991
           Modified by Awele Ndili, January 20, 1997

*/

//static char rcsid[] = "@(#)r250.c 1.2 15:50:31 11/21/94   EFC";

#include <limits.h>
#include "r250.h"
#include "defines.h"
#include "structs.h"
#include "funcprot.h"

/* set the following if you trust rand(), otherwise the minimal standard
   generator is used
*/
/* #define TRUST_RANDOM */

#ifndef TRUST_RANDOM
#include "randlcg.h"
#endif

/* defines to allow for 16 or 32 bit integers */
#define BITS 31

#if WORD_BIT == 32
#ifndef BITS
#define BITS      32
#endif
#else
#ifndef BITS
#define BITS      16
#endif
#endif

```

```

#if BITS == 31
#define MSB          0x40000000L
#define ALL_BITS     0x7fffffffL
#define HALF_RANGE   0x20000000L
#define STEP         7
#endif

#if BITS == 32
#define MSB          0x80000000L
#define ALL_BITS     0xffffffffL
#define HALF_RANGE   0x40000000L
#define STEP         7
#endif

#if BITS == 16
#define MSB          0x8000
#define ALL_BITS     0xffff
#define HALF_RANGE   0x4000
#define STEP         11
#endif

static unsigned int r250_buffer[ 250 ];
static int r250_index;

#ifdef NO_PROTO
void r250_init(sd)
int seed;
#else
void r250_init(int sd)
#endif
{
    int j, k;
    unsigned int mask, msb;

#ifdef TRUST_RANDOM
    if (BITS == 32 || BITS == 31)
        srand48( sd );
    else
        srand( sd );
#endif

    else
        set_seed( sd );
#endif

    r250_index = 0;
    for (j = 0; j < 250; j++) /* fill r250 buffer with BITS-1 bit
values */
#ifdef TRUST_RANDOM
    if (BITS == 32 || BITS == 31)
        r250_buffer[j] = (unsigned int)lrand48();
    else

```

```

        r250_buffer[j] = rand();
#endif
#else
        r250_buffer[j] = randlcg();
#endif

    for (j = 0; j < 250; j++)        /* set some MSBs to 1 */
#ifdef TRUST_RAND
        if ( rand() > HALF_RANGE )
            r250_buffer[j] |= MSB;
#else
        if ( randlcg() > HALF_RANGE )
            r250_buffer[j] |= MSB;
#endif

    msb = MSB;        /* turn on diagonal bit */
    mask = ALL_BITS;  /* turn off the leftmost bits */

    for (j = 0; j < BITS; j++)
    {
        k = STEP * j + 3; /* select a word to operate on */
        r250_buffer[k] &= mask; /* turn off bits left of the
diagonal */
        r250_buffer[k] |= msb; /* turn on the diagonal bit */
        mask >>= 1;
        msb >>= 1;
    }
}

unsigned int r250()        /* returns a random unsigned integer */
{
    register int      j;
    register unsigned int new_rand;

    if ( r250_index >= 147 )
        j = r250_index - 147;    /* wrap pointer around */
    else
        j = r250_index + 103;

    new_rand = r250_buffer[ r250_index ] ^ r250_buffer[ j ];
    r250_buffer[ r250_index ] = new_rand;

    if ( r250_index >= 249 )        /* increment pointer for next time
*/
        r250_index = 0;
    else
        r250_index++;

    return new_rand;
}

```

```

float fr250()          /* returns a random float in range 0..1 */
{
    register int      j;
    register unsigned int new_rand;

    if ( r250_index >= 147 )
        j = r250_index - 147;    /* wrap pointer around */
    else
        j = r250_index + 103;

    new_rand = r250_buffer[ r250_index ] ^ r250_buffer[ j ];
    r250_buffer[ r250_index ] = new_rand;

    if ( r250_index >= 249 )      /* increment pointer for next time
*/
        r250_index = 0;
    else
        r250_index++;

    return (float)new_rand / ALL_BITS;
}

#ifdef MAIN

/* test driver    prints out either NMR_RAND values or a histogram
*/

#include <stdio.h>

#define NMR_RAND 5000
#define MAX_BINS 500

#ifdef NO_PROTO
void main(argc, argv)
int argc;
char **argv;
#else
void main(int argc, char **argv)
#endif
{
    int j,k,nmr_bins,seed;
    int bins[MAX_BINS];
    double randm, bin_inc;
    double bin_limit[MAX_BINS];

    if ( argc != 3 )
    {
        printf("Usage -- %s nmr_bins seed\n", argv[0]);
        exit(1);
    }

```

```

nmr_bins = atoi( argv[1] );
if ( nmr_bins > MAX_BINS )
{
    printf("ERROR -- maximum number of bins is %d\n", MAX_BINS);
    exit(1);
}

seed = atoi( argv[2] );

r250_init( seed );

if ( nmr_bins < 1 )      /* just print out the numbers */
{
    for (j = 0; j < NMR_RAND; j++)
        printf("%f\n", dr250() );
    exit(0);
}

bin_inc = 1.0 / nmr_bins;
for (j = 0; j < nmr_bins; j++)      /* initialize bins to zero */
{
    bins[j] = 0;
    bin_limit[j] = (j + 1) * bin_inc;
}

bin_limit[nmr_bins-1] = 1.0e7;      /* make sure all others are in
last bin */

for (j = 0; j < NMR_RAND; j++)
{
    randm = r250() * (double)ALL_BITS;
    for (k = 0; k < nmr_bins; k++)
        if ( randm < bin_limit[k] )
        {
            bins[k]++;
            break;
        }
}

for (j = 0; j < nmr_bins; j++)
    printf("%d\n", bins[j]);
}

#endif

```

```

FILE:      RANDLCG.C
AUTHOR:    Park & Miller
DATE:      1988

```

```

/* randlcg          Linear Congruential Method, the "minimal standard
generator"          Park & Miller, 1988, Comm of the ACM, 31(10), pp.
1192-1201
*/

```

```

/*static char rcsid[] = "@(#)randlcg.c      1.1 15:48:15 11/21/94  EFC";

```

```

#include <math.h>
#include <limits.h>
#include "defines.h"
#include "structs.h"
#include "funcprot.h"

```

```

#define ALL_BITS      0xffffffff

```

```

static long int quotient  = LONG_MAX / 16807L;
static long int reMainder = LONG_MAX % 16807L;

```

```

static long int seed_val = 1L;

```

```

long set_seed(long int sd)
{
    return seed_val = sd;
}

```

```

long get_seed()
{
    return seed_val;
}

```

```

unsigned long int randlcg()          /* returns a random unsigned integer
*/

```

```

{
    if ( seed_val <= quotient )
        seed_val = (seed_val * 16807L) % LONG_MAX;
    else
    {
        long int high_part = seed_val / quotient;
        long int low_part  = seed_val % quotient;

        long int test = 16807L * low_part - reMainder *
high_part;

        if ( test > 0 )

```

```
        seed_val = test;
    else
        seed_val = test - LONG_MAX;
}

return seed_val;
}
```

```

FILE:      GLOBALS.H
AUTHOR:    Awele Ndili
DATE:      November 1996

```

```

//=====//
// globals.h                                     //
// Awele Ndili, November 1996                     //
// Global definitions GPS Recv Sim.               //
//=====//

#ifndef _includeGlobals
#define _includeGlobals

//----GEC Plessey Style
chanstruc CH[MAXCHANNELS];

//----Almanac & GPS Signal Structs
almstruc   almanac[SVS_IN_ALMANAC];
double user_latitude;  // in degrees
double user_longitude; // in degrees
double user_height;    // in meters

//----Incoming Signal Properties
SV_Signal  gpsSignal[SVS_IN_ALMANAC];
SV_Signal cwInterference;          970928 CW Interference

double      SV_Elevation_Mask;      // Elevation Mask of SVs
allowed in simulated signal

float awgn_std_dev;                  // Standard deviation of AWGN
float bg_awn_std_dev;                // Standard deviation of AWGN
float cw_interference_ampl;          // Amplitude of CW interference
float      sv_attenuation_ratio;     // SV Signal attenuation ratio
971016
float Ni, Nq;                        // Inphase and quadrature AWGN
float      AWGNSR_dB;                // AWGN-to-Signal Ratio in dB
float      BG_AWGNSR_dB;             // AWGN-to-Signal Ratio in dB
float      CWISR_dB;                 // Interference-to-Signal
Ratio in dB
float      CWInterferenceDopplerOffset; // CW
Interference doppler offset 970928
float      SV_ATTNUATN_dB;           // SV Signal attenuation, in
dB 971016
int      NumberOfSimulatedSVs;      // Number of SVs above Mask angle
int      MaxAlmanacSize;             // Maximum number of SVs to
include in almanac, default=25
///double DelayB4NoiseInclude;

//----Butterworth Filter Parameters, Passband=0.12 thru 0.86 of half the
sample freq., 5.714MHz
double inputX[FILTERBUFFERSIZE];    // filter input, x
double outputY[FILTERBUFFERSIZE];    // filter output, y

```



```

int filterSizeP1; // Size of filter plus 1
int filterIdx;    // filter index into inputX/Y
// 2nd order bwf: Passband=0.14 thru 0.84 of half the sample freq.,
5.714MHz
double butterwf2B[]={0.6625, 0.0, -0.6625};
double butterwf2A[]={1.0, -0.0397, -0.3249};

// 4th order bwf: Passband=0.12 thru 0.86 of half the sample freq.,
5.714MHz
//older: float butterwf4A[]={1.0, -0.086729506, -0.90081485,
0.020876351, 0.31973247};
//older: float butterwf4B[]={0.55578788, 0.0, -1.11157576, 0.0,
0.55578788};

// 4th order bwf: 970608 Passband=if3+-1 Mhz = 4.31+-1Mhz, 0.29 thru
0.46 of half the sample freq., 4x5.714MHz
double butterwf4B[]={0.05380207910216, 0.0, -0.10760415820432, 0.0,
0.05380207910216};
double butterwf4A[]={1.0, -1.270465952312, 1.65989506599545, -
0.84799285998031, 0.46042722879906};

// 6nd order bwf: Passband=0.14 thru 0.84 of half the sample freq.,
5.714MHz
double butterwf6B[]={0.3745, 0.0, -1.1234, 0.0, 1.1234, 0.0, -0.3745};
double butterwf6A[]={1.0, -0.1081, -1.1572, 0.0547, 0.6953, -0.0166, -
0.1378};

//----Noise & Interference Properties
int includeAWGN;           // Include AWGN
int includeBGAWGN;         // Include Background AWGN
int includeCWInterference; // Include CW interference
int includePULSEInterference; // Include Pulsed interference
int includeSVAttenuation;  // Include SV Signal attenuation
//971016
double delayB4AWGN;        // Delay before addition of
AWGN
double delayB4CWInterference; // Delay before addition of CW
Interference
double delayB4SVAttenuation; // Delay before SV Signal attenuation
//971016
float pulseDutyCycle;      // Pulse duty cycle {0 -> 1}
int pulsePositionIsRandom; // Pulsing Scheme
double pulseWidth;         // in seconds
double pulseStartWindow;   // Duration (in seconds), within the
accumulationTime, within which pulse can start

//----DCO Signal Properties
int goldCode[MAXGOLDCODECOUNT][1023];
int GOLDCODECOUNT;

//----Receiver Properties
double if4SampleRate;
double SamplePeriod;

```

```

double SubSamplePeriod; // For upstream filtering 970608
unsigned long Ta; // Acquisition Threshold
unsigned long Tl; // Loss-of-lock Threshold

//----Correlator Properties
double accumulationTime;
double corrSpacing; // in Seconds

//----Acquisition Properties
unsigned int dcoCAOffset;
double dcoMaxFreq;
double dcoMinFreq;
//unsigned long dcoDwellTimePerFreqBin; // in milliseconds or counts
(since lms=accumtime)
//unsigned long dcoDwellTime; // in milliseconds or counts (since
lms=accumtime)

//----Tracking Loop Properties
long Coast; // Coasting_Period, in epochs (or ms, if lepoche=lms)
int carrierSmoothingOn; // is Carrier smoothing of code DLL on? (yes=1,
no=0)

//----Adaptive Quantizer Properties
double AdativeQuantizerLevel1;
double AdativeQuantizerLevel2;
double AdativeQuantizerLevel3;
double qAdjustment;
unsigned long QuantizerTrack[4];
unsigned long QuantizerCnt; // keeps track of number of accesses, to
implement adaptability
unsigned long QuantizerEvaluationCount; // Period, in chips for re-
evaluation of quantizer levels

//----Generic Constants
double totalRunTime;
double realTime;
double twoPi;
int screenPrinting;

double stableQTZlevelP[QTZHISTORY];
double stableQTZlevelM[QTZHISTORY];
unsigned long maxPreAcquistionIQ2;
unsigned long maxIQ2;
unsigned long accumCnt;
unsigned long preAcqAccumCnt;
int whichFLL; // indicator for which FLL in
use: none;4-Quad;fine=0,1,2

// Formerly # defines, now Globals

//----Incoming Signal Properties
//double IF3_Carrier_Freq; // Hz

```

```

//double IF3_CA_Code_Rate;           // cycles per second
int IF3_INIT_CAOFFSET;              // Initial CA Code Offset

//----Receiver Properties
double IF4_Sample_Rate;              // Samples per second
//int MAXCHANNELS;                   // Number of channels
this receiver has

//----Correlator Properties
double INTEGRATIONTIME;              // Sample n Dump time
(seconds)
double EarlyLateCorrSpace;           // Chip Width (Note: 0.5=>.25
each side, for GEC Plessey)

//----Acquisition Properties
unsigned long DetectionThreshold;     // Signal acquisition threshold
unsigned long LockLossThreshold;     // Signal loss-of-lock threshold
double DCO_CA_Code_SRate;             // dco ca code rate during
search (cycles per second)
double DCO_CA_Code_Rate;              // dco ca code rate after lock
(cycles per second)
int DCO_INIT_CAOFFSET;               // Initial dco ca code offset
double FreqSearchBinWidth;           // Hz
double DCOInitFreq;                  // Hz
double DCOMaxDoppler;                // Hz
double Init_CarrFreqErr;              // Hz; Initial offset of GPS
Recv from Tracked signal frequency

//----Tracking Loop Properties
double COASTING_PERIOD;               //Coasting period, in seconds
int EMLREADINGS;                     // Number of EML readings to
integrate for code tracking

//----Adaptive Quantizer Properties
double ADAPTIVEQUANTIZERPERIOD;       // *** Time to reevaluate
quantizer levels, in secs
double A_QUANTIZER_INCREMENT;         // Adaptive Quantizer adjustment
increments

//----Generic Constants
double RUNTIME;                       // Total run time
(seconds)
long run_id;                          // Unique Id assigned to
this run

#endif

```

```

FILE:      DEFINES.H
AUTHOR:    Awele Ndili
DATE:      November 1996

```

```

//=====//
// defines.h
// Awele Ndili, November 1996
// Global definitions GPS Recv Sim.
//=====//

//----Macros
#define BEEP(x) {int beepCnt; for(beepCnt=0;beepCnt<x; beepCnt++)
printf("%c",7);}
#define OPEN_FILE(x,y,z) if((x=fopen(y,z))==NULL){printf("%cCould not
open File %s in %s mode.\n",7,y,z);return -1;}
#define OPEN_FILE2(x,y,z) if((x=fopen(y,z))==NULL){printf("%cCould not
openFile %s in %s mode.\n",7,y,z);return;}

#define iSAR(larg,rcount)      (((long)(larg))>>rcount)
#define ISquare(x)             ((unsigned long)((x)*(x)))

typedef int logical;
#define _TRUE 1
#define _FALSE 0
#define PI 3.1415926535898
#define DTR 0.01745329251994
#define miu 3.986005e14 // m3/sec2;
Gravitation constant
#define omega_earth 7.292115147e-5 // rad/sec; mean
earth rotation rate
#define radius_earth 6371006.84 // m; Average earth
radius. Equatorial r=6378137.0
#define SPEED_OF_LIGHT 2.99792458e8
#define L1 1575.42e6
#define L1_TO_CA 1540.0
#define L1_TO_IF3 1.571111111111111e+09 // Downconversion

//----GEC Plessey specific
#define NACCUM 50 //Max number of bufferable accumulations +21
#define NOISE_FLOOR_FLOAT 97142.0 /* expected accumulator noise
floor. */
#define CODE_DCO_ZERO_DOPPLER 0x2DD49518L
#define CARR_DCO_ZERO_DOPPLER 0x1F7B1B89L

//----Incoming Signal Properties
#define MAXGOLDCODECOUNT 32 // Max Number of
GPS SV signals to include in incoming signal
#define SVS_IN_ALMANAC 25 // Number of GPS
SVs in Almanac

```

```

#define QTZHISTORY 10    // To compute average stabilized qtz level
#define MAXCHANNELS      1    // Number of
channels this receiver has

//---Filtering
#define ORDEROFFILTER    4    // 4th order Butterworth Filter for IF4
#define dFilterA          butterwf4A
#define dFilterB          butterwf4B
#define FILTERBUFFERSIZE    10

/*
#define IF3_Carrier_Freq    100+(35.42-1400.0/45)*1e6    // Hz
#define IF3_CA_Code_Rate    1.023e6    // cycles per
second
#define IF3_INIT_CAOFFSET    14    // Initial CA Code
Offset

//---Receiver Properties
#define IF4_Sample_Rate    (40.0/7e-6)    // Samples per second
#define MAXCHANNELS      1    // Number of
channels this receiver has

//---Correlator Properties
#define INTEGRATIONTIME    0.001    // Sample n Dump time
(seconds)
#define EarlyLateCorrSpace    0.5    // Chip Width
(Note: 0.5=>.25 each side, for GEC Plessey)

//---Acquisition Properties
#define DetectionThreshold    1690000    // Signal
acquisition threshold
#define LockLossThreshold    360000    // Signal loss-of-
lock threshold
#define DCO_CA_Code_SRate    1.02325e6    // dco ca code rate
during search (cycles per second)
#define DCO_CA_Code_Rate    1.023e6    // dco ca code rate
after lock (cycles per second)
#define DCO_INIT_CAOFFSET    0    // Initial dco ca
code offset
#define FreqSearchBinWidth    300    // Hz
#define DCOInitFreq    (35.42-1400.0/45)*1e6    // Hz
#define DCOMaxDoppler    6000    // Hz

//---Tracking Loop Properties
#define COASTING_PERIOD    20    //Coasting period,
in seconds
#define EMLREADINGS    40    // Number of EML
readings to integrate for code tracking

//---Adaptive Quantizer Properties
#define ADAPTIVEQUANTIZERPERIOD    0.0005    // Time to reevaluate
quantizer levels, in secs

```

```
#define A_QUANTIZER_INCREMENT      0.01  // Adaptive Quantizer
adjustment increments

//----Generic Constants
#define RUNTIME                    0.100      // Total run time
(seconds)
*/
```

FILE: FUNCPROT.H
AUTHOR: Awele Ndili
DATE: November 1996

```

void readinGlobals(void);
void printoutGlobals(void);
void RunTimeLoop(void);
void doOneAccum(void);
void initializeVar       (void);
void CodeTrackingLoopOLD(void);
void CarrierTrackingLoop(void);
int get_tokenId(char *token);
void processToken( int dToken, char *value);
void initialize(void);
void setDefaults(void);
void readAlmanac( void );
void ProcAccumPhase(void);
void generateSVsignals ( void );
void updateGPSSignal( double timeOffsetFromToa );
void computeSVparameters( double timeOffsetFromToa, almstruc *dAlmanac,
SV_Signal *dGPSSignal, int recurseFlag);
double kepler(double Mo_now, double ecc);
double getSigStrengthfromElev(double SV_elevation);

void ProcAccum( chanstruc *CHPTR);
void UpdateLockIndicators( chanstruc *CHPTR, accumstruc *A);
void CoastMode( chanstruc *CHPTR);
void CodeLockMonitor( chanstruc *CHPTR, unsigned long I2pQ2);
void CarrierLockMonitor( chanstruc *CHPTR);
void CodeTrackingLoop( chanstruc *CHPTR, accumstruc *A);

// From goldcode.c
void generatePNcodes     ( char *prn1, char *prn2 );
void generateCACodes     ( void );

int if4Quantize (double dAnalogSig);

// From gaussian.c & co
void box_muller(void); /* normal random variate generator */
void box_muller2(void); /* normal random variate generator */
void r250_init(int sd);
long set_seed(long int sd);
long get_seed();
unsigned long int randlcg();
float           fr250( void );

```

```

FILE:      EXTERNS.H
AUTHOR:    Awele Ndili
DATE:      November 1996

```

```

//=====//
// externs.h
// Awele Ndili, November 1996
// Global definitions GPS Recv Sim.
//=====//

//----GEC Plessey Style
extern chanstruc CH[12];

//----ALMANAC info
extern almstruc almanac[SVS_IN_ALMANAC];

//----Incoming Signal Properties
extern SV_Signal gpsSignal[SVS_IN_ALMANAC];
extern SV_Signal cwInterference; // 970928 CW
Interference

extern int GOLDCODECOUNT;
//extern float PL_AMPLITUDE;
extern double SV_Elevation_Mask; // Elevation Mask of SVs
allowed in simulated signal
extern int NumberOfSimulatedSVs; // Number of SVs above Mask
angle
extern int MaxAlmanacSize; // Maximum number of SVs
to include in almanac, default=25
//extern double DelayB4NoiseInclude;

//----Butterworth Filter Parameters
extern double inputX[FILTERBUFFERSIZE]; // filter input, x
extern double outputY[FILTERBUFFERSIZE]; // filter output, y
extern int filterSizePl; // Size of filter plus 1
extern int filterIdx; // filter index into inputX/Y
extern double butterwf2A[];
extern double butterwf2B[];
extern double butterwf4A[];
extern double butterwf4B[];
extern double butterwf6A[];
extern double butterwf6B[];

//----Noise & Interference Properties
extern int includeAWGN; // Include AWGN
extern int includeBGAWGN; // Include Background
AWGN
extern int includeCWInterference; // Include CW interference
extern int includePULSEInterference; // Include Pulsed interference
extern int includeSVAttenuation; // Include SV Signal
attenuation

```



```

extern double delayB4AWGN;           // Delay before addition
of AWGN
extern double delayB4CWInterference; // Delay before addition of CW
Interference
extern double delayB4SVAttenuation;  // Delay before SV Signal
attenuation

extern float      BG_AWGN_SNR_dB;    // Background AWGN-to-
Signal Ratio in dB
extern float      AWGN_SNR_dB;       // AWGN-to-Signal Ratio
in dB
extern float      CWISR_dB;          // Interference-to-
Signal Ratio in dB
extern float      SV_ATTENUATION_dB; // SV Signal attenuation
in dB 971016
extern float      CWInterferenceDopplerOffset; // CW
Interference doppler offset 970923
extern float      Ni, Nq;           // Inphase and
quadrature AWGN
extern float      cw_interference_ampl; // Amplitude of CW
interference
extern float      sv_attenuation_ratio; // SV Signal attenuation ratio
971016
extern float      pulseDutyCycle;    // Pulse duty cycle (0 -
> 1)
extern int        pulsePositionIsRandom; // Pulsing Scheme
extern double     pulseWidth;
extern double     pulseStartWindow;   // Duration, within the
accumulationTime, within which pulse can start

//----DCO Signal Properties
extern int goldCode[MAXGOLDCODECOUNT][1023];

//----Receiver Properties
extern double if4SampleRate;
extern double SamplePeriod;
extern double SubSamplePeriod;      // For upstream filtering 970608
extern unsigned long Ta;             // Acquisition Threshold
extern unsigned long Tl;             // Loss-of-lock Threshold

//----Correlator Properties
extern double accumulationTime;
extern double corrSpacing;           // in Seconds

//----Acquisition Properties
extern unsigned int dcoCAOffset;
extern double dcoMaxFreq;
extern double dcoMinFreq;
//extern unsigned long dcoDwellTimePerFreqBin; // in milliseconds or
counts (since lms=accumtime)
//extern unsigned long dcoDwellTime;           // in milliseconds or counts
(since lms=accumtime)

```

```

//----Tracking Loop Properties
extern long Coast;           // Coasting_Period, in epochs (or ms, if
lepoche=1ms)
extern int carrierSmoothingOn; // is Carrier smoothing of code DLL
on? (yes=1, no=0)

//----Adaptive Quantizer Properties
extern double AdativeQuantizerLevel1;
extern double AdativeQuantizerLevel2;
extern double AdativeQuantizerLevel3;
extern double qAdjustment;
extern unsigned long QuantizerTrack[4];
extern unsigned long QuantizerCnt; // keeps track of number of
accesses, to implement adaptability
extern unsigned long QuantizerEvaluationCount; // Period, in chips for
re-evaluation of quantizer levels

//----Generic Constants
extern double totalRunTime;
extern double realTime;
extern double twoPi;
extern int screenPrinting;

extern double stableQTZlevelP[QTZHISTORY];
extern double stableQTZlevelM[QTZHISTORY];
extern unsigned long maxPreAcquistionIQ2;
extern unsigned long maxIQ2;
extern unsigned long accumCnt;
extern unsigned long preAcqAccumCnt;
extern int whichFLL; // indicator for which
FLL in use: none;4-Quad;fine=0,1,2

// ===== Formerly #defines
=====
//----Incoming Signal Properties
// extern double IF3_Carrier_Freq; // Hz
// extern double IF3_CA_Code_Rate; // cycles per second
extern int IF3_INIT_CAOFFSET; // Initial CA Code
Offset
extern float awgn_std_dev; // Standard deviation of
AWGN
extern float bg_awgn_std_dev; // Standard
deviation of AWGN

//----Receiver Properties
extern double IF4_Sample_Rate; // Samples per
second
//extern int MAXCHANNELS; // Number of
channels this receiver has

//----Correlator Properties
extern double INTEGRATIONTIME; // Sample n Dump
time (seconds)

```

```

FILE:      STRUCTS.H
AUTHOR:    Awele Ndili
DATE:      November 1996

```

```

//=====//
// structs.h
// Awele Ndili, November 1996
// Global Structures GPS Recv Sim.
//=====//

/*
 * Channel accumulation data - GEC Plessey
 */

typedef struct
{
    int IP; /* I (prompt
arm). */
    int QP; /* Q (prompt
arm). */
    int ID; /* I (dithered
arm). */
    int QD; /* Q (dithered
arm). */
    long IDataBit; /* I
(prompt+dithered). */
    long I_Prompt; /* IP +
ID. */
    long Q_Prompt; /* QP +
QD. */
    unsigned long I2_Plus_Q2; /* I_Prompt^2 +
Q_Prompt^2. */
    unsigned _1ms_epoch; /* Instantaneous 1ms epoch
count. */
    unsigned _20ms_epoch; /* Instantaneous 20ms epoch
count. */
} accumstruc;

/*
 * Channel control block. One for each allocated satellite tracking
channel.- GEC Plessey
 */

typedef struct
{
    int channel; /* Channel
number. */
    unsigned SV; /* Current allocated SV, 0=channel is
idle. */
#ifdef NOTINUSE

```

```

    unsigned _SATCNTL; /* Register
address. */
    unsigned _CARRIER_DCO_INCR_HIGH; /* Register
address. */
    unsigned _CARRIER_DCO_INCR_LOW; /* Register
address. */
    unsigned _CODE_DCO_INCR_HIGH; /* Register
address. */
    unsigned _CODE_DCO_INCR_LOW; /* Register
address. */
    unsigned _EPOCH_LOAD; /* Register
address. */
    unsigned _EPOCH_CHECK; /* Register
address. */
    unsigned _EPOCH_COUNT; /* Register
address. */
    unsigned _CODE_PHASE; /* Register
address. */
    unsigned _CODE_DCO_PHASE; /* Register
address. */
    unsigned _I_PROMPT; /* Register
address. */
    unsigned _Q_PROMPT; /* Register
address. */
    unsigned _I_TRACK; /* Register
address. */
    unsigned _Q_TRACK; /* Register
address. */
    unsigned _CARRIER_CYCLE_COUNTER_HIGH; /* Register
address. */
    unsigned _CARRIER_CYCLE_COUNTER_LOW; /* Register
address. */
    unsigned _CARRIER_DCO_PHASE; /* Register
address. */
    unsigned _ACCUM_RESET; /* Register
address. */
#endif

// ANN 961122
double dcoCarrierFreq; // Hz
double dcoCarrierPhaseE; // Radians
double dcoCarrierPhaseL; // Radians
double dcoCodeRate; // Code Rate (cps)
double dcoCodePhaseE; // Chips
double dcoCodePhaseL; // Chips
double dcoCodeRateSamplePeriod; // dcoCodeRate * SamplePeriod
(Chips)
double dcoCodeRateCorrSpacing; // dcoCodeRate * CorrSpacing
(Chips)
double twoPiFrSamplePeriod; // 2pi * dcoCarrierFreq *
SamplePeriod (radians)
double twoPiFrCorrSpacing; // 2pi * dcoCarrierFreq *
CorrelatorSpacing (radians)

```

```

    unsigned long CARRDCO;          /* Hardware carrier DCO increment */
16. /*
    unsigned long CODEDCO;          /* Hardware code DCO increment */
32. /*
    unsigned long pCARRDCO;         /* Hardware carrier DCO increment */
16. /*
    unsigned long pCODEDCO;         /* Hardware code DCO increment */
32. /*
    int CurrDoppBinIndex;           /* Index of Doppler bin now being
searched. */
    long CodeDoppBin;               /* Code DCO adjustment per Doppler
bin. */
    long CarrDoppBin;               /* Carrier DCO adjustment per Doppler
bin. */
    long TotalCodeMovement;         /* Code srchd in curr Dopp bin, 1/512
chips. */
    unsigned INXTACCUM;             /* Index of next available accumulation
entry. */
    unsigned iCURACCUM;             /* Index of last accumulation entry
processed. */
    unsigned AccumPending;          /* How many accums. are pending in the
buffer. */
    accumstruc ACCUM[NACCUM];       /* Accumulation data from
channel. */
    unsigned _1MS_EPOCH_SLEW;       /* Error in 1-ms epoch
counter. */
    unsigned _20MS_EPOCH_SLEW;      /* Error in 20-ms epoch
counter. */
    long EML;                       /* Current Early - Late
accumulation. */
    unsigned NEML;                  /* # samples contained in EML
accumulator. */
    long EMLOLD;                    /* Early - Late accum. at previous filter
cycle. */
    long IM1;                       /* I, previous
millisecond. */
    long QM1;                       /* Q, previous
millisecond. */
    long wdot_c;                    /* Carrier frequency dot, carrier
loop. */
    long phase_error;               /* Carrier phase, carrier
loop. */
    long avg_phase_change;          /* Carrier phase change per
ms. */
    long CdLI;                      /* Code lock indicator,
Av(I**2+Q**2). */
    long CrfrLI;                    /* Carrier frequency lock
indicator. */
    unsigned coasting;              /* Remaining # ms to coast before
unlocking. */
    logical corlk;                  /* 1=Correlation lock has been
achieved. */
    logical carfrlk;                /* 1=Carrier frequency lock has been
achieved. */

```

```

    logical bitlk;                /* 1=Bit sync has been
achieved. */
    int BitNumber;                /* Bit number within
subframe. */
    logical FrameSync;           /* 1=Frame sync has been
achieved */
    logical FrameSyncAndTIC;      /* 1=TIC occurred since frame
sync. */
    logical LostLockDuringLastTIC; /* 1=lost code or carrier lock
. */
    logical LostCodeLockDuringLastTIC; /* 1=lost code
lock. */
    logical LostCarrierLockDuringLastTIC; /* 1=lost carrier
lock. */
    long IPSUM20;                /* Sum of I over current & 19 previous
ms. */
    long DBINTEG[20];            /* ABS(20-ms I integrals), 1-ms 20
epochs. */
    int nDBINTEG;                /* Number of samples in DBINTEG, up to a
max. */
    int BESTEPOCH;               /* Index of biggest
DBINTEG[. */
    unsigned long BESTSUM;        /* Value of biggest
DBINTEG[. */
    unsigned BadDataCntr;        /* # data decode errors since last good
data. */
    // unsigned long wordbuff[FRAME_SIZE_PLUS_2]; /* Buffer for 12 GPS
words. */
    unsigned long codeph;        /* Code phase, this TIC, units 1/512 of a
chip. */
    unsigned long old_codeph;     /* Code phase, prev TIC, " " " "
". */
    unsigned carrdcophase;        /* Carr phase, this TIC, units  $\pi/512$ 
radians. */
    unsigned prevcarrdcophase;    /* Carr phase, prev TIC, " " "
. */
} chanstruc;

/*
 * Almanac data.
 */

typedef struct
{
    int prn;                     /* PRN Number of this satellite */
    float toa;                   /* Reference time, seconds of
refweek. */
    float sqrta;                 /* SQRT(orbital semimajor axis in
meters). */
    float ecc;                   /* Orbital
eccentricity. */
    float inclin;                /* Orbital inclination
(degrees). */

```

```

        float OMEGA;                /* Right ascension at ref time
(degrees). */
        float OMEGAdot;             /* Rate of right ascension
(degrees/sec). */
        float omega;                /* Argument of perigee at ref time
(degrees). */
        float Mo;                   /* Mean anomaly at ref time (degrees).
*/
    } almstruc;

/*
 * GPS Signal Structure
 */

typedef struct
{
    int prn;                        // PRN Number, 1 thru 32
    double codePhase;               // Phase of Gold code
    double carrierPhase;            // Phase of carrier
    double codeRate;                // Code frequency, including
dopper shift
    double carrierFrequency;        // Nominal frequency plus
dopper shift for this SV
    double twoPIfcSamplePeriod;    // Update for carrierphase
(increment each sample period =  $2\pi f_c T$ )
    double codeRateSamplePeriod;   // Update for codephase (increment
each sample period)
    double twoPIfcSubSamplePeriod; // Update for carrierphase
(increment each subsample period =  $2\pi f_c T$ )
    double codeRateSubSamplePeriod; // Update for codephase
(increment each subsample period)
    double carrierAmplitude;        // SNR indicator - function of
satellite elevation
    double elevation;               // in degrees
    double azimuth;                 // in degrees
    double x;                       // SV's position, in
meters, with ref. to user
    double y;                       // SV's position, in
meters
    double z;                       // SV's position, in
meters
} SV_Signal;

```

FILE: R250.H
 AUTHOR: Kirkpatrick, S., and E. Stoll
 DATE: 1981

/* r250.h prototypes for r250 random number generator,

Kirkpatrick, S., and E. Stoll, 1981; "A Very Fast
 Shift-Register Sequence Random Number Generator",
 Journal of Computational Physics, V.40

also:

see W.L. Maier, DDJ May 1991

Modified by Awele Ndili, January 20, 1997

*/

#ifndef _R250_H_
#define _R250_H_ 1.2

#ifdef __cplusplus
extern "C" {
#endif

#ifdef NO_PROTO
void r250_init();
unsigned int r250();
float fr250();

#else
void r250_init(int seed);
unsigned int r250(void);
float fr250(void);
#endif

#ifdef __cplusplus
}
#endif

#endif

FILE: RANDLCG.H
AUTHOR: Park & Miller
DATE: 1988

/* randlcg.h prototypes for the minimal standard random number
generator,

Linear Congruential Method, the "minimal standard generator"
Park & Miller, 1988, Comm of the ACM, 31(10), pp. 1192-1201

rcsid: @(#)randlcg.h 1.1 15:48:09 11/21/94 EFC

*/

#ifndef _RANDLCG_H_
#define _RANDLCG_H_ 1.1

#ifdef __cplusplus
extern "C" {
#endif

#ifdef NO_PROTO
long set_seed();
long get_seed();
unsigned long int randlcg();

#else
long set_seed(long);
long get_seed(long);
unsigned long int randlcg();

#endif

#ifdef __cplusplus
}
#endif

#endif

Appendix D

Hardware Configuration Input Files

This section contains the listings of the input configuration files. "rcv_spec.in". used to generate all results presented in this thesis. Each input file contains the default set of configurations used for one set of runs, associated with a single type of interference. There are seven input files, corresponding to the seven groups of runs, with associated interference types shown in the table below.

Interference Type Group Number	Description
Run 501	AWGN
Run 502	CW Interference at L1
Run 503	Pulsed AWGN
Run 504	Pulsed CW Interference
Run 505	Signal Attenuation
Run 506	CW Interference at L1 + 1 kHz
Run 507	CW Interference at L1 + 7 kHz

Run 501: AWGN

```

//=====
// Receiver Specifications      rcv_spec.in      //
// Awele Ndili, November 1996      //
// This file is read at the start of a run to      //
// set the properties of the receiver.      //
// Format:      //
// #define TOKENNAME      TOKENVALUE      //Optional comments      //
//=====

//----Generic Constants
#define RUNID      501      // Unique Id for this Run
#define RUNTIME      6.000      // Total run time (seconds)
#define PRINT2SCREEN      NO      // Display summary to screen
at 1kHz

//----Noise & Interference Input
#define INCLUDE_AWGN      YES      // Add AWGN? options are YES,
NO
#define DELAY_B4_AWGN      3.0      // Delay before addition of
AWGN, seconds
#define INCLUDE_BG_AWGN YES      // Add Background AWGN (rcv noise)?
// options are YES, NO
#define BG_AWNSR_DB      8      // Background AWGN level, in
dB
#define INCLUDE_CW      NO      // Add CW interference?
options are YES, NO
#define DELAY_B4_CW      3.0      // Delay before addition of CW
int., seconds
#define INCLUDE_PULSE      NO      // Add Pulsed (CW)
interference? options are
// YES, NO
#define PULSING_SCHEME      RANDOM      // Pulse Scheme
(pulse position) - options are FIXED, RANDOM
#define INCLUDE_SVATTENUATION      NO      // Attenuate SV signal?
(simulates fading, blockages, multipath, etc) options are YES, NO
#define DELAY_B4_SVATTENUATION      3.0      // Delay before SV
signal attenuation, seconds

//----Incoming Signal Properties
#define IF3_INIT_CAOFFSET      40      // Initial CA Code
Offset
#define Init_CarrFreqErr      100.0      // Initial Carrier Freq. Error
(Sig - DCO)
#define MAX_ALMANAC_SIZE      25      // Maximum number of GPS
Satellites in almanac, default=25

//----Receiver Properties
#define IF4_Sample_Rate      5.714285714285714e+06      // Samples
per second
#define MAXCHANNELS      1      // Number of channels
this receiver has

```

```

//----Correlator Properties
#define INTEGRATIONTIME      0.001      // Sample n Dump time
(seconds)
#define EarlyLateCorrSpace    0.5        // Chip Width (Note:
0.5=>.25 each side, for GEC Plessey)

//----Acquisition Properties
#define DetectionThreshold    971420 //10dB      // Signal
acquisition threshold
#define LockLossThreshold    191437 //360000      // Signal loss-of-
lock threshold
#define DCO_CA_Code_SRate    1.02325e6      // dco ca code rate
during search (cycles per second)
#define DCO_CA_Code_Rate     1.023e6        // dco ca code rate
after lock (cycles per second)
#define DCO_INIT_CAOFFSET    0              // Initial dco ca code
offset
#define FreqSearchBinWidth   300             // Hz
#define DCOMaxDoppler        6000           // Hz
#define CARRIERSMOOTHINGON YES             // Carrier smoothing on? on
code DLL (YES OR NO) 980417

//----Tracking Loop Properties
#define COASTING_PERIOD      20              //Coasting period, in
seconds
#define EMLREADINGS         40              // Number of EML
readings to integrate for code tracking

//----Adaptive Quantizer Properties
#define ADAPTIVEQUANTIZERPERIOD 0.0005      // Time to
reevaluate quantizer levels, in secs
#define A_QUANTIZER_INCREMENT 0.01         // Adaptive Quantizer
adjustment increments

```

Run 502: CW INTERFERENCE

```

//=====//
// Receiver Specifications      rcv_spec.in      //
// Awele Ndili, November 1996      //
// This file is read at the start of a run to      //
// set the properties of the receiver.      //
// Format:      //
// #define TOKENNAME      TOKENVALUE      //Optional comments      //
//=====//

//----Generic Constants
#define RUNID      502      Unique Id for this Run
#define RUNTIME      6.000      // Total run time (seconds)
#define PRINT2SCREEN      NO      // Display summary to screen
at 1kHz

//----Noise & Interference Input
#define INCLUDE_AWGN      NO      // Add AWGN? options are YES, NO
#define DELAY_B4_AWGN      3.0      // Delay before addition of
AWGN, seconds
#define INCLUDE_BG_AWGN      YES      // Add Background AWGN (rcv
noise)? options are YES, NO
#define BG_AWGNSR_DB      8      // Background AWGN level, in
dB
#define INCLUDE_CW      YES      // Add CW interference?
options are YES, NO
#define DELAY_B4_CW      3.0      // Delay before addition of CW
int., seconds
#define INCLUDE_PULSE      NO      // Add Pulsed (CW)
interference? options are YES, NO
#define PULSING_SCHEME      RANDOM      // Pulse Scheme
(pulse position) - options are FIXED, RANDOM
#define INCLUDE_SVATTENUATION      NO      // Attenuate SV signal?
(simulates fading, blockages, multipath, etc) options are YES, NO
#define DELAY_B4_SVATTENUATION      3.0      // Delay before SV
signal attenuation, seconds

//----Incoming Signal Properties
#define IF3_INIT_CAOFFSET      40      // Initial CA Code Offset
#define Init_CarrFreqErr      100.0      // Initial Carrier Freq. Error
(Sig - DCO)
#define MAX_ALMANAC_SIZE      25      // Maximum number of GPS
Satellites in almanac, default=25

//----Receiver Properties
#define IF4_Sample_Rate      5.714285714285714e+06      // Samples
per second
#define MAXCHANNELS      1      // Number of channels this
receiver has

//----Correlator Properties
#define INTEGRATIONTIME      0.001      // Sample n Dump time

```

```

(seconds)
#define EarlyLateCorrSpace    0.5          // Chip Width (Note: 0.5=>.25
each side, for GEC Plessey)

//----Acquisition Properties
#define DetectionThreshold    971420 //10dB          //          Signal
acquisition threshold
#define LockLossThreshold    191437 //360000          // Signal loss-of-
lock threshold
#define DCO_CA_Code_SRate    1.02325e6          // dco  ca  code  rate
during search (cycles per second)
#define DCO_CA_Code_Rate    1.023e6          // dco  ca  code  rate
after lock (cycles per second)
#define DCO_INIT_CAOFFSET    0          // Initial dco ca code offset
#define FreqSearchBinWidth    300          // Hz
#define DCOMaxDoppler        6000          // Hz
#define CARRIERSMOOTHINGON  YES          // Carrier smoothing on? on
code DLL (YES OR NO) 980417

//----Tracking Loop Properties
#define COASTING_PERIOD      20          //Coasting period, in seconds
#define EMLREADINGS         40          // Number of EML readings to
integrate for code tracking

//----Adaptive Quantizer Properties
#define ADAPTIVEQUANTIZERPERIOD    0.0005          //          Time          to
reevaluate quantizer levels, in secs
#define A_QUANTIZER_INCREMENT    0.01          //          Adaptive          Quantizer
adjustment increments

```


Run 503: PULSED AWGN

```

//=====//
// Receiver Specifications      rcv_spec.in      //
// Awele Ndili, November 1996      //
// This file is read at the start of a run to      //
// set the properties of the receiver.      //
// Format:      //
// #define TOKENNAME      TOKENVALUE      //Optional comments      //
//=====//

//----Generic Constants
#define RUNID      503      // Unique Id for this Run
#define RUNTIME      6.000      // Total run time (seconds)
#define PRINT2SCREEN      NO      // Display summary to screen
at 1kHz

//----Noise & Interference Input
#define INCLUDE_AWGN      YES      // Add AWGN? options are YES,
NO
#define DELAY_B4_AWGN      3.0      // Delay before addition of
AWGN, seconds
#define INCLUDE_BG_AWGN      YES      // Add Background AWGN (rcv
noise)? options are YES, NO
#define BG_AWGNSR_DB      8      // Background AWGN level, in
dB
#define INCLUDE_CW      NO      // Add CW interference?
options are YES, NO
#define DELAY_B4_CW      3.0      // Delay before addition of CW
int., seconds
#define INCLUDE_PULSE      YES      // Add Pulsed (CW)
interference? options are YES, NO
#define PULSING_SCHEME      RANDOM      // Pulse Scheme
(pulse position) - options are FIXED, RANDOM
#define INCLUDE_SVATTENUATION      NO      // Attenuate SV signal?
(simulates fading, blockages, multipath, etc) options are YES, NO
#define DELAY_B4_SVATTENUATION      3.0      // Delay before SV
signal attenuation, seconds

//----Incoming Signal Properties
#define IF3_INIT_CAOFFSET      40      // Initial CA Code Offset
#define Init_CarrFreqErr      100.0      // Initial Carrier Freq. Error
(Sig - DCO)
#define MAX_ALMANAC_SIZE      25      // Maximum number of GPS
Satellites in almanac, default=25

//----Receiver Properties
#define IF4_Sample_Rate      5.714285714285714e+06      // Samples
per second
#define MAXCHANNELS      1      // Number of channels this
receiver has

//----Correlator Properties

```

```

#define INTEGRATIONTIME      0.001      // Sample n Dump time
(seconds)
#define EarlyLateCorrSpace    0.5        // Chip Width (Note: 0.5=>.25
each side, for GEC Plessey)

//----Acquisition Properties
#define DetectionThreshold     971420 //10dB          // Signal
acquisition threshold
#define LockLossThreshold      191437 //360000        // Signal loss-of-
lock threshold
#define DCO_CA_Code_SRate      1.02325e6          // dco ca code rate
during search (cycles per second)
#define DCO_CA_Code_Rate       1.023e6           // dco ca code rate
after lock (cycles per second)
#define DCO_INIT_CAOFFSET      0                // Initial dco ca code offset
#define FreqSearchBinWidth     300              // Hz
#define DCOMaxDoppler          6000             // Hz
#define CARRIERSMOOTHINGON    YES              // Carrier smoothing on? on
code DLL (YES OR NO) 980417

//----Tracking Loop Properties
#define COASTING_PERIOD        20              //Coasting period, in seconds
#define EMLREADINGS            40              // Number of EML readings to
integrate for code tracking

//----Adaptive Quantizer Properties
#define ADAPTIVEQUANTIZERPERIOD 0.0005         // Time to
reevaluate quantizer levels, in secs
#define A_QUANTIZER_INCREMENT  0.01           // Adaptive Quantizer
adjustment increments

```

Run 504: PULSED CW INTERFERENCE

```

//=====//
// Receiver Specifications      rcv_spec.in      //
// Awele Ndili, November 1996      //
// This file is read at the start of a run to      //
// set the properties of the receiver.      //
// Format:      //
// #define TOKENNAME      TOKENVALUE      //Optional comments      //
//=====//

//----Generic Constants
#define RUNID      504      Unique Id for this Run
#define RUNTIME      6.000      // Total run time (seconds)
#define PRINT2SCREEN      NO      // Display summary to screen
at 1kHz

//----Noise & Interference Input
#define INCLUDE_AWGN      NO      // Add AWGN? options are YES,
NO
#define DELAY_B4_AWGN      3.0      // Delay before addition of
AWGN, seconds
#define INCLUDE_BG_AWGN      YES      // Add Background AWGN (rcv
noise)? options are YES, NO
#define BG_AWGN_SR_DB      3      // Background AWGN level, in
dB
#define INCLUDE_CW      YES      // Add CW interference?
options are YES, NO
#define DELAY_B4_CW      3.0      // Delay before addition of CW
int., seconds
#define INCLUDE_PULSE      YES      // Add Pulsed (CW)
interference? options are YES, NO
#define PULSING_SCHEME      RANDOM      // Pulse Scheme
(pulse position) - options are FIXED, RANDOM
#define INCLUDE_SVATTENUATION      NO      // Attenuate SV signal?
(simulates fading, blockages, multipath, etc) options are YES, NO
#define DELAY_B4_SVATTENUATION      3.0      // Delay before SV
signal attenuation, seconds

//----Incoming Signal Properties
#define IF3_INIT_CAOFFSET      40      // Initial CA Code Offset
#define Init_CarrFreqErr      100.0      // Initial Carrier Freq. Error
(Sig - DCO)
#define MAX_ALMANAC_SIZE      25      // Maximum number of GPS
Satellites in almanac, default=25

//----Receiver Properties
#define IF4_Sample_Rate      5.714285714285714e+06      // Samples
per second
#define MAXCHANNELS      1      // Number of channels this
receiver has

//----Correlator Properties

```

```

#define INTEGRATIONTIME      0.001      // Sample n Dump time
(seconds)
#define EarlyLateCorrSpace   0.5         // Chip Width (Note: 0.5=>.25
each side, for GEC Plessey)

//----Acquisition Properties
#define DetectionThreshold    971420 //10dB          // Signal
acquisition threshold
#define LockLossThreshold    191437 //360000         // Signal loss-of-
lock threshold
#define DCO_CA_Code_SRate    1.02325e6           // dco ca code rate
during search (cycles per second)
#define DCO_CA_Code_Rate     1.023e6           // dco ca code rate
after lock (cycles per second)
#define DCO_INIT_CAOFFSET    0                // Initial dco ca code offset
#define FreqSearchBinWidth   300              // Hz
#define DCOMaxDoppler        6000             // Hz
#define CARRIERSMOOTHINGON YES              // Carrier smoothing on? on
code DLL (YES OR NO) 980417

//----Tracking Loop Properties
#define COASTING_PERIOD      20              //Coasting period, in seconds
#define EMLREADINGS         40              // Number of EML readings to
integrate for code tracking

//----Adaptive Quantizer Properties
#define ADAPTIVEQUANTIZERPERIOD 0.0005      // Time to
reevaluate quantizer levels, in secs
#define A_QUANTIZER_INCREMENT 0.01         // Adaptive Quantizer
adjustment increments

```

Run 505: SIGNAL ATTENUATION

```

//=====//
// Receiver Specifications      rcv_spec.in      //
// Awele Ndili, November 1996      //
// This file is read at the start of a run to      //
// set the properties of the receiver.      //
// Format:      //
// #define TOKENNAME      TOKENVALUE      //Optional comments      //
//=====//

//----Generic Constants
#define RUNID      505      // Unique Id for this Run
#define RUNTIME      6.000      // Total run time (seconds)
#define PRINT2SCREEN      NO      // Display summary to screen
at 1kHz

//----Noise & Interference Input
#define INCLUDE_AWGN      NO      // Add AWGN? options are YES,
NO
#define DELAY_B4_AWGN      3.0      // Delay before addition of
AWGN, seconds
#define INCLUDE_BG_AWGN      YES      // Add Background AWGN (rcv
noise)? options are YES, NO
#define BG_AWGN_SR_DB      3      // Background AWGN level, in
dB
#define INCLUDE_CW      NO      // Add CW interference?
options are YES, NO
#define DELAY_B4_CW      3.0      // Delay before addition of CW
int., seconds
#define INCLUDE_PULSE      NO      // Add Pulsed (CW)
interference? options are YES, NO
#define PULSING_SCHEME      RANDOM      // Pulse Scheme
(pulse position) - options are FIXED, RANDOM
#define INCLUDE_SVATTENUATION      YES      // Attenuate SV signal?
(simulates fading, blockages, multipath, etc) options are YES, NO
#define DELAY_B4_SVATTENUATION      3.0      // Delay before SV
signal attenuation, seconds

//----Incoming Signal Properties
#define IF3_INIT_CAOFFSET      40      // Initial CA Code Offset
#define Init_CarrFreqErr      100.0      // Initial Carrier Freq. Error
(Sig - DCO)
#define MAX_ALMANAC_SIZE      25      // Maximum number of GPS
Satellites in almanac, default=25

//----Receiver Properties
#define IF4_Sample_Rate      5.714285714285714e+06      // Samples
per second
#define MAXCHANNELS      1      // Number of channels this
receiver has

//----Correlator Properties

```

```

#define INTEGRATIONTIME      0.001      // Sample n Dump time
(seconds)
#define EarlyLateCorrSpace    0.5        // Chip Width (Note: 0.5=>.25
each side, for GEC Plessey)

//----Acquisition Properties
#define DetectionThreshold     971420 //10dB          // Signal
acquisition threshold
#define LockLossThreshold     191437 //360000          // Signal loss-of-
lock threshold
#define DCO_CA_Code_SRate     1.02325e6          // dco ca code rate
during search (cycles per second)
#define DCO_CA_Code_Rate      1.023e6          // dco ca code rate
after lock (cycles per second)
#define DCO_INIT_CAOFFSET      0              // Initial dco ca code offset
#define FreqSearchBinWidth     300            // Hz
#define DCOMaxDoppler          6000           // Hz
#define CARRIERSMOOTHINGON    YES           // Carrier smoothing on? on
code DLL (YES OR NO) 980417

//----Tracking Loop Properties
#define COASTING_PERIOD        20             //Coasting period, in seconds
#define EMLREADINGS            40            // Number of EML readings to
integrate for code tracking

//----Adaptive Quantizer Properties
#define ADAPTIVEQUANTIZERPERIOD 0.0005        // Time to
reevaluate quantizer levels, in secs
#define A_QUANTIZER_INCREMENT  0.01          // Adaptive Quantizer
adjustment increments

```

Run 506: CW Interference at L1 + 1 KHz

```

//=====//
// Receiver Specifications      rcv_spec.in      //
// Awele Ndili, November 1996      //
// This file is read at the start of a run to      //
// set the properties of the receiver.      //
// Format:      //
// #define TOKENNAME      TOKENVALUE //Optional comments //
//=====//

//----Generic Constants
#define RUNID      506      // Unique Id for this Run
#define RUNTIME      6.000      // Total run time (seconds)
#define PRINT2SCREEN      NO      // Display summary to screen
at 1kHz

//----Noise & Interference Input
#define INCLUDE_AWGN      NO      // Add AWGN? options are YES,
NO
#define DELAY_B4_AWGN      3.0      // Delay before addition of
AWGN, seconds
#define INCLUDE_BG_AWGN      YES      // Add Background AWGN (rcv
noise)? options are YES, NO
#define BG_AWGN_SR_DB      8      // Background AWGN level, in
dB
#define INCLUDE_CW      YES      // Add CW interference?
options are YES, NO
#define DELAY_B4_CW      3.0      // Delay before addition of CW
int., seconds
#define INCLUDE_PULSE      NO      // Add Pulsed (CW)
interference? options are YES, NO
#define PULSING_SCHEME      RANDOM      // Pulse Scheme
(pulse position) - options are FIXED, RANDOM
#define INCLUDE_SVATTENUATION      NO      // Attenuate SV signal?
(simulates fading, blockages, multipath, etc) options are YES, NO
#define DELAY_B4_SVATTENUATION      3.0      // Delay before SV
signal attenuation, seconds

//----Incoming Signal Properties
#define IF3_INIT_CA_OFFSET      40      // Initial CA Code Offset
#define Init_CarrFreqErr      100.0      // Initial Carrier Freq. Error
(Sig - DCO)
#define MAX_ALMANAC_SIZE      25      // Maximum number of GPS
Satellites in almanac, default=25

//----Receiver Properties
#define IF4_Sample_Rate      5.714285714285714e+06      // Samples
per second
#define MAXCHANNELS      1      // Number of channels this
receiver has

//----Correlator Properties

```

```

#define INTEGRATIONTIME      0.001      // Sample n Dump time
(seconds)
#define EarlyLateCorrSpace   0.5         // Chip Width (Note: 0.5=>.25
each side, for GEC Plessey)

//----Acquisition Properties
#define DetectionThreshold    971420 //10dB           // Signal
acquisition threshold
#define LockLossThreshold    191437 //360000          // Signal loss-of-
lock threshold
#define DCO_CA_Code_SRate    1.02325e6           // dco ca code rate
during search (cycles per second)
#define DCO_CA_Code_Rate     1.023e6           // dco ca code rate
after lock (cycles per second)
#define DCO_INIT_CAOFFSET    0                // Initial dco ca code offset
#define FreqSearchBinWidth   300              // Hz
#define DCOMaxDoppler        6000             // Hz
#define CARRIERSMOOTHINGON YES              // Carrier smoothing on? on
code DLL (YES OR NO) 980417

//----Tracking Loop Properties
#define COASTING_PERIOD      20              //Coasting period, in seconds
#define EMLREADINGS         40              // Number of EML readings to
integrate for code tracking

//----Adaptive Quantizer Properties
#define ADAPTIVEQUANTIZERPERIOD 0.0005      Time to
reevaluate quantizer levels, in secs
#define A_QUANTIZER_INCREMENT 0.01         // Adaptive Quantizer
adjustment increments

```


Run 507: CW Interference at L1 + 7 KHz

```

//=====//
// Receiver Specifications      rcv_spec.in      //
// Awele Ndili, November 1996      //
// This file is read at the start of a run to      //
// set the properties of the receiver.      //
// Format:      //
// #define TOKENNAME      TOKENVALUE      //Optional comments      //
//=====//

//----Generic Constants
#define RUNID      507      Unique Id for this Run
#define RUNTIME      6.000      // Total run time (seconds)
#define PRINT2SCREEN      NO      // Display summary to screen
at 1kHz

//----Noise & Interference Input
#define INCLUDE_AWGN      NO      // Add AWGN? options are YES,
NO
#define DELAY_B4_AWGN      3.0      // Delay before addition of
AWGN, seconds
#define INCLUDE_BG_AWGN      YES      // Add Background AWGN (rcv
noise)? options are YES, NO
#define BG_AWGNSR_DB      8      // Background AWGN level, in
dB
#define INCLUDE_CW      YES      // Add CW interference?
options are YES, NO
#define DELAY_B4_CW      3.0      // Delay before addition of CW
int., seconds
#define INCLUDE_PULSE      NO      // Add Pulsed (CW)
interference? options are YES, NO
#define PULSING_SCHEME      RANDOM      // Pulse Scheme
(pulse position) - options are FIXED, RANDOM
#define INCLUDE_SVATTENUATION      NO      // Attenuate SV signal?
(simulates fading, blockages, multipath, etc) options are YES, NO
#define DELAY_B4_SVATTENUATION      3.0      // Delay before SV
signal attenuation, seconds

//----Incoming Signal Properties
#define IF3_INIT_CAOFFSET      40      // Initial CA Code Offset
#define Init_CarrFreqErr      100.0      // Initial Carrier Freq. Error
(Sig - DCO)
#define MAX_ALMANAC_SIZE      25      // Maximum number of GPS
Satellites in almanac, default=25

//----Receiver Properties
#define IF4_Sample_Rate      5.714285714285714e+06      // Samples
per second
#define MAXCHANNELS      1      // Number of channels this
receiver has

//----Correlator Properties

```

```

#define INTEGRATIONTIME      0.001      // Sample n Dump time
(seconds)
#define EarlyLateCorrSpace    0.5        // Chip Width (Note: 0.5=>.25
each side, for GEC Plessey)

//----Acquisition Properties
#define DetectionThreshold    971420 //10dB      // Signal
acquisition threshold
#define LockLossThreshold    191437 //360000      // Signal loss-of-
lock threshold
#define DCO_CA_Code_SRate    1.02325e6      // dco ca code rate
during search (cycles per second)
#define DCO_CA_Code_Rate     1.023e6        // dco ca code rate
after lock (cycles per second)
#define DCO_INIT_CAOFFSET    0              // Initial dco ca code offset
#define FreqSearchBinWidth   300            // Hz
#define DCOMaxDoppler        6000          // Hz
#define CARRIERSMOOTHINGON YES            // Carrier smoothing on? on
code DLL (YES OR NO) 980417

//----Tracking Loop Properties
#define COASTING_PERIOD      20             //Coasting period, in seconds
#define EMLREADINGS         40             // Number of EML readings to
integrate for code tracking

//----Adaptive Quantizer Properties
#define ADAPTIVEQUANTIZERPERIOD 0.0005      // Time to
reevaluate quantizer levels, in secs
#define A_QUANTIZER_INCREMENT 0.01         // Adaptive Quantizer
adjustment increments

```


Appendix E

Run Configuration Input Files

This section contains the listings of the run configuration files, "run_spec.in", used to generate all results presented in this thesis. Each file contains the list of all runs for one interference type, and enables entire groups of runs to be executed in 'batch' mode. There are seven input files, corresponding to the seven groups of runs, with associated interference types shown in the table below.

Interference Type	Description
Group Number	
Run 501	AWGN, 0 to 40 dB NSR
Run 502	CW Interference at L1, 0 to 40 dB JSR
Run 503	Pulsed AWGN, 150 dB, 0 to 100 % duty cycle
Run 504	Pulsed CW Interference at L1, 150 dB, 0 to 100 % duty cycle
Run 505	Signal Attenuation, 0 to 24 dB
Run 506	CW Interference at L1 + 1 kHz, 0 to 40 dB JSR
Run 507	CW Interference at L1 + 7 kHz, 0 to 40 dB JSR

Run 501: AWGN

0	0	0	0	0
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0
4	0	0	0	0
5	0	0	0	0
6	0	0	0	0
7	0	0	0	0
8	0	0	0	0
9	0	0	0	0
10	0	0	0	0
11	0	0	0	0
12	0	0	0	0
13	0	0	0	0
14	0	0	0	0
15	0	0	0	0
16	0	0	0	0
17	0	0	0	0
18	0	0	0	0
19	0	0	0	0
20	0	0	0	0
21	0	0	0	0
22	0	0	0	0
23	0	0	0	0
24	0	0	0	0
25	0	0	0	0
26	0	0	0	0
27	0	0	0	0
28	0	0	0	0
29	0	0	0	0
30	0	0	0	0
31	0	0	0	0
32	0	0	0	0
33	0	0	0	0
34	0	0	0	0
35	0	0	0	0
36	0	0	0	0
37	0	0	0	0
38	0	0	0	0
39	0	0	0	0
40	0	0	0	0
-1	-1	-1	-1	-1
AWGN_DB	CW_DB	PULSE_DC	DOPPLER_OF FSET	SV_ATTENUA TION_DB

Run 502: CW INTERFERENCE

0	0	0	0	0
0	1	0	0	0
0	2	0	0	0
0	3	0	0	0
0	4	0	0	0
0	5	0	0	0
0	6	0	0	0
0	7	0	0	0
0	8	0	0	0
0	9	0	0	0
0	10	0	0	0
0	11	0	0	0
0	12	0	0	0
0	13	0	0	0
0	14	0	0	0
0	15	0	0	0
0	16	0	0	0
0	17	0	0	0
0	18	0	0	0
0	19	0	0	0
0	20	0	0	0
0	21	0	0	0
0	22	0	0	0
0	23	0	0	0
0	24	0	0	0
0	25	0	0	0
0	26	0	0	0
0	27	0	0	0
0	28	0	0	0
0	29	0	0	0
0	30	0	0	0
0	31	0	0	0
0	32	0	0	0
0	33	0	0	0
0	34	0	0	0
0	35	0	0	0
0	36	0	0	0
0	37	0	0	0
0	38	0	0	0
0	39	0	0	0
0	40	0	0	0
-1	-1	-1	-1	-1
AWGN_DB	CW_DB	PULSE_DC	DOPPLER_OF FSET	SV_ATTENUA TION_DB

Run 503: PULSED AWGN

150	0	0	0	0
150	0	0.02	0	0
150	0	0.04	0	0
150	0	0.06	0	0
150	0	0.08	0	0
150	0	0.1	0	0
150	0	0.12	0	0
150	0	0.14	0	0
150	0	0.16	0	0
150	0	0.18	0	0
150	0	0.2	0	0
150	0	0.22	0	0
150	0	0.24	0	0
150	0	0.26	0	0
150	0	0.28	0	0
150	0	0.3	0	0
150	0	0.32	0	0
150	0	0.34	0	0
150	0	0.36	0	0
150	0	0.38	0	0
150	0	0.4	0	0
150	0	0.42	0	0
150	0	0.44	0	0
150	0	0.46	0	0
150	0	0.48	0	0
150	0	0.5	0	0
150	0	0.52	0	0
150	0	0.54	0	0
150	0	0.56	0	0
150	0	0.58	0	0
150	0	0.6	0	0
150	0	0.62	0	0
150	0	0.64	0	0
150	0	0.66	0	0
150	0	0.68	0	0
150	0	0.7	0	0
150	0	0.72	0	0
150	0	0.74	0	0
150	0	0.76	0	0
150	0	0.78	0	0
150	0	0.8	0	0
150	0	0.82	0	0
150	0	0.84	0	0
150	0	0.86	0	0
150	0	0.88	0	0
150	0	0.9	0	0
150	0	0.92	0	0

150	0	0.94	0	0
150	0	0.96	0	0
150	0	0.98	0	0
150	0	1	0	0
-1	-1	-1	-1	-1
AWGN_DB	CW_DB	PULSE_DC	DOPPLER_OF FSET	SV_ATTENUA TION_DB

Run 504: PULSED CW INTERFERENCE

0	150	0	0	0
0	150	0.02	0	0
0	150	0.04	0	0
0	150	0.06	0	0
0	150	0.08	0	0
0	150	0.1	0	0
0	150	0.12	0	0
0	150	0.14	0	0
0	150	0.16	0	0
0	150	0.18	0	0
0	150	0.2	0	0
0	150	0.22	0	0
0	150	0.24	0	0
0	150	0.26	0	0
0	150	0.28	0	0
0	150	0.3	0	0
0	150	0.32	0	0
0	150	0.34	0	0
0	150	0.36	0	0
0	150	0.38	0	0
0	150	0.4	0	0
0	150	0.42	0	0
0	150	0.44	0	0
0	150	0.46	0	0
0	150	0.48	0	0
0	150	0.5	0	0
0	150	0.52	0	0
0	150	0.54	0	0
0	150	0.56	0	0
0	150	0.58	0	0
0	150	0.6	0	0
0	150	0.62	0	0
0	150	0.64	0	0
0	150	0.66	0	0
0	150	0.68	0	0
0	150	0.7	0	0
0	150	0.72	0	0
0	150	0.74	0	0
0	150	0.76	0	0
0	150	0.78	0	0
0	150	0.8	0	0
0	150	0.82	0	0
0	150	0.84	0	0
0	150	0.86	0	0
0	150	0.88	0	0
0	150	0.9	0	0
0	150	0.92	0	0

0	150	0.94	0	0
0	150	0.96	0	0
0	150	0.98	0	0
0	150	1	0	0
-1	-1	-1	-1	-1
AWGN_DB	CW_DB	PULSE_DC	DOPPLER_OF FSET	SV_ATTENUA TION_DB

Run 505: SIGNAL ATTENUATION

0	0	0	0	0
0	0	0	0	1
0	0	0	0	2
0	0	0	0	3
0	0	0	0	4
0	0	0	0	5
0	0	0	0	6
0	0	0	0	7
0	0	0	0	8
0	0	0	0	9
0	0	0	0	10
0	0	0	0	11
0	0	0	0	12
0	0	0	0	13
0	0	0	0	14
0	0	0	0	15
0	0	0	0	16
0	0	0	0	17
0	0	0	0	18
0	0	0	0	19
0	0	0	0	20
0	0	0	0	21
0	0	0	0	22
0	0	0	0	23
0	0	0	0	24
0	0	0	0	25
0	0	0	0	26
0	0	0	0	27
0	0	0	0	28
0	0	0	0	29
0	0	0	0	30
-1	-1	-1	-1	-1
AWGN_DB	CW_DB	PULSE_DC	DOPPLER_OF FSET	SV_ATTENUA TION_DB

Run 506: CW Interference at L1 + 1 KHz

0	0	0	1000	0
0	1	0	1000	0
0	2	0	1000	0
0	3	0	1000	0
0	4	0	1000	0
0	5	0	1000	0
0	6	0	1000	0
0	7	0	1000	0
0	8	0	1000	0
0	9	0	1000	0
0	10	0	1000	0
0	11	0	1000	0
0	12	0	1000	0
0	13	0	1000	0
0	14	0	1000	0
0	15	0	1000	0
0	16	0	1000	0
0	17	0	1000	0
0	18	0	1000	0
0	19	0	1000	0
0	20	0	1000	0
0	21	0	1000	0
0	22	0	1000	0
0	23	0	1000	0
0	24	0	1000	0
0	25	0	1000	0
0	26	0	1000	0
0	27	0	1000	0
0	28	0	1000	0
0	29	0	1000	0
0	30	0	1000	0
0	31	0	1000	0
0	32	0	1000	0
0	33	0	1000	0
0	34	0	1000	0
0	35	0	1000	0
0	36	0	1000	0
0	37	0	1000	0
0	38	0	1000	0
0	39	0	1000	0
0	40	0	1000	0
-1	-1	-1	-1	-1
AWGN_DB	CW_DB	PULSE_DC	DOPPLER_OF FSET	SV_ATTENUA TION_DB

Run 507: CW Interference at L1 + 7 KHz

0	0	0	7000	0
0	1	0	7000	0
0	2	0	7000	0
0	3	0	7000	0
0	4	0	7000	0
0	5	0	7000	0
0	6	0	7000	0
0	7	0	7000	0
0	8	0	7000	0
0	9	0	7000	0
0	10	0	7000	0
0	11	0	7000	0
0	12	0	7000	0
0	13	0	7000	0
0	14	0	7000	0
0	15	0	7000	0
0	16	0	7000	0
0	17	0	7000	0
0	18	0	7000	0
0	19	0	7000	0
0	20	0	7000	0
0	21	0	7000	0
0	22	0	7000	0
0	23	0	7000	0
0	24	0	7000	0
0	25	0	7000	0
0	26	0	7000	0
0	27	0	7000	0
0	28	0	7000	0
0	29	0	7000	0
0	30	0	7000	0
0	31	0	7000	0
0	32	0	7000	0
0	33	0	7000	0
0	34	0	7000	0
0	35	0	7000	0
0	36	0	7000	0
0	37	0	7000	0
0	38	0	7000	0
0	39	0	7000	0
0	40	0	7000	0
-1	-1	-1	-1	-1
AWGN_DB	CW_DB	PULSE_DC	DOPPLER_OF FSET	SV_ATTENUA TION_DB

Bibliography

- [1] P. Enge, "Local Area Augmentation of GPS for the Precision Approach of Aircraft", Proceedings of the IEEE, 1997
- [2] Eschenbach, R. "GPS Applications in General Aviation", Global Positioning Systems, Theory and Applications, Vol. II, Parkinson, B. W., Spilker Jr, J. J., Axelrad, P., Enge, P., AIAA 1996
- [3] P. Enge, T. Walter, S. Pullen, C. Kee, YC Chao, and YJ Tsai, "Wide Area Augmentation of the Global Positioning System", Proceedings of the IEEE, vol 84, no. 8, August 1996.
- [4] Anon., "1992 Federal Radionavigation Plan," U.S. Departments of Transportation and Defense, DOT-VNTSC-RSPA-92-2/DOD-4650.5, Washington, DC, 1992. (as referenced by 5)
- [5] R. Braff et al, "Applications of GPS to Air Traffic Control", Global Positioning Systems, Theory and Applications, Vol. II, Parkinson, B. W., Spilker Jr, J. J., Axelrad, P., Enge, P., AIAA 1996
- [6] F. Amoroso, "Performance of the Adaptive A/D Converter in Combined CW and Gaussian Interference", IEEE Transactions on Communication, vol. COM-34, no. 3, March 1986.
- [7] R. Grover Brown, "Receiver Autonomous Integrity Monitoring", Global Positioning Systems, Theory and Applications, Vol. II, Parkinson, B. W., Spilker Jr, J. J., Axelrad, P., Enge, P., AIAA 1996
- [8] Parkinson, B. W., and Axelrad, P., "Autonomous GPS Integrity Monitoring Using the Pseudorange Residual," Navigation (Washington), Vol. 35, No. 2, 1988, pp. 255-274.
- [9] GPS Builder-2 Designer's Guide, GEC Plessey Semiconductors, GPS Group, Cheney Manor, Swindon, Wiltshire, UK
- [10] Cahn, C. R. et al, "Software Implementation of a PRN Spread Spectrum Receiver to Accommodate Dynamics", IEEE Transactions on Communications, Vol. COM-25, No. 8, August 1977.
- [11] Johnson, M. W., Erlandson, R., "GNSS Receiver Interference Susceptibility and Civil Aviation Impact", Presented at ION-GPS 95, September 1995.
- [12] J.J. Spilker Jr., "Signal Structure and Theoretical Performance", Global Positioning Systems, Theory and Applications, Vol. I, Parkinson, B. W., Spilker Jr, J. J., Axelrad, P., Enge, P., AIAA 1996
- [13] Hegarty, C., J., "Analytical Derivation of Maximum Tolerable In-Band Interference Levels for Aviation Applications of GNSS", Presented at the RTCA SC 159. Working Group 4A meeting, 1997.

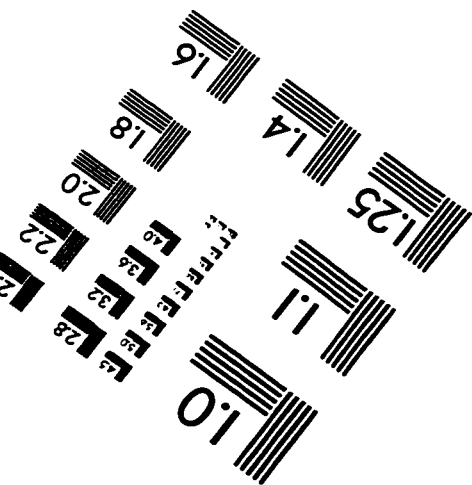
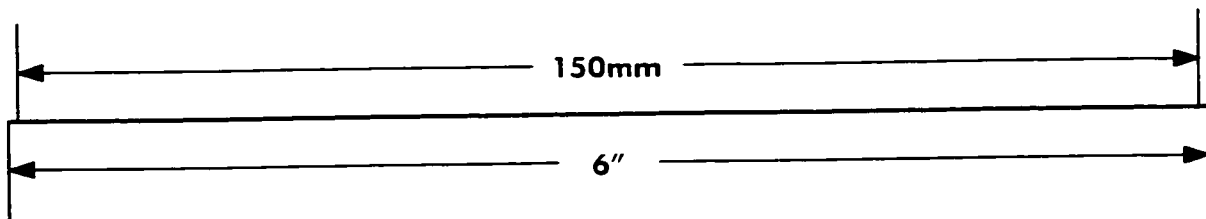
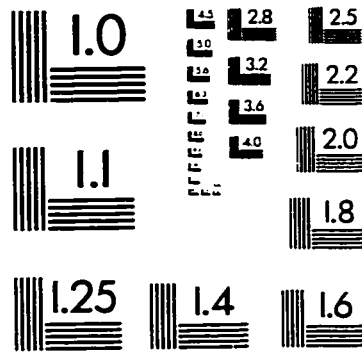
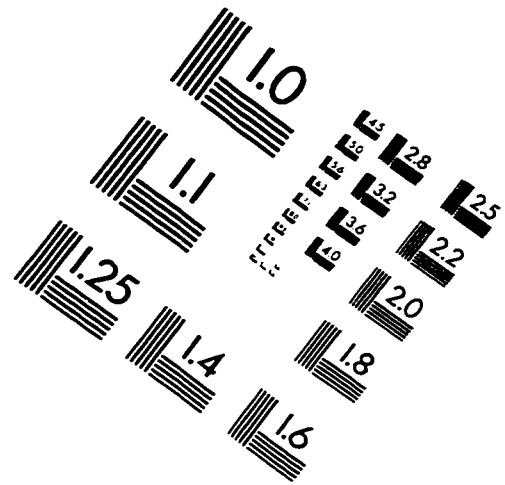
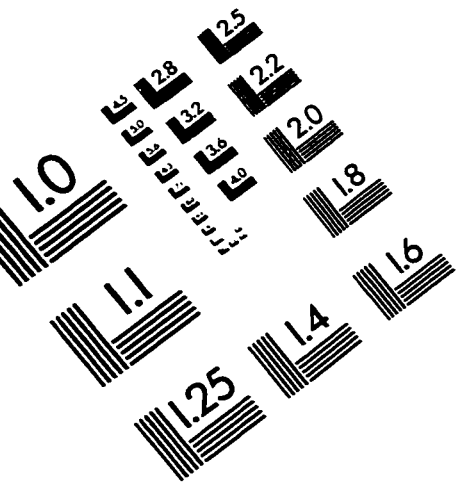
- [14] Enge, P.,K., "The Global Positioning System: Signals, Measurements, and Performance". International Journal of Wireless Information Networks. Vol. 1, No. 2, 1994.
- [15] Van Dierendonck, A. J., "GPS Receivers". Global Positioning Systems. Theory and Applications. Vol. 1, Parkinson, B. W., Spilker Jr. J. J., Axelrad, P., Enge, P., AIAA 1996
- [16] D. Lawrence, S. Cobb, B. Pervan, G. Opshaug, P. Enge, J.D. Powell, and B. Parkinson, "Performance Evaluation of On-Airport Local Augmentation System Architectures", Proceedings of ION GPS-95, The 8th International Technical Meeting of the Satellite Division of the Institute of Navigation, Palm Springs, Sept. 12-15, 1995.
- [17] B. Pervan, D. Lawrence, K. Gromov, G. Opshaug, J. Christie, P-y Ko, S. Pullen, P. Enge and B. Parkinson, "Flight Test Evaluation of a Prototype Local Area Augmentation System Architecture". Proceedings of ION GPS-97, The Tenth International Technical Meeting of the Satellite Division of the Institute of Navigation, Kansas City, Sept. 16-19, 1997.
- [18] D. Lawrence, "Aircraft Landing Using GPS: Development and Evaluation of a Real Time System for Kinematic Positioning Using the Global Positioning System". PhD dissertation, Stanford University, 1996.
- [19] C. Cohen, B. Pervan, H. S. Cobb, D. Lawrence, J. D. Powell, and B. Parkinson, "Precision Landing of Aircraft Using Integrity Beacons", Global Positioning System: Theory and Applications, editors Parkinson, Spilker, Axelrad and Enge, American Institute of Aeronautics and Astronautics, Washington D.C., 1996.
- [20] P. Enge and A. Ndili, "Interference Mitigation by Stand-Alone and Intrack APLS". Presented to the LAAS Architecture Review Committee, Stanford, Oct. 3, 4, 1996.
- [21] T. Walter and P. Enge, "Weighted RAIM for Precision Approach". Proceedings of ION GPS-95, The Eight International Technical Meeting of the Satellite Division of the Institute of Navigation, Palm Springs, CA, Sept. 12-15, 1995.
- [22] A. Ndili, "GPS Pseudolite Signal Design", Proceedings of ION GPS-94, The Seventh International Technical Meeting of the Satellite Division of the Institute of Navigation, Salt Lake City, Sept. 20-23, 1994.
- [23] F. Amoroso, "The Bandwidth of Digital Data Signals". IEEE Communications Society Magazine, Vol. 18, No. 6, November 1980, pp. 13-24.
- [24] A. Brown, "A GPS Precision Approach and Landing System". Presented at the 5th International Technical Meeting, Institute of Navigation GPS-92, September 1992.
- [25] C. E. Cohen, B. Pervan, H. S. Cobb, D. Lawrence, J. D. Powell, and B. W. Parkinson, "Real-time cycle ambiguity resolution using a pseudolite for precision landing of aircraft with GPS", Proceedings of the Second International Symposium on Differential Satellite Navigation Systems, Amsterdam, March 1993.

- [26] D. L. Chandler, "Embarrassing Lessons for Navigators". The Boston Globe, June 19, 1995.
- [27] R. Gold, "Maximal Recursive Sequences with 3-Valued Recursive Cross-Correlation Functions". IEEE Transactions on Information Theory, January 1968, pp. 154-156.
- [28] N. Levanon and A. Freedman, "Periodic Ambiguity Function of CW Signals with Perfect Periodic Autocorrelation", IEEE Transactions on Aerospace and Electronic Systems, Vol. 28, No. 2, April 1992, pp. 387-395.
- [29] F. J. MacWilliams, N. J. A. Sloane, "Pseudo-Random Sequences and Arrays". Proceedings of the IEEE, Vol. 64, No. 12, December 1976, pp. 1715-1727.
- [30] J. Nagle, A.J. Van Dierendonck, Quyen Dan Hua, "Inmarsat-3 Navigation Signal C/A Code Selection and Interference Analysis". Presented at the ION-TEK-92, San Diego CA., January 1992.
- [31] D. V. Sarwate and Michael B. Pursley, "Crosscorrelation Properties of Pseudorandom and Related Sequences", Proceedings of the IEEE, Vol. 68, No. 5, May 1980, pp. 593-619.
- [32] J. J. Spilker, Jr., "GPS Signal Structure and Performance Characteristics", NAVIGATION, Journal of the Institute of Navigation, Vol. I, June 1989.
- [33] J. J. Spilker, Jr., *Digital Communications by Satellite*, Prentice-Hall, Engelwood Cliffs, N.J., 1977.
- [34] T. A. Stansell, Jr., "RTCM SC 104 Recommended Pseudolite Signal Specifications", Global Positioning System, Vol. III, Institute of Navigation, 1986.
- [35] A. J. Van Dierendonck, "The Role of Pseudolites in the Implementation of Differential GPS". Presented at Position Location and Navigation Symposium, PLANS '90, March 1990.
- [36] A. J. Van Dierendonck, "Concepts for Replacing Shipboard TACAN with Differential GPS", Presented at the 3rd International Technical Meeting, Institute of Navigation GPS-90, September 1990.
- [37] A. J. Van Dierendonck, B. D. Elrod, "Testing and Evaluation of GPS Augmented with Pseudolites for Precision Approach Applications", Presented at DSNS Conference, Amsterdam, 1993.
- [38] RTCA, "Local Area Augmentation System Airport Pseudolite Signal Specification", Prepared by SC-159, WG 4a, APL subgroup, March 16, 1997.

- [39] Carlson, A. B., "Communication Systems, An Introduction to Signals and Noise in Electrical Communication", 3rd Ed. McGraw-Hill, Inc. San Francisco, 1986.
- [40] Gray, R. M., Goodman, J. W., "Fourier Transforms: An Introduction for Engineers", Kluwer Academic Publishers, Boston, 1995
- [41] Upadhyay, T., Dimos G., Ferzali W., Weed, D., "Test Results on Mitigation of SATCOM-Induced Interference to GPS Operation", Proceedings of ION GPS-95, The 8th International Technical Meeting of the Satellite Division of the Institute of Navigation, Palm Springs, CA, Sept. 12-15, 1995.
- [42] Nisner, P., and Owen, J., "Practical Measurements of Radio Frequency Interference to GPS Receivers and an Assessment of Interference Levels by Flight Trials in the European Regions", Proceedings of ION GPS-95, The 8th International Technical Meeting of the Satellite Division of the Institute of Navigation, Palm Springs, CA, Sept. 12-15, 1995.
- [43] P. Enge, E. Swanson, R. Mullin, K. Ganther, A. Bommarito, R. Kelly, "Terrestrial Radionavigation Technologies", NAVIGATION: Journal of the Institute of Navigation, Vol. 42, No. 1, Spring 1995.
- [44] J. Raquest and G. Lachapelle, "Determination and Reduction of GPS Reference Station Multipath Using Multiple Receivers", Proceedings of ION GPS-96, The Ninth International Technical Meeting of the Satellite Division of the Institute of Navigation, Kansas City, MO, Sept. 17-20, 1996.
- [45] Mattos, P., "Multipath Elimination for the Low-Cost Consumer GPS", Proceedings of ION GPS-96, The Ninth International Technical Meeting of the Satellite Division of the Institute of Navigation, Kansas City, MO, Sept. 17-20, 1996.
- [46] D. Doris, A. Benhallam, "On Correlation Processes Reducing Multipath Errors in the L1 GPS Receiver", Proceedings of ION GPS-96, The Ninth International Technical Meeting of the Satellite Division of the Institute of Navigation, Kansas City, MO, Sept. 17-20, 1996.
- [47] J. Auton, J. Cruz, "Simulating GPS Receiver Measurement Errors", Proceedings of ION GPS-96, The Ninth International Technical Meeting of the Satellite Division of the Institute of Navigation, Kansas City, MO, Sept. 17-20, 1996.
- [48] B. Preil, I.A.I. Tamam, B. Bobrovsky, "Jammer Design for GPS Signals", Proceedings of ION GPS-96, The Ninth International Technical Meeting of the Satellite Division of the Institute of Navigation, Kansas City, MO, Sept. 17-20, 1996.
- [49] M. Schwartz and L. Shaw, *Signal Processing, Discrete Spectral Analysis, Detection and Estimation*, McGraw-Hill, New York 1975.
- [50] J. Diesel and G. Dunn, "GPS/IRS AIME: Certification for Sole Means and Solution to RF Interference", Proceedings of ION GPS-96, The Ninth International Technical Meeting of the Satellite Division of the Institute of Navigation, Kansas City, MO, Sept. 17-20, 1996.

- [51] B. Schnauffer, G. McGraw, "WAAS Receiver Carrier Tracking Loop and Data Demodulation Performance in the Presence of Wideband Interference". Proceedings of ION GPS-96, The Ninth International Technical Meeting of the Satellite Division of the Institute of Navigation, Kansas City, MO, Sept. 17-20, 1996.
- [52] R. Walker, and K. Kubik, "Numerical Modelling of GPS Signal Propagation". Proceedings of ION GPS-96, The Ninth International Technical Meeting of the Satellite Division of the Institute of Navigation, Kansas City, MO, Sept. 17-20, 1996.
- [53] A. Ndili, and P. Enge, "GPS Receiver Autonomous Interference Detection", Proceeding of the IEEE 1998 Position, Location and Navigation Symposium, Palm Springs, CA, April 20-23, 1998.
- [54] P. Ward, "GPS Receiver Search Techniques", Proceeding of the IEEE 1996 Position, Location and Navigation Symposium, 1996.
- [55] A. Ndili, "Receiver Autonomous Interference Detection", Proceedings of the 53rd Annual Meeting of the Institute of Navigation, Albuquerque, New Mexico. June 1997.
- [56] G. McGraw and B. Schnauffer, "A Modified Frequency-Locked Loop for Improved WAAS Carrier Tracking", Proceedings of ION GPS-95, The Eight International Technical Meeting of the Satellite Division of the Institute of Navigation, Palm Springs, CA, Sept. 12-15, 1995.
- [57] R. Erlandson, "Global Navigation Satellite System Interference Analysis: Category II Precision Approach with Land-Mobile Satcom Interference", RTCA SC-159 Ad Hoc Interference Working Group, February 6-7, 1995.
- [58] R. Cole, "Simulation Testing of Precision Approach Integrity Algorithms". Proceedings of ION GPS-96, The Ninth International Technical Meeting of the Satellite Division of the Institute of Navigation, Kansas City, MO, Sept. 17-20, 1996.
- [59] L. Garin, F. Diggelen, J. Rousseau, "Strobe and Edge Correlator Multipath Mitigation for Code", Proceedings of ION GPS-96, The Ninth International Technical Meeting of the Satellite Division of the Institute of Navigation, Kansas City, MO, Sept. 17-20, 1996.
- [60] J.J. Spilker Jr., "Fundamentals of Signal Tracking Theory", Global Positioning Systems, Theory and Applications, Vol. I, Parkinson, B. W., Spilker Jr. J. J., Axelrad, P., Enge, P., AIAA 1996
- [61] D. Knuth, *The Art of Computer Programming*, 3rd Edition, Addison-Wesley, Reading, Massachusetts, 1997.

IMAGE EVALUATION TEST TARGET (QA-3)



APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989

© 1993, Applied Image, Inc., All Rights Reserved

