AD-A084 179   ARMY MATERIEL DEVELOPMENT AND READINESS COMMAND ADEL--ETC F/G 17/4
A COMPUTER PROGRAM TO ANALYZE SPREAD-SPECTRUM SYSTEM PERFORMANC--ETC(U)
DEC 79 H R HARRELSON
UNCLASSIFIED CM/CCM-79-5 NL

1 OF 1
AD
A084179

END
DATE
FILMED
6 80
DTIC

CM/CCM-79-5

LEVEL

December 1979

ADA084179

A Computer Program to Analyze Spread-Spectrum System Performance

DTIC ELECTE MAY 9 1980 S C

by Hal R. Harrelson

U.S. Army Materiel Development and Readiness Command

Countermeasures/
Counter-countermeasures Office
2800 Powder Mill Road
Adelphi, MD 20783

Editorial review and camera-ready copy by Technical Reports Branch, Harry Diamond Laboratories

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER (14) CM/CCM-79-5	2. GOVT ACCESSION NO. AD-A084179	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) (6) A Computer Program to Analyze Spread-Spectrum System Performance.		5. TYPE OF REPORT & PERIOD COVERED (9) Technical Report.
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) (10) Hal R. Harrelson		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Countermeasures/Counter-countermeasures Office, 2800 Powder Mill Road Adelphi, MD 20783		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Program Ele: 6.37.49.A
11. CONTROLLING OFFICE NAME AND ADDRESS US Army Materiel Development & Readiness Command Alexandria, VA 22333	(11)	12. REPORT DATE December 1979
		13. NUMBER OF PAGES 82 (12) 73
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

DRCMS Code: 643749.4620011
DA Project (16) 1S463749D462
ERADCOM Project: T499TC

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Frequency hopping	Computer analysis
Jamming	Computer graphics
Direct sequence spread spectrum	Electronic warfare
Pseudonoise spread spectrum	Electronic counter-countermeasures (ECCM)
Spread-spectrum communications	

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A computer program has been developed which analyzes and compares the performance of certain types of frequency hopping and pseudonoise (sometimes called "direct sequence") spread-spectrum communication systems in the presence of hostile jamming.

The program computes the probability of bit or word error as a function of the communication signal parameters (spectrum

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

411 -7

20. Abstract (Cont'd)

spreading methods, error-correcting coding, data modulation, bandwidth, etc) and the jamming parameters (jamming modulation, power, bandwidth). The output consists mainly of plots of error probability as a function of one or more of these parameters selected by the user. The plots are produced on an interactive computer graphics terminal, providing rapidly produced results and high user-computer interaction.

CONTENTS

FIGURES

1. INTRODUCTION

Success in modern military warfare depends heavily on maintaining adequate and secure lines of communication. Thus, the attempted interception and jamming of communications by the enemy seems inevitable, and measures must be taken by the communicators to ensure that their lines of communication remain open and secure. One method of decreasing the vulnerability of communication systems to interception and jamming is to employ spread-spectrum techniques. These techniques provide resistance to jamming and interception by distributing the transmitted energy over large bandwidths. The energy density of the transmitted waveform is therefore decreased, yielding a low probability of intercept (LPI) system. In addition, if unsophisticated jamming of at least a significant portion of the signal bandwidth is desired, then the bandwidth over which a jammer must operate is increased, forcing the jammer to dilute its available power. This dilution results in decreased jamming energy density and, consequently, decreased jamming effectiveness.

The quantification of the decrease in jamming effectiveness and the corresponding increase in the intelligibility of the communication signal is not a simple procedure, however. In fact, signal intelligibility is a highly complex function of the signal and jamming parameters, atmospheric and terrain conditions, and a number of other factors. Voice communication intelligibility is particularly difficult to theoretically quantify because it is heavily dependent on the communicators' speaking and listening abilities as well as on message content. Digital data intelligibility, on the other hand, is somewhat simpler to quantify in terms of bit error rates, but the use of bit-interleaving techniques or error-correcting codes in the communication system can complicate the determination of word error rates. Spread-spectrum techniques, because they tend to decrease transmission errors in the presence of jamming or other interference, further complicate the determination of digital message intelligibility.

This paper describes a computer program which analyzes the performance of digital communication systems that employ certain types of spread-spectrum techniques in the presence of interfering signals. The program computes the bit or word error rate as a function of the communication signal parameters (spectrum spreading technique, error-correcting codes, data modulation, bandwidth, etc.) and the interference parameters (thermal noise, jamming power, jamming modulation, jamming bandwidth, etc.). The program output consists mainly of plots of bit or word error rates as functions of one or more of the signal or interference parameters, as determined by the user. The plots are produced interactively on a Tektronix-type storage-tube graphics terminal connected to an IBM 370/168 computer which executes the program. The use of interactive computer graphics provides rapidly produced results and high user-computer interaction.

The computer model, as well as the theoretical treatment[1] on which the model is based, assumes that the spread-spectrum receiver has acquired the intended signal and that the transmitter and receiver are in perfect synchronization. Thus, the effects of jamming on signal acquisition and synchronization are not modeled in this program.

Since the computer model described in this report is based on the theoretical treatments of Torrieri[1,2] and since this report frequently references those treatments, users of this report should have CM/CCM-78-2 and CM/CCM-79-9 readily available.

2. PROGRAM OVERVIEW

The computer program described in this paper was designed to analyze and compare digital communication systems employing two types of spread-spectrum techniques: frequency hopping (FH) and pseudonoise (PN, also known as direct sequence or DS), in environments where considerable interference is present.

Frequency hopping is defined as the periodic changing of the communication signal-carrier frequency throughout a wide band of frequency choices. A system that changes frequency at a rate higher than the data rate of the digital information to be transmitted is called a fast-frequency hopping system. A system that hops slower than the information rate is called a slow-frequency hopping system. Although the bandwidth of a frequency hopping signal at any given instant during transmission is small (aproximately equal to the data rate in slow-hopping systems or the hopping rate in fast-hopping systems), the total bandwidth occupied by the signal over a long period of time is much greater than this "instantaneous" bandwidth because of the hopping carrier frequency (see fig. 1). The resistance of frequency hopping signals to intercept and jamming derives from the large total bandwidth generated by the hopping systems. In order to be effective against most practical hopping systems, a potential interceptor or jammer must either follow the hopping signal in frequency or increase its bandwidth to encompass at least a significant portion of the hopping bandwidth. The former tactic is a difficult task, especially for systems with high hopping rates; the latter tactic results in a reduction in the interceptor's signal-to-noise ratio (making interception more difficult) and also in the amount of jamming power present within the relatively narrow "instantaneous" bandwidth of the hopping signal.

---

[1]D. J. Torrieri, *Frequency Hopping in a Jamming Environment*, U.S. Army Development and Readiness Command, DARCOM Report CM/CCM-78-2 (December 1978).

[2]D. J. Torrieri, *Pseudonoise Spread-Spectrum Systems in Communication Warfare*, U.S. Army Development and Readiness Command, DARCOM Report CM/CCM-79-9 (December 1979).

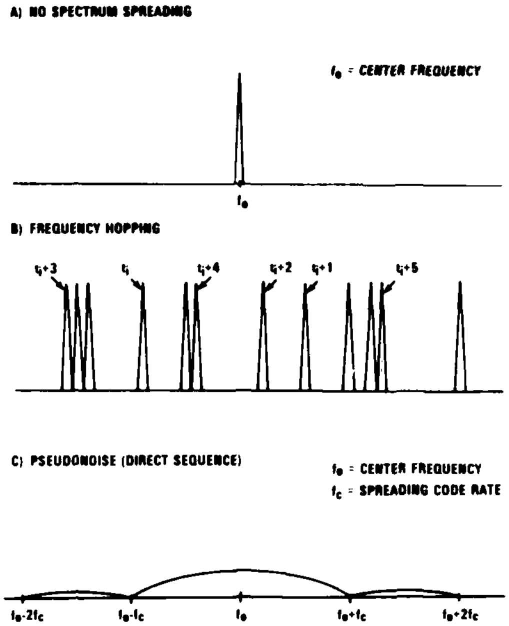

Figure 1. Frequency spectra of spread spectrum signals.

Pseudonoise modulation is defined in this report as the periodic alteration of the phase of the transmitted waveform achieved by phase-shift-keying the carrier after data modulation has been completed.* If the frequency at which phase alterations occur is considerably higher than the transmitted data rate, then the null-to-null bandwidth of the transmitted signal is increased to approximately twice the frequency at which phase alterations occur (see fig. 1). The resistance of pseudonoise signals to intercept and jamming derives from the large bandwidth generated by the pseudonoise system. Since the transmitted energy is spread over a large bandwidth, the spectral power density of the signal is reduced, and in some systems is below the noise level. The pseudonoise signal is noise-like, so a potential interceptor would have great difficulty even detecting the presence of the signal. For a given amount of total jamming power in the pseudonoise bandwidth, the resistance of the pseudonoise signal to jamming does not depend greatly on the bandwidth of the jamming because of the despreading function in

**Although other forms of pseudonoise modulation exist, phase-shift keyed pseudonoise modulation is the only type considered in this report because it is the most commonly used type of pseudonoise modulation.*

the receiver. This despreading function merely demodulates the phase-shift-keyed pseudonoise modulation imposed on the signal by the transmitter, thereby restoring the signal to its original form. However, the despreading function is actually the inverse of the spreading function, and the application of the despreading function to any signals other than the intended pseudonoise signal will result in the spreading of those signals. Thus, narrowband jamming is effectively spread by the despreading function, resulting in the wideband noise-like spectrum in figure 2. Wideband noise jamming is not affected by despreading because of the random nature of this type of jamming (that is, further randomization of an already random process has no effect). Thus, wideband jamming retains its reduced effectiveness (caused by its wide bandwidth and reduced power density), while narrowband jamming is altered by the despreading function to look like wideband jamming. Additional information about frequency hopping and pseudonoise modulation is available.[1-4]

In both frequency hopping and pseudonoise communication systems, there must be some method which tells the transmitter what frequency to change to or when to change phase. This method must also be known to the receiver so that it can follow the transmitted signal in frequency or phase in order to derive the transmitted data as they were before spectrum spreading was imposed. The method used in most frequency hopping and pseudonoise systems uses a sequence of binary digits called a spreading code. The spreading code is normally (although not necessarily) a pseudo-random generated code, so that interception of a portion of the communication signal will not allow the determination of the code-generation scheme. In typical frequency hopping systems, the spreading code bit generated for each hopping period is shifted into a shift register containing a number of previously generated spreading code bits, and the contents of the shift register are used to determine the next frequency in the hopping sequence. In typical binary pseudonoise systems, the new spreading code bit is compared to the previous bit, with a change in logical state between these two adjacent bits causing phase alteration of the carrier signal. The rate at which

---

[1]*D. J. Torrieri, Frequency Hopping in a Jamming Environment, U.S. Army Development and Readiness Command, DARCOM Report CM/CCM-78-2 (December 1978).*

[2]*D. J. Torrieri, Pseudonoise Spread-Spectrum Systems in Communication Warfare, U.S. Army Development and Readiness Command, Report CM/CCM-79-9 (December 1979).*

[3]*Spread Spectrum Operator's Handbook, National Security Agency, NSA Report W32-228-78 (30 May 1978).*

[4]*R. C. Dixon, Spread Spectrum Systems, John Wiley and Sons, Inc., New York (1976).*

the spreading code is generated determines the hopping rate in a frequency hopping system, but in general does not determine the total bandwidth of the hopping signal. (Although the hopping rate does have an effect on the "instantaneous" signal bandwidth and thus may alter the total bandwidth, this alteration is not significant, except in systems with very high hopping rates and relatively narrow total bandwidths.) The total bandwidth of a frequency hopping system is determined primarily by the range of possible frequency choices (or "channels") allowed by the system. As previously mentioned, the null-to-null bandwidth of a pseudonoise system depends on the spreading code and, specifically, the bandwidth is approximately twice the spreading-code generation rate.

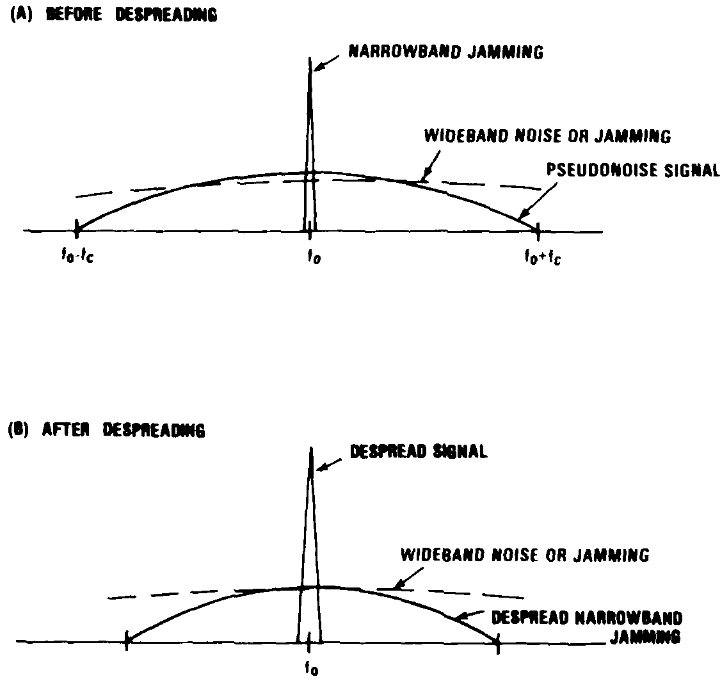

Figure 2. Effects of pseudonoise despreading on noise and jamming.

It is intuitively obvious that the use of frequency hopping or pseudonoise modulation on a signal decreases the vulnerability of the signal to narrowband interference by reducing the signal's power density and increasing its bandwidth. However, it is difficult for on to intuitively determine exactly how much improvement either of these techniques might offer in any particular communication application. Theoretical treatments of the effects of frequency hopping and pseudonoise modulation have been published.[1,2] The treatment uses strictly analytical models of these spread-spectrum techniques to derive expressions for bit and word error rates in terms of the characteristics of the spectrum spreading techniques being used, the information being transmitted, and the interference that is present.

The program described in this paper is a computer adaptation of portions of the theory presented by Torrieri.[1,2] The program models the following communication and interference characteristics.

(a) Spread-spectrum techniques:

1. Fast-frequency hopping--models equation (14) of CM/CCM-78-2.[1]

2. Slow-frequency hopping--models equations (20) and (59) of CM/CCM-78-2.

3. Pseudonoise (direct sequence)--models equations (54) and (55) of CM/CCM-79-9.[2]

(b) Data modulation techniques (frequency hopping only):

1. Binary frequency shift keyed (binary FSK)--This type of modulation uses two different frequency channels for each transmitted information bit; transmission at one frequency denotes a logical "1," while transmission at the other frequency denotes a "0." Demodulation of the received signal is done through a comparison of the energy in the two channels; the higher energy is selected. Thus, the receiver does not require any threshold circuitry. The advantages of this type of modulation are that it requires relatively simple and inexpensive hardware, and it does not require phase coherence of the carrier from bit to bit. Its main disadvantage is that it requires two channels for each transmitted bit and therefore uses twice the bandwidth of other modulation techniques.

---

[1] *D. J. Torrieri, Frequency Hopping in a Jamming Environment, U.S. Army Development and Readiness Command, DARCOM Report CM/CCM-78-2 (December 1978).*

[2] *D. J. Torrieri, Pseudonoise Spread-Spectrum Systems in Communication Warfare, U.S. Army Development and Readiness Command, Report CM/CCM-79-9 (December 1979).*

2. Coherent phase shift keyed (CPSK). This type of modulation encodes the information bit to be transmitted in the phase of the carrier. The receiver demodulates this signal by measuring the received phase and comparing it to a reference; an in-phase or out-of-phase condition determines the value of the received bit. The advantages of this type of modulation are its reduced bandwidth requirements (half that of binary FSK) and its higher resistance (with respect to FSK) to bit errors due to interference. Its main disadvantages are that it requires complex circuitry in the receiver for thresholding and phase measurement and that carrier phase coherence must be maintained by the transmitter when generating the information bits.

(c) Error-correcting codes: The program allows the use of repetition coding, where each information bit is transmitted more than once and at more than one hopping frequency. The receiver uses majority rule to determine the correct value of the information bit. The program also allows Hamming-type error-correcting codes, such as (7,4) coding. This type of coding converts a 4-bit information word to a 7-bit transmitted bit stream. The receiver is able to detect up to two bit errors in the 7-bit stream, and can correct one bit error in the stream, thereby correctly restoring the 4-bit information word in spite of the single bit error. The program accepts any type of (m,n) coding; the user need supply only m (the total number of bits after coding), n (the number of information bits per word before coding), and the maximum number of bit errors that the code can correct in each m-bit block. (For [7,4] coding, these numbers are 7,4, and 1, respectively.)

(d) Jamming techniques:

1. Wideband Gaussian noise jamming with variable bandwidth.
2. Narrowband (continuous-tone) jamming.
3. Narrowband repeater jamming (sometimes called follower jamming).
4. White noise repeater jamming.

The user of the program also specifies the following parameters.

(e) Information bandwidth: The bandwidth of the information signal before spread-spectrum techniques are applied. This value is also used by the program as the frequency hopping channel bandwidth.

(f) Total bandwidth: The bandwidth of the transmitted signal after spread-spectrum techniques are applied.

(g) Number of hopping channels: The total number of frequency choices (or pairs of frequency choices for binary FSK modulation) available for frequency hopping. The number may either be specified by the user or derived in the program by dividing the total bandwidth (f, above) by the information bandwidth (e, above). The latter derivation results in the use of the maximum number of hopping channels available in the total bandwidth.

(h) Signal-to-noise ratio (SNR): The ratio in decibels of the total signal power to the thermal noise power.

(i) Jamming-to-signal ratio (JSR): The ratio in decibels of the total jamming power present over the total spread-spectrum bandwidth to the total signal power. In other words, JSR is the jamming-to-signal ratio that would exist if no spectrum spreading techniques were used and if the total jamming power were concentrated within the information bandwidth.

The program does not consider signal acquisition and synchronization in its analyses of spread-spectrum communication systems. Synchronization is the procedure that aligns the pseudo-random codes used in generating the transmitted spread-spectrum signal with the codes used in the despreading portion of the receiver to allow proper despreading to occur. The problem of synchronization in spread-spectrum systems is similar to the problem of synchronization in enciphered systems because enciphered systems also employ pseudo-random codes that must be properly aligned in the receiver before deciphering can be accomplished. Since transmitter-receiver synchronization is required before any communication may take place, this is often considered one of the more vulnerable links in a system. In particular, hostile jamming may be targeted on a communication system specifically to deny or disrupt the synchronization process. Although this process is an important part of any spread-spectrum system, it is difficult and complicated to analyze or model because of the many variables involved. Therefore, both the program described in this paper and the theoretical treatments of Torrieri[1,2] on which this program is based assume that transmitter-receiver synchronization has been accomplished and is maintained throughout the duration of the communication in question.

3. PROGRAM DESCRIPTION

The computer program described in this report employs interactive computer graphics to provide improved user-computer interaction and a simple, easily understood graphical display of the results of an analysis. Dialogue between the user and the program is generally in a question-answer format. The user is first supplied with a short

---

[1] *D. J. Torrieri, Frequency Hopping in a Jamming Environment, U.S. Army Development and Readiness Command, DARCOM Report CM/CCM-78-2 (December 1978).*

[2] *D. J. Torrieri, Pseudonoise Spread-Spectrum Systems in Communication Warfare, U.S. Army Development and Readiness Command, Report CM/CCM-79-9 (December 1979)*

overview of the program. The user is then asked a series of questions concerning the characteristics of the communication system and the interference environment to be analyzed. The program displays each question, and then waits after each question until the user has provided an answer. Some questions require only a yes or no answer; others require the user to choose from a list of options, in which case invalid answers will be rejected and the question repeated; still others require a numerical value such as SNR, and any number will be accepted. The answers to some questions will control the sequence of succeeding questions. All questions have a default answer, so that any question which is not given an answer by the user will nonetheless be assigned the default answer (usually a typical answer or an answer which allows the user maximum flexibility in succeeding questions).

The question-answer section of the program is composed of three parts, which are discussed in the following.

(1) The first part requests some general information about the analysis, i.e., the type of spread-spectrum technique to be used, the type of jamming present, and the type of error-correcting coding to be used (if any).

(2) The second part requests the format of the graphical output desired by the user. The user may select both the x-axis variable and the family variable. The x-axis variable is the signal or jamming parameter that the user wishes to vary in order to evaluate its effect on the bit or word error rate of the communication system. This variable appears as the x-axis variable on the graphical plot output produced by the analysis. The family variable is the signal or jamming parameter which is allowed to assume a number of discrete values (maximum of 9 values) in order to produce a family of curves on a single plot, with each curve representing one of the specified family variable values. The user is prompted by the program for the x-axis variable and its range of values, and for the family variable and its discrete values. The parameters that follow may be specified as the x-axis or family variables.

(a) Number of repetitions of an information bit when repetition coding is used.

(b) Number of hopping channels that are receiving an interfering (i.e., jamming) signal.

(c) Signal-to-noise ratio (SNR) in decibels.

(d) Jamming-to-signal ratio (JSR) in decibels.

(e) Portion of total spread-spectrum bandwidth over which an interfering signal is present (i.e., portion of band that is jammed). (This parameter is directly related to parameter (b), and as such, parameters (b) and (e) are mutually exclusive.)

These five parameters are the only ones that may be varied within a particular analysis, and only two of these (the parameters selected to be the x-axis and family variables) may be varied at any one time. All other parameters (such as bandwidths) are held constant during an analysis.

(3) The third portion of the question-answer section of the program allows the user to set the values of the constant parameters and the parameters not selected as the x-axis or family variable. All of these remaining parameters have default values which are used if the user declines to specify a value.

After the question-answer section of the program has been completed, the user is provided with a tabular synopsis of the parameter values that will be used in the analysis. The program then runs the analysis and produces the x-y plot containing the results of the analysis; finally, the program returns to the beginning of the question-answer section to allow the definition of the parameters for a new analysis, if desired by the user.

A listing of the program source code appears in appendix A.

Appendix B gives an example of the interactive dialogue between the user and the program for a typical analysis, along with the results of the analysis.

All program variables of interest are described in detail in appendix C, and appendix D describes all library subroutines and functions required by the program. Finally, appendix E gives default values for certain program variables. These values are used unless the user specifies other values when asked.

The remainder of section 3 is devoted to descriptions of the internal structures of the various routines used in the program. All the routines are written in FORTRAN IV and may be executed, with minor modifications, on most machines with sufficient memory and a FORTRAN IV compiler.

3.1 Main Program

The main program consists almost entirely of calls to subroutines which perform the actual work required for an error analysis. The main program first calls subroutines INITP (which performs various initialization functions) and ELT1 (which zeros the central processor unit timer). The primary program loop, starting at ISN 15, is then entered. Each iteration of this loop constitutes a complete analysis of a system, including plots and tabular output. The loop begins by reinitializing the graphics screen (ISN 15-17) then displays the amount of CPU time used during the previous loop iteration (ISN 18-19) and reinitializes the CPU timer (ISN 20). Then subroutine MODE is called in ISN 21 to determine the type of analysis to be performed during the current loop iteration (bit or word error rate, type of jamming, type of data modulation, etc). Subroutine DFAM, called in ISN 22, determines which parameter (if any) is to be varied to produce a family of curves and the value of the parameter to be used for each curve. Subroutine DXAX, called in ISN 23, similarly determines which parameter is to be the x-axis variable and the range of values for the x-axis. Subroutine PARSET (ISN 24) then determines the values of all other parameters. Subroutine TABOUT (ISN 25) produces the tabular output, including the important characteristics of the analysis and the values of all input parameters. Subroutine PCALC then calculates the bit/word error rate curves (for frequency hopping and optionally for PN modulation, respectively, in ISN 26 and 31) . These curves are then plotted on the graphics screen by subroutine FHPLOT (ISN 27 and 35). The loop is then terminated in ISN 37 by a transfer to statement 10.

3.2 Subroutine FHPLOT

Subroutine FHPLOT actually plots the analysis results on the screen of the graphics terminal. The routine begins by clearing the screen and setting graphics display options concerning the actual plot appearance (ISN 7-17). The data to be plotted are then scaled (ISN 18-37) and plotted (ISN 38-56), and axis labels are added (ISN 57-61).

3.3 Entry INITP

This entry point in subroutine FHPLOT initializes the plot library when the program is first begun. Axis labels defined in block data are processed for use by the plot library (ISN 66-72). The actual plotting area on the terminal screen is then determined (ISN 76-86). Finally, the program introduction is displayed on the screen if the user so desires (ISN 87-96).

3.4 Subroutine MODE

This subroutine determines the basic signal and jamming characteristics to be used in the analysis. The type of jamming to be analyzed and some associated parameters are requested from the user (ISN 25-40). The coding parameters are then requested (ISN 42-65). Finally, some information is requested concerning the type of frequency hopping and type of data modulation to be analyzed (ISN 69-83).

3.5 Subroutine DFAM

This routine determines the parameter to be used as the family variable. The family variable is first requested (ISN 16-31); then the number of family curves and their corresponding parameter values are obtained from the user (ISN 33-44).

3.6 Entry FAMC

This entry point in subroutine DFAM automatically determines the family parameter values when repetition coding is selected in subroutine PARSET.

3.7 Subroutine DXAX

This routine determines the parameter to be used as the x-axis variable. The x-axis variable is requested in ISN 11-22, then the range of values for the x-axis is obtained (ISN 24-30).

3.8 Entry XAXC

This entry point in subroutine DXAX automatically determines the x-axis parameter range when repetition coding is selected in subroutine PARSET.

3.9 Subroutine PARSET

This routine obtains from the user the values of all parameters not previously set. Default values are assigned in ISN 17-27. Defaults are overridden by the user, if desired, in ISN 28-83. The use of repetition coding is then determined (ISN 84-90) as follows: (1) if no other coding is specified in subroutine MODE, repetition coding is assumed, and (2) the number of repetitions is determined by the chip bandwidth (FCM) divided by the information bandwidth (FB). This latter quotient is, by default, one, unless otherwise specified by FCM and FB, because FCM and FB are equal by default. Thus simple bit error rate (one repetition) is calculated as the default. Finally, if the number of repetitions is to be the family or x-axis variable, then the appropriate routine is called (ISN 92-94).

3.10 Subroutine TABOUT

This routine provides a tabular summary of the important parameters to be used in the analysis. Constants are printed in ISN 13-34. Family and x-axis parameters and their values are then printed (ISN 35-41). Finally, values that are in units of decibels, kilohertz, or megahertz are converted to their absolute values for computational use (ISN 42-46).

3.11 Subroutine PCALC

This routine directs the calculation of the data points on each of the family curves. The correct family and x-axis values are assigned to the family and x-axis variables for each data point to be computed. Then each data point is actually calculated by calling the appropriate function (FHOP for frequency hopping, PRPN for pseudonoise) as determined by the argument SUBR, which is assigned a value of either FHOP or PRPN by the main program based on the type of plot desired by the user. The data point is placed in the array Y for subsequent plotting.

3.12 Function FHOP

This routine actually calculates the bit or word error rates for fast and slow frequency hopping. Signal and jamming parameters are set and checked in ISN 10-31. Intermediate probabilities for use in the error rate evaluation are then calculated (with different probabilities for different types of signals and jamming) in ISN 33-65. The error rate is then calculated for FSK modulated fast-frequency hopping with narrowband jamming in ISN 66-97, which evaluates equation (14) of Torrieri.[1] Actually, this equation has seven nested sums, four of which are calculated here, while the inner three are calculated in function PXKI (ISN 90). Error rates for repeater jamming, slow frequency hopping, white noise repeater jamming, and coherent PSK modulation are determined in ISN 98-137.

3.13 Subroutine QINT

This routine is used by the integration routine NL9 (see function FHOP, ISN 52) to integrate a variation of Marcum's Q function for use in determining one of the intermediate probabilities needed to calculate error rates.

3.14 Function PXKI

This function is called by function FHOP (see ISN 90 of FHOP) to evaluate the inner three sums of equation (14) of Torrieri.[1]

---

[1]*D. J. Torrieri, Frequency Hopping in a Jamming Environment, U.S. Army Development and Readiness Command, DARCOM Report CM/CCM-78-2 (December 1978).*

3.15 Function PRPN

This function is called by subroutine PCALC to calculate the error rates for pseudonoise modulated signals. This routine evaluates equations (54) and (55) of Torrieri.[2]

3.16 Subroutine FREAD

This routine reads one real number from the terminal without performing a line feed before the read takes place. This routine may be replaced by a READ statement, but this will not allow the user's answer to a question to follow the question on the same line on the terminal screen.

3.17 Subroutine IREAD

This routine reads one integer number from the terminal without performing a line feed before the read takes place. This routine may be replaced by a READ statement.

3.18 Function LOGCOM

This routine computes the natural logarithm of (I,J) where I and J are the arguments of the function and

$$(I,J) = \frac{I!}{J!\ (I - J)!} \quad .$$

3.19 Function FACT

This routine calculates I!, where I is the argument of the function.

3.20 Block Data

This routine initializes the x- and y-axis label alphanumeric values for use by the plotting routines.

---

[2] *D. J. Torrieri, Pseudonoise Spread-Spectrum Systems in Communication Warfare, U.S. Army Development and Readiness Command, Report CM/CCM-79-9 (December 1979).*

4. SUMMARY

A computer program has been developed which analyzes and compares the performance of two types of spread-spectrum techniques in interfering or jamming environments. The program does not consider synchronization problems or signal and jamming propagation effects. Instead, the program assumes that the transmitter and receiver are in synchronization and that there is a defined signal-to-noise ratio and jamming-to-signal ratio at the receiver. The program can give considerable insight into the performance increases and relative values that might be achieved by the incorporation of spread-spectrum techniques into existing or postulated communication systems.

APPENDIX A.--PROGRAM LISTING

PRECEDING PAGE BLANK-NOT FILMED

APPENDIX A--PROGRAM LISTING

This appendix contains a listing of the spread-spectrum analysis program. Further details concerning the use of the program as well as descriptions of subroutines and variables used in the program are given in appendices B, C, D and E.

APPENDIX A

```
                C
                C   THIS PROGRAM ESTIMATES THE PROBABILITY OF ERROR (I.E. THE
                C      BIT ERROR RATE) FOR AN FSK FREQUENCY HOPPING COMMUNICATION
                C      SYSTEM.  NOISE AND JAMMING ARE CONSIDERED IN THE MODEL.
                C      THE USER IS PROMPTED FOR ALL ESSENTIAL INFORMATION REQUIRED
                C      FOR AN ANALYSIS.  OUTPUT CONSISTS OF A TABULATION OF THE
                C      IMPORTANT MODEL PARAMETERS FOLLOWED BY ANY PLOTS REQUESTED
                C      BY THE USER.
                C
ISN 0002              EXTERNAL FHOP,PRPN
ISN 0003              LOGICAL*1 T(8),ITIME(12)
ISN 0004              EQUIVALENCE (ITIME(5),T(1))
ISN 0005              COMMON / PLOT / X(100),Y(100,9)
ISN 0006              COMMON / INPAR / NJAM,BETA1,GAMMA,PB,HW,FCM,A,B,D,NBPC,IWN,IFP,
                     +                 G,NBPW,NERR,JTP,UPBJ,IEST,NCH,ICON,ISAME,ISLOW
ISN 0007              COMMON / FAMPAR / IFAM,NFAM,FAMV(9)
ISN 0008              COMMON / XAXPAR / IXAX,XAXS,XAXE,XAXI,NPTSX
ISN 0009              COMMON / LABELS / LAB(35),LAB2(35),LAB3(140),LABY2(16),
                     +                  LABY(4),LEN(5),LENY
                C
                C   INITIALIZATION OF PROGRAM
                C
ISN 0010              I=0
ISN 0011              IF(I.NE.0) GO TO 20
ISN 0013              CALL INITP(IP)
ISN 0014              CALL ELT1
                C
                C   LOOP INIT
                C
ISN 0015           10 CALL ERASE
ISN 0016              CALL MOVABS(0,760)
ISN 0017              CALL ANMODE
ISN 0018              CALL ELT3(ITIME)
ISN 0019              WRITE(6,2000) T
ISN 0020              CALL ELT1
                C
                C   JAMMING, BIT/WORD MODE SELECT
                C
ISN 0021              CALL MODE
                C
                C   FAMILY DETERMINATION
                C
ISN 0022              CALL DFAM
```

```
                    C
                    C   X-AXIS DETERMINATION
                    C
ISN 0023                CALL DXAX(IFAM)
                    C
                    C   SET PARAMETER VALUES
                    C
ISN 0024                CALL PARSET(IFAM,IXAX)
                    C
                    C   TABULATED OUTPUT
                    C
ISN 0025                CALL TABOUT
                    C
                    C   CALC PROB OF ERROR CURVES FOR FREQ HOPPING
                    C
ISN 0026                CALL PCALC(FHOP,0)
                    C
                    C   PLOT FREQ HOPPING CURVES
                    C
ISN 0027                IF(ISAME.EQ.0.OR.JTP.NE.1) CALL FHPLOT(NFAM,NPTSX,IXAX,0)
                    C
                    C   CALC PROB OF ERROR OR TABULATED OUTPUT FOR PN MODULATION
                    C
ISN 0029                IF(JTP.NE.1) GO TO 10
ISN 0031                CALL PCALC(PRPN,ISAME)
                    C
                    C   PLOT PN CURVES
                    C
ISN 0032                NF2=NFAM
ISN 0033                IF(ISAME.EQ.1) NF2=2*NFAM
ISN 0035                CALL FHPLOT(NF2,NPTSX,IXAX,ISAME)
ISN 0036                WRITE(6,1000)
ISN 0037                GO TO 10
                    C
                    C   FORMATS
                    C
ISN 0038              20 STOP
ISN 0039            1000 FORMAT(//)
ISN 0040            2000 FORMAT(6X,'TIME = ',8A1)
ISN 0041                END
```

```
ISN 0002          SUBROUTINE FHPLOT(LINF,LIMP,IXAX,ISAME)
ISN 0003          COMMON / PLOT / X(100),Y(100,9)
ISN 0004          COMMON / LABELS / LAB(35),LAB2(35),LAB3(140),LABY2(16),
                 +                  LABY(4),LEN(5),LENY
ISN 0005          INTEGER LFPR/32/,PFPR(8)/'TYPE',' Y P','OR O','VERV','IEW
                 +     'OF P','ROGR','AM. '/
ISN 0006          INTEGER YES/'Y   '/, MAJOR/4/, MINOR/3/
              C
              C   CLEAR SCREEN
              C
ISN 0007          CALL MOVABS(0,760)
ISN 0008          CALL ERASE
ISN 0009          CALL ANMODE
              C
              C   SCALE DATA
              C
ISN 0010          CALL BINITT
              C
              C   TIC MARKS
              C
ISN 0011          CALL XFRM(MAJOR)
ISN 0012          CALL YFRM(MAJOR)
ISN 0013          CALL XMFRM(MINOR)
ISN 0014          CALL YMFRM(MINOR)
              C
              C   LINE, AXIS TYPES
              C
ISN 0015          CALL YTYPE(2)
ISN 0016          CALL NPTS(LIMP)
ISN 0017          CALL LINE(0)
              C
              C   SCALING
              C
ISN 0018          XMI=X(1)
ISN 0019          XMA=X(1)
ISN 0020          YMI=Y(1,1)
ISN 0021          YMA=Y(1,1)
ISN 0022          DO 20 I=1,LIMP
ISN 0023             IF(X(I).LT.XMI) XMI=X(I)
ISN 0025             IF(X(I).GT.XMA) XMA=X(I)
ISN 0027             DO 10 J=1,LINF
ISN 0028                IF(Y(I,J).LT.YMI) YMI=Y(I,J)
ISN 0030                IF(Y(I,J).GT.YMA) YMA=Y(I,J)
```

```
ISN 0032        10    CONTINUE                                              00012700
ISN 0033        20 CONTINUE                                                 00012800
              C                                                             00012900
              C   SET PLOT SCALE MAX, MIN                                   00013000
              C                                                             00013100
ISN 0034           YMI=AMAX1(YMI,1.E-8)                                     00013200
ISN 0035           YMA=AMIN1(YMA,1.)                                        00013300
ISN 0036           CALL DLINX(XMI,XMA)                                      00013400
ISN 0037           CALL DLINY(YMI,YMA)                                      00013500
              C                                                             00013600

              C   DRAW FIRST PLOT                                           00013700
              C                                                             00013800
ISN 0038           CALL CHECK(X,Y)                                          00013900
ISN 0039           CALL DSPLAY(X,Y)                                         00014000
              C                                                             00014100
              C   DRAW REMAINING FAMILY CURVES                              00014200
              C                                                             00014300
ISN 0040           IF(LIMF.LE.1) GO TO 40                                   00014400
ISN 0042           LIMF2=LIMF                                               00014500
ISN 0043           IF(ISAME.EQ.1) LIMF2=LIMF/2                              00014600
ISN 0045           IF(LIMF2.LE.1) GO TO 35                                  00014700
ISN 0047           DO 30 I=2,LIMF2                                          00014800
ISN 0048              CALL CPLOT(X,Y(1,I))                                  00014900
ISN 0049        30 CONTINUE                                                 00015000
ISN 0050        35 IF(ISAME.NE.1) GO TO 40                                  00015100
ISN 0052           LIMF3=LIMF2+1                                            00015200
ISN 0053           CALL LINE(2)                                             00015300
ISN 0054           DO 38 I=LIMF3,LIMF                                       00015400
ISN 0055              CALL CPLOT(X,Y(1,I))                                  00015500
ISN 0056        38 CONTINUE                                                 00015600
              C                                                             00015700
              C   DRAW LABELS                                               00015800
              C                                                             00015900
ISN 0057        40 CALL JUSTER(LEN(IXAX),LAB3((IXAX-1)*28+1),0,32,LDUM,IOFF) 00016000
ISN 0058           CALL NOTATE(IXCEN+IOFF,IFIX(YMIN)-75,LDUM,LAB3((IXAX-1)*28+1)) 00016100
ISN 0059           CALL MOVABS(IFIX(XMIN)-110,IFIX(YMAX)-125)               00016200
ISN 0060           CALL VLABEL(LENY,LABY2)                                  00016300
ISN 0061           CALL ANMODE                                              00016400
              C                                                             00016500
              C   FINISHED                                                  00016600
              C                                                             00016700
```

```
ISN 0062          CALL TREAD(IDUM,LD,4)                                  00016800
ISN 0063          RETURN                                                 00016900
            C                                                            00017000
            C   INITIALIZATION ROUTINE                                   00017100
            C                                                            00017200
ISN 0064          ENTRY INITP(IP)                                        00017300
            C                                                            00017400
            C   INITIALIZE PLOT LIBRARY                                  00017500
            C                                                            00017600
ISN 0065          CALL PGMASK                                            00017700
            C                                                            00017800
            C   TRANSLATP LABELS FROM EBCDIC TO ASCII FOR PLOTTING       00017900
            C                                                            00018000
ISN 0066          DO 50 I=1,35                                           00018100
ISN 0067             LAB2(I)=LAB(I)                                      00018200
ISN 0068       50 CONTINUE                                               00018300
ISN 0069          CALL TREA(4*35,LAB)                                    00018400
ISN 0070          CALL TREA(LENY,LABY)                                   00018500
ISN 0071          CALL UNPAK(140,LAB,LAB3)                               00018600
ISN 0072          CALL UNPAK(16,LABY,LABY2)                              00018700
            C                                                            00018800
            C   INIT PLOTTING                                            00018900
            C                                                            00019000
ISN 0073          IBAUD=120                                              00019100
ISN 0074          CALL INITT(IBAUD)                                      00019200
ISN 0075          CALL BINITT                                            00019300
            C                                                            00019400
            C   GET SCREEN WINDOW MAX, MIN                               00019500
            C                                                            00019600
ISN 0076          ITEM=IBASEX(13)                                        00019700
ISN 0077          XMIN=COMGET(ITEM)                                      00019800
ISN 0078          ITEM=IBASEX(14)                                        00019900
ISN 0079          XMAX=COMGET(ITEM)                                      00020000
ISN 0080          ITEM=IBASEY(13)                                        00020100
ISN 0081          YMIN=COMGET(ITEM)                                      00020200
ISN 0082          ITEM=IBASEY(14)                                        00020300
ISN 0083          YMAX=COMGET(ITEM)                                      00020400
            C                                                            00020500
            C   CALCULATE CENTER OF WINDOW                               00020600
            C                                                            00020700
ISN 0084          IXCEN=(XMAX+XMIN)/2.                                   00020800
ISN 0085          IYCEN=(YMAX+YMIN)/2.                                   00020900
```

```
ISN 0086          WRITE(6,2000)
              C
              C   PULL OR SHORT PROMPTING
              C
ISN 0087          CALL ERASE
ISN 0088          CALL ANMODE
ISN 0089          CALL TWRITE(PFPR,LFPR)
ISN 0090          READ(5,3000) IP
              C
              C   PRINT BASIC INFORMATION
              C
ISN 0091          IF(IP.NE.YES) RETURN
ISN 0093          WRITE(6,4000)
ISN 0094          WRITE(6,1000)
ISN 0095          CALL TREAD(IDUN,LD,4)
ISN 0096          RETURN
              C
              C   FORMATS
              C
ISN 0097       1000 FORMAT(/' TYPE RETURN TO CONTINUE')
ISN 0098       2000 FORMAT(//)
ISN 0099       3000 FORMAT(A1)
ISN 0100       4000 FORMAT(/' THIS PROGRAM ESTIMATES THE BIT ERROR RATE FOR ',
                 +     'FSK FREQUENCY'/4X,'HOPPING AND PN COMMUNICATION ',
                 +     'SYSTEMS.  NOISE AND JAMMING ARE'/4X,'CONSIDERED IN ',
                 +     'THE MODEL.  THIS PROGRAM MUST BE RUN FROM A TEKTRONIX'/
                 +     4X,'GRAPHICS TERMINAL OR EQUIVALENT.'//
                 +     ' TO HALT EXECUTION OF THIS PROGRAM, TYPE ',1H#,
                 +     ' IN RESPONSE'/'    TO ANY REQUEST FOR ',
                 +     'NUMERICAL INPUT.')
ISN 0101          END
ISN 0002          SUBROUTINE MODE
ISN 0003          INTEGER LJTP/14/, PJTP(4)/'JAMM','ING ','TYPE',':   '/
ISN 0004          INTEGER LEXP/41/, PEXP(11)/'DO Y','OU W','ANT ','EXPA','NDED',
                 +     ' INT','EGRA','LS (','Y OR',' N)?','    '/
ISN 0005          INTEGER LCON/33/, PCON(9)/'CONS','TANT',' JAM','MING',' POW',
                 +     'ER (','Y OR',' N)?','    '/
ISN 0006          INTEGER LBPW/21/, PBPW(6)/'WORD',' LEN','GTH ','IN B','ITS:',
                 +     '    '/
ISN 0007          INTEGER LCHIP/16/, PCHIP(4)/'CHIP','S PE','R WO','RD: '/
ISN 0008          INTEGER LNER/22/, PNER(6)/'CHIP',' ERR','OR T','HRES','HOLD',
                 +     ':   '/
```

```
ISN 0009        INTEGER LSAME/33/, PSAME(9)/'HOPP','ING,',' PN ','CURV','ES O',
               +     'N SA','ME P','LOT?','    '/
ISN 0010        INTEGER LSLOW/19/, PSLOW(5)/'SLOW',' FRE','Q HO','PPIN','G?  '/
ISN 0011        INTEGER LWN/22/, PWN(6)/'WHIT','E NO','ISE ','REPE','ATER',
               +     '?   '/
ISN 0012        INTEGER LPPP/9/, PPP(3)/'COH ','PSK?','    '/
ISN 0013        INTEGER YES/'Y   '/
ISN 0014        COMMON / INPAR / NJAM,BETA1,GAMMA,FB,BW,FCM,A,B,D,NBPC,IWN,IPP,
               +                  G,NBPW,NERR,JTP,UPBJ,IEST,NCH,ICON,ISAME,ISLOW
          C
          C   SET MODE VARIABLE DEFAULTS
          C
ISN 0015        NBPW=1
ISN 0016        NBPC=1
ISN 0017        NERR=1
ISN 0018        UPBJ=0.
ISN 0019        IEST=0
ISN 0020        ICON=0
ISN 0021        ISAME=0
ISN 0022        ISLOW=0
ISN 0023        IWN=0
ISN 0024        IPP=0
          C
          C   JAMMING TYPE
          C
ISN 0025     10 WRITE(6,1000)
ISN 0026        CALL TWRITE(PJTP,LJTP)
ISN 0027        CALL IREAD(JTP)
ISN 0028        IF(JTP.LT.1.OR.JTP.GT.3) GO TO 10
          C
          C   PARTIAL BAND JAMMING PARAMETERS
          C
ISN 0030        IF(JTP.NE.3) GO TO 20
ISN 0032        WRITE(6,3000)
ISN 0033        CALL TWRITE(PWN,LWN)
ISN 0034        CALL TREADC(IP,LD,4)
ISN 0035        IF(IP.EQ.YES) IWN=1
ISN 0037     20 WRITE(6,3000)
ISN 0038        CALL TWRITE(PCON,LCON)
ISN 0039        CALL TREADC(IP,LD,4)
ISN 0040        IF(IP.EQ.YES) ICON=1
          C
```

```
                C    BIT/WORD MODE SELECT (WORD LEN = 1  ==>  BIT MODE)
                C
ISN 0042              IF(JTP.EQ.3.AND.IWN.EQ.1) RETURN
ISN 0044              WRITE(6,2000)
ISN 0045           40 CALL TWRITE(PBPW,LBPW)
ISN 0046              CALL IREAD(NBPW)
ISN 0047              IF(NBPW.LE.0) GO TO 40
ISN 0049              IF(NBPW.EQ.1) GO TO 70
                C
                C    WORD MODE - GET CODING PARAMETERS
                C
ISN 0051           50 CALL TWRITE(PCHIP,LCHIP)
ISN 0052              CALL IREAD(NBPC)
ISN 0053              IF(NBPC.LE.0) GO TO 50
ISN 0055           60 CALL TWRITE(PNER,LNER)
ISN 0056              CALL IREAD(NERR)
ISN 0057              IF(NERR.LT.0.OR.NERR.GT.NBPC) GO TO 60
ISN 0059           70 IF(JTP.EQ.3) RETURN
ISN 0061              IF(JTP.NE.1) GO TO 80
ISN 0063              WRITE(6,3000)
ISN 0064              CALL TWRITE(PSAME,LSAME)
ISN 0065              CALL TREADC(IP,LD,4)
                C
                C    PLOT MODE - PH, PN ON SAME OR SEPARATE GRAPHS
                C
ISN 0066              IF(IP.EQ.YES) ISAME=1
ISN 0068              WRITE(6,3000)
                C
                C    INTEGRAL COMPUTATION MODE SELECT
                C
ISN 0069              CALL TWRITE(PEXP,LEXP)
ISN 0070              CALL TREADC(IP,LD,4)
ISN 0071              IF(IP.EQ.YES) IEST=1
ISN 0073           80 WRITE(6,3000)
                C
                C    DETERMINE FAST OR SLOW HOPPING
                C
ISN 0074              CALL TWRITE(PSLOW,LSLOW)
ISN 0075              CALL TREADC(IP,LD,4)
ISN 0076              IF(IP.EQ.YES) ISLOW=1
ISN 0078              IF(IP.NE.YES) GO TO 100
                C
                C    COHERENT PSK
```

```
              C                                                                          00033700
ISN 0080            WRITE(6,3000)                                                        00033800
ISN 0081            CALL TWRITE(PPP,LPPP)                                                00033900
ISN 0082            CALL TREADC(IP,LD,4)                                                 00034000
ISN 0083            IF(IP.EQ.YES) IPP=1                                                  00034100
              C                                                                          00034200
              C   FORMATS                                                                00034300
              C                                                                          00034400
ISN 0085        100 RETURN                                                               00034500
ISN 0086       1000 FORMAT(/' JAMMING OPTIONS:'/5X,'1   NARROW BAND'/                    00034600
                   +     5X,'2   PARTIAL BAND'/5X,'3   REPEATER')                        00034700
ISN 0087       2000 FORMAT(/' IF WORD LENGTH IS ONE BIT, BIT ERROR RATE ',               00034800
                   +     'IS CALCULATED.'/'   OTHERWISE WORD ERROR RATE ',               00034900
                   +     'IS CALCULATED.'/)                                              00035000
ISN 0088       3000 FORMAT(1X)                                                           00035100
ISN 0089            END                                                                  00035200
ISN 0002            SUBROUTINE DFAM                                                      00035300
ISN 0003            INTEGER LPFAM/20/, PFAM(5)/'FAMI','LY (','# TO',' STO','P): '/       00035400
ISN 0004            INTEGER LPFN/26/, PFAMN(7)/'    ','NBR ','OF F','AMIL','Y CU',       00035500
                   +     'RVES',':   '/                                                  00035600
ISN 0005            INTEGER PFAM1(7)/'    ','NBR ','OF J','AMME','D CH','ANNE','LS: '/00035700
ISN 0006            INTEGER PFAM2(7)/'    ','SNR(','FSK)',' IN ','DB: ','    ','    '/00035800
ISN 0007            INTEGER PFAM3(7)/'    ','JSR ','IN D','B:  ','    ','    ','    '/00035900
ISN 0008            INTEGER PFAM4(7)/'    ','PORT','ION ','OF B','AND ','JAMM','ED: '/00036000
ISN 0009            INTEGER PLEN/7/, LPFV(5)/0,28,20,15,28/, PFAMV(28)                   00036100
ISN 0010            EQUIVALENCE (PFAMV(1),PFAM1(1)), (PFAMV(8),PFAM2(1))                 00036200
ISN 0011            EQUIVALENCE (PFAMV(15),PFAM3(1)), (PFAMV(22),PFAM4(1))               00036300
ISN 0012            COMMON / INPAR / NJAM,BETA1,GAMMA,FB,BW,PCM,A,B,D,NBPC,IWN,IFP,      00036400
                   +                 G,NBPW,NERR,JTP,UPBJ,IEST,NCH,ICON,ISAME,ISLOW      00036500
ISN 0013            COMMON / FAMPAR / IFAM,NFAM,FAMV(9)                                  00036600
              C                                                                          00036700
              C   FAMILY VARIABLE DEFAULTS                                               00036800
              C                                                                          00036900
ISN 0014            IFAM=0                                                               00037000
ISN 0015            NFAM=1                                                               00037100
              C                                                                          00037200
              C   PROMPT FOR FAMILY TYPE                                                 00037300
              C                                                                          00037400
ISN 0016            CALL MOVABS(0,760)                                                   00037500
ISN 0017            CALL ERASE                                                           00037600
ISN 0018            CALL ANMODE                                                          00037700
```

```
ISN 0019          WRITE(6,1000)
ISN 0020          IF(JTP.EQ.2) WRITE(6,2000)
ISN 0022          WRITE(6,3000)
            C
            C   FAMILY TYPE SELECT
            C
ISN 0023       10 CALL TWRITE(PFAM,LPFAM)
ISN 0024          CALL IREAD(IFAM)
ISN 0025          IF(IFAM.LT.0.OR.IFAM.GT.5) GO TO 10
ISN 0027          IF(IFAM.EQ.1.AND.NBPW.GT.1) GO TO 10
ISN 0029          IF(IFAM.LE.1) RETURN
ISN 0031          IF(IFAM.EQ.5.AND.JTP.NE.2) GO TO 10
            C
            C   GET NBR OF CURVES
            C
ISN 0033       20 CALL TWRITE(PFAMN,LPFN)
ISN 0034          CALL IREAD(NFAM)
ISN 0035          IF(NFAM.LT.1.OR.NFAM.GT.9) GO TO 20
ISN 0037          IF(NFAM.GT.4.AND.ISAME.NE.0) GO TO 20
            C
            C   GET FAMILY VARIABLE VALUES
            C
ISN 0039          J=1+(IFAM-2)*PLEN
ISN 0040          DO 30 I=1,NFAM
ISN 0041             CALL TWRITE(PFAMV(J),LPFV(IFAM))
ISN 0042             CALL FREAD(FAMV(I))
ISN 0043       30 CONTINUE
ISN 0044          RETURN
            C
            C   SET VALUES FOR FAMILY = CHIPS-PER-BIT
            C
ISN 0045          ENTRY FAMC
ISN 0046          NFAM=(NBPC+1)/2
ISN 0047          IF(NFAM.GT.9) NFAM=9
ISN 0049          DO 40 I=1,NFAM
ISN 0050             FAMV(I)=1+(I-1)*2
ISN 0051       40 CONTINUE
ISN 0052          RETURN
ISN 0053     1000 FORMAT(/ ' VALID FAMILY AND X-AXIS VARIABLES ARE:'//4X,
              +      '0   NONE (FAMILY ONLY)'/4X,'1   NBR OF CHIPS PER BIT'/
              +      4X,'2   NBR OF JAMMED CHANNELS'/4X,'3   SIGNAL-TO-NOISE ',
              +      'RATIO (PSK)'/4X,'4   JAMMING-TO-SIGNAL RATIO')
```

```
ISN 0054     2000 FORMAT(4X,'S · PORTION OF BAND JAMMED')
ISN 0055     3000 FORMAT(1X)
ISN 0056          END

ISN 0002          SUBROUTINE DXAX(IFAM)
ISN 0003          INTEGER LPXAX/8/, PXAX(2)/'X-AX','IS: '/
ISN 0004          INTEGER LPXS/18/, PXAXS(5)/'    ','X-AX','IS S','TART',':   '/
ISN 0005          INTEGER LPXE/16/, PXAXE(4)/'    ','X-AX','IS E','ND: '/
ISN 0006          INTEGER LPXI/17/, PXAXI(5)/'    ','X-AX','IS I','NCR:','    '/
ISN 0007          COMMON / PLOT / X(100),Y(100,9)
ISN 0008          COMMON / INPAR / NJAM,BETA1,GAMMA,FB,BW,FCM,A,B,D,NBPC,IWN,IFP,
                 +                 G,NBPW,NERR,JTP,UPBJ,IEST,NCH,ICON,ISAME,ISLOW
ISN 0009          COMMON / XAXPAR / IXAX,XAXS,XAXE,XAXI,NPTSX
             C
             C   SET X-AXIS VARIABLE DEFAULTS
             C
ISN 0010          IXAX=4
             C
             C   X-AXIS TYPE SELECT
             C
ISN 0011          WRITE(6,1000)
ISN 0012       10 CALL TWRITE(PXAX,LPXAX)
ISN 0013          CALL IREAD(IXAX)
ISN 0014          IF(IXAX.EQ.IFAM) GO TO 10
ISN 0016          IF(IXAX.LT.1.OR.IXAX.GT.5) GO TO 10
ISN 0018          IF(IXAX.EQ.1.AND.NBPW.GT.1) GO TO 10
ISN 0020          IF(IXAX.EQ.1) RETURN
ISN 0022          IF(IXAX.EQ.5.AND.JTP.NE.2) GO TO 10
             C
             C   GET X-AXIS START, END, INCR VALUES
             C
ISN 0024          CALL TWRITE(PXAXS,LPXS)
ISN 0025          CALL FREAD(XAXS)
ISN 0026          CALL TWRITE(PXAXE,LPXE)
ISN 0027          CALL FREAD(XAXE)
ISN 0028          CALL TWRITE(PXAXI,LPXI)
ISN 0029          CALL FREAD(XAXI)
ISN 0030          GO TO 20
             C
             C   SET VALUES FOR X-AXIS = CHIPS-PER-BIT
             C
ISN 0031          ENTRY XAXC
ISN 0032          XAXS=1.
```

```
ISN 0033          XAXI=2.
ISN 0034          XAXE=NBPC
              C
              C   SET NBR OF POINTS, X-AXIS PLOT ARRAY
              C
ISN 0035       20 NPTSX=ABS(XAXE-XAXS)/ABS(XAXI)+1
ISN 0036          IF(NPTSX.GT.100) NPTSX=100
ISN 0038          IF(XAXS.GT.XAXE) XAXI=-ABS(XAXI)
ISN 0040          DO 30 I=1,NPTSX
ISN 0041             X(I)=XAXS+(I-1)*XAXI
ISN 0042       30 CONTINUE
ISN 0043          RETURN
ISN 0044     1000 FORMAT(1X)
ISN 0045          END

ISN 0002          SUBROUTINE PARSET(IFAM,IXAX)
ISN 0003          INTEGER YES/'Y   '/
ISN 0004          INTEGER LDEP/26/, PDEP(7)/'ARE ','DEPA','ULTS',' OK ','(Y O',
                 +     'R N)','?   '/
ISN 0005          INTEGER LNJAM/24/, PNJAM(6)/'NBR ','OF J','AMME','D CH','ANNE',
                 +     'LS: '/
ISN 0006          INTEGER LBETA/16/, PBETA(4)/'SNR(','PSK)',' IN ','DB: '/
ISN 0007          INTEGER LGAMMA/11/, PGAMMA(3)/'JSR ','IN D','B:  '/
ISN 0008          INTEGER LUPB/24/, PUPB(6)/'PORT','ION ','OF B','AND ','JAMM',
                 +     'ED: '/
ISN 0009          INTEGER LFB/32/, PFB(8)/'INFO',' BIT',' BAN','DWID','TH (',
                 +     'FR I','N KH','Z): '/
ISN 0010          INTEGER LG/30/, PG(8)/'PN P','ROCE','SSIN','G GA','IN (',
                 +     'G IN',' DB)',':   '/
ISN 0011          INTEGER LBW/31/, PBW(8)/'HOPP','ING ','BAND','WIDT','H (B',
                 +     'W IN',' MHZ','):  '/
ISN 0012          INTEGER LFCM/29/, PFCM(8)/'CHIP',' BAN','DWID','TH (','FCM ',
                 +     'IN K','HZ):',' '    '/
ISN 0013          INTEGER LPA/28/, PA(7)/'HOPP','ING ','OVER','LAP ','FACT',
                 +     'OR (','A): '/
ISN 0014          INTEGER LPB/34/, PB(9)/'COMM','-TO-','JAM ','DIFF',' FRE',
                 +     'Q FA','CTOR',' (B)',':   '/
ISN 0015          INTEGER LPD/26/, PD(7)/'PN R','CVR ','LOSS',' FAC','TOR ','(D) ',
                 +     ':   '/
ISN 0016          COMMON / INPAR / NJAM,BETA1,GAMMA,FB,BW,FCM,A,B,D,NBPC,IWN,IFP,
                 +                   G,NHPW,NERR,JTP,UPBJ,IBST,NCH,ICON,ISAME,ISLOW
              C
              C   SET DEFAULT VALUES FOR PARAMETERS
              C
```

```
ISN 0017         BETA1=13.
ISN 0018         GAMMA=0.
ISN 0019         FB=25.
ISN 0020         BW=25.
ISN 0021         NJAM=1
ISN 0022         FCM=25.
ISN 0023         UPBJ=0.25
ISN 0024         A=1.
ISN 0025         B=1.
ISN 0026         D=1.
ISN 0027         G=30.
           C
           C   ARE DEFAULTS OK?
           C
ISN 0028         CALL MOVABS(0,760)
ISN 0029         CALL ERASE
ISN 0030         CALL ANMODE
ISN 0031         WRITE(6,1000)
ISN 0032         IF(IFAM.NE.2.AND.IXAX.NE.2) WRITE(6,1010) NJAM
ISN 0034         IF(IFAM.NE.3.AND.IXAX.NE.3) WRITE(6,1020) BETA1
ISN 0036         IF(IFAM.NE.4.AND.IXAX.NE.4) WRITE(6,1030) GAMMA
ISN 0038         IF(IFAM.NE.5.AND.IXAX.NE.5.AND.IFAM.NE.2.AND.IXAX.NE.2)
                +     WRITE(6,1040) UPBJ
ISN 0040         WRITE(6,3000) FB,G,BW,FCM,A,B,D
ISN 0041         CALL TWRITE(PDEF,LDEF)
ISN 0042         CALL TREADC(KEY,LD,4)
ISN 0043         IF(KEY.EQ.YES) GO TO 50
           C
           C   DEFAULTS NOT ACCEPTED - GET DESIRED VALURS
           C
           C   GET NBR OF JAMMED CHANNELS
           C
ISN 0045         WRITE(6,2000)
ISN 0046         IF(IFAM.EQ.2.OR.IXAX.EQ.2) GO TO 20
ISN 0048         IF(JTP.EQ.2) GO TO 20
ISN 0050         CALL TWRITE(PNJAM,LNJAM)
ISN 0051         CALL IREAD(NJAM)
           C
           C   GET SIG-TO-NOISE RATIO
           C
ISN 0052      20 IF(IFAM.EQ.3.OR.IXAX.EQ.3) GO TO 30
ISN 0054         CALL TWRITE(PBETA,LBETA)
```

```
ISN 0055          CALL FREAD(BETA1)
            C
            C   GET JAM-TO-SIG RATIO
            C
ISN 0056       30 IF(IPAM.EQ.4.OR.IXAX.EQ.4) GO TO 40
ISN 0058          CALL TWRITE(PGAMMA,LGAMMA)
ISN 0059          CALL FREAD(GAMMA)
            C
            C   PORTION OF BAND JAMMED
            C
ISN 0060       40 IF(IPAM.EQ.2.OR.IXAX.EQ.2) GO TO 45
ISN 0062          IF(IPAM.EQ.5.OR.IXAX.EQ.5) GO TO 45
ISN 0064          IF(JTP.NE.2) GO TO 45
ISN 0066          CALL TWRITE(PUPB,LUPB)
ISN 0067          CALL FREAD(UPBJ)
            C
            C   GET REMAINING PARAMETERS
            C
ISN 0068       45 CALL TWRITE(PFB,LFB)
ISN 0069          CALL FREAD(FB)
ISN 0070          CALL TWRITE(PG,LG)
ISN 0071          CALL FREAD(G)
ISN 0072          CALL TWRITE(PBW,LBW)
ISN 0073          CALL FREAD(BW)
ISN 0074          IF(ISLOW.EQ.1.OR.NBPW.NE.1) GO TO 47
ISN 0076          CALL TWRITE(PFCM,LFCM)
ISN 0077          CALL FREAD(FCM)
ISN 0078       47 CALL TWRITE(PA,LPA)
ISN 0079          CALL FREAD(A)
ISN 0080          CALL TWRITE(PB,LPB)
ISN 0081          CALL FREAD(B)
ISN 0082          CALL TWRITE(PD,LPD)
ISN 0083          CALL FREAD(D)
            C
            C   CALC CHIPS-PER-BIT IF NEEDED
            C
ISN 0084       50 IF(NBPW.GT.1.OR.ISLOW.EQ.1) RETURN
ISN 0086          C=FCM/FB
ISN 0087          NBPC=C
ISN 0088          IF(NBPC/2.EQ.(NBPC+1)/2) NBPC=NBPC-1
ISN 0090          IF(NBPC.LT.0) NBPC=1
            C
```

```
                  C   SET FAMILY, X-AXIS VARIABLES IF CHIPS-PER-BIT
                  C
ISN 0092                IF(IFAM.EQ.1) CALL FAMC
ISN 0094                IF(IXAX.EQ.1) CALL XAXC
ISN 0096                RETURN
ISN 0097           1000 FORMAT(/'    *** VARIABLE DEFAULTS ***'/)
ISN 0098           1010 FORMAT(4X,'NBR OF JAMMED CHANNELS (NJAM) = ',I4)
ISN 0099           1020 FORMAT(4X,'SNR (BETA1,FSK) = ',F6.1,' DB')
ISN 0100           1030 FORMAT(4X,'JSR (GAMMA) = ',F6.1,' DB')
ISN 0101           1040 FORMAT(4X,'PORTION JAMMED (UPBJ) = ',F6.4)
ISN 0102           3000 FORMAT(4X,'INFO BIT BANDWIDTH (FB) = ',F6.1,' KHZ'/
                       +      4X,'PN PROCESSING GAIN (G) = ',F6.1,' DB'/
                       +      4X,'HOPPING BANDWIDTH (BW) = ',F6.1,' MHZ'/
                       +      4X,'CHIP BANDWIDTH (FCM) = ',F6.1,' KHZ'/
                       +      4X,'HOPPING OVERLAP FACTOR (A) = ',F6.4/
                       +      4X,'COMM-TO-JAM DIFF FREQ FACTOR (B) = ',F6.4/
                       +      4X,'PN RCVR LOSS FACTOR (D) = ',F6.4/)
ISN 0103           2000 FORMAT(1X)
ISN 0104                END

ISN 0002                SUBROUTINE TABOUT
ISN 0003                INTEGER JTYPE(9)/'NARR','OW B','AND ','PART','IAL ','BAND',
                       +     'REPE','ATER','    '/
ISN 0004                INTEGER MODE(2)/'BIT ','WORD'/, RATE(2)/'FAST','SLOW'/
ISN 0005                COMMON / INPAR / NJAM,BETA1,GAMMA,FB,BW,FCM,A,B,D,NBPC,IWN,IPP,
                       +                 G,NBPW,NERR,JTP,UPBJ,IEST,NCH,ICON,ISAME,ISLOW
ISN 0006                COMMON / FAMPAR / IFAM,NFAM,FAMV(9)
ISN 0007                COMMON / XAXPAR / IXAX,XAXS,XAXE,XAXI,NPTSX
ISN 0008                COMMON / LABELS / LAB(35),LAB2(35),LAB3(140),LABY2(16),
                       +                  LABY(4),LEN(5),LENY
                  C
                  C   CLEAR SCREEN
                  C
ISN 0009                CALL MOVABS(0,760)
ISN 0010                CALL ERASE
ISN 0011                CALL ANMODE
ISN 0012                WRITE(6,1000)
                  C
                  C   BEGIN OUTPUT
                  C
ISN 0013                MODEW=MIN0(2,NBPW)
ISN 0014                LINL=(JTP-1)*3+1
ISN 0015                IF(IPP.EQ.1) WRITE(6,15000)
```

```
ISN 0017          WRITE(6,12000) RATE(ISLOW+1)
ISN 0018          WRITE(6,2000) JTYPE(LIML),JTYPE(LIML+1),JTYPE(LIML+2),MODE(MODEW)
ISN 0019          IF(NBPW.GT.1) WRITE(6,3000) NBPW,NERR
ISN 0021          IF(IFAM.NE.1.AND.IXAX.NE.1) WRITE(6,4000) NBPC
ISN 0023          IF(IFAM.NE.2.AND.IXAX.NE.2.AND.IFAM.NE.5.AND.IXAX.NE.5)
                 +     WRITE(6,5000) NJAM
ISN 0025          IF(IFAM.NE.3.AND.IXAX.NE.3) WRITE(6,6000) BETA1
ISN 0027          IF(IFAM.NE.4.AND.IXAX.NE.4) WRITE(6,7000) GAMMA
ISN 0029          IF(IFAM.NE.5.AND.IXAX.NE.5.AND.IFAM.NE.2.AND.IXAX.NE.2
                 +     .AND.JTP.EQ.2) WRITE(6,11000) UPBJ
ISN 0031          WRITE(6,8000) FB,G,BW
ISN 0032          IF(ISLOW.NE.1.AND.NBPW.EQ.1) WRITE(6,13000) FCM
ISN 0034          WRITE(6,14000) A,B,D
               C
               C  FAMILY, X-AXIS OUTPUT
               C
ISN 0035          LIML=(IFAM-1)*7+1
ISN 0036          LIMU=LIML+6
ISN 0037          IF(IFAM.GT.0) WRITE(6,9000) (LAB2(I),I=LIML,LIMU),
                 +      (FAMV(I),I=1,NFAM)
ISN 0039          LIML=(IXAX-1)*7+1
ISN 0040          LIMU=LIML+6
ISN 0041          WRITE(6,10000) (LAB2(I),I=LIML,LIMU),XAXS,XAXE
               C
               C  CONVERT UNITS ON PARAMETERS
               C
ISN 0042          BETA1=10.**(BETA1/10.)/NBPW
ISN 0043          GAMMA=10.**(GAMMA/10.)
ISN 0044          FB=1000.*FB
ISN 0045          G=10.**(G/10.)

ISN 0046          BW=1.E6*BW
ISN 0047          CALL TREAD(IDUM,LD,4)
ISN 0048          RETURN
               C
               C  FORMATS
               C
ISN 0049        1000 FORMAT(//)
ISN 0050        2000 FORMAT(' JAMMER TYPE = ',3A4,8X,'ERROR RATE = ',A4/)
ISN 0051        3000 FORMAT(3X,I2,' BITS PER WORD',8X,I2,' WRONG FOR WORD ERROR'/)
ISN 0052        4000 FORMAT(4X,'NBR OF CHIPS PER WORD = ',I4)
ISN 0053        5000 FORMAT(4X,'NBR OF JAMMED CHANNELS = ',I4)
ISN 0054        6000 FORMAT(4X,'SIGNAL-TO-NOISE RATIO = ',F6.1,' DB')
```

```
ISN 0055     7000 FORMAT(4X,'JAMMING-TO-SIGNAL RATIO = ',F6.1,' DB')
ISN 0056     8000 FORMAT(4X,'INFO BIT BANDWIDTH = ',F5.1,' KHZ'/
                +        4X,'PN PROCESSING GAIN = ',F5.1,' DB'/
                +        4X,'HOPPING BANDWIDTH = ',F5.1,' MHZ')
ISN 0057    13000 FORMAT(4X,'CHIP BANDWIDTH = ',F5.1,' KHZ')
ISN 0058    14000 FORMAT(4X,'HOPPING OVERLAP FACTOR = ',F4.2/
                +        4X,'COMM-TO-JAM DIFF FREQ FACTOR = ',F4.2/
                +        4X,'PN RCVR LOSS FACTOR = ',F4.2)
ISN 0059     9000 FORMAT(/4X,'FAMILY: ',7A4/8X,'VALUES WERE: ',9(F5.1,','))
ISN 0060    10000 FORMAT(/4X,'X-AXIS: ',7A4/8X,'START: ',F6.1,4X,'END: ',F6.1)
ISN 0061    11000 FORMAT(4X,'PORTION JAMMED = ',F5.2)
ISN 0062    12000 FORMAT(' HOPPING RATE = ',A4/)
ISN 0063    15000 FORMAT(' COHERENT PSK MODULATION'/)
ISN 0064          END

ISN 0002          SUBROUTINE PCALC(SUBR,IOFF)
ISN 0003          EXTERNAL SUBR
ISN 0004          COMMON / PLOT / X(100),Y(100,9)
ISN 0005          COMMON / INPAR / NJAM,BETA1,GAMMA,FB,BW,FCM,A,B,D,NBPC,IWN,IFP,
                +                  G,NBPW,NERR,JTP,UPBJ,IEST,NCH,ICON,ISAME,ISLOW
ISN 0006          COMMON / FAMPAR / IFAM,NFAM,FAMV(9)
ISN 0007          COMMON / XAXPAR / IXAX,XAXS,XAXE,XAXI,NPTSX
            C
            C   CALC PROB OF ERROR CURVES
            C
ISN 0008          DO 30 I=1,NFAM
ISN 0009             IF(IFAM.EQ.0) GO TO 10
ISN 0011             IF(IFAM.EQ.1) NBPC=FAMV(I)+0.1
ISN 0013             IF(IFAM.EQ.2) NJAM=FAMV(I)+0.1
ISN 0015             IF(IFAM.EQ.3) BETA1=10.**(FAMV(I)/10.)/NBPW
ISN 0017             IF(IFAM.EQ.4) GAMMA=10.**(FAMV(I)/10.)
ISN 0019             IF(IFAM.EQ.5) UPBJ=FAMV(I)
ISN 0021       10    DO 20 J=1,NPTSX
ISN 0022                IF(IXAX.EQ.1) NBPC=X(J)+0.1
ISN 0024                IF(IXAX.EQ.2) NJAM=X(J)+0.1
ISN 0026                IF(IXAX.EQ.3) BETA1=10.**(X(J)/10.)/NBPW
ISN 0028                IF(IXAX.EQ.4) GAMMA=10.**(X(J)/10.)
ISN 0030                IF(IXAX.EQ.5) UPBJ=X(J)
ISN 0032                Y(J,I+IOFF*NFAM)=SUBR(IFAM,IXAX)
ISN 0033       20    CONTINUE
ISN 0034       30 CONTINUE
ISN 0035          RETURN
ISN 0036          END
```

```
ISN 0002        FUNCTION FHOP(IFAM,IXAX)
ISN 0003        EXTERNAL Q,QINT
ISN 0004        REAL*8 Q,PI,QVAL
ISN 0005        REAL JNR, JNROLD, LOGCOM, S(3)
ISN 0006        DATA PI/3.14159265350D0/
ISN 0007        DATA JNROLD/0./, SNROLD/0./
ISN 0008        COMMON / QVAR / SNR,JNR
ISN 0009        COMMON / INPAR / NJAM,BETA1,GAMMA,FB,BW,PCM,A,B,D,NBPC,IWN,IFP,
               +                    G,NBPW,NERR,JTP,UPBJ,IEST,NCH,ICON,ISAME,ISLOW
             C
             C  CALC FREQ HOPPING VARIABLES
             C
ISN 0010        FC=NBPC*FB/NBPW
ISN 0011        NCH=A*BW/FC
ISN 0012        IF(JTP.EQ.2.AND.IFAM.NE.2.AND.IXAX.NE.2) NJAM=UPBJ*NCH
ISN 0014        IF(JTP.EQ.2.AND.(IFAM.EQ.2.OR.IXAX.EQ.2)) UPBJ=FLOAT(NJAM)/NCH
             C
             C  CHECK FOR VALID VALUES
             C
ISN 0016        IF(2*NBPC.GT.NCH) GO TO 90
ISN 0018        IF(NJAM.GT.NCH) GO TO 100
ISN 0020        IF(NBPW.LE.1) NERR=(NBPC+2)/2
             C
             C  CALC MORE FREQ HOPPING VARIABLES
             C
ISN 0022        GAMMAT=GAMMA
ISN 0023        IF(ICON.EQ.1.AND.NJAM.GT.0) GAMMAT=GAMMA/NJAM
ISN 0025        ALPHA1=BETA1*GAMMAT
ISN 0026        SNR=BETA1*FB/PC
ISN 0027        JNR=GAMMAT*SNR
ISN 0028        PERR=0.
             C
             C  CALC INTERMEDIATE PROBS FOR JAMMING TYPES
             C
ISN 0029        IF(IFP.EQ.1) GO TO 120
ISN 0031        IF(JTP.EQ.2) GO TO 10
             C
             C  PROBS FOR NARROW-BAND, REPEATER
             C
ISN 0033        IF(JTP.EQ.3.AND.IWN.EQ.1) GO TO 110
```

```
ISN 0035          P1=0.                                                          00075500
ISN 0036          ROOT=SQRT(JNR*SNR)                                             00075600
ISN 0037          IF(ROOT.GT.174.) GO TO 5                                       00075700
ISN 0039          CALL I0(ROOT,BESI)                                             00075800
ISN 0040          P1=0.5*EXP((-JNR-SNR)/2.)*BESI                                 00075900
ISN 0041        5 IF(JTP.EQ.3) GO TO 70                                          00076000
ISN 0043          P2=Q(SQRT(JNR),SQRT(SNR))-P1                                   00076100
ISN 0044          IF(IEST.LE.0) P4=0.                                            00076200
ISN 0046          IF(IEST.LE.0) GO TO 20                                         00076300
ISN 0048          IF(ABS(JNROLD-JNR).LT.ABS(JNR*0.001).AND.                      00076400
               +     ABS(SNROLD-SNR).LT.ABS(SNR*0.001)) GO TO 20                 00076500
ISN 0050          SNROLD=SNR                                                     00076600
ISN 0051          JNROLD=JNR                                                     00076700
ISN 0052          CALL NLS(QINT,0.D0,2.D0*PI,-1.E-5,1,200,QVAL,ERRQ,NFUN,IER)    00076800
ISN 0053          P4=0.5+QVAL/(4.*PI)                                            00076900
ISN 0054          GO TO 20                                                       00077000
             C                                                                   00077100
             C    PROBS FOR PARTIAL BAND                                         00077200
             C                                                                   00077300
ISN 0055       10 P1=EXP(-SNR/(2.+JNR))/(2.+JNR)                                 00077400
ISN 0056          P2=(1.+JNR)*P1                                                 00077500
ISN 0057          P4=0.5*EXP(-SNR/(2.+2.*JNR))                                   00077600
             C                                                                   00077700
             C    PROB FOR NARROW-BAND, PARTIAL BAND                             00077800
             C                                                                   00077900
ISN 0058       20 P3=0.5*EXP(-SNR/2.)                                            00078000
             C                                                                   00078100
             C    CALC PROB OF ERROR FOR NARROW, PARTIAL BAND                    00078200
             C                                                                   00078300
ISN 0059          IF(ISLOW.EQ.1) GO TO 85                                        00078400
ISN 0061          NJ1=MIN0(NJAM,NBPC)+1                                          00078500
ISN 0062          IL2=MAX0(NBPC-NCH+NJAM+1,1)                                    00078600
ISN 0063          IF(IL2.GT.NJ1) GO TO 65                                        00078700
ISN 0065          C1=LOGCOM(NCH,NJAM)                                            00078800
             C                                                                   00078900
             C    FAST HOPPING SUM - EQ 14 IN TORRIERI REPORT                    00079000
             C                                                                   00079100
ISN 0066          DO 60 IX=NERR,NBPC                                             00079200
ISN 0067             DO 50 I2=IL2,NJ1                                            00079300
ISN 0068                K=I2-1                                                   00079400
ISN 0069                IL3=MAX0(IL2,NJAM-K+2*NBPC-NCH+1)                        00079500
ISN 0070                NJ2=MIN0(NBPC,NJAM-K)+1                                  00079600
ISN 0071                IF(IL3.GT.NJ2) GO TO 50                                  00079700
```

```
ISN 0073                C2=LOGCOM(NBPC,K)
ISN 0074                DO 40 I3=IL3,NJ2
ISN 0075                   I=I3-1
ISN 0076                   MIN1=1+MINO(1,K)
ISN 0077                   I4L=MAXO(I+K-NBPC+1,1)
ISN 0078                   IF(IEST.GT.0.OR.JTP.NE.1) GO TO 25
ISN 0080                   MIN1=1
ISN 0081                   I4L=1
ISN 0082                   IF(I4L.LT.I+K-NBPC+1) GO TO 40
ISN 0084       25          IF(I4L.GT.MIN1) GO TO 40
ISN 0086                   C3=LOGCOM(NCH-2*NBPC,NJAM-K-I)
ISN 0087                   DO 30 I4=I4L,MIN1
ISN 0088                      IQ=I4-1
ISN 0089                      PC1=EXP(LOGCOM(K,IQ)+LOGCOM(NBPC-K,I-IQ)+C2+C3-C1)
ISN 0090                      PTEMP=PXKI(IX,K,I,IQ,NBPC,P1,P2,P3,P4,
              +                     IEST,JTP)*PC1
ISN 0091                      PERR=PERR+PTEMP
ISN 0092       30          CONTINUE
ISN 0093       40       CONTINUE
ISN 0094       50    CONTINUE
ISN 0095       60 CONTINUE
ISN 0096       65 FHOP=PERR
ISN 0097          RETURN
            C
            C    PROB OF ERROR FOR REPEATER
            C
ISN 0098       70 PERR=0.
ISN 0099          DO 80 IX=NERR,NBPC
ISN 0100             PERR=PERR+BINOM(NBPC,IX,P1)
ISN 0101       80 CONTINUE
ISN 0102          FHOP=PERR
ISN 0103          RETURN
            C
            C    SLOW FREQ HOPPING
            C
ISN 0104       85 S(1)=P3
ISN 0105          S(2)=0.5*(P1+P2)
ISN 0106          S(3)=P4
ISN 0107          PA=LOGCOM(NCH,NJAM)
ISN 0108          NL=MAXO(0,NJAM+2-NCH)+1
ISN 0109          NU=MINO(2,NJAM)+1
ISN 0110          PERR=0.
ISN 0111          IF(NU.LT.NL) GO TO 89
```

```
ISN 0113          DO 88 IX=NERR,NBPC
ISN 0114             DO 86 IN=NL,NU
ISN 0115                N=IN-1
ISN 0116                PERR=PERR+COMB(2,N)*EXP(LOGCOM(NCH-2,NJAM-N)-PA)*
               +              BINOM(NBPC,IX,S(IN))
ISN 0117       86    CONTINUE
ISN 0118       88 CONTINUE
ISN 0119       89 FHOP=PERR
ISN 0120          RETURN
             C
             C   ERROR RETURNS
             C
ISN 0121       90 WRITE(6,1000) NBPC,NCH
ISN 0122          FHOP=0.
ISN 0123          RETURN
ISN 0124      100 WRITE(6,2000) NJAM,NCH
ISN 0125          FHOP=0.
ISN 0126          RETURN
             C
             C   WHITE NOISE REPEATER
             C
ISN 0127      110 FHOP=EXP(-BETA1/(2.+ALPHA1))/(2.+ALPHA1)
ISN 0128          RETURN
             C
             C   COHERENT PSK
             C
ISN 0129      120 S0=0.5*ERFC(SQRT(SNR))
ISN 0130          S1=0.5*ERFC(SQRT(SNR/(1.+ALPHA1)))
ISN 0131          PERR=0.
ISN 0132          DO 130 IX=NERR,NBPC
ISN 0133             TEMP=COMB(NBPC,IX)
ISN 0134             PERR=PERR+TEMP*(FLOAT(NJAM)/NCH*S1**IX*(1.-S1)**(NBPC-IX)+
               +          (1.-FLOAT(NJAM)/NCH)*S0**IX*(1.-S0)**(NBPC-IX))
ISN 0135      130 CONTINUE
ISN 0136          FHOP=PERR
ISN 0137          RETURN
             C
             C   FORMATS
             C
ISN 0138     1000 FORMAT('  ** C=',I6,' TOO LARGE, NCH=',I6)
ISN 0139     2000 FORMAT('  ** J=',I6,' TOO LARGE, NCH=',I6)
ISN 0140          END
```

```
ISN 0002          SUBROUTINE QINT(X,F)                                     00088300
              C                                                            00088400
              C   PROGRAM USED BY INTEGRATION ROUTINE NL9 TO EVALUATE      00088500
              C      Q FUNCTION INTEGRAL                                   00088600
              C                                                            00088700
ISN 0003          EXTERNAL Q                                               00088800
ISN 0004          REAL*8 X,F,Q,DARG                                        00088900
ISN 0005          REAL JNR                                                 00089000
ISN 0006          COMMON / QVAR / SNR,JNR                                  00089100
ISN 0007          A=SQRT(JNR)                                              00089200
ISN 0008          B=SQRT(SNR)                                              00089300
ISN 0009          DARG=JNR+SNR+2.*A*B*DCOS(X)                              00089400
ISN 0010          IF(DARG.LT.0.D0) DARG=0.D0                               00089500
ISN 0012          AB2=DSQRT(DARG)                                          00089600
ISN 0013          F=Q(A,AB2)-Q(AB2,A)                                      00089700
ISN 0014          RETURN                                                   00089800
ISN 0015          END                                                      00089900

ISN 0002          FUNCTION PXXI(X,K,I,Q,NBPC,P1,P2,P3,P4,IEST,JTP)         00090000
ISN 0003          INTEGER X,X1,X2,X3,Q                                     00090100
              C                                                            00090200
              C   CALC PROB SUMS                                           00090300
              C                                                            00090400
ISN 0004          J1=MINO(X,K-Q)+1                                         00090500
ISN 0005          IA=NBPC-I-K+Q                                            00090600
ISN 0006          SUM=0.                                                   00090700
ISN 0007          DO 30 IX1=1,J1                                           00090800
ISN 0008             X1=IX1-1                                              00090900
ISN 0009             PC3=BINOM(K-Q,X1,P1)                                  00091000
ISN 0010             J2=MINO(X-X1,I-Q)+1                                   00091100
ISN 0011             DO 20 IX2=1,J2                                        00091200
ISN 0012                X2=IX2-1                                           00091300
ISN 0013                PC4=BINOM(I-Q,X2,P2)                               00091400
ISN 0014                IBT=X-X1-X2                                        00091500
ISN 0015                I3L=MAX0(IBT-IA+1,1)                               00091600
ISN 0016                J3=MINO(IBT,Q)+1                                   00091700
ISN 0017                IF(I3L.GT.J3) GO TO 20                             00091800
ISN 0019                DO 10 IX3=I3L,J3                                   00091900
ISN 0020                   X3=IX3-1                                        00092000
ISN 0021                   PC5=1.                                          00092100
ISN 0022                   IF(IEST.GT.0.OR.JTP.NE.1) PC5=BINOM(Q,X3,P4)    00092200
ISN 0024                   IF(PC5.LT.0.) GO TO 10                          00092300
ISN 0026                   IB=IBT-X3                                       00092400
```

```
ISN 0027                    PC2=BINOM(IA,IB,P3)
ISN 0028                    SUM=SUM+PC2*PC3*PC4*PC5
ISN 0029         10      CONTINUE
ISN 0030         20    CONTINUE
ISN 0031         30 CONTINUE
ISN 0032            PXKI=SUM
ISN 0033            RETURN
ISN 0034            END
ISN 0002            FUNCTION PRPN(IFAM,IXAX)
               C
               C  ROUTINE WHICH CALCULATES ERROR RATE FOR PN
               C
ISN 0003            COMMON / INPAR / NJAM,BETA1,GAMMA,PB,BW,FCM,A,B,D,NBPC,IWN,IFP,
                   +                  G,NBPW,NERR,JTP,UPBJ,IEST,NCH,ICON,ISAME,ISLOW
ISN 0004            WC=FLOAT(NBPW)/FLOAT(NBPC)
ISN 0005            EBNO=BETA1*D
ISN 0006            ARG=B*GAMMA/G+1./EBNO
ISN 0007            PTEMP=0.5*ERFC(SQRT(WC/ARG))
ISN 0008            IF(PTEMP.EQ.0..OR.PTEMP.EQ.1.) GO TO 20
ISN 0010            IF(NBPW.LE.1) NERR=(NBPC+2)/2
ISN 0012            PSUM=0.
ISN 0013            DO 10 I=NERR,NBPC
ISN 0014               PSUM=PSUM+BINOM(NBPC,I,PTEMP)
ISN 0015         10 CONTINUE
ISN 0016            PRPN=PSUM
ISN 0017            RETURN
ISN 0018         20 PRPN=PTEMP
ISN 0019            RETURN
ISN 0020            END
ISN 0002            FUNCTION BINOM(I,J,P)
               C
               C  COMPUTES BINOMIAL COEFFICIENT
               C
ISN 0003            BINOM=1.
ISN 0004            IF((P.NE.0..OR.J.NE.0).AND.(P.NE.1..OR.J.NE.I))
                   +     BINOM=COMB(I,J)*P**J*(1.-P)**(I-J)
ISN 0006            RETURN
ISN 0007            END
               C
               C  SUBROUTINE FREAD READS A REAL NUMBER FORM THE TERMINAL
               C
ISN 0002            SUBROUTINE FREAD(FARG)
ISN 0003            LOGICAL*1 INPUT.NPT
```

```
ISN 0004          INTEGER PSIGN/'#   '/
ISN 0005          DIMENSION INPUT(21)
ISN 0006          EQUIVALENCE (INTEMP,INPUT(1))
ISN 0007          DATA NPT/'.'/
ISN 0008          CALL TREAD(INPUT,INLEN,20)
ISN 0009          IF(INLEN.LE.0) RETURN
ISN 0011          IF(INTEMP.EQ.PSIGN) CALL FINITT(0,700)
ISN 0013          CALL STGLOC(LOC,INPUT,NPT,'EQ',1,INLEN)
ISN 0014          IF(LOC.NE.0) GO TO 10
ISN 0016          INPUT(INLEN+1)=NPT
ISN 0017          LOC=INLEN
ISN 0018       10 NDEC=INLEN-LOC
ISN 0019          FARG=FCNVT(INPUT,INLEN,NDEC)
ISN 0020          RETURN
ISN 0021          END
              C
              C   IREAD READS AN INTEGER FROM THE TERMINAL
              C
ISN 0002          SUBROUTINE IREAD(IARG)
ISN 0003          DIMENSION INPUT(5)
ISN 0004          INTEGER PSIGN/'#   '/
ISN 0005          CALL TREAD(INPUT,INLEN,20)
ISN 0006          IF(INPUT(1).EQ.PSIGN) CALL FINITT(0,700)
ISN 0008          IF(INLEN.GT.0) IARG=ICNVT(INPUT,INLEN)
ISN 0010          RETURN
ISN 0011          END
ISN 0002          REAL FUNCTION LOGCOM(I,J)
              C
              C   COMPUTES NATURAL LOG OF COMBINATION OF (I,J)
              C
ISN 0003          DATA PI/3.14159265/
ISN 0004          STIR(L)=ALOG(SQRT(2.*PI*L))+L*ALOG(FLOAT(L))-L
ISN 0005          LOGCOM=0.
ISN 0006          IF(J.GT.I.OR.J.LT.0) GO TO 30
ISN 0008          IF(I.EQ.J.OR.J.EQ.0) RETURN
ISN 0010          LLIM=MAX0(J,I-J)+1
ISN 0011          F=1.
ISN 0012          DO 10 K=LLIM,I
ISN 0013             F=F/FLOAT(I-K+1)*FLOAT(K)
ISN 0014             IF(F.GT.1.E60) GO TO 20
ISN 0016       10 CONTINUE
ISN 0017          LOGCOM=ALOG(F)
ISN 0018          RETURN
```

```
                C
                C   ASYMPTOTIC FORM
                C
ISN 0019          20 IF(I.GT.50) A=STIR(I)
ISN 0021             IF(I.LE.50) A=ALOG(FACT(I))
ISN 0023             IF(J.GT.50) B=STIR(J)
ISN 0025             IF(J.LE.50) B=ALOG(FACT(J))
ISN 0027             IF(I-J.GT.50) C=STIR(I-J)
ISN 0029             IF(I-J.LE.50) C=ALOG(FACT(I-J))
ISN 0031             LOGCOM=A-B-C
ISN 0032             RETURN
ISN 0033          30 WRITE(6,1000) I,J
ISN 0034             RETURN
ISN 0035        1000 FORMAT(' *** INVALID ARGS IN LOGCOM    I =',I5,'    J =',I5)
ISN 0036             END
ISN 0002             FUNCTION FACT(I)
                C
                C   COMPUTES I FACTORIAL
                C
ISN 0003             FACT=1.
ISN 0004             IF(I.LT.0) GO TO 20
ISN 0006             IF(I.LE.1) RETURN
ISN 0008             DO 10 J=1,I
ISN 0009                FACT=FACT*FLOAT(J)
ISN 0010          10 CONTINUE
ISN 0011             RETURN
ISN 0012          20 WRITE(6,1000) I
ISN 0013             RETURN
ISN 0014        1000 FORMAT(' *** INVALID ARG IN FACT    I =',I5)
ISN 0015             END
ISN 0002             BLOCK DATA
ISN 0003             COMMON / LABELS / LAB(35),LAB2(35),LAB3(140),LABY2(16),
                    +                LABY(4),LEN(5),LENY
ISN 0004             DATA LEN/14,22,26,28,22/, LENY/13/
ISN 0005             DATA LAB/'CHIP','S PE','R WO','RD  ','    ','    ','    ',
                    +        'NBR ','OF C','HANN','ELS ','JAMM','ED  ','    ',
                    +        'SIGN','AL/N','OISE',' PER',' WOR','D (D','B)  ',
                    +        'JAMN','ING-','TO-S','IGNA','L RA','TIO ','(DB)',
                    +        'PORT','ION ','OF B','AND ','JAMM','ED  ','    '/
ISN 0006             DATA LABY/'PROB',' OF ','ERRO','R   '/
ISN 0007             END
```

APPENDIX B.--SAMPLE PROGRAM DIALOGUE

APPENDIX B.—SAMPLE PROGRAM DIALOGUE

The program begins by asking if the user wishes an overview of the program (see fig. B-1). If the user answers "y," then the brief program description is displayed before the program proceeds to the question-answer part of the program. If the user answers "n," then the description is not displayed. (All user responses in the dialogue are underlined.)

```
TYPE Y FOR OVERVIEW OF PROGRAM. y

THIS PROGRAM ESTIMATES THE BIT ERROR RATE FOR FSK FREQUENCY
   HOPPING AND PN COMMUNICATION SYSTEMS.  NOISE AND JAMMING ARE
   CONSIDERED IN THE MODEL.  THIS PROGRAM MUST BE RUN FROM A TEKTRONIX
   GRAPHICS TERMINAL OR EQUIVALENT.

TO HALT EXECUTION OF THIS PROGRAM, TYPE $ IN RESPONSE
   TO ANY REQUEST FOR NUMERICAL INPUT.

TYPE RETURN TO CONTINUE ____
```

Figure B-1. Sample brief program description.

The question-answer program begins by displaying the amount of Central Processing Unit (CPU) time used by the previous analysis (if there was one), and then begins asking questions in order to define the analysis parameters (see fig. B-2). The type of jamming to be used is first determined. Next, the user is asked, "Constant jamming power?" If the answer to this question is yes, then the total amount of jamming power is held constant in any analyses which vary the number of jammed channels or the portion of the band to be jammed (either as the x-axis or the family variable). Thus, when varying the number of jammed channels or the portion of the band to be jammed, the program dilutes or concentrates the jamming power density accordingly, keeping the total jamming-to-signal ration (JSR) a constant over the total bandwidth. If the user answers "no" to this question, the jamming power density is held constant, and an increase or decrease in the number of jammed channels or portion of the band jammed will cause a corresponding increase or decrease in the total JSR over the total bandwidth.

The user is then asked to specify the data-coding parameters to be used. Requests are made for word length (the number of bits per word before coding), chips per word (where a "chip" is a coded bit and, thus, chips per word is the number of bits per word after coding), and chip error threshold (the minimum number of coded bits which must be in error in order to prevent the coding from correcting these errors). If the word length is specified to be one, then bit error rates are calculated and the remaining questions concerning coding are skipped. (It should be noted that bit error rates may be calculated for either uncoded data or repetition-coded data. Repetition coding and the method for specifying its use in the program are explained in the description of the third part of the question-answer portion, below). The user is then asked if frequency hopping and pseudonoise curves are to be plotted on the same set of axes or on two different sets. Following this, the user is asked "Do you want expanded integrals?" A "no" answer causes certain integrals used in the calculation of the bit error rates to be estimated rather than calculated. This estimation is used primarily in special cases either where the estimation is known to be valid or where the evaluation of the integrals is too costly in CPU time. Next, the user is asked if slow frequency hopping is to be analyzed instead of fast hopping. If slow hopping is selected, the user is finally asked if coherent phase shift keying (PSK) data modulation is to be analyzed instead of frequency shift keying (FSK) modulation (the default).

```
     TIME = 0:00.01?

JAMMING OPTIONS:
    1   NARROW BAND
    2   PARTIAL BAND
    3   REPEATER
JAMMING TYPE: 1

CONSTANT JAMMING POWER (Y OR N)? y

IF WORD LENGTH IS ONE BIT, BIT ERROR RATE IS CALCULATED.
  OTHERWISE WORD ERROR RATE IS CALCULATED.

WORD LENGTH IN BITS: 4
CHIPS PER WORD: 4
CHIP ERROR THRESHOLD: 1

HOPPING, PN CURVES ON SAME PLOT? y

DO YOU WANT EXPANDED INTEGRALS (Y OR N)? y

SLOW FREQ HOPPING? n
```

Figure B-2. Question-answer portion--part 1.

The second part of the question-answer portion of the program is shown in figure B-3. The user is asked to specify the parameters to be used for the x-axis and family, and their values or ranges. The x-axis increment is the interval between adjacent points evaluated along the x-axis in producing the data curves. If the x-axis limits and increments indicate that more than 100 points are to be evaluated, then the x-axis upper limit is changed so that only 100 points are evaluated.

```
VALID FAMILY AND X-AXIS VARIABLES ARE:

    0   NONE (FAMILY ONLY)
    1   NBR OF CHIPS PER BIT
    2   NBR OF JAMMED CHANNELS
    3   SIGNAL-TO-NOISE RATIO (FSK)
    4   JAMMING-TO-SIGNAL RATIO

FAMILY ($ TO STOP): 4
    NBR OF FAMILY CURVES: 3
    JSR IN DB: 10
    JSR IN DB: 20
    JSR IN DB: 30

X-AXIS: 3
    X-AXIS START: 13
    X-AXIS END: 26
    X-AXIS INCR: .25
```

Figure B-3. Question-answer portion--part 2.

The third part of the question-answer portion is shown in figure B-4. All variables not previously assigned values are displayed, along with their default values. If all these default values are satisfactory, then the user need only respond "yes" to the question, "Are defaults OK?" If the user answers "no," then he will be prompted for values for each of the parameters that may legally be specified at this time. If the user types only a return in response to any prompt, the default value for that parameter will be assumed.

The parameter values specified or defaulted in the third question-answer portion are used to determine certain other parameter values. The number of hopping channels available to the frequency hopping system to be analyzed is determined by

$$NCH = A * (BW * NBPW)/(NBPC * FB)$$

```
*** VARIABLE DEFAULTS ***

 NBR OF JAMMED CHANNELS (NJAM) =    1
 PORTION JAMMED (UPBJ) = 0.2500
 INFO BIT BANDWIDTH (FB) =   25.0 KHZ
 PN PROCESSING GAIN (G) =   30.0 DB
 HOPPING BANDWIDTH (BW) =   25.0 MHZ
 CHIP BANDWIDTH (FCM) =   25.0 KHZ
 HOPPING OVERLAP FACTOR (A) = 1.0000
 COMM-TO-JAM DIFF FREQ FACTOR (B) = 1.0000
 PN RCVR LOSS FACTOR (D) = 1.0000

ARE DEFAULTS OK (Y OR N)? n

NBR OF JAMMED CHANNELS:__
INFO BIT BANDWIDTH (FB IN KHZ):__
PN PROCESSING GAIN (G IN DB):__
HOPPING BANDWIDTH (BW IN MHZ):__
HOPPING OVERLAP FACTOR (A):__
COMM-TO-JAM DIFF FREQ FACTOR (B):__
PN RCVR LOSS FACTOR (D) :__
```

Figure B-4. Question-answer portion--part 3.

where NBPW and NBPC are the number of information bits per word and the number of coded bits per word, respectively, and where A (hopping channel overlap factor), BW (hopping bandwidth), and FB (information bit bandwidth) are set or defaulted in the third question-answer portion. In addition, repetition coding may also be specified in this portion. Repetition coding is used to improve bit error rates by transmitting each information bit a number of times with a change in frequency (in hopping systems) between each repetition, and using majority logic in the receiver to determine the true value of the information bit. The use of repetition coding is controlled by the values of FCM (coded bit bandwidth) and FB (information bit bandwidth) when bit error rates (rather than word error rates) have been previously specified. If FCM and FB are equal (which is the default) then no repetition coding is used, and simple bit error rates are calculated. If FCM and FB are not equal, then repetition coding is assumed, with the number of repetitions specified by

$$NBPC = FCM/FB.$$

NBPC is an integer and is therefore truncated to an integar value if the ratio of FCM and FB is not a whole number. If FCM is smaller than FB, then NBPC is set to one and the simple bit error rate is calculated.

Figure B-5 shows the tabular output produced by the program and summarizes the important analysis parameters and their values. The tabular output will remain on the display screen until the user hits the return key and allows the program to continue. Figure B-6 shows the analysis results in the form of a plot of the requested data. The solid line on the plot is actually an overlapping of the three family curves computed for frequency hopping. This overlapping occurred because the variation of the family variable (jamming-to-signal ratio) had no effect on the performance of the frequency hopping signal. The three family curves for pseudonoise modulation appear as dashed lines in order to differentiate them from the frequency hopping curves when both sets of curves are plotted on the same set of axes. The curves are not automatically labeled by the program with their corresponding family variable values because of the difficulty in determining where such labeling should be placed. Thus, the user must watch the display screen as the plot is generated in order to determine the correct family variable value for each curve (the curves are drawn in the order in which the family values were specified by the user in the question-answer portion of figure B-3). The plot will remain on the display screen until the user hits the return key. The program will then return to the first part of the question-answer portion (fig. B-2) to allow the user to begin a new analysis.

```
HOPPING RATE = FAST

JAMMER TYPE = NARROW BAND          ERROR RATE = WORD

   4 BITS PER WORD          1 WRONG FOR WORD ERROR

   NBR OF CHIPS PER WORD =      4
   NBR OF JAMMED CHANNELS =      1
   INFO BIT BANDWIDTH =  25.0 KHZ
   PN PROCESSING GAIN =  30.0 DB
   HOPPING BANDWIDTH =  25.0 MHZ
   HOPPING OVERLAP FACTOR = 1.00
   COMM-TO-JAM DIFF FREQ FACTOR = 1.00
   PN RCVR LOSS FACTOR = 1.00

   FAMILY: JAMMING-TO-SIGNAL RATIO (DB)
       VALUES WERE:  10.0, 20.0, 30.0,

   X-AXIS: SIGNAL/NOISE PER WORD (DB)
       START:   13.0    END:   26.0
```

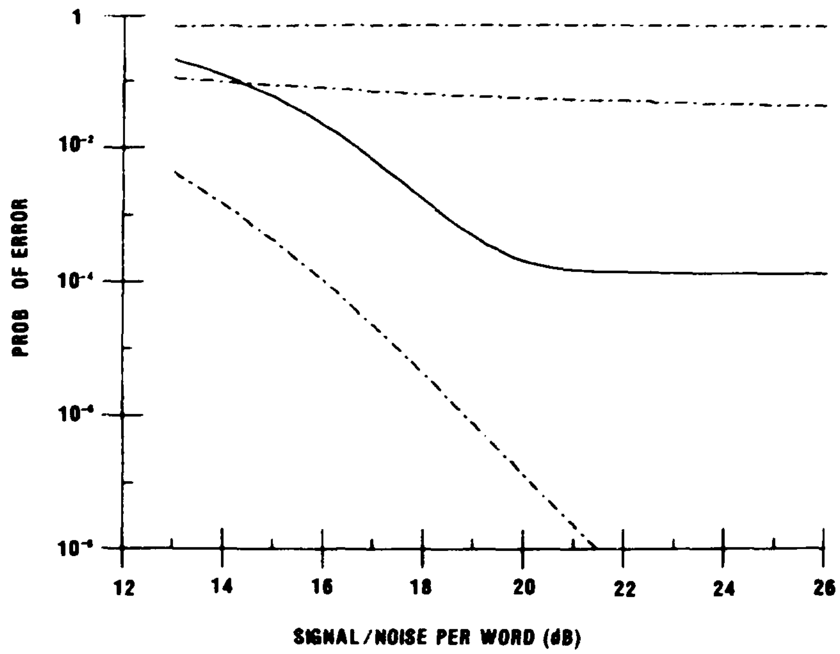

Figure B-6. Plot of analysis results.

APPENDIX C.--COMMON VARIABLES

APPENDIX C.--COMMON VARIABLES

All program variables of interest to the user are described in detail in the following appendix.

COMMON BLOCK PLOT

X(100) — Real array containing up to 100 x coordinates (i.e., x-axis values) to be plotted. The exact number of points in X is given by the variable NPTSX in COMMON Block XAXPAR. The values in X are set in subroutine DXAX (ISN 40-42) and are used in subroutine PCALC to calculate the error probability curves.

Y(100,9) — Real array containing up to nine family curves with up to 100 y coordinates (i.e., y-axis values) to be plotted. The exact number of family curves to be plotted is given by the variable NFAM in COMMON block FAMPAR. The exact number of points in each curve is given by the variable NPTSX in COMMON block XAXPAR. The values in Y are set in subroutine PCALC. This variable corresponds to the probability of error, P_w, in equation (14) of Torrieri.[1]

COMMON BLOCK INPAR

NJAM — Integer variable whose value is the number of distinct frequency hopping channels which are being jammed by CW (tone) or wide-band noise jammers. This variable may be the x-axis variable, the family variable, or a constant. NJAM must be less than the total number of hopping channels, and must not be less than zero. This variable corresponds to the number of channels jammed, j, in Torrieri.[1]

BETA1 — Real variable whose value is the received signal-to-noise ratio (SNR) per coded word in decibels. It is converted to SNR per coded bit in subroutine TABOUT (ISN 42) or subroutine PCALC (ISN 15 or 26), and is converted to SNR per uncoded (i.e., information) bit in subroutine FHOP (ISN 26). This variable may be the x-axis variable, the family variable, or a

---

[1] *D. J. Torrieri, Frequency Hopping in a Jamming Environment, U.S. Army Development and Readiness Command, DARCOM Report CM/CCM-78-2 (December 1978).*

constant. BETA1 corresponds to wR_s/N_{tu} for frequency hopping (Torrieri,[1] p 21) and to R_s/N_oB_m for PN modulation. [R_s/N_oB_m is related to N_o/E_b in equation (55) of Torrieri[2] by $E_b/N_o = \left(R_s/N_oB_m\right)B_mT_m$, where B_mT_m corresponds to program variable D]

GAMMA — Real variable whose value is the received jamming-to-signal ratio (JSR) in decibels. For frequency hopping, GAMMA may be either the JSR per frequency hopping channel or the JSR over a portion or all of the entire hopping bandwidth (i.e., the jammer power is spread out over a number of hopping channels), determined by the value of the variable ICON in this COMMON block. The value of ICON is set by the use in subroutine MODE. For PN modulation, GAMMA is the total JSR due to all jamming signals in the PN bandwidth. GAMMA may be the x-axis variable, the family variable, or a constant. GAMMA corresponds to R_j/R_s in Torrieri.[1,2]

FB — Real variable whose value is the information (uncoded) bit bandwidth in kilohertz (i.e., the information rate in kilobits per second of the transmitted bit stream if it were uncoded). This variable is a constant set by the user in subroutine PARSET, and corresponds to $2f_b$ in equation (3) and B_m in equation (48) of Torrieri.[1]

BW — Real variable whose value is the total bandwidth in megahertz available for frequency hopping communications. This variable is a constant set by the user in subroutine PARSET, and corresponds to B_o in equation (3) of Torrieri.[1]

FCM — Real variable whose value is the coded bit rate of the transmitted bit stream (i.e., the coded bit bandwidth) in kilohertz for repetition coding only. The number of coded bits per uncoded bit when repetition coding is used is calculated as NBPC = FCM/FB, where NBPC and FB are as specified in this COMMON block description. This variable (FCM) is a constant set by the user in subroutine PARSET, and corresponds to $2f_c$ in equation (3) of Torrieri.[1]

---

[1] *D. J. Torrieri, Frequency Hopping in a Jamming Environment, U.S. Army Development and Readiness Command, DARCOM Report CM/CCM-78-2 (December 1978).*

[2] *D. J. Torrieri, Pseudonoise Spread-Spectrum Systems in Communication Warfare, U.S. Army Development and Readiness Command, Report CM/CCM-79-9 (December 1979).*

A — Real variable whose value is a parameter which accounts for any hopping channel overlap or separation (A = 1 for adjacent channels, A > 1 for overlapping channels, A < 1 for separated channels). This variable is a constant set by the user in subroutine PARSET, and corresponds to "a" in equation (3) of Torrieri.[1]

B — Real variable whose value is a parameter expressing the type of jamming against PN systems. The worst type of jamming (with respect to the PN system) is continuous wave jamming at the center frequency of the PN transmission; in this case, B = 2. For other types of jamming, B < 2. This variable is a constant whose value is set by the user in subroutine PARSET, and corresponds to "b" in equation (55) of Torrieri.[2]

D — Real variable whose value is a parameter expressing the receiver losses in a PN system. The value of D typically varies between 1 and 2, with 2 being the worst case (with respect to the PN receiver). This variable is a constant whose value is set by the user in subroutine PARSET, and corresponds to "$B_m T_m$" (time-bandwidth product) in the expression for N_o/E_b as described in this section as part of the discussion of variable BETA1.

NBPC — Integer variable whose value is the number of "chips" (coded bits) per word. For example, if (7,4) coding is used (where 4 information bits are coded into 7 transmitted bits), NBPC = 7. If coding is not employed, NBPC = NBPW (see the description of NBPW in this COMMON block). If bit error rate is desired, NBPC = NBPW = 1. If repetition coding is used, NBPC is the number of times each bit is repeated. This variable is a constant (except in repetition coding, where it may be the family or x-axis variable) set by the user in subroutine MODE, and corresponds to "C" in equations (1) and (50) of Torrieri.[1]

---

[1] *D. J. Torrieri, Frequency Hopping in a Jamming Environment, U.S. Army Development and Readiness Command, DARCOM Report CM/CCM-78-2 (December 1978).*

[2] *D. J. Torrieri, Pseudonoise Spread-Spectrum Systems in Communication Warfare, U.S. Army Development and Readiness Command, Report CM/CCM-79-9 (December 1979).*

APPENDIX C

IWN	Integer variable whose value specifies the type of jammer modulation used by a repeater jammer. If IWN = 1, then white-noise modulation is used. If IWN = 0, then no modulation is used (i.e., the jammer merely retransmits its received signal). This variable is a flag set by the user in subroutine MODE.
IFP	Integer variable whose value specifies the use of coherent phase-shift-keyed (PSK) data modulation rather than frequency-shift-keyed (FSK) modulation for frequency hopping. If IFP = 1, then coherent PSK modulation is used; if IFP = 0, FSK is used. This variable is a flag set by the user in subroutine MODE.
G	Real variable whose value is the processing gain in decibels of the PN system. This variable is a constant set by the user in subroutine PARSET, and corresponds to G in equation (55) of Torrieri.[2]
NBPW	Integer variable whose value is the number of information (i.e., uncoded) bits per word in the data transmission. For example, if (7,4) coding is used, NBPW = 4. If NBPW = 1, then bit error rates are computed in the program; otherwise, word error rates are computed, and the user is issued requests for coded word length and coding parameters. This variable is a constant set by the user in subroutine MODE, and corresponds to "w" in equation (1) of Torrieri.[1]
NERR	Integer variable whose value is the smallest number of coded bits per word which must be received in error in order to cause a word error at the receiver. For example, (7,4) coding allows correct determination of the four information bits if no more than one of the seven coded bits is received in error. Thus, at least two bit errors must occur in a word in order for a word error to occur; NERR = 2 in this case. This variable is a constant set by the user in subroutine MODE, and corresponds to "r" in equation (2) and (50) of Torrieri.[1]

[1] *D. J. Torrieri, Frequency Hopping in a Jamming Environment, U.S. Army Development and Readiness Command, DARCOM Report CM/CCM-78-2 (December 1978).*

[2] *D. J. Torrieri, Pseudonoise Spread-Spectrum Systems in Communication Warfare, U.S. Army Development and Readiness Command, Report CM/CCM-79-9 (December 1979).*

JTP

Integer variable whose value specifies the type of jamming employed. The possible values of JTP are JTP = 1 (narrowband, or continuous wave, jamming); JTP = 2 (partial band or wideband jamming); JTP = 3 (repeater jamming). This variable is a flag set by the user in subroutine MODE.

UPBJ

Real variable whose value is the portion of the total hopping or PN bandwidth (BW or BS in this COMMON block) that is being jammed by one or more wideband noise jammers. This variable is functionally related to the variable NJAM (see the description of NJAM in this COMMON block). Thus, only one of the two may be set by the user--the other is determined from the value of the one that is set (see lines ISN 12 and 14 in subroutine FHOP). This variable may be the x-axis variable, the family variable, or a constant. The value of UPBJ must not be less than zero nor greater than one. This variable corresponds to "μ" in equation (46) of Torrieri.[1]

IEST

Integer variable whose value specifies whether or not an approximation to equation (14) in Torrieri[1] is used. If IEST = 1, the approximation is not used, and equation (14) is calculated in its entirety. If IEST = 0, then the summation over q in equation (14) is not performed (i.e., q is set to zero), and the value of P_3 in equation (13) is set to one. This approximation considerably speeds up program execution since S_2 in equation (13) need not be computed (the value of S_2 is determined by equation (28) and involves the time-consuming integration of a highly complex function). However, the accuracy of the results suffers, particularly where large numbers of jammers are present or a significant portion of the band is being jammed. The approximation is normally not used unless it is certain that no significant computation errors will occur. This variable is a flag set by the user in subroutine MODE.

---

[1] *D. J. Torrieri, Frequency Hopping in a Jamming Environment, U.S. Army Development and Readiness Command, DARCOM Report CM/CCM-78-2 (December 1978).*

APPENDIX C

NCH	Integer variable whose value is the number of hopping channels available to the frequency hopping system. This variable is calculated using equation (3) in Torrieri[1] (see ISN 11 in subroutine FHOP), and corresponds to the variable M in that equation.
ICON	Integer variable whose value specifies whether or not total jamming power is to remain constant (spread over either a portion of the band or a number of continuous wave jammers). If ICON = 0, total jamming power is not constant, and as the number of jammers (or portion of the band jammed) increases, the amount of jamming power increases proportionally. If ICON = 1, then changes in the number or bandwidth of the jammers does not change the total jamming power (i.e., more jammers result in less power per jammer; thus, jamming power is "diluted"). This latter case is used to determine optimum jamming strategies when the jammers are limited in total power available (this is the case in the real world). The variable ICON is a flag set by the user in subroutine MODE.
ISAME	Integer variable whose value determines whether or not frequency hopping and PN error probability curves are to be plotted on the same plot for comparison. If ISAME = 0, then frequency hopping curves are plotted separately from the PN curves. If ISAME = 1, then both sets of curves are plotted on the same plot. This variable is a flag set by the user in subroutine MODE.
ISLOW	Integer variable whose value specifies either fast or slow frequency hopping. If ISLOW = 0, then the equations for fast frequency hopping (see section 2 of Torrieri[1]) are used. If ISLOW = 1, then the equations for slow frequency hopping (section 3, reference 1) are used. This variable is a flag set by the user in subroutine MODE.

[1]*D. J. Torrieri, Frequency Hopping in a Jamming Environment, U.S. Army Development and Readiness Command, DARCOM Report CM/CCM-78-2 (December 1978).*

COMMON BLOCK FAMPAR

IFAM	Integer variable whose value determines which program variable is to be varied in order to produce a family (i.e., a series) of curves. The allowable family variables and their corresponding values of IFAM are none (IFAM = 0), number of chips per bit (NBPC, for repetition coding only) (IFAM = 1), number of jammed channels (NJAM, cw jamming only) (IFAM = 2), signal-to-noise ratio (BETA1) per word (IFAM = 3), jamming-to-signal ratio (GAMMA) (IFAM = 4), and portion of band jammed (UPBJ, for partial band jamming only) (IFAM = 5). This variable is a flag set by the user in subroutine DFAM.
NFAM	Integer variable whose value is the number of family curves to be calculated. NFAM may be no greater than nine. This variable is a constant set by the user in subroutine DFAM.
FAMV	Real array containing up to nine family variable parameter values. The exact number of parameter values in FAMV is given by the variable NFAM. A separate error probability curve is generated for each of the NFAM values in FAMV. The parameter to be assigned the values in FAMV is given by the variable IFAM. The actual assignment of the values to the parameter occurs in ISN 11-19 of subroutine PCALC. The values of FAMV are set by the user in subroutine DFAM.

COMMON BLOCK XAXPAR

IXAX	Integer variable whose value determines which program variable is to be the x-axis variable. The allowable x-axis variables and their corresponding values of IXAX are the same as for IFAM in COMMON block FAMPAR except that IXAX = 0 (no x-axis) is not allowed. The variable selected to be the x-axis by IXAX is the variable whose value is varied to calculate each point on an error probability curve. The values of the x-axis variable to be used by the program are calculated using the variables XAXS, XAXE, and XAXI in this COMMON block. The actual values are then placed in the array X in COMMON block PLOT; there are NPTSX (see this COMMON block) of these values. This variable is a flag set by the user in subroutine DXAX.

XAXI — Real variable whose value is the desired increment between successive x-axis variables. Thus, the n^{th} x-axis value (in array X) is calculated by adding XAXI to the $(n-1)^{th}$ x-axis value. It should be noted that since successive x-axis value are calculated by adding the increment to the previous x-axis value, the number of x-axis values will exceed 100 (the maximum allowed) if XAXI is less than one one-hundredth of the span of the x-axis (XAXE-XAXS). In this case the number of x-axis values is truncated to 100, and the x-axis will not reach XAXE. The variable XAXI is a constant set by the user in subroutine DXAX.

NPTSX — Integer variable whose value is the number of x-axis values that are in the array X (in COMMON block PLOT) to be plotted. This variable must not be less than one nor greater than 100. The value of NPTSX is calculated in ISN 35 of subroutine DXAX, using XAXS, XAXE, and XAXI.

COMMON BLOCK LABELS

LAB — Integer array dimensioned to 35 words and containing the five titles for the x-axis (see description of IXAX in COMMON block XAXPAR). Each title is seven words long, and the characters are packed four to a word. The titles are placed in the array at compile time in the BLOCK DATA subroutine (ISN 5). The title characters are initially in EBCDIC but are translated to ASCII in ISN 69 of entry point INITP (subroutine FHPLOT).

LAB2 — This array contains an exact copy of the contents of array LAB before LAB is translated to ASCII. LAB2 is used in subroutine TABOUT to display the family and x-axis variables to the user.

LAB3 — Integer array dimensiond to 140 and containing the unpacked x-axis titles. The titles are unpacked from four to a word (in array LAB) to one to a word (LAB3) in ISN 7 of entry point INITP (subroutine FHPLOT). Unpacking is performed after translation from EBCDIC to ASCII character codes has taken place.

LABY — Integer array containing the four-word y-axis title. The title is packed four characters to a word, and is initialy placed in LABY in the BLOCK

DATA subroutine (ISN 6). The characters are initially in EBCDIC but are translated to ASCII in ISN 71 of entry point INITP (subroutine FHPLOT).

LABY2 — Integer array dimensioned to 16 and containing the unpacked y-axis titles. The titles are unpacked from four to a word (in array LABY) to one to a word (LABY2) in ISN 72 of entry point INITP (subroutine FHPLOT). Unpacking is performed only after translation from EBCDIC to ASCII has taken place.

LEN — Integer array dimensiond to five words, containing the number of characters in each of the five x-axis titles. The index (i.e., subscript) of LEN required to retrieve the length of a specific title is the value of the variable IXAX (see COMMON block XAXPAR) for that title. LEN is assigned values in the BLOCK DATA subroutine.

LENY — Integer variable whose value is the number of characters in the y-axis title.

APPENDIX D.--LIBRARY ROUTINES

APPENDIX D.--LIBRARY ROUTINES

All library subroutines and functions required by the program are described in the following appendix.

GENERAL-PURPOSE ROUTINES

ELT1	Initializes CPU timer to 0 for determination of CPU time used in program execution.
ELT3 (ITIME)	Places CPU time used since last call to ELT1 into 12-byte array ITIME in format HH:MM:SS.FFF.
TREAD (INPUT, INLEN, LENBUF)	Reads characters from terminal into array INPUT. INLEN is the number of characters read, and LENBUF is the length of INPUT in bytes. If the number of characters typed is greater than LENBUF, then the input will be truncated to length LENBUF, and an error message will be printed.
PGMASK	Eliminates error message when exponent underflow occurs.
TREA (LEN,IBUF)	Translates LEN characters in array IBUF from EBCDIC to ASCII character codes. Characters must be packed one to a byte in IBUF.
UNPAK (LEN, IBUF1 IBUF2)	Unpacks LEN characters in array IBUF1 and places unpacked characters in array IBUF2 for use by plotting routines. Characters must be packed one to a byte in IBUF1, and will be placed in IBUF2 one to a word, right-justified. IBUF2 must be dimensioned to at least LEN words.
TWRITE (IBUF, LEN)	Writes LEN characters in array IBUF to terminal, without preceding or succeeding carriage return/line feed. Characters in IBUF are packed one to a byte. This routine is used mainly for input prompting. Example: CALL TWRITE ('INPUR LAMBDA:' 14)
IREAD (INTEG)	Reads a single-integer number from the terminal. This routine uses TREAD to read the terminal input, and then uses FORTRAN integer conversion routines. The integer input is

	placed in the variable INTEG on return. If the character '#' is entered as the first character from the terminal, IREAD executes a STOP command, halting execution of the program and returning control to the operating system.
TREADC (INPUT, INLEN, LENBUF)	Same as TREAD, except all lower-case characters in INPUT are forced to upper case.
FREAD (FLOAT)	Reads a single floating-point number from the terminal. This routine uses TREAD to read the terminal input, and then uses FORTRAN floating-point conversion routines. A decimal point or "E"-type exponent is optional. The floating-point number is placed in the variable FLOAT on return. If the character "#" is entered as the first character from the terminal, FREAD executes a STOP command, halting execution of the program and returning control to the operating system.
Q (A,B)	Function which returns the value of Marcum's Q-function for arguments A and B.
IO(ARG, RESULT)	Computes the zero-order modified Bessel function of ARG, returning the result in RESULT.
NL9 (SUBR, XL, XH, ACCUR, INT, LIM, RESULT, ERROR, NFUN, IER)	Estimates the integral of a function (calculated in subroutine SUBR, supplied by user) between limits XL and XH, and returns estimate in RESULT.
LOGCOM (I, J)	Function which computes $\log_e$ (I,J) where (I,J) = I!/[J! (I-J)!]. The real *4 result is returned as the value of the function LOGCOM. LOGCOM must be declared REAL *4.
COMB (I,J)	Function which computes (I,J) = I!/[J!(I-J)!]. The real *4 result is returned as the value of the function.
BINOM (I,J,P)	Function which computes the binomial coefficient BINOM = $(I,J)P^{J}(1 - P)^{I-J}$.
ERFC (X)	FORTRAN library function which computes the complementary error function for argument X. The result is returned as the value of the function.

FACT (I) — Function which return FACT(I) = I!.

<u>PLOTTING ROUTINES</u> — For Tektronix graphics terminals only.

INITT (IBAUD)	Initializes PLOT-10 library. IBAUD is the character transmission rate at which the terminal will communicate with the host computer (in characters per second). Must be called exactly once during program execution.
ERASE	Erases the screen of the Tektronix terminal.
ANMODE	Converts the terminal to the alphanumeric mode.
MOVABS (IX,IY)	Moves the cursor invisibly to the absolute screen position represented by the coordinates (IX,IY). This routine is used in order to position the cursor at the desired starting point of a visible line or character string.
BINITT	Resets PLOT-10 option flags back to their default values.
XFRM(I)	Selects options for x-axis major grid lines and tic marks based on value of argument I.
YFRM(I)	Selects options for y-axis major grid lines and tic marks based on value argument I.
XMFRM(I)	Selects options for x-axis minor tic marks based on value argument I.
YMFRM(I)	Selects options for y-axis minor tic marks based on value of argument I.
YTYPE(I)	Selects y-axis scale type (i.e., linear, logarithmic, etc.) based on value of argument I. For I = 2, logarithmic scaling is selected. Default is linear scaling.
NPTS(INPT)	Passes to the plot library the number of (x,y) data points to be plotted on a given plot. INPT is the number of points that are passed.

APPENDIX D

LINE(I)	Selects line type to be used by the plot library when connecting the points being plotted. I = 0 selects a solid line; I = 2 selects a dashed line.
DLIMX(XMIN,XMAX)	Passes to the plot library the minimum and maximum values of X coordinates of the points to be plotted.
DLIMY(YMIN,YMAX)	Passes to the plot library the minimum and maximum values of the Y coordinates of the points to be plotted.
CHECK (X,Y)	Scales the data to fit in the plotting window, calculates x- and y-axis labels and increments, performs other checking functions. X and Y are the arrays containing the X and Y coordinates of the data to be plotted. The number of points in X and Y to be plotted are specified by calling subroutine NPTS.
DISPLAY(X,Y)	Plots the (x,y) data on the screen, and draws the axes and axis labels.
CPLOT(X,Y)	Plots an additional set of (x,y) data points on an existing plot, scaling the new data to the existing plot and clipping points, if necessary. X and Y are arrays containing the X and Y coordinates of the additional data points.
JUSTER (LEN, LABEL, JUST, IFILL, LENF, IOFF)	Left-, right- or center-justifies (selected by JUST) the character string of length LEN in array LABEL (packed one character to a word). The length of the character string without fill characters (LENF) and the distance in screen raster units between the justification point (e.g., center) of the string and the starting point of the string (IOFF) are returned. This routine is used to center x-axis titles.
NOTATE (IX, IY, LEN, LABEL)	Writes the character string of length LEN in array LABEL (packed one to a word) at screen location (IX,IY). This routine is used to write x-axis titles.

VLABEL (LEN,LABEL)

Writes the character string of length LEN in array LABEL (packed one to a word) vertically (i.e., a line feed and backspace are performed between each character). The writing of the vertical label begins at the current cursor position, and may be specified by performing a MOVABS immediately prior to the call to VLABEL. This routine is used to write y-axis titles.

IBASEX(I)

Function which return as its value the address of the PLOT-10 x-axis COMMON variable specified by the value of the argument I. Used to retrieve previously set PLOT-10 options and variable values.

IBASEY(I)

Same as IBASEX, except for y-axis variables.

COMGET (IBASE)

Function which returns as its value the value of the COMMON variable whose address is the value of the argument IBASE (obtained by call to IBASEX or IBASEY). Used to retrieve PLOT-10 values such as axis lengths and locations.

APPENDIX E.--VARIABLE DEFAULT VALUES

PRECEDING PAGE BLANK-NOT FILMED

APPENDIX E.--VARIABLE DEFAULT VALUES

The following appendix names, describes, and gives the default values for certain program variables.

Variable Name	Variable Description	Default Value
A	Frequency hopping channel overlap factor.	1.
B	Signal-jamming center frequency difference factor.	2.
BETA1	Signal-to-noise ratio per word in decibels.	13.
BW	Frequency hopping bandwidth in Megahertz.	25.
D	PN receiver loss factor (also called the receiver time-bandwidth product).	1.
FB	Information bit bandwidth in Kilohertz.	25.
FCM	Chip (coded bit) bandwidth in Kilohertz.	25.
G	PN processing gain in decibels.	30.
GAMMA	Jamming-to-signal ratio in decibels.	0.
NBPC	Chips (coded bits) per word	1
NBPW	Word length in bits	1
NERR	Chip error threshold (the number of bit errors required in a coded word in order to cause a word error).	1
NJAM	Number of jammed channels.	1
UPBJ	Portion of band jammed (partial band jamming).	0.25

DISTRIBUTION

ADMINISTRATOR
DEFENSE DOCUMENTATION CENTER
ATTN DDC-TCA (12 COPIES)
CAMERON STATION, BUILDING 5
ALEXANDRIA, VA 22314

COMMANDER
US ARMY MISSILE & MUNITIONS
CENTER AND SCHOOL
ATTN ATSK-CTD-F
REDSTONE ARSENAL, AL 35809

DIRECTOR
US ARMY MATERIEL SYSTEMS ANALYSIS
ACTIVITY
ATTN DRXSY-MP
ATTN DRXSY-CT
ABERDEEN PROVING GROUND, MD 21005

DIRECTOR
DEFENSE ADVANCED RESEARCH PROJECTS
AGENCY
ATTN DIR, TACTICAL TECHNOLOGY OFFICE
ARCHITECT BUILDING
1400 WILSON BLVD
ARLINGTON, VA 22209

DIRECTOR
DEFENSE COMMUNICATIONS ENGINEERING
CENTER
ATTN R&D OFFICE, ASST DIR FOR TECH
1860 WIEHLE AVE
RESTON, VA 22090

DIRECTOR OF DEFENSE
RESEARCH & ENGINEERING
ATTN DEP DIR (TACTICAL WARFARE PROGRAM)
WASHINGTON, DC 20301

ASSISTANT SECRETARY OF THE ARMY
(RES, DEV, & ACQ)
ATTN DEP FOR COMM & TARGET ACQ
ATTN DEP FOR AIR & MISSILE DEFENSE
WASHINGTON, DC 20310

COMMANDER
US ARMY COMMUNICATIONS-ELEC. COMMAND
ATTN STEEP-MT-M
FORT HUACHUCA, AZ 85613

OFFICE, DEPUTY CHIEF OF STAFF FOR
OPERATIONS & PLANS
DEPARTMENT OF THE ARMY
ATTN DAMO-TCD, ELECTRONIC/WARFARE
SIGNAL SECURITY
ATTN DAMO-RQZ
WASHINGTON, DC 20310

COMMANDER
US ARMY CONCEPTS ANALYSIS AGENCY
8120 WOODMONT AVENUE
ATTN MDCA-SMS
BETHESDA, MD 20014

COMMANDER
US ARMY COMMUNICATIONS R&D COMMAND
ATTN DRSEL-CE, COMMUNICATIONS-ELECTRONIC
SYS INTEG OFFICE
FORT MONMOUTH, NJ 07703

DIRECTOR, ELECTRONIC WARFARE LABORATORY
ATTN DELEW-V
ATTN DELEW-C
ATTN DELEW-E
ATTN DELEW-M-ST
FORT MONMOUTH, NJ 07703

COMMANDER
ELECTRONICS WARFARE LABORATORY
OFFICE OF MISSILE ELECTRONIC WARFARE
WHITE SANDS MISSILE RANGE, NM 88002

COMMANDER
NAVAL WEAPONS CENTER
ATTN CODE 35, ELECTRONIC WARFARE DEPT
CHINA LAKE, CA 93555

DIRECTOR
NAVAL RESEARCH LABORATORY
ATTN CODE 5700, TACTICAL ELEC
WARFARE DIVISION
WASHINGTON, DC 20375

COMMANDER
NAVAL SURFACE WEAPONS CENTER
ATTN DF-20, ELECTRONICS WARFARE DIV
ATTN DK, WARFARE ANALYSIS DEPT
DAHLGREN, VA 22448

DIRECTOR
AF AVIONICS LABORATORY
ATTN EL (WR), ELECTRONIC WARFARE DIV
WRIGHT-PATTERSON AFB, OH 45433

COMMANDER
HQ, TACTICAL AIR COMMAND
ATTN DOR, DIR OF ELECTRONIC
WARFARE OPNS
LANGLEY AFB, VA 23665

COMMANDER
HQ USAF TACTICAL AIR WARFARE
CENTER (TAC)
ATTN ER, DCS/ELECTRONIC WARFARE
AND RECONNAISSANCE
ATTN ERW, DIR OF ELECTRONIC
WARFARE
EGLIN AFB, FL 32542

DISTRIBUTION (Cont'd)

US ARMY ELECTRONICS RESEARCH &
DEVELOPMENT COMMAND
ATTN TECHNICAL DIRECTOR, DRDEL-CT
ATTN DRDEL-CCM (3 COPIES)
ATTN DRDEL-ST
ATTN DRDEL-OP
ATTN HARRELSON, H. R., DRDEL-CCM
(20 COPIES)
ATTN HARMAN, R., DRDEL-MA
ADELPHI, MD 20783

INSTITUTE FOR DEFENSE ANALYSIS
400 ARMY NAVY DRIVE
ARLINGTON, VA 22209

DIA
DEP DIR OF SCIENTIFIC AND TECH INST
ELECTRONICS WARFARE BRANCH
1735 N. LYNN STREET
ARLINGTON, VA 22209

DEPT OF NAVY
OFFICE OF RES, DEV, TEST & EVAL
ATTN TACTICAL AIR SURFACE & EW DEV DIV
(NOP-982E5)
ATTN C&C EW AND SENSORS SEC
(NOP-982F3)
THE PENTAGON
WASHINGTON, DC 20350

COMMANDER
US ARMY TRAINING & DOCTRINE COMMAND
ATTN ATDC (DCS, COMBAT DEVELOPMENTS)
FT MONROE, VA 23651

OFFICE OF THE DEPUTY CHIEF OF STAFF
FOR RES, DEV, & ACQ
DEPARTMENT OF THE ARMY
ATTN DAMA-WS
ATTN DAMA-CS
ATTN DAMA-AR
ATTN DAMA-SCS, ELECTRONIC WARFARE TEAM
WASHINGTON, DC 20310

US ARMY COMBINED ARMS COMBAT DEV ACTIVITY
ATTN ATZLCA-CA
ATTN ATZLCA-CO
ATTN ATZLCA-FS
ATTN ATZLCA-SW
ATTN ATZLCA-COM-G
FT LEAVENWORTH, KS 66027

DIRECTOR
ELECTRONICS TECHNOLOGY & DEV LAB
ATTN DELET
FT MONMOUTH, NJ 07703

COMMANDER
US ARMY MATERIEL DEV & READINESS COMMAND
ATTN DRCPP
ATTN DRCPS
ATTN DRCDE
ATTN DRCDE-D
ATTN DRCBSI
5001 EISENHOWER AVENUE
ALEXANDRIA, VA 22333

DIRECTOR
US ARMY NIGHT VISION AND ELECTRO-OPTICS
LABORATORY
FT BELVOIR, VA 22060

COMMANDER
US ARMY COMBAT SURVEILLANCE AND
TARGET ACQUISITION LAB
FT MONMOUTH, NJ 07703

DIRECTOR
US ARMY SIGNALS WARFARE LAB
VINT HILL FARMS STATION
WARRENTON, VA 22186

COMMANDER
US ARMY INTELLIGENCE AND SECURITY COMMAND
ARLINGTON HALL STATION
ATTN IARDA (DCS, RDA)
ATTN IAITA (DIR, THREAT ANALYSIS)
4000 ARLINGTON BLVD
ARLINGTON, VA 22212

US ARMY TRADOC SYSTEMS ANALYSIS
ACTIVITY
ATTN ATAA-TDB
WHITE SANDS MISSILE RANGE, NM 88002

DIRECTOR
NATIONAL SECURITY AGENCY
ATTN S65
FT MEADE, MD 20755

HARRY DIAMOND LABORATORIES
ATTN 00100, COMMANDER/TECH DIR/TSO
ATTN CHIEF, 00210
ATTN CHIEF, 10000
ATTN CHIEF, 20000
ATTN CHIEF, 30000
ATTN CHIEF, 40000
ATTN RECORD COPY, 81200
ATTN HDL LIBRARY, 81100 (3 COPIES)
ATTN HDL LIBRARY, 81100 (WRF)
ATTN TECHNICAL REPORTS BR, 81300
ATTN SANN, K. H., 11110
ATTN DANDO, J., 21400