NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

ANALYSIS OF PERIODIC BINARY SEQUENCES

by

Jarratt M. Mowery

June 1986

Thesis Advisor: Professor Harold Fredricksen

Approved for public release; distribution is unlimited.

CURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.
. DECLASSIFICATION / DOWNGRADING SCHEDULE	
PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)

. NAME OF PERFORMING ORGANIZATION Naval Postgraduate School	6b OFFICE SYMBOL *(If applicable)* 62	7a. NAME OF MONITORING ORGANIZATION Naval Postgraduate School
. ADDRESS *(City, State, and ZIP Code)* Monterey, California 93943-5000		7b. ADDRESS *(City, State, and ZIP Code)* Monterey, California 93943-5000
. NAME OF FUNDING / SPONSORING ORGANIZATION	8b. OFFICE SYMBOL *(If applicable)*	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

. ADDRESS *(City, State, and ZIP Code)*

10 SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO	TASK NO	WORK UNIT ACCESSION NO.

TITLE *(Include Security Classification)*

ANALYSIS OF PERIODIC BINARY SEQUENCES

PERSONAL AUTHOR(S)

Mowery, Jarratt, M.

a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT *(Year, Month, Day)*	15 PAGE COUNT
Master's Thesis	FROM ________ TO ________	1986, June 20	87

SUPPLEMENTARY NOTATION

COSATI CODES			18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
FIELD	GROUP	SUB-GROUP	Periodic Binary Sequences; Pseudo-Random Sequences; Harmonic Analysis; Fisher's Test; Estimating Period; Discrete Fourier Transform

ABSTRACT *(Continue on reverse if necessary and identify by block number)*

This thesis is concerned with the harmonic analysis of periodic binary sequences using the Discrete Fourier Transform. The effects of various types of noise on the spectral content of a sequence are investigated. Assuming independent identical Normal distributions for errors, a method for the application of Fisher's test is proposed to provide a quantitative measure of the significance of spectral components. This proposed method is implemented by computer program and applied to the problem of estimating the period of noise garbled pseudo-random binary sequences.

0 DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED	
2a NAME OF RESPONSIBLE INDIVIDUAL Professor Harold Fredricksen	22b TELEPHONE *(Include Area Code)* (408) 646-2235	22c. OFFICE SYMBOL 53

D FORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete — SECURITY CLASSIFICATION OF THIS PAGE

Analysis of Periodic Binary Sequences

by

Jarratt M. Mowery
Lieutenant, United States Navy
B.S., Western Washington University, 1979

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
June 1986

ABSTRACT

This thesis is concerned with the harmonic analysis of periodic binary sequences using the Discrete Fourier Transform. The effects of various types of noise on the spectral content of a sequence are investigated. Assuming independent identical Normal distributions for errors, a method for the application of Fisher's test is proposed to provide a quantitative measure of the significance of spectral components. This proposed method is implemented by computer program and applied to the problem of estimating the period of noise garbled pseudo-random binary sequences.

TABLE OF CONTENTS

I. INTRODUCTION 8

II. DATA SEQUENCES 11

A. M-SEQUENCES 11

B. GOLD CODES 14

C. FULL SEQUENCES 16

III. NOISE EFFECTS 19

A. ADDITIVE BIT ERRORS 19

B. BURST ERRORS 20

C. TIMING ERRORS 23

D. CONCLUSIONS ON NOISE EFFECTS 28

IV. FISHER'S METHOD AND ITS APPLICATIONS 29

A. FISHER'S METHOD 30

B. APPLICATION BY NOWROOZI 33

1. First Application 33

2. Second Application 33

C. APPLICATION BY SHIMSHONI 34

V. A PROPOSED METHOD FOR THE APPLICATION OF FISHER'S TEST 36

A. DEVELOPMENT OF A PROPOSED METHOD 36

B. CALCULATION OF THE MEASURE OF SIGNIFICANCE . . 38

C. PROGRAM FOR THE APPLICATION OF FISHER'S TEST 40

VI. ESTIMATING THE PERIOD OF A PSEUDO-RANDOM SEQUENCE 42

A. ESTIMATION PROCEDURE 42

B. CASE ONE: FULL-SEQUENCES 43

1. Example One 43

2. Example Two 46

C. CASE TWO: M-SEQUENCES 48

1. Example Three 49

D. CASE THREE: GOLD CODES 51

1. Example Four 52

VII. CONCLUSION 55

A. METHOD FOR DETERMINING SPECTRAL SIGNIFICANCE 55

B. SUGGESTIONS FOR FURTHER RESEARCH 56

APPENDIX A: PROGRAM TO GENERATE M-SEQUENCES 57

APPENDIX B: PROGRAM TO GENERATE GOLD CODES 59

APPENDIX C: PROGRAM TO GENERATE A FULL SEQUENCE 62

APPENDIX D: PROGRAM FOR RANDOM BIT ERRORS 64

APPENDIX E: PROGRAM FOR BURST ERRORS 68

APPENDIX F: PROGRAM FOR RANDOM DELETIONS 72

APPENDIX G: PROGRAM FOR RANDOM INSERTIONS 75

APPENDIX H: PROGRAM FOR TIMING ERRORS 78

APPENDIX I: PROGRAM FOR FISHER'S TEST 81

LIST OF REFERENCES 85

INITIAL DISTRIBUTION LIST 86

LIST OF TABLES

1. PERCENT ERROR OF HARMONIC SIGNIFICANCE COMPUTATION 40
2. SIGNIFICANT COMPONENTS IN EXAMPLE ONE 46
3. SIGNIFICANT COMPONENTS IN EXAMPLE TWO 48
4. SIGNIFICANT COMPONENTS IN EXAMPLE THREE 51
5. SIGNIFICANT COMPONENTS IN EXAMPLE FOUR 54

LIST OF FIGURES

1.1 Typical Noise Corrupted Spectrum 9
2.1 Feedback Shift Register 13
2.2 Spectrum of an m-Sequence 14
2.3 Gold Code Generator 15
2.4 Spectrum of a Gold Code 17
2.5 Spectrum of a Full Sequence 18
3.1 Additive Bit Errors: 16.6% error rate 21
3.2 Additive Bit Errors: 25% error rate 21
3.3 Additive Bit Errors: 33% error rate 22
3.4 Burst Errors 23
3.5 Random Bit Deletions 25
3.6 Random Bit Insertions 26
3.7 Timing Errors 27
6.1 Spectrum for Example One 45
6.2 Spectrum for Example Two 47
6.3 Spectrum for Example Three 50
6.4 Spectrum for Example Four 53

I. INTRODUCTION

The effect of noise on the spectrum of a signal is obvious to the observer. It appears to include many random components of various sizes. In other words, it looks "noisy". Figure 1.1 shows the spectrum of a noise corrupted binary sequence. The ordinates of this spectrum are the squared magnitudes of the DFT components of the sequence. Throughout this thesis, spectra will be presented in this manner. They are commonly refered to as periodograms. When analyzing the spectrum of a signal, the question naturally arises of the significance of spectral components. The researcher must distinguish the spectral components which represent the signal being studied from those which are merely random perturbations caused by noise. This research is primarily concerned with finding a method to quantitatively evaluate the statistical significance of the components of a spectrum.

This research will restrict attention to the analysis of a small, but important class of periodic binary sequences, namely, pseudo-random sequences and some close relatives. These sequences, their properties and their applications will be described briefly. Some methods used to generate these sequences will also be explained.

The possible effects of noise on the spectral content of such sequences will be briefly investigated. It is assumed that the sequence is subjected to the types of noise which might be expected to occur in a communications system making use of such a sequence. No detailed knowledge of the system is assumed. The only information available is a garbled version of the original sequence. Note that this is a highly simplified situation in that no provision is made for data modulation. Future research will do well to consider

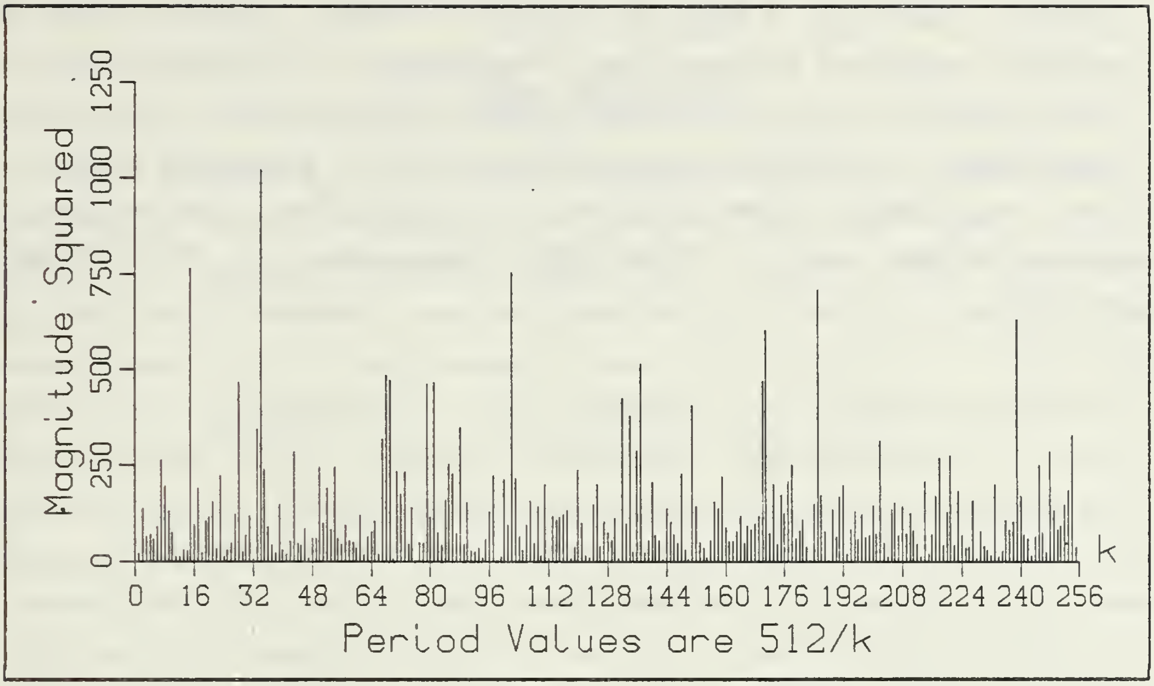

Figure 1.1 Typical Noise Corrupted Spectrum.

various methods of data modulation as well as possible noise sources. In order to facilitate proceeding with the quantification of spectral component significance, the further simplifying assumption is made that all noise effects can be modeled as independent identically distributed Normal random variables.

In conducting harmonic analysis to reveal the periodic structure of a time series in the presence of noise, random fluctuations alone can account for some harmonic components being greater than others. The researcher must therefore use some means to determine the plausibility that a particular component represents a real periodicity. In 1929, R.A. Fisher developed a test for the significance of harmonic components [Ref. 1]. Over the past 20 years several researchers have applied Fisher's test in a variety of different ways [Refs. 2,3]. Fisher's test will be briefly presented along with some of these more recent applications. This thesis proposes the application of Fisher's test in a new way which is more flexible than the methods employed by previous researchers. This will provide the researcher with more meaningful quantitative information upon which to base his conclusions. This proposed method is then applied to the problem of estimating the period of pseudo-random and related binary sequences in the presence of noise.

II. DATA SEQUENCES

This research is first concerned with the analysis of periodic binary sequences which may be referred to in general as pseudo-random sequences. These sequences are called pseudo-random because they exhibit certain properties associated with randomness, namely, balance, run and correlation properties. The property of balance means that each period of a sequence contains approximately as many zeros as ones. The run property refers to the occurrence of strings of consecutive ones and zeros. One half of the runs present are of length one, one forth are of length two, one eight are of length three and so on. Correlation refers to the property that if a sequence is compared with a cyclic shift of itself, there will be an approximately equal number of agreements and disagreements, except for the case when the cyclic shift is a multiple of the sequence period. In that case all bits will agree. As such their spectra appear similar to that of noise or similar to a truly random sequence. These sequences and near relatives were chosen for study because of their importance in a wide variety of applications. Examples of applications of these sequences include spread spectrum systems and secure communications among many others. Specifically, the sequences considered are (maximal) m-sequences, Gold codes and de Bruijn or full sequences. [Ref. 4: pp. 24-27]

A. M-SEQUENCES

Any periodic sequence may be generated by a linear feedback shift register [Ref. 5: p. 411]. For a shift register of n stages, an m-sequence of length 2**n -1 may be generated. m-sequences are termed maximal because they are the longest sequence which may be generated with a specified number of feedback shift register stages using a linear feedback rule. To further describe the structure of m-sequences, consider the following. If a window of width n equal to the number of stages in the shift register is slid along the m-sequence, in one period of the sequence all possible non-zero binary n-tuples would be observed. The various n-tuples can be viewed as elements of a finite field. The successive n-tuples represent powers of a primitive element in the multiplicative group of the field. Thus the m-sequences represent the structure necessary to determine multiplication within the field. Addition is accomplished by modulo-two component-wise addition of the binary n-tuples. m-sequences play an important part in spread spectrum systems as well as in other applications.

The extensive mathematical underpinnings of the design process of an m-sequence will not be discussed here. Suffice it to say that a feedback shift register may be designed to generate the m-sequence using a primitive polynomial described by Galois field theory. At each clock signal the storage registers pass their bits along to their successor locations. The bit computed from the current contents of the registers using the polynomial feedback rule is returned to the leftmost register. An example of a feedback shift register designed to generate the period 7 m-sequence ...00101110010111OO... using the polynomial $f(x)=x^3+x+1$ is shown in Figure 2.1. In this figure, the boxes represent storage devices (flip-flops) and the circle represents a modulo-two adder. Upon receipt of each clock pulse, the storage devices simultaneously shift their present contents to the right. The output of the adder is fed back to the left most storage device.

The m-sequences used in this research are generated by an interactive FORTRAN program. Appendix A contains a copy of this program. This program prompts the user to input the

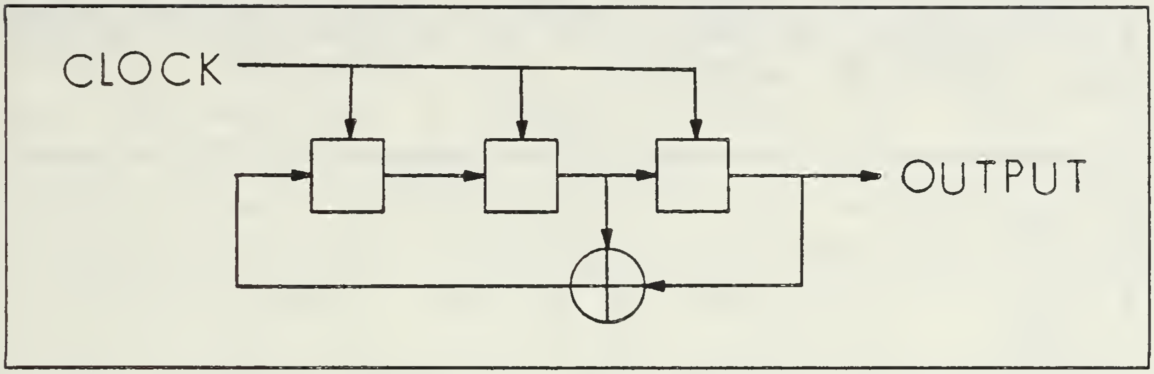

Figure 2.1 Feedback Shift Register.

coefficients of a primitive polynomial which defines the feedback paths of the feedback shift register. The program then generates as many bits of the m-sequence as the user desires. Tables of polynomials can be used to input polynomial coefficients for any desired sequence period (less than 2**15 -1) [Ref. 4: pp. 62-65]. Longer sequences can be constructed, but are not germane to this study.

Figure 2.2 shows the spectrum of a period 15 m-sequence derived using a 512 point DFT. Signal components are smeared over several adjacent frequencies because the period of the sequence does not evenly divide the number of points in the DFT. It should also be noted that the spectrum appears quite well structured. This simple, orderly spectrum can be anticipated because of the simplicity of the theoretical power spectral density. The power spectral density is easily obtained by taking the Fourier transform of the periodic autocorrelation function (applying the Wiener-Khinchin theorem). The periodic autocorrelation function is composed of a constant value of -1/period with a triangle function of height one occuring once each period. Transformed, the resulting power spectral density is composed of discrete elements occuring at intervals of 1/period within a sinc squared envelope. Although the spectrum obtained by the DFT is not a good estimate of the theoretical power spectral density, its simple structure is a reflection of the simplicity of the theoretical spectrum. [Ref. 5: pp. 387-388]

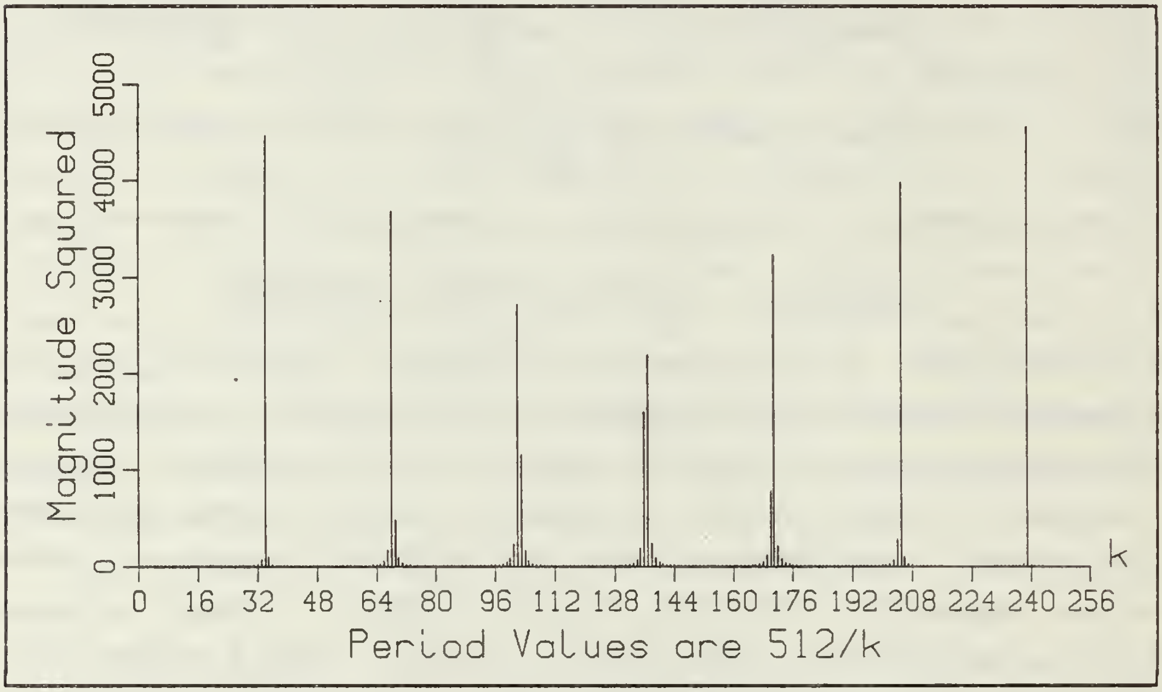

Figure 2.2 Spectrum of an m-Sequence.

B. GOLD CODES

Gold codes are designed chiefly for application in spread spectrum, multiple access systems. They are useful in this application because of their well controlled cross correlation properties, (i.e. they have a three valued cross-correlation function). A Gold code is simply the modulo two sum of an appropriately chosen pair of m-sequences of the same period. These sequences are termed a preferred pair. For a preferred pair of m-sequences an entire family of Gold codes can be generated by shifting the relative phases of the m-sequences. A complete Gold code family consists of 2**n +1 codes, where n is the number of stages in the feedback shift registers. An example of a Gold code generator is shown in Figure 2.3. Here the preferred pair of sequences are generated by the polynomials $f(x)=x^5+x^2+1$ and $f(x)=x^5+x^4+x^3+x^2+1$, both of which produce m-sequences of period 31 and hence the resulting Gold code is also of period 31. [Ref. 5: pp. 404-407]

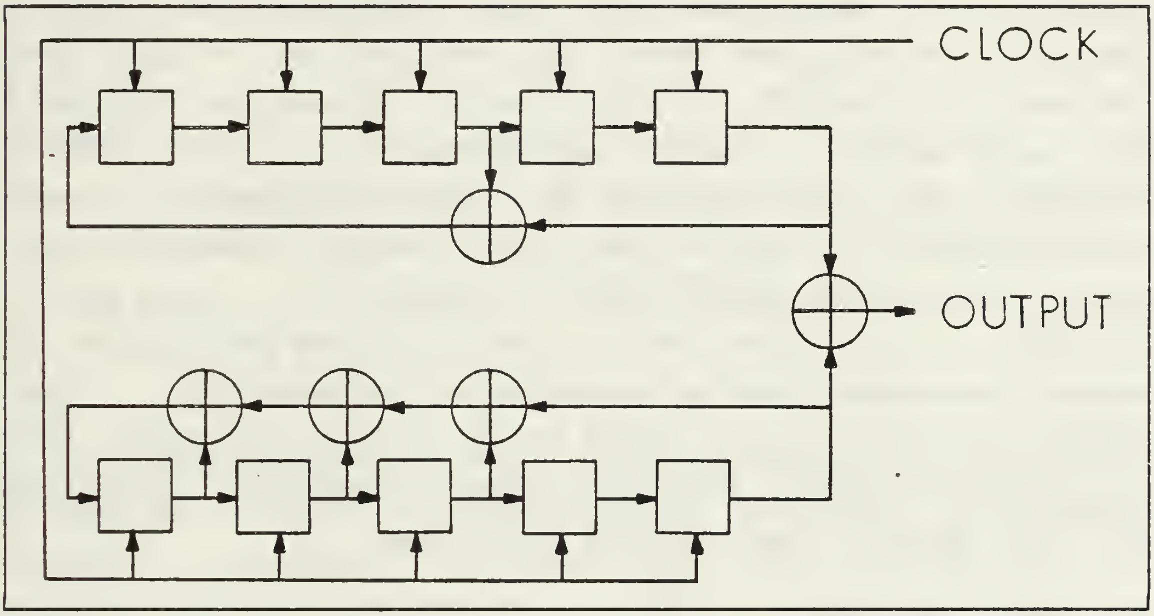

Figure 2.3 Gold Code Generator.

An interactive FORTRAN program was written to generate Gold codes. A copy of this program may be found in Appendix B. This program is composed of two m-sequence generators and a modulo-two adder. Depending upon the initial load specified for the feedback shift registers of these two generators, m-sequences of different phases are produced. When the m-sequences are added together, a Gold code is produced. An example of a Gold code spectrum is shown in Figure 2.4. This spectrum was derived using a 512 point DFT. This Gold code is a period 31 sequence generated by the generator shown in Figure 2.3. It is important to note that the spectrum of a Gold code is not nearly as well structured as that of an m-sequence. This fact may again be anticipated by considering the more complex nature of the theoretical periodic autocorrelation function of a Gold code. It is a many valued function which depends on the particular Gold code for the determination of those values. Taking the Fourier transform results in a spectrum which is difficult to describe mathematically. The spectrum also depends on the particular Gold code being examined. It is therefore no surprise that the spectrum of a Gold code obtained by the DFT is not nearly as simple and well structured as that found for an m-sequence. In the example pictured, the Gold code has 12 ones and 19 zeros in each period instead of the 16 ones and 15 zeros present in each period of the m-sequences used to generate it. It is poorly balanced. Gold codes do not exhibit the properties of pseudo-randomness nearly as well as m-sequences. Even though Gold codes may be found which are well balanced, they still do not exhibit the other pseudo-randomness properties and their spectra are not well structured.

C. FULL SEQUENCES

Full sequences are a third example of shift register sequences considered. These sequences are of the non-linear feedback type. Because these sequences are more complex to analyze and in general more complex to generate, they find application in secure communications. In structure, full sequences are similar to m-sequences. They are sequences of length 2**n with every binary n-tuple appearing in one period of the sequence. A full sequence can be constructed from any m-sequence by simply adding one additional zero to the longest string of zeros present. This is by no means the only method available to generate full sequences. Numerous algorithms for generating full sequences are available in the literature. [Ref. 4: pp. 128-141]

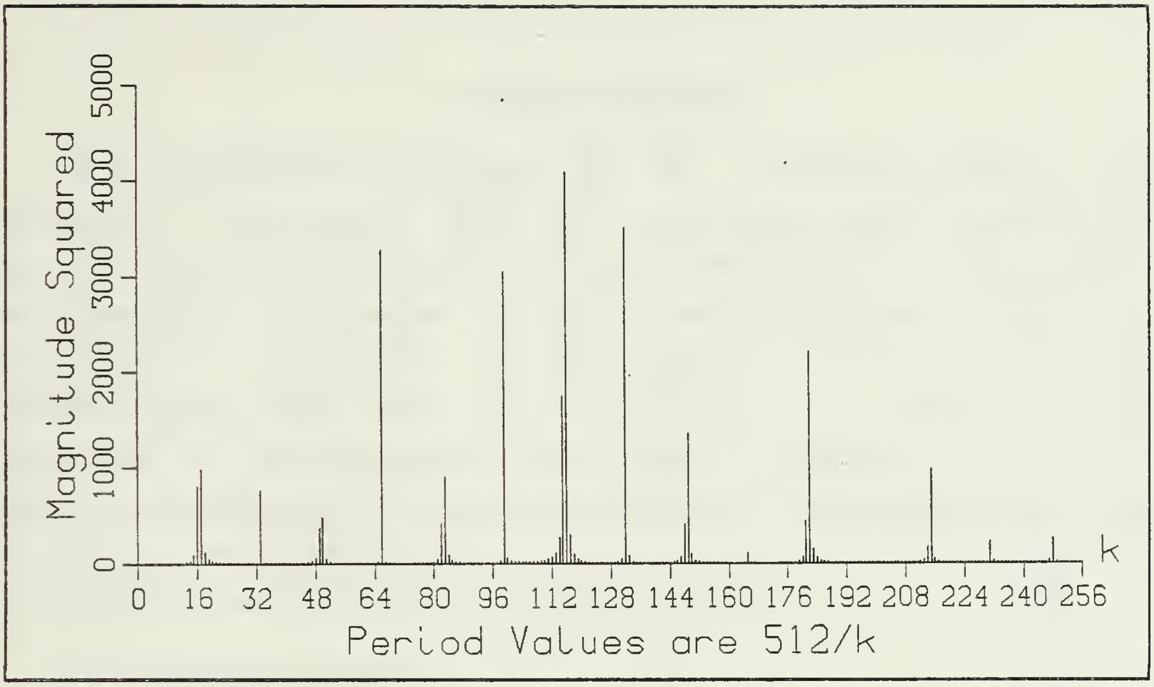

Figure 2.4 Spectrum of a Gold Code.

FORTRAN Programs were written to generate full sequences by adding a zero to a previously generated m-sequence. Appendix C contains an example of such a program. An example of the spectrum of a full sequence is shown in Figure 2.5, derived using a 512 point DFT. In this case the full sequence was found by modifying the m-sequence generated using the polynomial $f(x)=x^4+x+1$. The resulting full sequence is of period 16. Note that since the period of the full sequence evenly divides the length of the DFT, no spectral smearing is observed.

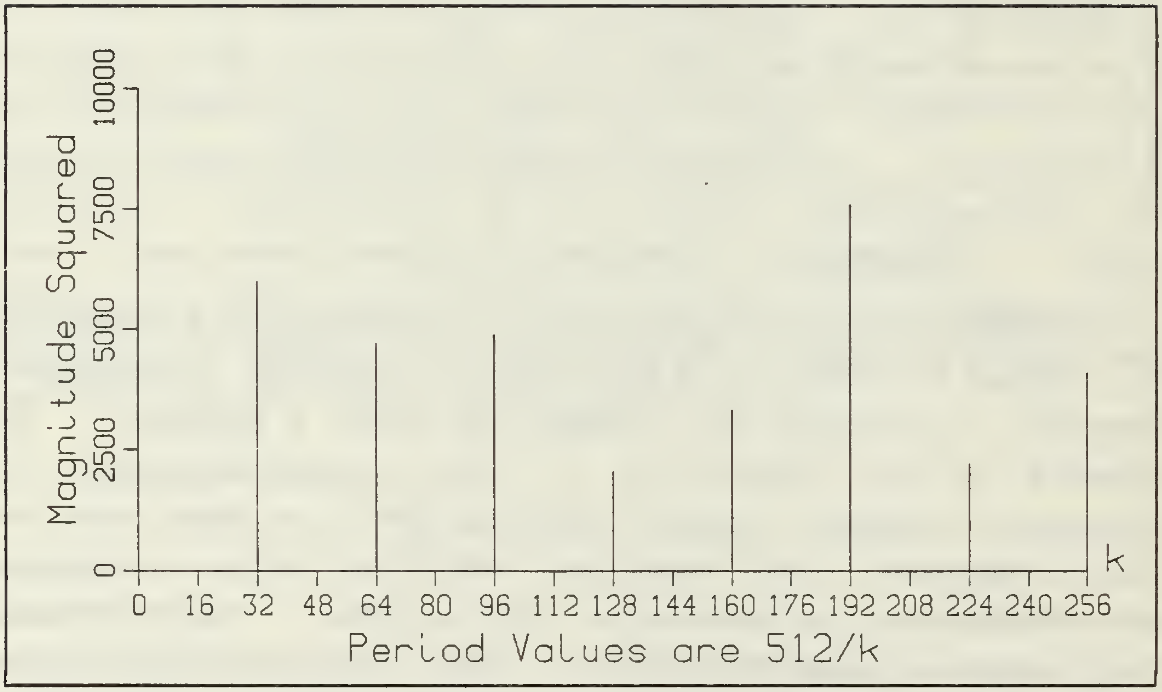

Figure 2.5 Spectrum of a Full Sequence.

III. NOISE EFFECTS

The sequences described in the previous chapter are assumed to have been used in a communications system and garbled versions of them have been recovered. Once again, no specific knowledge of the communications system or channel is assumed or developed in this thesis. Observations are made of the effects of noise on the spectrum of the sequences discussed in Chapter II. Many error scenarios by which the original sequence could be altered are investigated. Some of the more important situations are described in what follows.

A. ADDITIVE BIT ERRORS

One possible manner in which a sequence could be altered is that individual bits are subjected to noise and therefore are incorrectly recovered. Receiver thermal noise or channel noise could conceivably cause such individual bit errors to occur in some random fashion.

A FORTRAN program was written to simulate this error pattern and to study its effect on the spectrum of a sequence. Appendix D contains a copy of this program. The rate at which errors are introduced is set interactively by the user. The IMSL subroutine GGUBFS is used to generate a random number for each bit of the input data sequence. Each random number is tested to see if it falls within a range defined by the user specified error rate. If it does appear in that range, the corresponding bit is reversed. In this manner, each data bit is subjected to the same probability of error. The original sequence is modified to reflect these random bit errors and its DFT is computed using the IMSL subroutine FFT2C and is plotted.

Thorough testing was carried out for the three classes of sequences of chapter II at various error rates. Figures 3.1, 3.2 and 3.3 show examples of these tests for the simple case of a period 16 full sequence formed from the m-sequence generated by $f(x)=x^4+x+1$. The probability of error used was 16.6%, 25% and 33% respectively. The introduction of random bit errors is reflected in the noise floor which may be observed at all frequencies. As more errors are introduced, the noise floor increases while the signal components decrease until the signal is eventually lost in the noise. As the error rate changes, the location of the signal components remains constant while their magnitudes, relative to the noise components, change. Note that magnitudes on the ordinate axis decrease as the error rate increases in each successive example. Note the large component at k=148 in Figure 3.3. This component is due to the noise alone. In this way errors can be made in spectral analysis due to the effects of additive bit errors.

B. BURST ERRORS

Another means by which a sequence can be altered by noise is that bursts of errors might occur. Channel noise could account for such an event. This type of noise may also more closely model the case of a structured noise source such as an intelligent jammer.

A FORTRAN program was written to simulate the occurance of bursts of errors. This program may be found in Appendix E. To run this program the user must input the probability of a burst occuring, the length of each burst and the probability of bit errors occuring within a burst. As the program moves sequentially through the data sequence, a random number is generated for each data bit using the IMSL subroutine GGUBFS. If that random number falls within a range defined by the specified probability of a burst occuring, the program jumps into a burst error loop. The

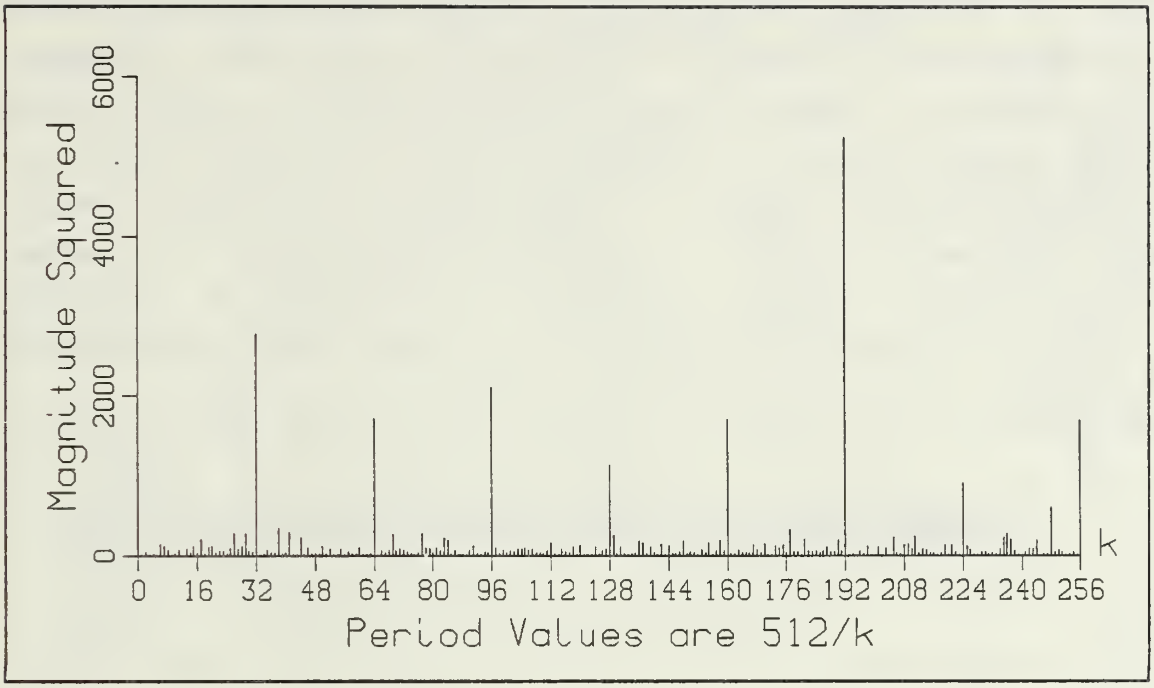

Figure 3.1 Additive Bit Errors: 16.6% error rate.

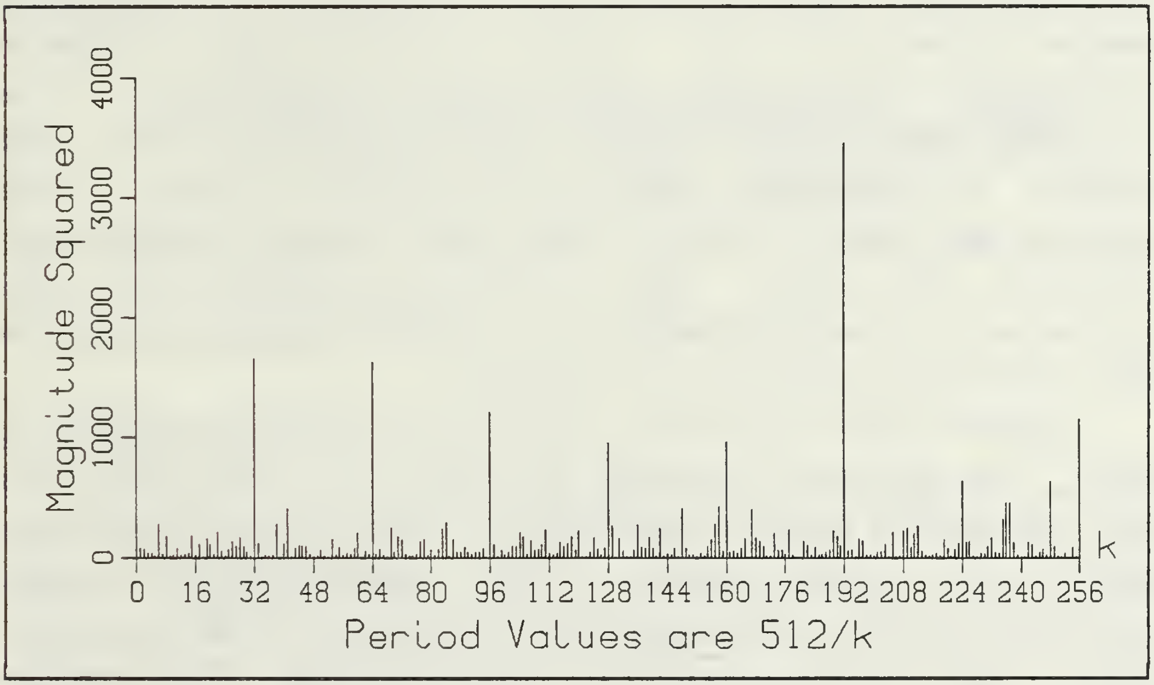

Figure 3.2 Additive Bit Errors: 25% error rate.

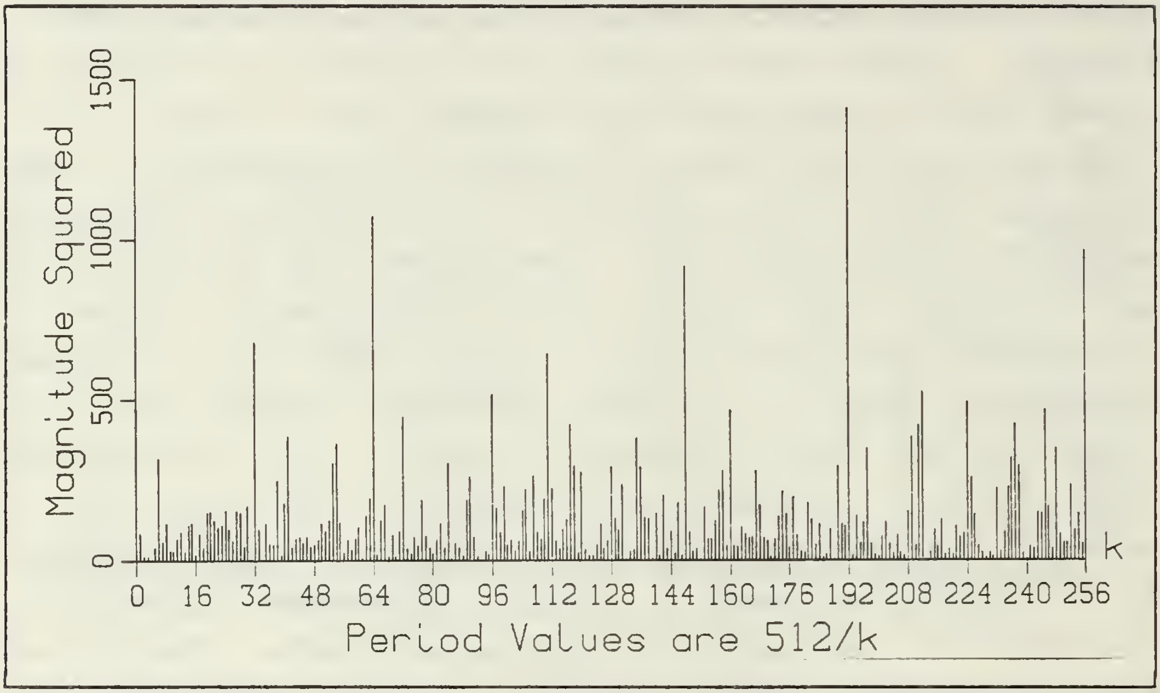

Figure 3.3 Additive Bit Errors: 33% error rate.

first and last bits of the burst are changed to define the bounds of the error burst. All data bits within the burst are subject to error with the probability specified. This is accomplished using a random number generator in the same fashion as described above for a burst occuring. The program then jumps back into the data sequence immediately following the burst. When the entire sequence has been subjected to bursts of errors its DFT is computed using the IMSL subroutine FFT2C and is plotted.

Numerous tests were also carried out for the burst error case. One such test is shown in Figure 3.4. For this test the same period 16 full sequence is used as was used in the additive bit error examples. The probability of a burst occuring is 10%, the length of the burst is 17 bits and the probability of error within each burst is 33%. This results in 131 errors occuring, or an overall error rate of 25%. This is comparable to the situation in Figure 3.3 for the additive bit error case. As can be observed in this example, the effect of bursts of errors on the spectrum is quite similar to that of additive bit errors. A noise floor is again observed at all frequencies, with the magnitude of signal components diminished. Signal components still appear in their proper locations. In all respects, the effects of bursts of errors on the spectrum is indistinguishable from that of additive bit errors.

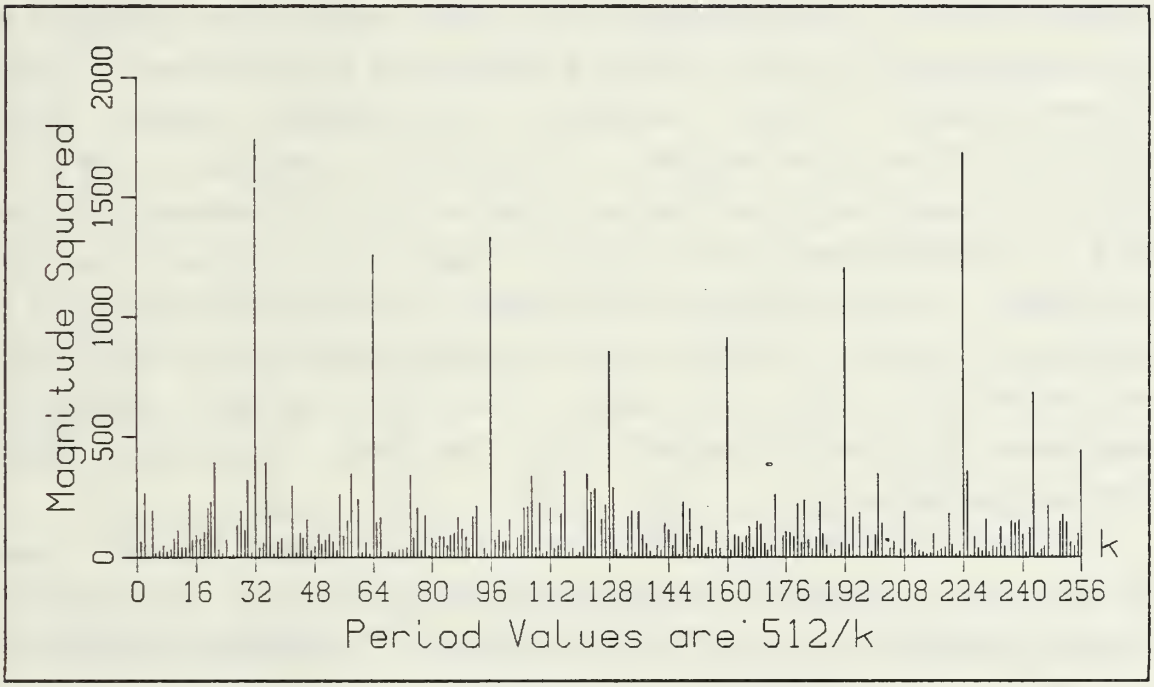

Figure 3.4 Burst Errors.

C. TIMING ERRORS

Another class of error through which a sequence could be garbled is that a bit of the sequence might occasionally be lost or inserted. For example, poor synchronization in the receiver could result in sampling outside the proper bit interval and thereby result in the loss of a bit. Poor timing could also lead to sampling twice during the same bit interval and thereby result in the insertion of a bit.

Timing errors resulting in the loss of bits in a random fashion are modeled by a FORTRAN program. This program may be found in Appendix F. To simulate this situation the program assumes that sampling normally takes place at the exact middle of a bit interval. At the prompting of the program the user sets a bound on how far the sampling instant could possibly slip forward in one bit interval. The program then moves iteratively through the sequence and generates a random number during each bit interval. This random number is generated by the IMSL subroutine GGUBFS and is constrained to lie within a bound set by the user. This random number is then added to the sampling time, which starts off at the middle of the bit interval. If the modified sampling time falls outside the bit interval, that bit is considered lost and is therefore removed from the sequence. In this manner the timing interval is continually sliding forward a random amount during each bit interval and occasionally a bit is lost. The DFT of the resulting sequence is then computed using the IMSL subroutine FFT2C and is plotted.

Tests carried out with this error model reveal some significant differences from the previous models considered. Random removal of bits from a periodic sequence causes the period of the sequence to fluctuate. Inspection of the signal spectrum reveals that the location of signal harmonic components is changed. For the removal of even a few bits, the fundamental frequency can be shifted out of its proper location. This shifting becomes progressively worse at higher harmonics of the fundamental.

Spectral smearing is another effect observed. The signal components are spread out over several adjacent frequencies. This effect also becomes progressively worse in higher harmonics.

Figure 3.5 shows an example of these phenomena for the same period 16 full sequence used in previous examples. The bound on how far the sampling time can slip forward in one sampling interval is set at 0.3. This results in the deletion of 119 bits of the sequence. Note that the higher harmonics of the fundamental frequency are lost in the noise.

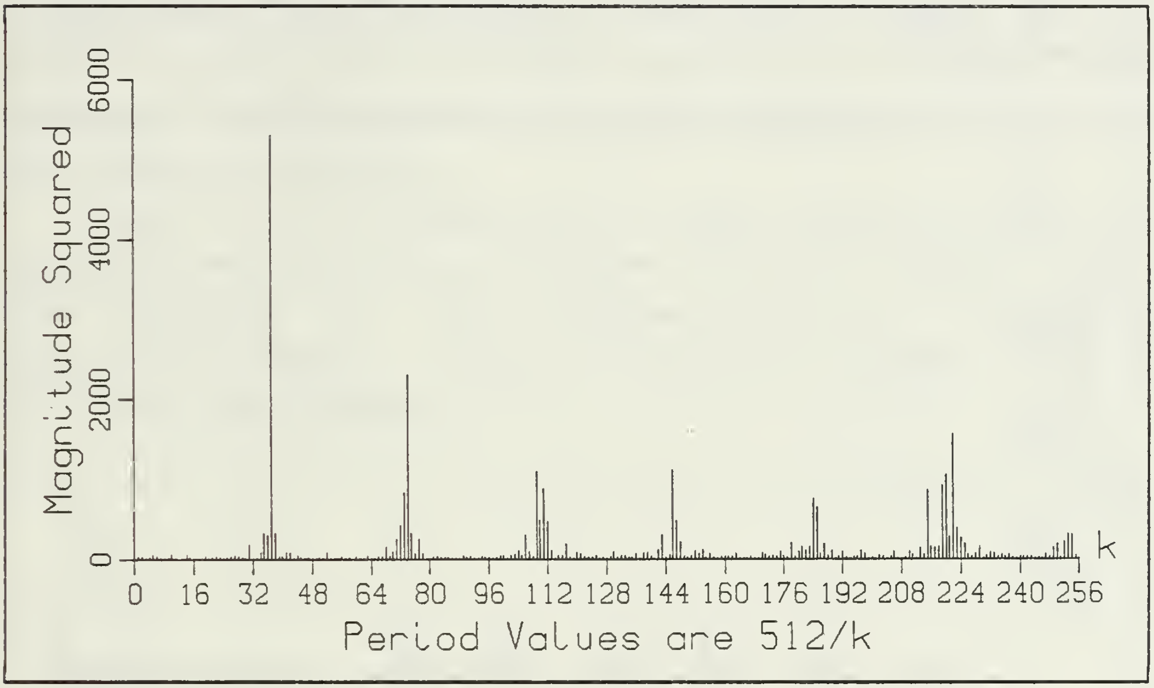

Figure 3.5 Random Bit Deletions.

In a similar fashion, a FORTRAN program was also written to model the random insertion of bits into the sequence. This program may be found in Appendix G. In this case, the sampling instant is allowed to slip back a small random amount during each bit interval. If the sampling time falls within the previous bit interval, that bit is inserted into the sequence again. The timing interval is therefore continually sliding back in a random fashion and a bit is occasionally repeated. A DFT is computed of the resulting sequence and the spectrum is plotted.

In the case of random bit insertions, tests produced results analogous to those found in the case of random bit deletions. The only difference is that the period of the sequence is tending to grow longer and therefore the spectral peaks shift in the opposite direction.

As an example of the effect of random bit insertions, Figure 3.6 shows the spectrum of the same period 16 full sequence with 119 bits inserted at random. The bound on how far the sampling time can slip back in one sampling interval is set at 0.3.

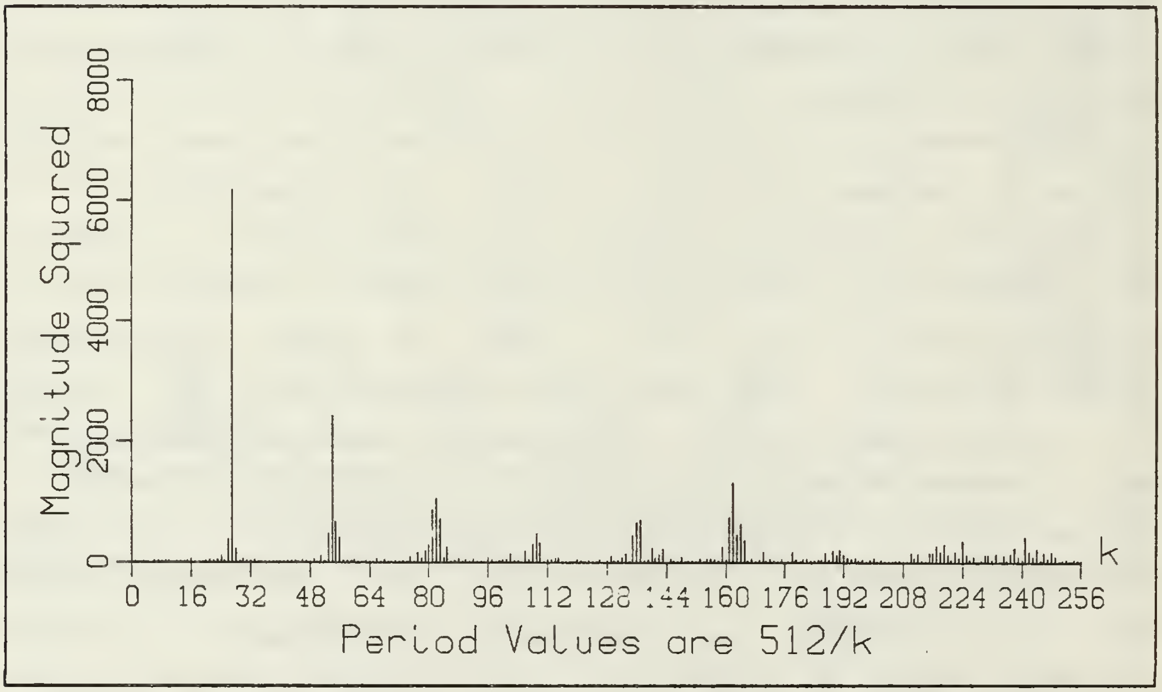

Figure 3.6 Random Bit Insertions.

Another FORTRAN program was written to combine these two timing error models. Appendix H contains this program. This program functions in exactly the same manner as the two previous programs except that during each bit interval it is equally likely that the sampling time slip forward as back. Consequently, a bit is occasionally lost from the sequence and a bit is occasionally inserted into the sequence. The net effect is that of an instability or "jitter" in the timing.

This case produced some interesting results. Since the net effect of deletions and insertions left the period relatively unchanged, the harmonic signal components remained in their proper locations. There is however considerable spectral smearing present and this effect is progressively worse in higher signal harmonics. The higher harmonics are also progressively attenuated. Compared to to effects of additive bit errors and bursts of errors on the spectrum, timing errors cause some very different and much more severe alterations.

Figure 3.7 is the spectrum of the same period 16 full sequence used in previous examples. In this example, it has been subjected to both random deletions and random insertions of bits. 52 bits have been deleted, while 45 bits have been inserted.

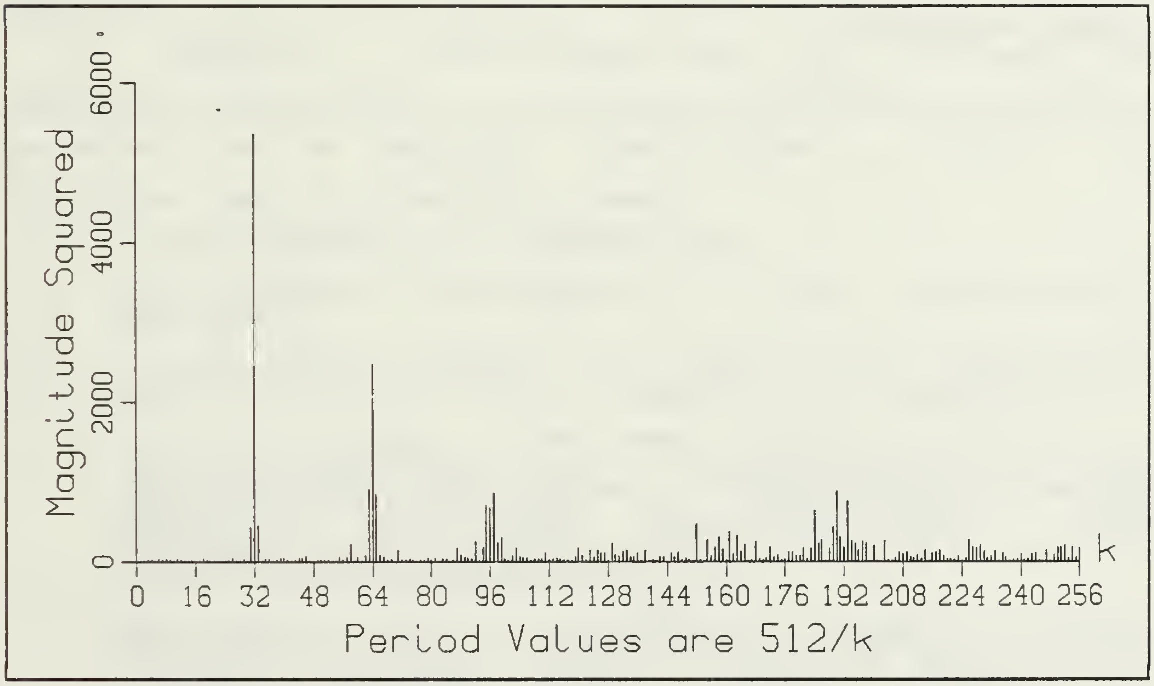

Figure 3.7 Timing Errors.

D. CONCLUSIONS ON NOISE EFFECTS

From the discussion above it is obvious that attempting to gather information from the spectrum of a periodic sequence subject to the effects of timing errors might prove to be a frustrating endeavor. Future studies could be conducted to examine sequences subjected to such effects. It should be easy to tell the difference between synchronization errors and additive errors for an analysis of the respective spectra. A further limiting assumption about noise effects must be made in order to proceed towards the goal of being able to quantitatively evaluate the significance of spectral components of a periodic sequence. All noise processes considered in this thesis will hereafter be assumed to result in independent identical Normal distributions for errors in the periodic sequences being studied. That is, the only error effects allowed will be random bit errors. Perfect bit synchronization will be assumed, and hence, the possibility of timing errors is no longer considered.

IV. FISHER'S METHOD AND ITS APPLICATIONS

In harmonic analysis, conclusions concerning the periodic nature of a time series are based on the magnitudes of harmonic components of the spectrum of the time series. It can be a difficult and often misleading task to look for peaks in such a spectrum. Components at the Fourier frequencies are bound to show many peaks and troughs due to the fact that they are approximately independent [Ref. 6: p. 110]. The spectrum of a set of purely random numbers will consist of some harmonic components which are larger than others by chance alone. One approach to determining the reliability of the harmonic components of a time series might therefore be to compare them to the harmonic components derived from a purely random time series. This is the approach of Fisher's test of significance in harmonic analysis.

In general, a significance test is concerned with deciding whether or not a hypothesis concerning statistical parameters of a sampling distribution is true. The following steps are typically taken:

1. A null hypothesis is decided upon.
2. An alternative hypothesis to the null hypothesis is developed.
3. A statistic, (which is a function of observations made), is decided upon to test the null hypothesis.
4. A critical region of the sample space is chosen such that the probability of a particular sample being observed within that region, conditioned on the null hypothesis being true, is very small. This probability is called the significance level. It is sometimes expressed as a percentile, which is found by taking 100*(1-probability).
5. Applying the test of significance involves rejecting the null hypothesis when an observed sample falls within the critical region. Since the probability of a sample appearing is quite small, when the null hypothesis is assumed true, that appearance is regarded as evidence against the null hypothesis. [Ref. 7: pp. 103-105]

A. FISHER'S METHOD

In 1929 R.A. Fisher, [Ref. 1] developed a test for significance in harmonic analysis. To provide the necessary background, a brief outline of Fisher's test is presented. For a detailed derivation of the formulas used in Fisher's test see Grenander and Rosenblatt [Ref. 8: pp. 91-94].

Consider a series, $x(t)=s(t)+n(t)$, $t=1,2,...N$, which has been sampled at equal time intervals. The series $s(t)$ is deterministic, while the series $n(t)$ is composed of independent identically distributed Normal random variables. $n(t)$ is $N(0,var)$, where the variance is unknown. The objective of this test is to make some statistical inference concerning the periodicity of $s(t)$. The null hypothesis is that only $n(t)$ is present, that is, the observed sample has no periodic activity. The alternative hypothesis is that periodic activity is present in the observed sample.

The sequence $x(t)$ may be decomposed into its harmonic components using the Fourier series representation. This is accomplished in the following manner:

$$x(t) = a_o/2 + \sum_{k=1}^{m} \left[a_k \cos(2\pi kt/N) + b_k \sin(2\pi kt/N)\right]$$

where m is the total number of harmonic components ($m=(N-2)/2$). Also, the coefficients a_k and b_k and the constant a_o are computed as follows:

$$a_o = 2/N \sum_{t=1}^{N} x(t)$$

$$a_k = 2/N \sum_{t=1}^{N} x(t)\cos(2\pi kt/N)$$

$$b_k = 2/N \sum_{t=1}^{N} x(t)\sin(2\pi kt/N)$$

The harmonic amplitude, c_k, is defined as:

$$c_k = (a^2+b^2)^{\frac{1}{2}}$$

The test statistic used in Fisher's test is the normalized harmonic amplitude, g_k, defined as:

$$g_k = c_k^2 \Big/ \sum_{k=1}^{m} c_k^2$$

A normalized quantity is used to remove the effect of the unknown variance and to restrict the values of this test statistic to lie between zero and one. The values of g_k are re-ordered according to size with $g_1 > g_2 > \cdots > g_m$.

Fisher derived the following expression for the probability that the largest normalized harmonic amplitude, g_1, is greater than a parameter x. This probability is conditioned on the null hypothesis being true, that is, only noise is assumed to be present.

$$P(g_1 > x) = m(1-x)^{m-1} - \frac{m(m-1)(1-2x)^{m-1}}{2} + \cdots$$

$$\cdots + \frac{(-1)^{m-1} m!(1-Lx)^{m-1}}{L!(m-L)!} \qquad (4.1)$$

The variable L in this equation represents the largest integer less than 1/x. This equation is solved for x in terms of p and m. It is then used to generate tables of x for a few particular values of p and for various values of m. Fisher recommends using p=0.05 and various values of m depending on the number of data points available.

Since p represents the probability that noise alone could account for g_1 being greater than x, 1-p suggests how much confidence could be placed in the assumption that a periodicity of the signal caused this. Therefore, 100*(1-p) is called the significance level percentile. In recommending the use of p=0.05, Fisher suggests that harmonic components within the 95th percentile are of interest.

A simple example can best illustrate how Fisher's test is applied. Suppose that a data set is examined to determine the presence of a periodic signal within the 95th percentile. The value of g_1 is computed from the data. Entering the table for p=0.05, the value of x is extracted which corresponds to the number of data points used. If it is observed that $g_1 < x$, no signal periodicity is present: the null hypothesis is accepted. If $g_1 > x$, a signal periodicity is assumed to be present in the 95th percentile: the null hypothesis is rejected and the alternate hypothesis is accepted.

Later, Fisher's test was extended to test for the significance of the g values of lesser magnitude [Ref. 9]. For the rth largest normalized harmonic amplitude, g_r, the probability that g_r exceeds a parameter x is given by:

$$p(g_r > x) = \frac{m!}{(r-1)!} \sum_{j=r}^{L} \frac{(-1)^{j-r}(1-jx)^{m-1}}{j(m-j)!(j-r)!} \qquad (4.2)$$

Once again, this is conditioned on the presence of noise alone. This formula actually indicates the probability that the r components g_1, g_2, $\cdots$ g_r, are greater than a parameter x. This extended version of Fisher's test is used in the same manner as Fisher's original test.

If independent identical normal distributions for errors cannot be assumed, then Fisher's test still provides a reasonable approximation to a measure of significance [Ref. 6: p. 111].

B. APPLICATION BY NOWROOZI

In 1966, Fisher's test was applied by A.A. Nowroozi [Ref. 2] to the problem of estimating the period of eigenvibrations of the earth. To facilitate his analysis, Nowroozi first developed tables of x values for the p and m values of interest in his study. His tables were designed to test for the presence of the largest harmonic amplitude only.

Nowroozi proposed two practical applications of Fisher's test.

1. First Application

The first practical use of Fisher's test proposed was almost identical to the method presented by Fisher. To briefly mention it's key points:

1. A decision is made prior to application of the test about the significance level of interest.
2. Next g_1 is computed and compared to the tabulated parameter x. If $g_1 > x$ the associated period is within the percentile indicated and if $g_1 < x$ it is not.
3. The same test is applied to g_2, g_3, etc.

This application of Fisher's test has a few weaknesses. The application of the test for g_1 is made to succeedingly smaller values of g_r instead of using the extended version of Fisher's test. This has the effect of possibly rejecting harmonic components which would have been accepted as significant by the extended version. Also, selecting a confidence level prior to application of the test is a somewhat arbitrary decision which could exclude some important data.

2. Second Application

As an alternative to the first method discussed above, Nowroozi suggested the following:

1. Plot the squared harmonic amplitudes, (c_k's squared), against the corresponding periods which they represent.
2. Decide upon a significance level.

3. Calculate a minimum significant squared amplitude, C, from the appropriate tabulated x using the relationship:

$$C = x \cdot \sum_{k=1}^{m} c_k^2$$

4. Draw a horizontal line across the plot of step (1) at the level of C.

5. Any spectral peaks having squared amplitudes above this line are significant at the level chosen previously.

Nowroozi's second approach is simpler to implement than the first, however, it shares the same weaknesses. It is this application of Fisher's test which Nowroozi actually used to estimate the periods of eigenvibrations of the earth. Nowroozi himself labeled some peaks as plausible which actually failed his test. He was therefore not totally confident himself in the method for applying Fisher's test which he developed, but felt that it was perhaps too severe.

C. APPLICATION BY SHIMSHONI

In an effort to improve the effectiveness of the analysis carried out by Nowroozi, Shimshoni (in 1971) suggested a more proper application of Fisher's test [Ref. 3]. He pointed out that in Nowroozi's application of Fisher's test, every component below a certain level of significance is rejected. In this he failed to take into account that the test actually refers only to the largest component. For this reason, some components which seemed quite plausible were rejected by the test which Nowroozi performed.

Shimshoni suggested that Fisher's test should be applied in its extended form. To facilitate this, he developed tables of x values for several of the largest normalized harmonic amplitudes instead of only the first. Again these tables allowed for various values of p and m. Using these extended tables, he proposed the following practical application of Fisher's test:

1. Decide upon a significance level of interest.
2. Compute and sort normalized harmonic amplitudes (g's) in decreasing order (with g_r =the rth largest amplitude).
3. In the table for the significance level chosen, look up x for r=1.
4. Accept all components greater than x as being significant at the desired level.
5. For the first amplitude in the list which fails, look up the x which corresponds to it's position in the sorted list.
6. Continue to accept amplitudes which are greater than this new value of x.
7. Repeat this process until reaching an amplitude which is smaller than the tabulated x value which corresponds to it. This amplitude and all succeeding amplitudes are thus failed at the chosen level of significance.

This method proposed by Shimshoni is a more proper application of Fisher's test in that it makes use of the extended form of his test. As a result, Shimshoni's method does accept some of the plausible periods which Nowroozi is forced to reject. His method does however suffer from the weakness of having to select a level of significance prior to applying the test. This could possibly result in the rejection of valuable data. For example, a spectral component within the 94.9th percentile will be rejected if only the 95th percentile is considered. This algorithm is also cumbersome to implement due to the necessity for several table look-ups and the possible need for interpolation.

V. A PROPOSED METHOD FOR THE APPLICATION OF FISHER'S TEST

A. DEVELOPMENT OF A PROPOSED METHOD

Having discussed Fisher's test and the various methods by which it has been applied, the question which naturally arises is one of optimization. How might Fisher's test "best" be applied? First, the criterion by which "best" is judged must be specified:

- The most applicable; that is, the method best suited to the analysis of pseudo-random sequences.
- The most straightforward; that is, the simplest and most direct method of performing calculations.
- The most exhaustive; that is, using the test to its fullest possible extent, not disregarding any useful information which it might provide.

In terms of applicability, Fisher's test should be used in its extended form in the analysis of pseudo-random sequences. Fisher himself suggested conditions under which the extension of his test might prove especially useful;

> "The second may be used in a test whether the second largest is significant, such as might be useful if, when the largest is doubtfully significant, it may still be suspected that the two largest are due to some systematic causes." [Ref. 9: p. 16]

It was shown earlier that in the case of the sequences of chapter II, their spectra consists of several harmonically related components of comparable magnitude. For the situation of interest, several of the larger components can be expected to be due to the same systematic cause; the periodicity of the sequence. Applying Fisher's test in the extended form appears to be best suited for the analysis of the sequences considered in this thesis.

Fisher's test could be applied in a straightforward manner, without the use of tables, by direct calculation of probabilities from Eq. 4.2. Since this is a calculation of the probability that g_r exceeds the value of x, a suitable parameter x must be chosen. Selecting x equal to the value of g_r observed would result in calculating the probability of the r largest normalized harmonic components occuring at higher amplitudes than those observed. In this manner, the calculation of the probability that g_r exceeds the value of g_r observed provides a measure of the significance of that particular value of g_r observed. Applying Fisher's test in this way, the difficulties involved in developing extensive tables to meet the needs of each particular situation and in using these tables in calculations are avoided. It is also no longer necessary to decide upon a level of significance in advance, a practice which can result in the loss of valuable data.

Finally, making use of the foregoing suggestions should result in the most exhaustive use of Fisher's test. Applying the test in it's extended form to as many of the larger components as possible would return the maximum amount of useful information. Components slightly smaller in magnitude than the largest will not be routinely disregarded for failing to exceed a predetermined level of significance. Applied in this manner, Fisher's test gives the spectrum analyst a quantitative basis upon which to not disregard any "good" information or accept any "poor" information.

The specific steps in this proposed method of applying Fisher's test can now be outlined:

1. Compute the FFT of the data sequence to find the magnitudes of harmonic components (c's).
2. Calculate the normalized harmonic components (g's).
3. Sort the normalized harmonic components in order of decreasing magnitude ($g_1 > g_2 > \cdots > g_m$).
4. Calculate the measure of significance of as many of the largest normalized harmonic components as is desired for analysis.
5. Sort the harmonic components according to their respective measures of significance, from the smallest to the largest probability calculated. The smallest probability is the highest measure of significance.

6. Perform analysis of the signal spectrum on the basis of the calculated measures of the significance of harmonic components.

Most of the steps listed are straightforward and simple to execute. The only step which presents a computational challenge is the actual calculation of the measures of significance. This step will therefore be discussed in more detail.

B. CALCULATION OF THE MEASURE OF SIGNIFICANCE

The measure of significance of a particular normalized harmonic component is calculated by direct application of Fisher's test in its extended form. Since no level of significance is set beforehand, what is actually calculated is the probability that a particular normalized harmonic component will exceed its measured level. Once again, this is under the assumption that only noise is present. This provides a measure of the significance of a spectral component. A small probability corresponds to a high level of significance.

The formula used to calculate harmonic significance is restated as follows:

$$P(g_r > x) = \frac{m!}{(r-1)!} \sum_{j=r}^{L} \frac{(-1)^{j-r}(1-jx)^{m-1}}{j(m-j)!(j-r)!}$$

To facilitate programming, this equation is expressed as:

$$p(g_r > x) = \frac{1}{\prod_{k=1}^{r-1}(r-k)} \sum_{j=r}^{L} \frac{(-1)^{j-r}\left[\prod_{k=1}^{m-1}(1-jx)\right]\left[\prod_{k=1}^{j}(m-k+1)\right]}{\left[\prod_{k=1}^{j-r}(j-r-k+1)\right]} \qquad (5.1)$$

Expressed in this form, the factorials and powers within the summation can be calculated simultaneously, by combining one term from each product at a time. In this way, the loss of accuracy associated with floating point arithmetic involving very large and very small numbers can be avoided.

A FORTRAN program was written to test the calculation of harmonic significance using Eq. 5.1. Extended precision was used in calculations in an effort to preserve as much accuracy as possible.

Despite precautions taken in programming, making test computations often led to an underflow condition (magnitude less than 10**-87). In an effort to prevent this, the product series was truncated when terms of magnitude less than 10**-75 were encountered. Since the product series is strictly decreasing, this was not detrimental to the accuracy of computations. Although computations involving smaller arguments occasionally led to an underflow condition, this did not adversely affect the accuracy of computations.

To prove the accuracy of the computer algorithm developed, a comparison test was conducted. The data used for this comparison was obtained from Shimshoni [Ref. 3: pp. 374-375]. Shimshoni obtained his data by iteratively solving Eq. 5.1 for x. Although he does not specify the degree of accuracy of his data, it is assumed that figures are accurate to five significant figures as listed. The results of these tests are summarized in Table 1. In this table, m represents the total number of harmonic components and r represents the order of a normalized harmonic component in the sorted list.

As Table 1 indicates, a quite acceptable error rate of less than 1.5 percent was achieved even for large values of m. The m values of interest in this thesis are 255 or less.

TABLE 1

PERCENT ERROR OF HARMONIC SIGNIFICANCE COMPUTATION

m	r=1	r=2	r=5	r=10
20	0.003	0.021	0.020	-----
50	0.011	0.057	0.046	0.022
100	0.005	0.071	0.071	0.070
300	0.021	0.170	0.136	0.860
500	0.093	0.065	0.867	0.230
1000	0.400	0.195	1.460	1.180

C. PROGRAM FOR THE APPLICATION OF FISHER'S TEST

A FORTRAN program was developed to apply Fisher's test to the analysis of noise corrupted, pseudo-random sequences. This program is contained in Appendix I. This program was designed to work in conjunction with previously written data generation and noise introduction programs. The program accepts as input the magnitudes of harmonic components computed by the DFT in the additive bit error program. This program then performs the sequence of steps outlined in section A of this chapter, the proposed method for the application of Fisher's test. As output, the program plots the spectrum and provides a table summarizing the results. The most significant components are also labeled with their respective measures of significance.

Fisher's test can calculate a measure of significance for the largest normalized harmonic components for which Eq. 5.1 can calculate a probability. As smaller components are used in Eq. 5.1, the probabilities computed approach unity. This implies that the component is almost certainly due to the noise alone. Attempting to calculate probabilities for even smaller components results in values outside the defined range for probabilities. For that reason, the program was designed to ignore any such results.

Experimentation suggests that Fisher's test could generally find a measure of significance for harmonic components of magnitude greater than 1.5 standard deviations from the mean.

Another question to be addressed is that of determining how large the DFT must be in comparison to the period of the sequence being studied. Experimentation shows that, in general, the DFT should be applied to data at least four times as long as the period of the sequence being studied. Otherwise Fisher's test as implemented will not produce any useful information. If only four periods or less are available, there will be so many multiples of the fundamental frequency present that the normalized harmonic components will be too small to allow for calculation of probabilities. On the other hand, the larger the DFT is with respect to the sequence period, the larger the amount of information available will be.

VI. ESTIMATING THE PERIOD OF A PSEUDO-RANDOM SEQUENCE

Estimating the period of a noise garbled, pseudo-random sequence is a critical factor in the analysis of such sequences. Once an accurate estimate of the period has been ascertained, further analysis into the method by which the sequence was generated is possible.

A. ESTIMATION PROCEDURE

The spectrum of a noise garbled, pseudo-random sequence is analyzed to produce an estimate of the sequence period. The analytic method adopted is Fisher's test as proposed in the previous chapter. Pseudo-random and related sequences analyzed include those discussed in chapter II. Noise effects are limited to the random bit errors (IID, N(0,Var)) for which Fisher's test was designed.

As was observed in the discussion of pseudo-random sequences, their spectra should, with adequate signal to noise ratio, consist of larger amplitude components at the fundamental frequency and at several multiples of it. It is often the case, however, that the fundamental frequency component is much smaller than some of its multiples. This situation was observed in the case of Gold codes. The fundamental frequency component may even be totally lost in the noise floor. It will therefore be necessary to determine the greatest common divisor (GCD) of the most significant components identified by Fisher's test. If these components are assumed to be at multiples of the fundamental frequency, the GCD may then be used as an estimate of the fundamental frequency and from it an estimate of the period will be computed.

Even so, it is possible to arrive at an incorrect estimate. Suppose, for example, that the second and forth multiples of the fundamental frequency are the only reliable components found. Since the GCD is the frequency of the second multiple, the estimate of the period will be one half the actual value. For this reason, the greater the number of reliable components available for analysis, the better the resulting estimate of the period. If only a limited number of highly significant (say, 99th percentile) components are available, it would be beneficial to make use of any components of slightly lower significance (say, 95th percentile) in deriving an estimate of the period.

This analysis will make use of the interactive FORTRAN programs which have been introduced at various points throughout this paper. These programs are designed to work in conjunction with one another in the following manner:

- Phase A: These programs generate the raw data sequences (see Appendices A,B and C).
- Phase B: This program introduces random bit errors into the data sequence and computes the signal spectrum by the DFT (see Appendix D).
- Phase C: This program applies Fisher's test to the signal spectrum using the method proposed in chapter 5 (see Appendix I).

B. CASE ONE: FULL-SEQUENCES

Full-sequences are analyzed first due to the simplicity of their spectra. Because a full-sequence may be chosen with a period which divides the length of the DFT evenly, no smearing of spectral components occurs. This situation results from the fact that the FFT algorithm used to compute the DFT is most easily implemented on power of two sized data sets.

1. <u>Example One</u>

For example, choose the data sequence to be a period 16 full-sequence. Let the error rate be 25 % and compute a 512 point DFT.

Figure 6.1 shows the signal spectrum with the components of highest significance labeled with the probabilities that noise could account for a component of greater amplitude. Note that only the first half of the spectrum is displayed, since the other half is a mirror image. Only frequency components from k=1 to k=255 are considered by Fisher's test, for in this case m=(512-2)/2=255. Table 2 lists the components of highest significance with their magnitudes squared and their respective percentiles.

Two components, at k=192 and k=64 are observed within the 99th percentile. Their greatest common divisor is 64. Basing an estimate of the sequence period on this information would yield a period of 8. This is unfortunately incorrect. Observing Table 2, reveals a component with significance within the respectable 95th percentile at k=32. Including this component yields a GCD of 32 and hence, a period of 16, which is correct.

This example serves to illustrate the effectiveness of the proposed method. Had a method been employed which pre-determined the use of a 99th percentile significance level, an incorrect conclusion would have resulted. This is despite the fact of having detected two components within the 99th percentile.

An interesting situation can be observed in this particular example. The component at k=32 is slightly larger than the component at k=64, yet it is calculated to be at a considerably lower significance level. This situation occurs whenever two components are found to be nearly equal in magnitude. The significance calculation for the larger component yields the probability that random noise could account for the appearance of two components at or above the measured value of that component. In the case of the smaller component, the calculation yielded the probability of three such components appearing at or above its nearly equal measured value. The probability of three

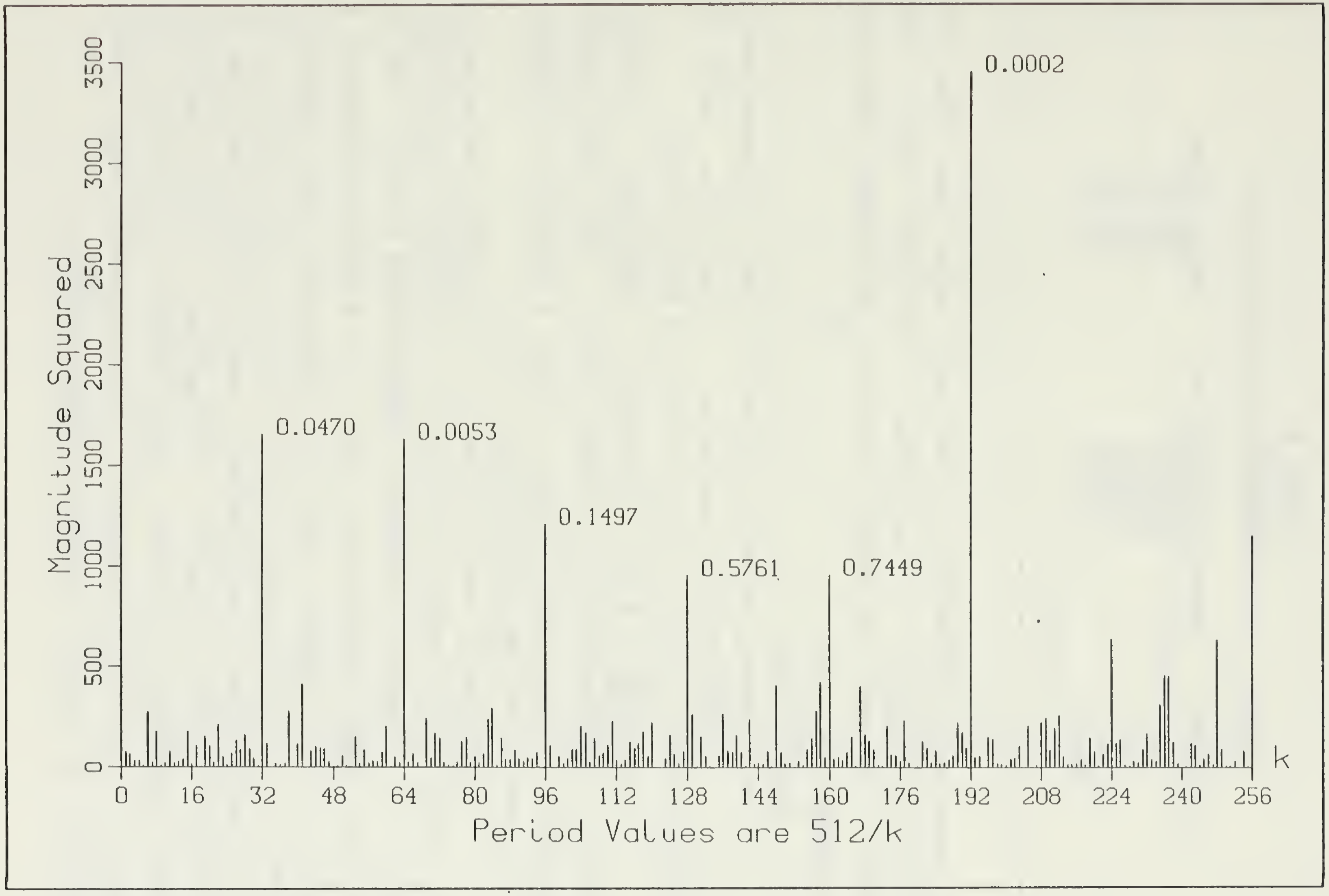

Figure 6.1 Spectrum for Example One.

TABLE 2

SIGNIFICANT COMPONENTS IN EXAMPLE ONE

Component	Magnitude Squared	Percentile
192	3461.0	99.98
64	1631.0	99.47
32	1653.7	95.30
96	1210.3	85.30
128	954.0	42.39
160	959.5	25.51

such components appearing is obviously much less than the probability of only two appearing. For that reason, the smaller component is found to be at a higher level of significance.

2. <u>Example Two</u>

For a second example, choose the data sequence again to be a period 16 full-sequence. The error rate is increased to 33 % and a 512 point DFT is again used.

In Figure 6.2 the spectrum is displayed, labeled again with the appropriate probabilities. Note the smaller magnitudes on the ordinate axes. Table 3 provides a summary of the results of applying Fisher's test.

This example was included to illustrate the value of this method of analysis in preventing erroneous conclusions. The two components of highest significance present are at k=192 and k=64, which have a GCD of 64 and hence indicate a period of 8. Though these two components were actually caused by the periodicity of the sequence, they are shown to be of insufficient significance to realistically use them in arriving at any conclusion. Attempting to use the third largest component, at k=148, would result in a GCD of 4 and hence an incorrect estimate for the period of 128. This component is actually caused by noise effects.

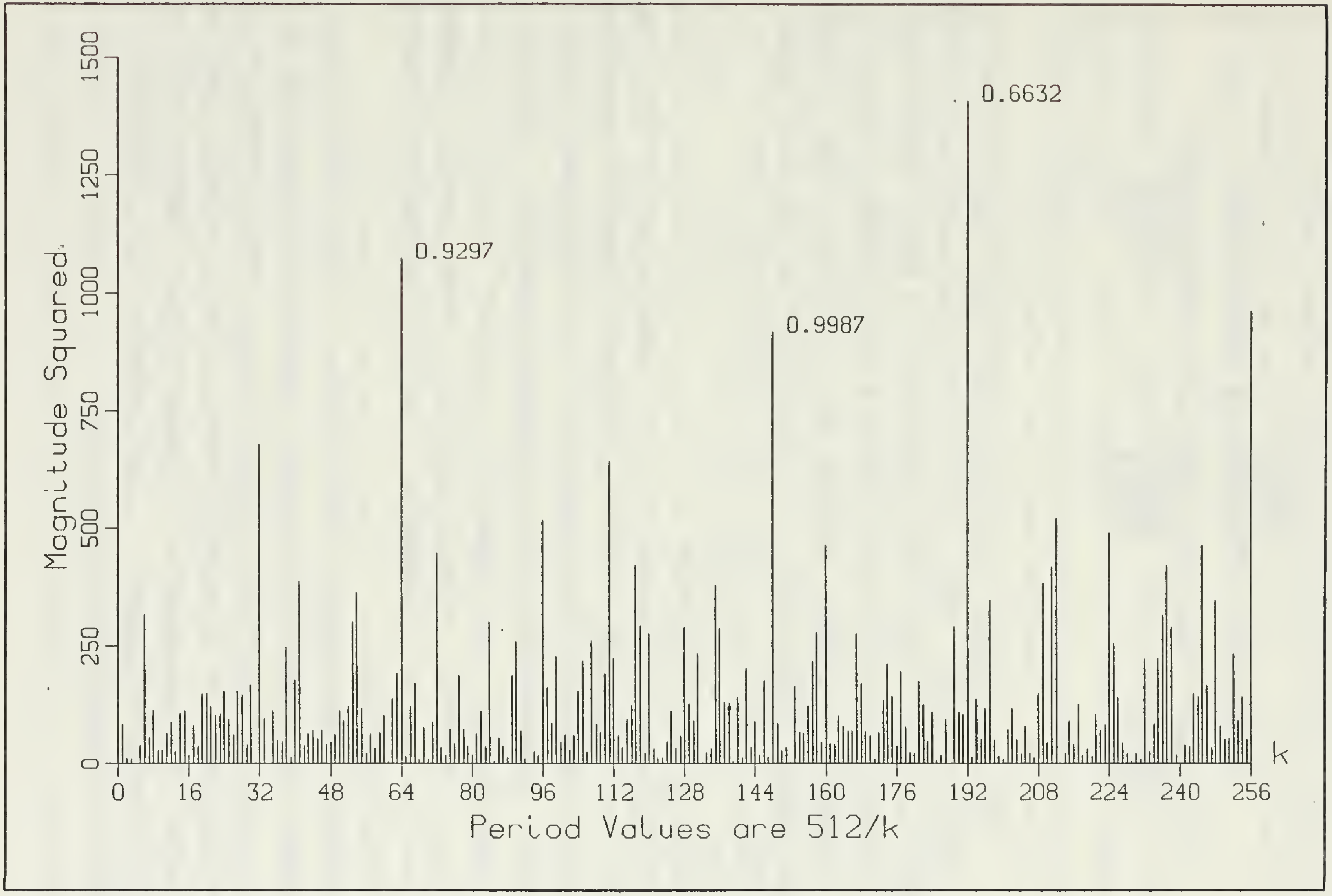

Figure 6.2 Spectrum for Example Two.

TABLE 3

SIGNIFICANT COMPONENTS IN EXAMPLE TWO

Component	Magnitude Squared	Percentile
192	1405.9	33.68
64	1072.1	7.03
148	915.5	0.00

If conclusions had been drawn from observations of the spectrum alone, using the two or three most conspicuous looking values, then the possibility of error would have been great. Note that the ordinate axis is again scaled to the largest component present and therefore the larger values appear conspicuous. In reality, as the test shows, they are not of a very high significance level.

C. CASE TWO: M-SEQUENCES

In the case of m-sequences, the sequence period does not divide the DFT length evenly. Therefore spectral smearing occurs. Since the signal components are spread over several adjacent frequencies, the normalized harmonic amplitudes will be smaller. Consequently, Fisher's test will be weaker. Another effect is that since only integer frequencies can be represented, only an approximate period can be ascertained.

Despite anticipated difficulties, the method still performs well. Tests involving a variety of m-sequences and error rates confirmed that Fisher's test remains adequate in computing measures of significance. Also, simply rounding estimates of the period to the nearest whole number leads to satisfactory conclusions. An example is included to illustrate these results.

1. Example Three

In this example the data sequence used is a period 15 m-sequence with an error rate of 25 %. A 512 point DFT is used to generate the spectrum.

The spectrum is shown in Figure 6.3, with the components of highest significance again labeled with their probabilities. Table 4 summarizes the results of applying Fisher's test.

In this example, only one component, at k=34, was found in the 99th percentile and one more, at k=68, was found in the 95th percentile. The GCD of these two frequencies is 34, leading to a period of 15.06. Rounding this to the nearest whole number yields an estimate for the period of 15, which is the correct value. Attempting to make use of the component of the next highest significance (in the 88th percentile), at k=171, leads to complete failure. The GCD of the three components is one, implying that no periodicity is present in the data. Due to spectral smearing components can bleed into adjacent frequencies, especially at higher harmonics. For this reason, a little dithering applied to the component at k=171 might be helpful. If it is assumed to be located at k=170, then it could represent the fifth harmonic and can be used to further confirm the conclusion based on the two components found at higher levels of significance.

This example serves to point out how this method of analysis is weaker in the case of sequence periods which do not evenly divide the DFT length. By comparison, Example One was conducted under exactly the same conditions as Example Three except that the period 16 full-sequence did divide the DFT length evenly. Therefore, no spectral smearing occurred. In that case, Fisher's test was more effective because it identified components of higher significance on which to base an estimate of the period.

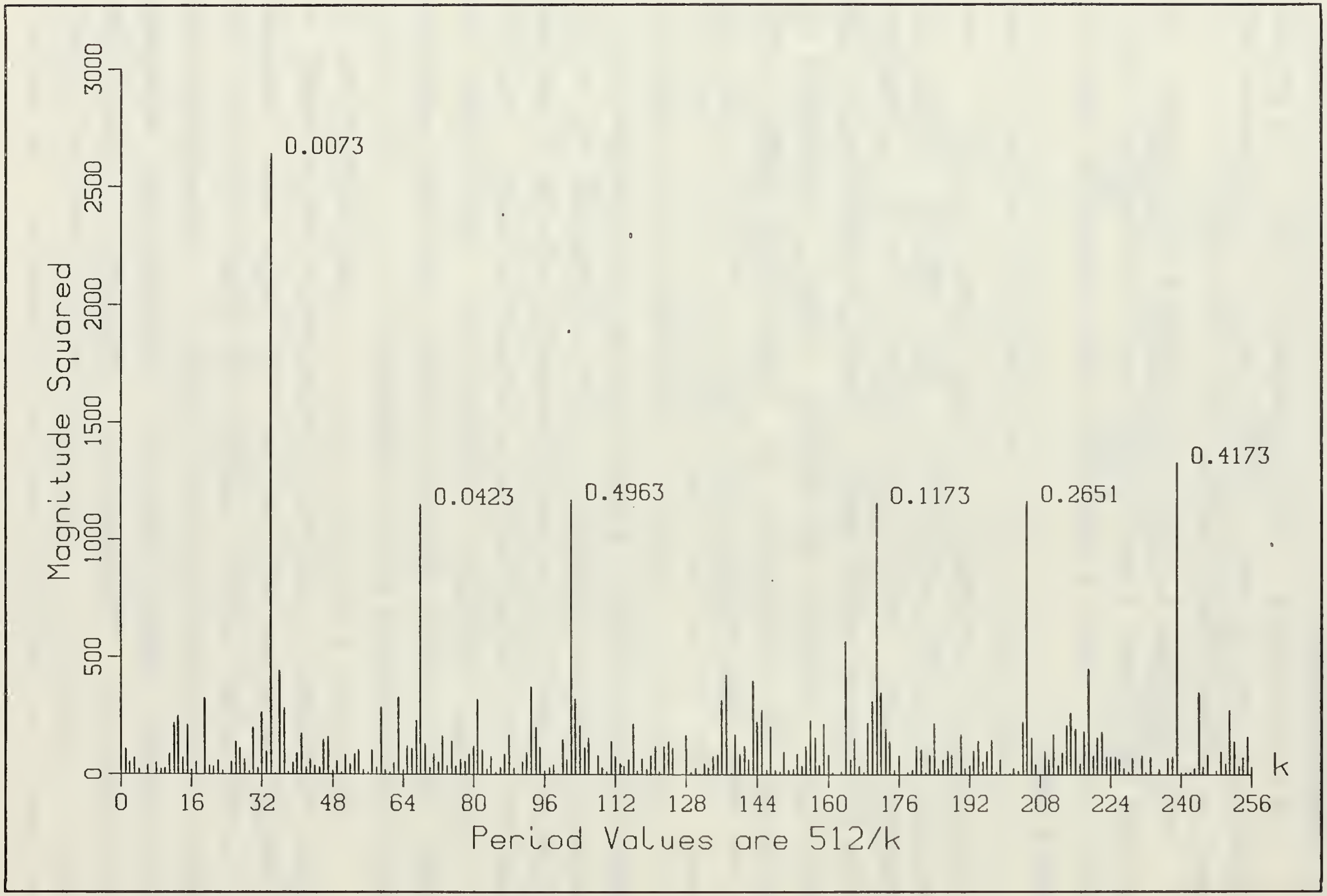

Figure 6.3 Spectrum for Example Three.

TABLE 4

SIGNIFICANT COMPONENTS IN EXAMPLE THREE

Component	Magnitude Squared	Percentile
34	2641.7	99.27
68	1150.8	95.77
171	1157.3	88.27
205	1165.2	73.49
239	1328.5	58.27
102	1172.3	50.37

Two components appeared in the 99th percentile and one in the 95th percentile. Several components of lower significance were multiples of the fundamental frequency and some could be used if necessary to improve confidence in the estimate, without the use of dithering. Even with these inherent difficulties, the method of analysis yields satisfactory results in the case of non-evenly dividing sequences. In this situation, conclusions should be based only on highly significant components. Any components appearing within the vicinity of the 95th percentile may be considered highly significant. Dithering may also be necessary in the case of components of slightly lower significance, especially when they represent higher harmonics of the fundamental frequency.

D. CASE THREE: GOLD CODES

Since Gold codes are formed by summing two m-sequences of the same period, their common period also does not divide the DFT evenly. As a result, spectral smearing is present. Besides sharing all the difficulties observed for m-sequences, Gold codes introduce some further problems of their own. Because Gold codes do not exhibit the properties associated with pseudo-randomness nearly as well as m-sequences, (ie. balance, run and correlation), their spectra are not nearly as well structured. The spectrum of a Gold code is much more "jumbled" in its appearance. This has a detrimental effect on the analysis of such sequences as demonstrated in the next example.

1. Example Four

This example analyzes the period 31 Gold code formed by summing the two m-sequences generated by the polynomials $f(x)=x^5+x^2+1$ and $f(x)=x^5+x^4+x^3+x^2+1$. The initial load for the first shift register is 1,0,0,0,0 and for the second shift register is 0,0,0,0,1. This particular Gold code has a very low density of ones (12 ones and 19 zeros) and therefore exhibits poor pseudo-randomness properties. The error rate used is 20 %. A 512 point DFT is used to compute the spectral components. The signal spectrum, labeled with the probabilities computed is shown in Figure 6.4. Table 5 summarizes the results of Fisher's test.

In this example, the two components of highest significance, at k=33 and k=165, have a GCD of 33. This leads to an estimated period of 15.5. This estimated period is exactly one half of the correct value. Observation of the spectrum reveals that the odd multiples of the fundamental frequency, including the fundamental itself, are suppressed. It is therefore not possible to generate a reliable estimate of the period by the proposed method.

Example Four is typical of attempts to estimate the period of a Gold code by the method proposed in this thesis. Though the suppression of odd harmonics observed in this example is not typical of all Gold codes, their spectra are in general much less well structured than the spectra of the m-sequences from which they were derived. Due to the difficulties discussed earlier and confirmed by numerous experiments, this method of analysis was found to be less effective for the analysis of Gold codes.

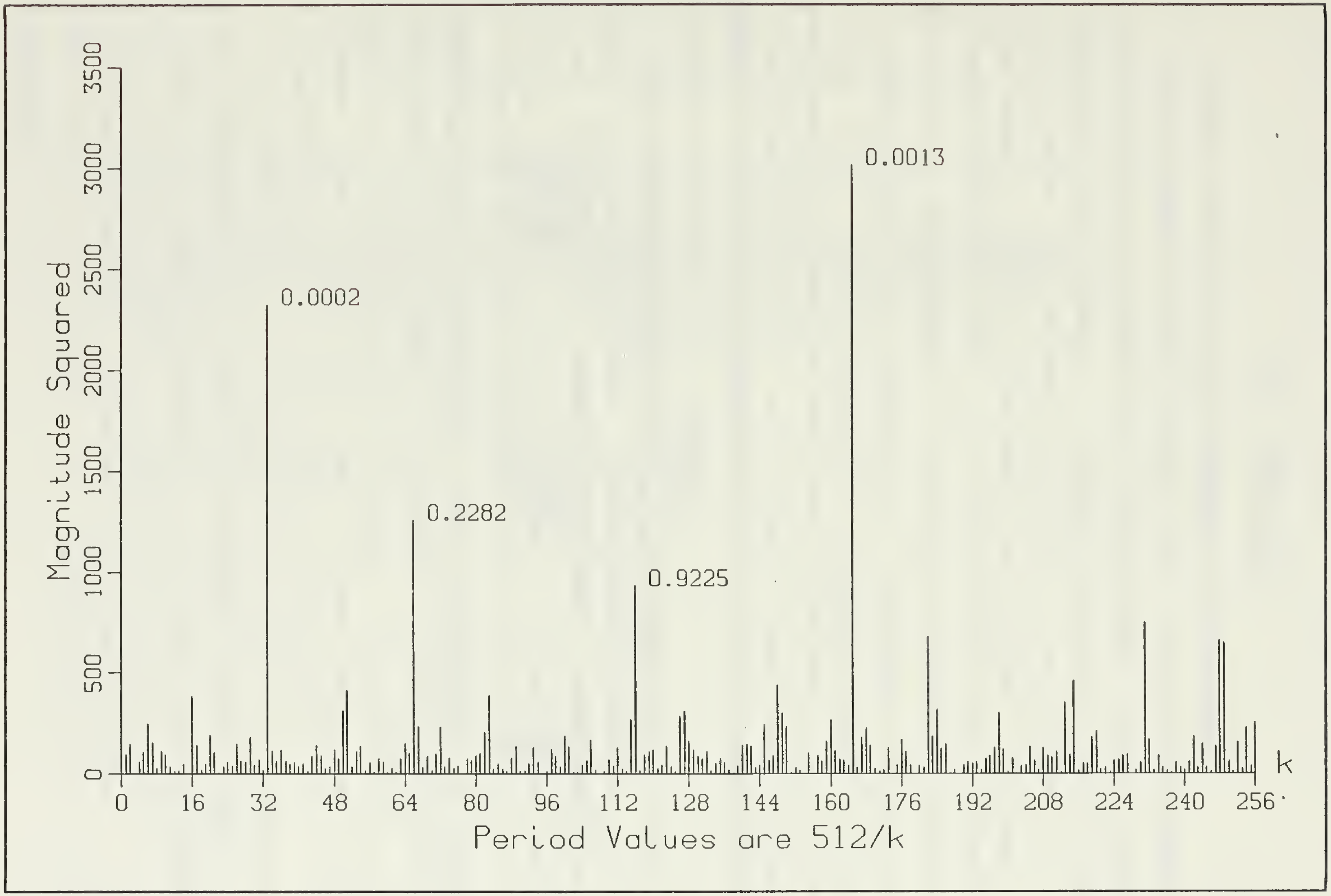

Figure 6.4 Spectrum for Example Four.

TABLE 5

SIGNIFICANT COMPONENTS IN EXAMPLE FOUR

Component	Magnitude Squared	Percentile
33	2324.7	99.98
165	3015.0	99.87
66	1257.3	77.18
116	930.6	7.75

VII. CONCLUSION

A. METHOD FOR DETERMINING SPECTRAL SIGNIFICANCE

The purpose of this research was to develop a method to quantitatively evaluate the significance of the spectral components of a signal. A method known as Fisher's Test of Significance in Harmonic Analysis was found in the literature, which promised to provide a means of accomplishing this [Ref. 1]. Fisher's test had been applied by previous researchers in a variety of different ways, none of which took full advantage of the power of this test [Refs. 2,3]. A new method for applying Fisher's test is proposed which uses the test more effectively and in a more direct manner. In this way a simple, flexible and effective method is found for determining the significance of the spectral components of a signal.

Applying this method to the problem of estimating the period of a noise garbled pseudo-random sequence met with mixed results. In some situations a good estimate is readily obtainable, while in others the method of analysis is found to be less effective. The failure is not in the methods ability to determine the significance of spectral components, but rather in the manner in which the method is applied to this particular problem. The lack of a well ordered spectrum in the case of some sequences studied proved detrimental to this method of analysis.

In conclusion , a method is developed for determining the significance of spectral components and is shown to be of value in harmonic analysis. As the scope of problems dealt with by harmonic analysis is quite broad, it is conceivable that this method may find applications in areas other than the analysis of pseudo-random and related sequences.

B. SUGGESTIONS FOR FURTHER RESEARCH

There are several promising avenues for further research. To begin with, the method developed in this research could be applied to other problems in harmonic analysis. Any situation involving the analysis of a simple periodic signal in the presence of noise could be approached using this method. One such example is the detection of the doppler shifted blade rate of a torpedo propeller in the noisy environment of a sonar signal.

Assuming different parameters, such as different noise generators, the problem of evaluating the significance of spectral components may be solved again. This may involve rederiving the probability expression under a different assumption concerning noise statistics.

A DFT program might be developed to operate on sequences of zeros and ones more efficiently, possibly allowing fast computations to be performed on longer sequences. Other fast algorithms could also be considered.

Dithering and other techniques might be developed in order to resolve the problems encountered when sequence periods did not divide the transform length evenly.

The challenging problem of analyzing periodic sequences when only a portion of the sequence is available might also be attacked by these methods. Here less than one period will be available for analysis so other properties of the sequence will have to be considered.

APPENDIX A

PROGRAM TO GENERATE M-SEQUENCES

```
C      PURPOSE:  THE PURPOSE OF THIS PROGRAM IS TO SIMULATE A LINEAR
C      SHIFT REGISTER IN ORDER TO GENERATE M-SEQUENCE DATA.
C
C      INSTRUCTIONS:  THE PROGRAM WILL PROMPT THE USER TO INPUT THE
C      DEGREE OF THE PRIMITIVE POLYNOMIAL WHICH DEFINES THE FEEDBACK
C      PATHS AND THE NUMBER OF DATA POINTS DESIRED.  THE USER WILL THEN
C      BE PROMPTED TO INPUT THE COEFFICIENTS OF THE POLYNOMIAL.  SOME
C      EXAMPLES OF PRIMITIVE POLYNOMIALS ARE:
C         F(X)=X**3+X+1, F(X)=X**4+X+1, F(x)=X**5+X**2+1
C         (FOR A MORE EXTENSIVE LIST SEE REF.4)
C
C      FILE USED IN THIS PROGRAM:
C      40:  DATA SEQUENCE GENERATED
C
       INTEGER X(15)/15*0./,F(15)/15*0./,BUF,NUM
       REAL A(2000)/2000*0./
       COMPLEX AC(2000)
C
C      INPUT DATA
C
       WRITE (6,1)
       READ(5,*) N
       WRITE(6,2)
       READ(5,*) NUM
C
C      PUT INITIAL LOAD IN SHIFT REGISTER
C
       X(N)=1
C
C      INPUT POLYNOMIAL COEFFICIENTS
C
       DO 10 I=1,N
          J=I-1
          WRITE(6,3) J
          READ(5,*) F(I)
 10       CONTINUE
C
C      GENERATE PN SEQUENCE
C
       DO 100 I=1,NUM
          A(I)=X(1)
          BUF=X(1)
          DO 110 J=2,N
```

```
          IF(F(J).EQ.1) GO TO 111
          GO TO 110
C
C         THE NEXT 4 STATEMENTS PREFORM MOD 2 ADDITION
C
 111      IF(X(J).EQ.BUF) GO TO 112
          IF(X(J).NE.BUF) BUF=1
          GO TO 110
 112      BUF=0
 110      CONTINUE
C
C       SHIFT THE SHIFT REGISTER
C
        DO 120 J=2,N
          K=J-1
          X(K)=X(J)
 120      CONTINUE
        X(N)=BUF
 100    CONTINUE
C
C     WRITE THE PN SEQUENCE INTO FILE 40
C
      DO 200 I=1,NUM
        AC(I)=CMPLX(A(I),0.0)
        WRITE(40,500) I,AC(I)
C       WRITE(6,501) I,A(I)
 200    CONTINUE
C
 1    FORMAT(' ','INPUT DEGREE OF POLYNOMIAL')
 2    FORMAT(' ','INPUT NUMBER OF POINTS DESIRED')
 3    FORMAT(' ','INPUT THE',I2,' POWER COEFFICIENT')
 500  FORMAT(I4,F12.5,F12.5)
 501  FORMAT(I4,2X,'A= ',F3.1)
      STOP
      END
```

APPENDIX B

PROGRAM TO GENERATE GOLD CODES

```
C     PURPOSE: THE PURPOSE OF THIS PROGRAM IS TO SIMULATE A GOLD CODE
C     GENERATOR IN ORDER TO GENERATE GOLD CODE SEQUENCE DATA.
C
C     INSTRUCTIONS:  AT THE PROMPTING OF THE PROGRAM THE USER INPUTS
C     THE DEGREE OF POLYNOMIALS TO BE USED TO DEFINE FEEDBACK PATHS
C     AND THE NUMBER OF DATA POINTS DESIRED.  THE USER THEN INPUTS
C     INITIAL LOADS FOR BOTH SHIFT REGISTERS AND THE COEFFICIENTS OF
C     THE POLYNOMIALS FOR BOTH SHIFT REGISTERS.  AN EXAMPLE OF
C     POLYNOMIALS WHICH MAY BE USED TO GENERATE A GOLD CODE IS:
C       F(X)=X**5+X**2+1 AND G(X)=X**5+X**4+X**3+X**2+1
C       (SEE REF.5 FOR CRITERIA USED TO SELECT POLYNOMIALS)
C
C     FILES USED IN THIS PROGRAM:
C      4:   M-SEQUENCES AND GOLD CODE GENERATED
C     40:   DATA SEQUENCE GENERATED
C
C
      INTEGER X(15)/15*0./,Y(15)/15*0./,F(15)/15*0./,G(15)/15*0./
      INTEGER BUF1,BUF2,NUM
      REAL A(2000)/2000*0./,B(2000)/2000*0./,C(2000)/2000*0./
      COMPLEX CC(2000)
C
C     INPUT DATA
C
      WRITE (6,1)
      READ(5,*) N
      WRITE(6,2)
      READ(5,*) NUM
C
C     PUT INITIAL LOAD IN FEEDBACK SHIFT REGISTERS
C
      DO 10 I=1,N
         J=I-1
         WRITE(6,3) J
         READ(5,*) X(I)
 10      CONTINUE
C
      DO 11 I=1,N
         J=I-1
         WRITE(6,4) J
         READ(5,*) Y(I)
 11      CONTINUE
C
```

```
C     INPUT POLYNOMIAL COEFFICIENTS FOR BOTH FSR'S
C
      DO 20 I=1,N
         J=I-1
         WRITE(6,5) J
         READ(5,*) F(I)
 20      CONTINUE
C
      DO 21 I=1,N
         J=I-1
         WRITE(6,6) J
         READ(5,*) G(I)
 21      CONTINUE
C
C     GENERATE GOLD CODE
C
C     COMPUTE M-SEQUENCE NR.1
C
      DO 100 I=1,NUM
         A(I)=X(1)
         BUF1=X(1)
         DO 110 J=2,N
            IF(F(J).EQ.1) GO TO 111
            GO TO 110
c
C           THE NEXT 4 STATEMENTS PERFORM MOD 2 ADDITION
c
 111        IF(X(J).EQ.BUF1) GO TO 112
            IF(X(J).NE.BUF1) BUF1=1
            GO TO 110
 112        BUF1=0
 110        CONTINUE
C
C        SHIFT THE SHIFT REGISTER
C
         DO 120 J=2,N
            K=J-1
            X(K)=X(J)
 120        CONTINUE
         X(N)=BUF1
 100     CONTINUE
C
C     COMPUTE M-SEQUENCE NR.2
C
      DO 200 I=1,NUM
         B(I)=Y(1)
         BUF2=Y(1)
         DO 210 J=2,N
            IF(G(J).EQ.1) GO TO 211
            GO TO 210
```

```
C
C           THE NEXT 4 STATEMENTS PREFORM MOD 2 ADDITION
C
 211        IF(Y(J).EQ.BUF2) GO TO 212
            IF(Y(J).NE.BUF2) BUF2=1
            GO TO 210
 212        BUF2=0
 210        CONTINUE
C
C         SHIFT THE SHIFT REGISTER
C
          DO 220 J=2,N
            K=J-1
            Y(K)=Y(J)
 220        CONTINUE
          Y(N)=BUF2
 200      CONTINUE
C
C       SUM M-SEQ NR.1 & M-SEQ NR.2 TO FORM GOLD CODE
C
        DO 300 I=1,NUM
          IF(A(I).EQ.B(I)) C(I)=0
          IF(A(I).NE.B(I)) C(I)=1
 300      CONTINUE
C
C       WRITE THE GOLD CODE INTO FILE 40
C
        DO 400 I=1,NUM
          CC(I)=CMPLX(C(I),0.0)
          WRITE(40,500) I,CC(I)

 400      CONTINUE
        DO 401 I=1,75
          WRITE(4,501) I,A(I),B(I),C(I)
 401      CONTINUE
C
 1      FORMAT(' ','INPUT DEGREE OF POLYNOMIALS')
 2      FORMAT(' ','INPUT NUMBER OF POINTS DESIRED')
 3      FORMAT('1','INPUT THE',I2,' STAGE INITIAL LOAD FOR FSR 1')
 4      FORMAT('1','INPUT THE',I2,' STAGE INITIAL LOAD FOR FSR 2')
 5      FORMAT('1','INPUT THE',I2,' POWER COEFFICIENT FOR FSR 1')
 6      FORMAT('1','INPUT THE',I2,' POWER COEFFICIENT FOR FSR 2')
 500    FORMAT(I4,F12.5,F12.5)
 501    FORMAT(I4,2X,'A= ',F3.1,2X,'B= ',F3.1,2X,'C= ',F3.1)
        STOP
        END
```

APPENDIX C

PROGRAM TO GENERATE A FULL SEQUENCE

```
C     PURPOSE:  THE PURPOSE OF THIS PROGRAM IS TO GENERATE FULL SEQUENCE
C     DATA.  A PERIOD 16 FULL SEQUENCE IS FORMED BY ADDING A ZERO TO
C     THE PERIOD 15 M-SEQUENCE GENERATED USING F(X)=X**4+x+1.
C
C     FILE USED IN THIS PROGRAM:
C     40:  DATA SEQUENCE GENERATED
C
      REAL A(2000)/2000*0.0/
      COMPLEX AC(2000)
      DO 10 J=1,1024,16
         I=J
         A(I)=0.0
         I=I+1
         A(I)=0.0
         I=I+1
         A(I)=0.0
         I=I+1
         A(I)=0.0
         I=I+1
         A(I)=1.0
         I=I+1
         A(I)=0.0
         I=I+1
         A(I)=0.0
         I=I+1
         A(I)=1.0
         I=I+1
         A(I)=1.0
         I=I+1
         A(I)=0.0
         I=I+1
         A(I)=1.0
         I=I+1
         A(I)=0.0
         I=I+1
         A(I)=1.0
         I=I+1
         A(I)=1.0
         I=I+1
         A(I)=1.0
         I=I+1
         A(I)=1.0
10       CONTINUE
```

```
      DO 11 I=1,1024
         AC(I)=CMPLX(A(I),0.0)
         WRITE(40,500) I,AC(I)
11       CONTINUE
500   FORMAT(I4,F12.5,F12.5)
      STOP
      END
```

APPENDIX D

PROGRAM FOR RANDOM BIT ERRORS

```
C     PURPOSE:  THE PURPOSE OF THIS PROGRAM IS TO INTRODUCE RANDOM BIT
C     ERRORS INTO A DATA SEQUENCE.
C
C     INSTRUCTIONS:  AT THE PROMPTING OF THE PROGRAM THE USER FIRST
C     INPUTS THE POWER OF TWO OF THE FFT DESIRED AND THEN THE ERROR
C     PERCENTAGE DESIRED.  THE PROGRAM THEN COMPUTES THE FFTS OF THE
C     INPUT DATA AND OF THE INPUT DATA WITH NOISE AND PLOTS THE RESULTS.
C
C     FILES USED IN THIS PROGRAM:
C     40:  INPUT DATA
C     42:  INPUT DATA WITH NOISE
C     50:  FFT OF INPUT DATA
C     51:  FFT OF DATA WITH NOISE
C
C     VARIABLE DECLARATIONS
C
      COMPLEX HOFK(2000),TA,TB,INDAT(2000)
      REAL RNO,ERPCT,MH1(2000)/2000*0./,MH2(2000)/2000*0./
      REAL XH(2000)/2000*0./
      REAL MAX1,MAX2,XMIN,XMAX,XINC,XLIM,DC1,DC2
      INTEGER ERR,IWK(20)
      DOUBLE PRECISION DSEED
      CHARACTER IANS
C
      CALL TEK618
      CALL PAGE(11.0,8.5)
      CALL MX1ALF('STANDARD','&')
      CALL MX2ALF('L/CSTD','#')
      CALL MX3ALF('L/CGREEK','@')
C
C     INITIALIZE VARIABLES
C
1000  WRITE(6,1)
      READ(5,*) M
      N=2**M
      WRITE(6,2)
      READ(5,*) ERPCT
      WRITE(6,3)
      READ(5,4) IANS
      IF (IANS.EQ.'Y') GO TO 1000
C
C     INPUT DATA
C
```

```
      DO 100 I=1,N
         READ(40,500) HOFK(I)
         INDAT(I)=HOFK(I)
 100     CONTINUE
C
C     INTRODUCE ERRORS
C
      TA=CMPLX(0.0,0.0)
      TB=CMPLX(1.0,0.0)
      ERR=0
      DSEED=123547.D0
      DO 120 I=1,N
         RNO=GGUBFS(DSEED)
         RNO=RNO*100
         IF (RNO.GT.ERPCT) GO TO 125
         IF (HOFK(I).EQ.TA) GO TO 121
         IF (HOFK(I).EQ.TB) GO TO 122
121      HOFK(I)=TB
         ERR=ERR+1
         GO TO 125
122      HOFK(I)=TA
         ERR=ERR+1
125      CONTINUE
120      CONTINUE
C
C     OUTPUT DATA WITH ERRORS
C
 200  DO 142 I=1,N
C        WRITE(42,502) I,HOFK(I)
 142     CONTINUE
C
C     COMPUTE FFTS
C
      CALL FFT2C(INDAT,M,IWK)
      CALL FFT2C(HOFK,M,IWK)
      DO 201 I=1,N
         INDAT(I)=CONJG(INDAT(I))
         HOFK(I)=CONJG(HOFK(I))
         MH1(I)=CABS(INDAT(I))
         MH2(I)=CABS(HOFK(I))
         XH(I)=I/1.0
201      CONTINUE
C
C     OUTPUT FFT MAGNITUDES
C
      DO 202 I=1,N
C        WRITE(50,501) I,MH1(I)
         WRITE(51,501) I,MH2(I)
202      CONTINUE
C
```

```
C     DISSPLA PLOTTING ROUTINE
C     (DISSPLA IS A TRADEMARK FOR DISPLAY INTEGRATED SOFTWARE SYSTEM
C     AND PLOTTING LANGUAGE, A PRODUCT OF THE INTEGRATED SOFTWARE
C     SYSTEMS CORPORATION)
C
      DC1=MH1(1)
      DC2=MH2(1)
      DO 399 I=2,N
         J=I-1
         MH1(J)=MH1(I)
         MH2(J)=MH2(I)
399      CONTINUE
      MH1(N)=0.0
      MH2(N)=0.0
      XMIN=0
      XMAX=N/2.0
      XINC=16
      XLIM=XINC/4.0
      MAX1=0.0
      MAX2=0.0
      DO 300 I=1,N
         IF (MH1(I).GT.MAX1) MAX1=MH1(I)
         IF (MH2(I).GT.MAX2) MAX2=MH2(I)
300      CONTINUE
      CALL AREA2D(9.5,6.5)
      CALL XINTAX
      CALL XNAME('B#IN&$',100)
      CALL YNAME('M#AGNITUDE&$',100)
      CALL HEADIN('FFT MAGNITUDE$',100,-1.2,4)
      CALL HEADIN('(INPUT DATA)$',100,1.2,4)
      CALL HEADIN('      $',100,1.2,4)
      CALL HEADIN('      $',100,1.2,4)
      CALL GRAF(XMIN,XINC,XMAX,0.0,'SCALE',MAX1)
      CALL RLMESS('#K&$',100,260.5,0.0)
      DO 301 I=1,XMAX
         CALL RLVEC(XH(I),0.0,XH(I),MH1(I),0)
301      CONTINUE
      CALL ENDPL(0)
      CALL AREA2D(9.5,6.5)
      CALL XINTAX
      CALL XNAME('B#IN&$',100)
      CALL YNAME('M#AGNITUDE&$',100)
      CALL HEADIN('FFT MAGNITUDE$',100,-1.2,4)
      CALL HEADIN('(DATA WITH NOISE)$',100,1.2,4)
      CALL HEADIN('      $',100,1.2,4)
      CALL HEADIN('      $',100,1.2,4)
      CALL HEIGHT(0.12)
      CALL MESSAG('ERROR RATE:$',100,0.5,7.0)
      CALL REALNO(ERPCT,1,'ABUT','ABUT')
      CALL MESSAG(' PCT$',100,'ABUT','ABUT')
```

```
      CALL MESSAG('TOTAL NUM. ERRORS:$',100,6.5,7.0)
      CALL INTNO(ERR,'ABUT','ABUT')
      CALL RESET('HEIGHT')
      CALL GRAF(XMIN,XINC,XMAX,0.0,'SCALE',MAX2)
      CALL RLMESS('#K&$',100,260.5,0.0)
      DO 302 I=1,XMAX
         CALL RLVEC(XH(I),0.0,XH(I),MH2(I),0)
302      CONTINUE
      CALL ENDPL(0)
      CALL DONEPL
1     FORMAT('1','ENTER M, THE POWER OF TWO OF THE FFT')
2     FORMAT('-','ENTER ERROR PERCENTAGE')
3     FORMAT('-','ANY CHANGES? Y OR N')
4     FORMAT(A)
500   FORMAT(4X,F12.5,F12.5)
501   FORMAT(I4,F12.5,F12.5)
502   FORMAT(I4,F12.5,F12.5)
      STOP
      END
```

APPENDIX E

PROGRAM FOR BURST ERRORS

```
C       PURPOSE:  THE PURPOSE OF THIS PROGRAM IS TO INSERT RANDOM BURSTS
C       OF ERRORS INTO A DATA SEQUENCE.
C
C       INSTRUCTIONS:  AT THE PROMPTING OF THE PROGRAM THE USER INPUTS
C       THE POWER OF TWO OF THE DESIRED FFT, THE LENGTH OF BURST, THE
C       PROBABILITY OF A BURST OCCURING AND THE PROBABILITY OF BIT
C       ERRORS WITHIN A BURST.  THE PROGRAM THEN COMPUTES THE FFT OF THE
C       DATA WITH ERRORS AND PLOTS THE RESULT.
C
C       FILES USED IN THE PROGRAM:
C       40:  INPUT DATA
C       41:  INPUT DATA WITH ERRORS
C       50:  FFT OF DATA WITH ERRORS
C
C       VARIABLE DECLARATIONS
        COMPLEX HOFK(2000),TA,TB
        REAL RN1,RN2,ERP1,ERP2,MH1(2000)/2000*0./,XH(2000)/2000*0./
        REAL MAX1,XMIN,XMAX,XINC
        INTEGER NOERR,IWK(20)
        DOUBLE PRECISION DSEED
        CHARACTER IANS
C
        CALL TEK618
        CALL PAGE(11.0,8.5)
        CALL MX1ALF('STANDARD','&')
        CALL MX2ALF('L/CSTD','#')
        CALL MX3ALF('L/CGREEK','@')
C
C       INITIALIZE VARIABLES
C
 10     WRITE(6,1)
        READ(5,*) M
        N=2**M
        WRITE(6,2)
        READ(5,*) L
        WRITE(6,3)
        READ(5,*) ERP1
        WRITE(6,4)
        READ(5,*) ERP2
        WRITE(6,5)
        READ(5,6) IANS
        IF (IANS.EQ.'Y') GO TO 10
C
```

```
C      INPUT DATA
C
       DO 20 I=1,N
          READ(40,500) HOFK(I)
 20       CONTINUE
C
C      INTRODUCE BURSTS OF ERRORS
C
       TA=CMPLX(0.0,0.0)
       TB=CMPLX(1.0,0.0)
       NOERR=0
       DSEED=123547.D0
       I=1
 100   IF (I.GT.N) GO TO 199
          RN1=GGUBFS(DSEED)
          RN1=RN1*100
          IF (RN1.LE.ERP1) GO TO 110
          I=I+1
          GO TO 100
C
C      INSERT BURST OF ERRORS
C
C      CHANGE FIRST BIT OF BURST
C
 110   IF (HOFK(I).EQ.TA) THEN
          HOFK(I)=TB
       ELSE
          HOFK(I)=TA
       END IF
       NOERR=NOERR+1
C
C      BURST OF HIGH PROBABILITY OF ERROR
C
       DO 120 J=I+1,I+L-2
          RN2=GGUBFS(DSEED)
          RN2=RN2*100
          IF (RN2.GE.ERP2) GO TO 120
             IF(HOFK(J).EQ.TA) THEN
                HOFK(J)=TB
             ELSE
                HOFK(J)=TA
             END IF
             NOERR=NOERR+1
 120   CONTINUE
C
C      CHANGE LAST BIT OF BURST
C
       K=I+L-1
       IF (HOFK(K).EQ.TA) THEN
          HOFK(K)=TB
```

```
      ELSE
         HOFK(K)=TA
      END IF
      NOERR=NOERR+1
      I=I+L
      GO TO 100
 199  CONTINUE
C
C     OUTPUT NUMBER OF ERRORS AND SEQUENCE WITH ERRORS
C
      WRITE(6,501) NOERR
      DO 200 I=1,N
         WRITE(41,502) I,HOFK(I)
 200     CONTINUE
C
C     COMPUTE FFT
C
      CALL FFT2C(HOFK,M,IWK)
      DO 210 I=1,N
         HOFK(I)=CONJG(HOFK(I))
         MH1(I)=CABS(HOFK(I))
         XH(I)=I/1.0
210      CONTINUE
      DO 220 I=1,N
         WRITE(50,503) I,MH1(I)
220      CONTINUE
C
C     DISSPLA PLOTTING ROUTINE
C     (DISSPLA IS A TRADEMARK FOR DISPLAY INTEGRATED SOFTWARE SYSTEM
C     AND PLOTTING LANGUAGE, A PRODUCT OF THE INTEGRATED SOFTWARE
C     SYSTEMS CORPORATION)
C
      DO 300 I=2,N
         J=I-1
         MH1(J)=MH1(I)
300      CONTINUE
      MH1(N)=0.0
      XMIN=0
      XMAX=N/2.0
      XINC=16
      MAX1=0.0
      DO 310 I=1,N
         IF (MH1(I).GT.MAX1) MAX1=MH1(I)
310      CONTINUE
      CALL AREA2D(9.5,6.5)
      CALL XINTAX
      CALL XNAME('C#OMPONENT&$',100)
      CALL YNAME('M#AGNITUDE&$',100)
      CALL HEADIN('FFT MAGNITUDE$',100,-1.2,4)
      CALL HEADIN('                $',100,1.2,4)
```

```
      CALL HEADIN('      $',100,1.2,4)
      CALL HEADIN('      $',100,1.2,4)
      CALL GRAF(XMIN,XINC,XMAX,0.0,'SCALE',MAX1)
      CALL RLMESS('#K&$',100,260.5,0.0)
      DO 320 I=1,XMAX
         CALL RLVEC(XH(I),0.0,XH(I),MH1(I),0)
320      CONTINUE
      CALL ENDPL(0)
      CALL DONEPL
1     FORMAT('1','ENTER N, THE POWER OF TWO OF THE FFT')
2     FORMAT('-','ENTER L, THE LENGTH OF BURST')
3     FORMAT('-','ENTER ERROR PERCENTAGE FOR OCCURANCE OF BURST')
4     FORMAT('-','ENTER ERROR PERCENTAGE WITHIN BURST')
5     FORMAT('-','ANY CHANGES? Y OR N')
6     FORMAT(A)
500   FORMAT(4X,F12.5,F12.5)
501   FORMAT(1H ,17HNUMBER OF ERRORS:,I4)
502   FORMAT(I4,F12.5,F12.5)
503   FORMAT(I4,F12.5,F12.5)
      STOP
      END
```

APPENDIX E

PROGRAM FOR RANDOM DELETIONS

```
C       PURPOSE: THE PURPOSE OF THIS PROGRAM IS TO SIMULATE TIMING ERRORS
C       THAT CAUSE BITS IN THE DATA STREAM TO BE DELETED AT RANDOM.
C
C       INSTRUCTIONS: THE USER SETS THE MAXIMUM BIT TIMING ERROR ALLOWED,
C       ASSUMING A BIT PERIOD OF ONE AND THAT SAMPLING OCCURS AT THE
C       MIDDLE OF THE BIT PERIOD.  THE PROGRAM SELECTS A RANDOM ERROR
C       BETWEEN ZERO AND THE MAXIMUM SPECIFIED AND APPLIES IT TO EACH
C       SAMPLING TIME. WHEN A SAMPLE IS TAKEN OUTSIDE THE BIT PERIOD, THE
C       BIT IS DELETED.  THE PROGRAM THEN COMPUTES THE FFT OF THE DATA
C       WITH ERRORS AND PLOTS THE RESULTS.
C
C       FILES USED IN THE PROGRAM:
C       40:  INPUT DATA
C       41:  INPUT DATA WITH ERRORS
C       50:  FFT OF DATA WITH ERRORS
C
C       VARIABLE DECLARATIONS
C
        COMPLEX HOFK(2000)
        REAL MXERR,MH1(2000),MAX1
        REAL XH(2000),XMIN,XMAX,XINC,TS,RN
        INTEGER NOCUT,TN
        DOUBLE PRECISION DSEED
        CHARACTER IANS
C
        CALL TEK618
        CALL PAGE(11.0,8.5)
        CALL MX1ALF('STANDARD','&')
        CALL MX2ALF('L/CSTD','#')
C
C       INITIALIZE VARIABLES
C
1000    WRITE(6,1)
        READ(5,*) M
        WRITE(6,2)
        READ(5,*) MXERR
        WRITE(6,3)
        READ(5,4) IANS
        IF (IANS.EQ.'Y') GO TO 1000
        N=2**M
        TN=N+(N/2)
        DO 100 I=1,N*1.75
           READ(40,500) HOFK(I)
```

```
 100     CONTINUE
C
C     INTRODUCE ERRORS
C
      NOCUT=0
      TS=0.5
      DSEED=123478.DO
      DO 50 I=1,TN
         RN=GGUBFS(DSEED)
         RN=RN*MXERR
         TS=TS+RN
         IF (TS.LE.1) GO TO 50
         DO 60 J=I,TN
            HOFK(J)=HOFK(J+1)
 60         CONTINUE
         NOCUT=NOCUT+1
         TS=TS-1.0
 50      CONTINUE
C
C     CHECK NUMBER OF BITS CUT AND OUTPUT DATA WITH ERRORS
C
      WRITE(6,503) NOCUT
      DO 70 I=1,N
         WRITE(41,501) I,HOFK(I)
 70      CONTINUE
C
C     COMPUTE FFT
C
      CALL FFT2C(HOFK,M,IWK)
      DO 201 I=1,N
         HOFK(I)=CONJG(HOFK(I))
         MH1(I)=CABS(HOFK(I))
         XH(I)=I/1.0
201      CONTINUE
      DO 202 I=2,N
         J=I-1
         MH1(J)=MH1(I)
202      CONTINUE
C
C     OUTPUT FFT OF DATA WITH ERRORS
C
      MH1(N)=0.0
      DO 203 I=1,N
         WRITE(50,502) I,MH1(I)
 203     CONTINUE
C
C     DISSPLA PLOTTING ROUTINE
C     (DISSPLA IS A TRADEMARK FOR DISPLAY INTEGRATED SOFTWARE SYSTEM
C     AND PLOTTING LANGUAGE, A PRODUCT OF THE INTEGRATED SOFTWARE
C     SYSTEMS CORPORATION)
```

```
C
      XMIN=0
      XMAX=N/2.0
      XINC=16
      MAX1=0.0
      DO 300 I=1,XMAX
         IF (MH1(I).GT.MAX1) MAX1=MH1(I)
300      CONTINUE
      CALL AREA2D(9.5,6.5)
      CALL XINTAX
      CALL XNAME('C#OMPONENT&$',100)
      CALL YNAME('M#AGNITUDE&$',100)
      CALL HEADIN('FFT MAGNITUDE$',100,-1.2,4)
      CALL HEADIN('     $',100,1.2,4)
      CALL HEADIN('     $',100,1.2,4)
      CALL HEADIN('     $',100,1.2,4)
      CALL MESSAG('MAX BIT ERROR:$',100,0.5,7.0)
      CALL REALNO(MXERR,2,'ABUT','ABUT')
      CALL MESSAG('TOTAL NUMBER OF DELETIONS:$',100,6.0,7.0)
      CALL INTNO(NOCUT,'ABUT','ABUT')
      CALL GRAF(XMIN,XINC,XMAX,0.0,'SCALE',MAX1)
      CALL RLMESS('#K&$',100,260.5,0.0)
      DO 301 I=1,XMAX
         CALL RLVEC(XH(I),0.0,XH(I),MH1(I),0)
301      CONTINUE
      CALL ENDPL(0)
      CALL DONEPL
1     FORMAT('1','ENTER POWER OF TWO OF FFT')
2     FORMAT('-','ENTER MAX ERROR')
3     FORMAT('-','ANY CHANGES? Y OR N')
4     FORMAT(A)
5     FORMAT('-','CHECK OF NO. CUTS AND MAX ERROR')
500   FORMAT(4X,F12.5,F12.5)
501   FORMAT(I4,F12.5,F12.5)
502   FORMAT(I4,F12.5)
503   FORMAT(1H ,17HNO. BITS DELETED:,I3)
      STOP
      END
```

APPENDIX G

PROGRAM FOR RANDOM INSERTIONS

```
C       PURPOSE:  THE PURPOSE OF THIS PROGRAM IS TO SIMULATE TIMING ERRORS
C       THAT CAUSE BITS IN THE DATA STREAM TO BE INSERTED AT RANDOM.
C
C       INSTRUCTIONS:  AT THE PROMPTING OF THE PROGRAM, THE USER SETS THE
C       MAXIMUM BIT TIMING ERROR ALLOWED, ASSUMING A BIT PERIOD OF ONE AND
C       THAT SAMPLING OCCURS AT THE MIDDLE OF THE BIT PERIOD.  THE PROGRAM
C       SELECTS A RANDOM ERROR BETWEEN ZERO AND THE MAXIMUM AND APPLIES
C       IT TO EACH SAMPLING TIME.  WHEN A SAMPLE IS TAKEN INSIDE THE SAME
C       BIT PERIOD, THE BIT IS INSERTED.  THE PROGRAM THEN COMPUTES THE
C       FFT OF THE DATA WITH ERRORS AND PLOTS THE RESULTS.
C
C       FILES USED IN THE PROGRAM:
C       40:  INPUT DATA (TIME SAMPLES)
C       41:  INPUT DATA WITH ERRORS
C       50:  FFT MAGNITUDE
C
C       VARIABLE DECLARATIONS
        COMPLEX HOFK(2000),TEMP(2000)
        REAL MXERR,MH1(2000),MAX1
        REAL XH(2000),XMIN,XMAX,XINC,TS,RN
        INTEGER NOADD,TN
        DOUBLE PRECISION DSEED
        CHARACTER IANS
C
        CALL TEK618
        CALL PAGE(11.0,8.5)
        CALL MX1ALF('STANDARD','&')
        CALL MX2ALF('L/CSTD','#')
C
C       INITIALIZE VARIABLES
C
1000    WRITE(6,1)
        READ(5,*) M
        WRITE(6,2)
        READ(5,*) MXERR
        WRITE(6,3)
        READ(5,4) IANS
        IF (IANS.EQ.'Y') GO TO 1000
        N=2**M
        TN=N+(N/2)
        DO 100 I=1,N*1.75
           READ(40,500) HOFK(I)
 100       CONTINUE
```

```
C
C     INTRODUCE ERRORS
C
      NOADD=0
      TS=0.5
      DSEED=123478.DO
      DO 50 I=1,TN
         RN=GGUBFS(DSEED)
         RN=RN*MXERR
         TS=TS-RN
         IF (TS.GE.0) GO TO 50
         DO 60 J=I,TN
            TEMP(J+1)=HOFK(J)
 60         CONTINUE
         DO 61 J=I+1,TN
            HOFK(J)=TEMP(J)
 61         CONTINUE
         NOADD=NOADD+1
         TS=TS+1.0
 50      CONTINUE
C
C     CHECK NUMBER OF BITS INSERTED AND OUTPUT DATA WITH ERRORS
C
      WRITE(6,503) NOADD
      DO 70 I=1,N
         WRITE(41,501) I,HOFK(I)
 70      CONTINUE
C
C     COMPUTE FFT
C
      CALL FFT2C(HOFK,M,IWK)
      DO 201 I=1,N
         HOFK(I)=CONJG(HOFK(I))
         MH1(I)=CABS(HOFK(I))
         XH(I)=I/1.0
201      CONTINUE
      DO 202 I=2,N
         J=I-1
         MH1(J)=MH1(I)
202      CONTINUE
      MH1(N)=0.0
      DO 203 I=1,N
         WRITE(50,502) I,MH1(I)
 203     CONTINUE
C
C     DISSPLA PLOTTING ROUTINE
C     (DISSPLA IS A TRADEMARK FOR DISPLAY INTEGRATED SOFTWARE SYSTEM
C     AND PLOTTING LANGUAGE, A PRODUCT OF THE INTEGRATED SOFTWARE
C     SYSTEMS CORPORATION)
C
```

```
      XMIN=0
      XMAX=N/2.0
      XINC=16
      MAX1=0.0
      DO 300 I=1,XMAX
         IF (MH1(I).GT.MAX1) MAX1=MH1(I)
300      CONTINUE
      CALL AREA2D(9.5,6.5)
      CALL XINTAX
      CALL XNAME('C#OMPONENT&$',100)
      CALL YNAME('M#AGNITUDE&$',100)
      CALL HEADIN('FFT MAGNITUDE$',100,-1.2,4)
      CALL HEADIN('      $',100,1.2,4)
      CALL HEADIN('      $',100,1.2,4)
      CALL HEADIN('      $',100,1.2,4)
      CALL MESSAG('MAX BIT ERROR:$',100,0.5,7.0)
      CALL REALNO(MXERR,2,'ABUT','ABUT')
      CALL MESSAG('TOTAL NUMBER OF INSERTIONS:$',100,6.0,7.0)
      CALL INTNO(NOADD,'ABUT','ABUT')
      CALL GRAF(XMIN,XINC,XMAX,0.0,'SCALE',MAX1)
      CALL RLMESS('#K&$',100,260.5,0.0)
      DO 301 I=1,XMAX
         CALL RLVEC(XH(I),0.0,XH(I),MH1(I),0)
301      CONTINUE
      CALL ENDPL(0)
      CALL DONEPL
1     FORMAT('1','ENTER POWER OF TWO OF FFT')
2     FORMAT('-','ENTER MAX ERROR')
3     FORMAT('-','ANY CHANGES? Y OR N')
4     FORMAT(A)
500   FORMAT(4X,F12.5,F12.5)
501   FORMAT(I4,F12.5,F12.5)
502   FORMAT(I4,F12.5)
503   FORMAT(1H ,15HNO. BITS ADDED:,I3)
      STOP
      END
```

APPENDIX H

PROGRAM FOR TIMING ERRORS

```
C     PURPOSE: THE PURPOSE OF THIS PROGRAM IS TO SIMULATE TIMING ERRORS
C     THAT CAUSE BITS TO BE DELETED OR INSERTED IN A RANDOM FASHION.
C
C     INSTRUCTIONS:  AT THE PROMPTING OF THE PROGRAM, THE USER SETS THE
C     MAXIMUM BIT TIMING ERROR ALLOWED, ASSUMING A BIT PERIOD OF ONE
C     AND THAT SAMPLING OCCURS AT THE MIDDLE OF THE BIT PERIOD.  THE
C     PROGRAM SELECTS A RANDOM ERROR BETWEEN ZERO AND THE MAXIMUM AND
C     APPLIES IT TO EACH SAMPLING TIME. WHEN A SAMPLE IS TAKEN OUTSIDE
C     THE BIT PERIOD, THE BIT IS DELETED.  WHEN A SAMPLE IS TAKEN
C     WITHIN THE PREVIOUS BIT PERIOD, THAT BIT IS INSERTED.  THE PROGRAM
C     THEN COMPUTES THE FFT AND PLOTS THE RESULTS.
C
C     FILES USED IN THE PROGRAM:
C     40:  INPUT DATA
C     41:  INPUT DATA WITH ERRORS
C     50:  FFT MAGNITUDE
C
C     VARIABLE DECLARATIONS
      COMPLEX HOFK(2000),TEMP(2000)
      REAL MXERR,MH1(2000),MAX1
      REAL XH(2000),XMIN,XMAX,XINC,TS,RN
      INTEGER NOCUT,TN,NOADD
      DOUBLE PRECISION DSEED
      CHARACTER IANS
C
      CALL TEK618
      CALL PAGE(11.0,8.5)
      CALL MX1ALF('STANDARD','&')
      CALL MX2ALF('L/CSTD','#')
C
C     INITIALIZE VARIABLES
C
1000  WRITE(6,1)
      READ(5,*) M
      WRITE(6,2)
      READ(5,*) MXERR
      WRITE(6,3)
      READ(5,4) IANS
      IF (IANS.EQ.'Y') GO TO 1000
      N=2**M
      TN=N+(N/2)
      DO 100 I=1,N*1.75
         READ(40,500) HOFK(I)
```

```
100       CONTINUE
C
C      INTRODUCE ERRORS
C
       NOCUT=0
       NOADD=0
       TS=0.5
       DSEED=123478.D0
       DO 50 I=1,TN
          RN=GGUBFS(DSEED)
          IF (RN.GE.0.5) GO TO 60
          GO TO 65
 60       RN=(RN-0.5)*MXERR
          TS=TS+RN
          IF (TS.LE.1) GO TO 50
          DO 61 J=I,TN
             HOFK(J)=HOFK(J+1)
 61          CONTINUE
          NOCUT=NOCUT+1
          TS=TS-1.0
          GO TO 50
 65       RN=(0.5-RN)*MXERR
          TS=TS-RN
          IF(TS.GE.0) GO TO 50
          DO 66 J=I,TN
             TEMP(J+1)=HOFK(J)
 66          CONTINUE
          DO 67 J=I+1,TN
             HOFK(J)=TEMP(J)
 67          CONTINUE
          NOADD=NOADD+1
          TS=TS+1
 50       CONTINUE
C
C      CHECK NUMBER OF BITS CUT AND ADDED AND OUTPUT DATA WITH ERRORS
C
       WRITE(6,503) NOCUT
       WRITE(6,504) NOADD
       DO 70 I=1,N
          WRITE(41,501) I,HOFK(I)
 70       CONTINUE
C
C      COMPUTE FFT
C
       CALL FFT2C(HOFK,M,IWK)
       DO 201 I=1,N
          HOFK(I)=CONJG(HOFK(I))
          MH1(I)=CABS(HOFK(I))
          XH(I)=I/1.0
201       CONTINUE
```

```
      DO 202 I=2,N
         J=I-1
         MH1(J)=MH1(I)
202      CONTINUE
      MH1(N)=0.0
      DO 203 I=1,N
         WRITE(50,502) I,MH1(I)
 203     CONTINUE
C
C     DISSPLA PLOTTING ROUTINE
C     (DISSPLA IS A TRADEMARK FOR DISPLAY INTEGRATED SOFTWARE SYSTEM
C     AND PLOTTING LANGUAGE, A PRODUCT OF THE INTEGRATED SOFTWARE
C     SYSTEMS CORPORATION)
C
      XMIN=0
      XMAX=N/2.0
      XINC=16
      MAX1=0.0
      DO 300 I=1,XMAX
         IF (MH1(I).GT.MAX1) MAX1=MH1(I)
300      CONTINUE
      CALL AREA2D(9.5,6.5)
      CALL XINTAX
      CALL XNAME('C#OMPONENT&$',100)
      CALL YNAME('M#AGNITUDE&$',100)
      CALL HEADIN('FFT MAGNITUDE$',100,-1.2,1)
      CALL MESSAG('MAX BIT ERROR:$',100,0.5,7.0)
      CALL REALNO(MXERR,2,'ABUT','ABUT')
      CALL MESSAG('TOTAL NUMBER OF DELETIONS:$',100,6.0,7.0)
      CALL INTNO(NOCUT,'ABUT','ABUT')
      CALL GRAF(XMIN,XINC,XMAX,0.0,'SCALE',MAX1)
      CALL RLMESS('#K&$',100,260.5,0.0)
      DO 301 I=1,XMAX
         CALL RLVEC(XH(I),0.0,XH(I),MH1(I),0)
301      CONTINUE
      CALL ENDPL(0)
      CALL DONEPL
1     FORMAT('1','ENTER POWER OF TWO OF FFT')
2     FORMAT('-','ENTER MAX ERROR')
3     FORMAT('-','ANY CHANGES? Y OR N')
4     FORMAT(A)
500   FORMAT(4X,F12.5,F12.5)
501   FORMAT(I4,F12.5,F12.5)
502   FORMAT(I4,F12.5)
503   FORMAT(1H ,13HNO. BITS CUT:,I3)
504   FORMAT(1H ,18HNO. BITS INSERTED:,I3)
      STOP
      END
```

APPENDIX I

PROGRAM FOR FISHER'S TEST

```
C     PURPOSE: THE PURPOSE OF THIS PROGRAM IS TO APPLY FISHER'S TEST
C     OF SIGNIFICANCE IN HARMONIC ANALYSIS.
C
C     INSTRUCTIONS: AT THE PROMPTING OF THE PROGRAM, THE USER INPUTS THE
C     NUMBER OF DATA POINTS AND THE NUMBER OF SIGNIFICANCE CALCULATIONS
C     DESIRED.  THE PROGRAM CALCULATES SIGNIFICANCE MEASURES AND PLOTS
C     THE SPECTRUM LABELED WITH APPROPRIATE SIGNIFICANCE MEASURES.
C     A TABLE OF OUTPUT DATA IS ALSO PROVIDED.
C
C     FILES USED IN THIS PROGRAM:
C      8: TABULAR OUTPUT
C     51: INPUT DATA (FFT MAGNITUDES OF DATA SEQUENCE)
C
      INTEGER I,J,K,L,M,N,R,NUM,KEY1(25),KEY2(25),KP(25)
      REAL G(512),SG(25)/25*0.0/,P(25),SP(25)/25*1.0/,MEANC
      REAL PROB(25),PER,C(512),SUMC,DC,LIM
      REAL XMIN,XMAX,XINC,MH1(512),MAX1,XH(512)
      REAL*16 PROD,SUM,FACT
C
      CALL TEK618
      CALL PAGE(11.0,8.5)
      CALL MX1ALF('STANDARD','&')
      CALL MX2ALF('L/CSTD','#')
C
C     INITIALIZE VARIABLES
C
      WRITE(6,1)
      READ(5,*) N
      M=(N-2)/2.0
      WRITE(6,2)
      READ(5,*) NUM
C
C     INPUT C'S AND CALCULATE G'S
C
      DO 100 I=1,N
         READ(51,500) C(I)
100      CONTINUE
      DC=C(1)
      DO 101 I=2,N
         J=I-1
         C(J)=C(I)
         MH1(J)=C(J)
         XH(J)=J/1.0
```

```
101      CONTINUE
      C(N)=0.0
      XH(N)=N/1.0
      SUMC=0.0
      MEANC=0.0
      DO 102 I=1,N
         SUMC=SUMC+C(I)**2
         MEANC=MEANC+C(I)
102      CONTINUE
      MEANC=MEANC/(N*1.0)
      WRITE(8,602) MEANC
      DO 103 I=1,N
         G(I)=(C(I)**2)/SUMC
103      CONTINUE
C
C     SORT DESIRED NUMBER OF G'S
C
      DO 110 I=1,NUM
         DO 111 J=1,M
            IF(SG(I).LT.G(J)) GO TO 112
            GO TO 111
112         SG(I)=G(J)
            KEY1(I)=J
111         CONTINUE
         K=KEY1(I)
         G(K)=0.0
110      CONTINUE
      WRITE(8,603)
      DO 113 I=1,NUM
         WRITE(8,599) I,KEY1(I),SG(I)
113      CONTINUE
C
C     CALCULATE PROBABILITIES
C
      LIM=1.0E-50
      DO 115 R=1,NUM
         SUM=0.0
         L=INT(1/SG(R))
         WRITE(8,504) L
         DO 120 J=R,L
            PROD=1.0
            DO 130 K=1,(M-1)
               IF(J.EQ.R) GO TO 131
               IF(K.GT.(J-R)) GO TO 131
               PROD=PROD*(1-(J*SG(R)))*(M-K+1)/((J-R-K+1)*1.0)
               GO TO 130
131            IF(K.GT.J) GO TO 132
               PROD=PROD*(1-(J*SG(R)))*(M-K+1)
               GO TO 130
132            PROD=PROD*(1-(J*SG(R)))
```

```
          IF(PROD.LE.LIM) GO TO 199
 130      CONTINUE
 199    PROD=PROD*((-1)**(J-R))/(J*1.0)
        SUM=SUM+PROD
 120    CONTINUE
      FACT=1.0
      IF(R.EQ.1) GO TO 122
      DO 121 J=1,(R-1)
        FACT=FACT*(R-J)
 121    CONTINUE
 122  P(R)=SUM/FACT
      IF(P(R).GT.1.0) P(R)=1.0
      IF(P(R).LT.0.0) P(R)=1.0
      PROB(R)=P(R)
 115  CONTINUE
C
C     OUTPUT PROBABILITIES TO TABLE
C
      WRITE(8,601)
      DO 116 I=1,NUM
        WRITE(8,600) I,P(I)
 116    CONTINUE
C
C     SORT PROBABILITIES
C
      DO 140 I=1,NUM
        DO 141 J=1,NUM
          IF(SP(I).GT.P(J)) GO TO 142
          GO TO 141
 142      SP(I)=P(J)
          KEY2(I)=J
 141      CONTINUE
        K=KEY2(I)
        P(K)=1.0
 140    CONTINUE
C
C     OUTPUT SORTED PROBABILITIES TO TABLE
C
      WRITE(8,501)
      DO 200 I=1,NUM
        J=KEY2(I)
        K=KEY1(J)
C       PER=N/(K*1.0)
        KP(I)=KEY1(J)
        WRITE(8,502) K,SP(I)
 200    CONTINUE
C
C     DISSPLA PLOTTING ROUTINE
C     (DISSPLA IS A TRADEMARK FOR DISPLAY INTEGRATED SOFTWARE SYSTEM
C     AND PLOTTING LANGUAGE, A PRODUCT OF THE INTEGRATED SOFTWARE
```

```
C      SYSTEMS CORPORATION)
C
       MH1(N)=0.0
       XMIN=0
       XMAX=N/2.0
       XINC=16
       MAX1=0.0
       DO 300 I=1,N
          IF (MH1(I).GT.MAX1) MAX1=MH1(I)
 300      CONTINUE
       CALL AREA2D(9.5,6.5)
       CALL XINTAX
       CALL YINTAX
       CALL XNAME('P#ERIOD &V#ALUES ARE 512/K&$',100)
       CALL YNAME('M#AGNITUDE& S#QUARED&$',100)
       CALL GRAF(XMIN,XINC,XMAX,0.0,'SCALE',MAX1)
       CALL RLMESS('#K&$',100,260.5,0.0)
       DO 301 I=1,XMAX
          CALL RLVEC(XH(I),0.0,XH(I),MH1(I),0)
 301      CONTINUE
       CALL HEIGHT(0.08)
       DO 310 I=1,NUM
          IF(SP(I).GE.1.0) GO TO 310
          IF(SP(I).LT.0.0) GO TO 310
          K=KP(I)
          CALL RLREAL(SP(I),8,K,C(K))
 310      CONTINUE
       CALL RESET('HEIGHT')
       CALL ENDPL(0)
       CALL DONEPL
 1     FORMAT('1','INPUT NUMBER OF DATA POINTS: N')
 2     FORMAT('-','INPUT THE NUMBER OF PROBABILITIES DESIRED')
 500   FORMAT(4X,F12.5)
 501   FORMAT('-','  C      PROBABILITY')
 502   FORMAT(I4,2X,F10.5)
 504   FORMAT(1H ,2HL=,I4)
 599   FORMAT(I3,3X,I3,3X,F12.5)
 600   FORMAT(I4,3X,F12.5)
 601   FORMAT('-','  R          PROBABILITY')
 602   FORMAT('-',22HTHE MEAN OF THE C'S IS,F12.4)
 603   FORMAT('-',' I     K         VALUE OF G')
       STOP
       END
```

LIST OF REFERENCES

1. Fisher, R.A., Tests of Significance in Harmonic Analysis, Proc. R. Soc., A, 125, 1929.

2. Nowroozi, A.A., Table for Fisher's Test of Significance in Harmonic Analysis, Geophys. J.R. Astr. Soc., 12, 1967.

3. Shimshoni, M., On Fisher's Test of Significance in Harmonic Analysis, Geophys. J.R. Astr. Soc., 23, 1971.

4. Golomb, S.W., Shift Register Sequences, Aegean Park Press, 1982.

5. Ziemer, R.E. and Peterson, R.L., Digital Communications and Spread Spectrum Systems, Macmillan Publishing Company, 1985.

6. Bloomfield, P., Fourier Analysis of Time Series: An Introduction, John Wiley and Sons, Inc., 1976.

7. Jenkins, G.M. and Watts, D.G., Spectral Analysis and Its Applications, Holden-Day, 1968.

8. Grenander, U., and Rosenblatt, M., Statistical Analysis of Stationary Time Series, John Wiley and Sons, Inc., 1957.

9. Fisher, R.A., On the Similarity of the Distributions Found for the Test of Significance in Harmonic Analysis, and in Steven's Problem in Geometrical Probability, Annals of Eugenics, 10, 1940.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Superintendent Attn: Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5000	2
3.	Prof. Harold Fredricksen, Code 53Fs Naval Postgraduate School Monterey, California 93943	10
4.	Prof. Charles Therrien, Code 62Ti Naval Postgraduate School Monterey, California 93943	1
5.	Prof. Harriet Rigas, Code 62Rr Naval Postgraduate School Monterey, California 93943	1
6.	Mario C. Neiva Brazilian Naval Commission 4706 Wisconsin Ave., NW Washington, DC, 20016	1
7.	Lt. Jarratt M. Mowery U.S. Naval Ship Repair Facility Box 34 FPO San Francisco, California 96651-1400	3

219557

Thesis
M8653 Mowery
c.1 Analysis of periodic
binary sequences.