Technical Document 1753
March 1990

Specification for HF Lowest Usable Frequency (LUF) Model

T. N. Roy

NAVAL OCEAN SYSTEMS CENTER
San Diego, California 92152-5000

J. D. FONTANA, CAPT, USN
Commander

R. M. HILLYER
Technical Director

ADMINISTRATIVE INFORMATION

This work, sponsored by the Naval Oceanographic Office, was performed from June 1989 through September 1989.

Released by
J. A. Ferguson, Head
Ionospheric Branch

Under Authority of
J. H. Richter, Head
Ocean and Atmospheric
Sciences Division

ACKNOWLEDGMENTS

The author wishes to acknowledge the work of R.B. Rose, J.R. Hill, M.P. Bleiweiss, P.E. Argo, J.N. Martin, P.H. Levine, D.B. Sailors, and W.K. Moision, who developed and improved the HF lowest usable frequency (LUF) model.

CONTENTS

1.0 INTRODUCTION 1

2.0 LUF MODEL INPUT AND OUTPUT PARAMETERS 2

3.0 LUF MODEL COMPUTER PROGRAM, SUBROUTINES, AND FUNCTIONS 4

4.0 LUF MODEL ALGORITHMS 6

5.0 LUF MODEL TEST CASES 12

6.0 REFERENCES 16

APPENDIX: FORTRAN 77 PROGRAM AND SUBROUTINE LISTINGS FOR LUF MODEL. 17

TABLES

2-1. LUF model input and output parameters. 2

2-2. LUF model antenna types. 3

5-1. LUF model season results (MHz). 12

5-2. LUF model 1-8 Å X-ray flux results (MHz). 12

5-3. LUF model signal-to-noise ratio results (MHz). 13

5-4. LUF model results (MHz) for transmitter 75 degrees north latitude (1.3090 radians) and 150 degrees west longitude (2.6180 radians). 13

5-5. LUF model results (MHz) for transmitter 35 degrees north latitude (0.6109 radian) and 150 degrees west longitude (2.6180 radians). 13

5-6. LUF model results (MHz) for transmitter 0 degrees north latitude (0.0 radians) and 150 degrees west longitude (2.6180 radians). 13

5-7. LUF model results (MHz) for transmitter −35 degrees north latitude (−0.6109 radian) and 150 degrees west longitude (2.6180 radians). 14

5-8. LUF model results (MHz) for transmitter −75 degrees north latitude (−1.3090 radians) and 150 degrees west longitude (2.6180 radians). 14

5-9. LUF model results (MHz) for transmitter and receiver antenna types. 14

5-10. LUF model results (MHz) for transmitter and receiver antenna bearings. 15

1.0 INTRODUCTION

Any communication system operating in the high-frequency (HF) spectrum between 2 and 32 MHz is subject to certain physical limitations on the propagation of its signal. These limitations determine propagation boundaries which are unique and definable for any given point in time and over any path. During daylight undisturbed solar conditions, the lower boundary of the propagating spectrum, the lowest usable frequency (LUF), is determined by the amount of ionospheric D-region absorption. The LUF over any communications path is also a function of such HF system parameters as transmitted power, required signal-to-noise ratio of the receiver, and transmitter and receiver antenna gains. For this reason, LUF values will vary slightly for different paths and HF systems during undisturbed ionospheric conditions. When a solar flare occurs, the increased absorption in the D-region becomes the dominant factor in determining the LUF, causing it to rise. The amount it rises is proportional to the peak value of the X-ray enhancement and the solar zenith angle. A model developed for prediction of the LUF must be able to deal with these parameters for both undisturbed and disturbed ionospheric conditions.

Work on developing a model of the undisturbed LUF began at the Naval Ocean Systems Center in the early 1970's (Ref. 1 and 2). In 1977 a model of the LUF was developed which included seasonal, solar cycle, latitude, and diurnal effects as well as nonionospheric system-dependent parameters such as receiver signal-to-noise ratio and transmitted power (Ref. 3).

In 1979 a computer program called QLOF was developed for predicting the LUF for undisturbed ionospheric conditions (Ref. 4). In 1986 the model was improved to enable the calculated LUF to be adjusted for transmitter power, antenna gains, range, and required signal-to-noise ratio. This improved LUF model, called QLOF 2.0, also contained corrections to antenna takeoff angle calculations at ranges greater than 3300 km, LUF calculations for long paths, antenna gain calculations, and ionospheric absorption values at high latitudes. The accuracy of this model was evaluated in 1987 using a data base of oblique sounder data (Ref. 5).

A model for predicting the LUF for disturbed ionospheric conditions was developed in 1974 (Ref. 6). A computer program called DLOF used this model to determine the effects of short wave fade (SWF) disturbances caused by increases in solar X-ray radiation. HF oblique sounder data were used in 1986 to evaluate the accuracy of this model (Ref. 7).

Several user's manuals have been published which contain specific information on how to use the HF propagation prediction models developed at NOSC (Ref. 8 and 9).

2.0 LUF MODEL INPUT AND OUTPUT PARAMETERS

Input and output parameters and parameter limits for the LUF model are listed in Table 2-1.

Table 2-1. LUF model input and output parameters.

Parameter Description	Parameter Limits
INPUT	
Transmitter Latitude	$-\pi/2$ to $\pi/2$ radians (90 deg south to 90 deg north latitude)
Transmitter West Longitude	0 to 2π radians (0 deg to 360 deg west longitude)
Receiver Latitude	$-\pi/2$ to $\pi/2$ radians (90 deg south to 90 deg north latitude)
Receiver West Longitude	0 to 2π radians (0 deg to 360 deg west longitude)
Month	1 to 12
Day	1 to 31
Hour	0 to 23 (Universal time)
Minute	0 to 59 (Universal time)
Julian Day	1 to 366
Year	Example: 89 for 1989
Transmitter Antenna Bearing	0 deg to 360 deg (referenced to geographic north)
Receiver Antenna Bearing	0 degrees to 360 deg (referenced to geographic north)
Required Signal-to-Noise Ratio for the System (1KHz Bandwidth)	−30 dB minimum
1–8 Å Solar X-Ray Flux	1.0×10^{-6} to 1.0 erg/cm • s
OUTPUT	
Lowest Usable Frequency	2.0 to 48.0 MHz
Distance Between Transmitter and Receiver	0 to π radians
Latitude of the Path Midpoint	$-\pi/2$ to $\pi/2$ radians
West Longitude of the Path Midpoint	0 to 2π radians
Latitude of a Point 1000 km from Receiver	$-\pi/2$ to $\pi/2$ radians
West Longitude of a Point 1000 km from Receiver	0 to 2π radians
Latitude of a Point 1000 km from Transmitter	$-\pi/2$ to $\pi/2$ radians
West Longitude of a Point 1000 km from Transmitter	0 to 2π radians
Azimuth from Receiver to Transmitter	0 to 2π radians

The required signal-to-noise ratio depends upon the type of communications circuit being evaluated. Examples are given for several systems in Ref. 5. Values for the 1–8 Å solar X-ray flux can be obtained from the Space Environment Services Center (SESC) in Boulder, Colorado.

Antenna gain is determined from the antenna bearings and antenna types which are selected from a list of eleven antennas supported by the LUF model. Table 2-2 shows a listing of antenna types available.

Table 2-2. LUF model antenna types.

Antenna Number	Antenna Description
001	FRD-10 CDAA (receive only)
041	OE316/TSC99 Hermes LP (receive only)
101	Quarter Wave Vertical
102	Loaded Whip (Short)
121	Half Wave Horizontal Dipole
122	Inverted L
141	Terminated Rhombic
142	Terminated Sloping Vee
144	Horizontal LPA (54 degrees)
161	Vertical Log Periodic
000	Isotropic or Unknown

3.0 LUF MODEL COMPUTER PROGRAM, SUBROUTINES, AND FUNCTIONS

The lowest usable frequency is calculated by using the following procedure. A program called LUF passes input parameters, initializes model values, and receives the calculated LUF and geographic path information by calling subroutines GTABLE, GCRAZ, PATH, SUBSOL, QLOF, ADJUST, and DLOF and the function COSLAW. The LUF program initializes the antenna gain table by calling the subroutine GTABLE. If the antenna azimuth default flag is true, the transmitter and receiver antenna bearings are calculated, by using subroutine GCRAZ, so that the antennas are pointed toward each other. If the antenna azimuth default flag is false, then user input values for the antenna bearings are used. Values of transmitter and receiver antenna type, antenna bearings, transmitter power, and required signal-to-noise ratio are stored in common SYSDAT. Geographic path information is calculated next by using subroutine PATH. Subroutine SUBSOL then calculates the sun's subsolar point for the given time, and the function COSLAW is used to determine what part of the path is sunlit. If the 1–8 Å X-ray flux is less than 5.0 X 10^{-3} erg/cm^2 • s, subroutine QLOF is used to calculate the LUF; otherwise subroutine DLOF is used. Finally, if QLOF was used, subroutine ADJUST is called to adjust the calculated LUF for transmitter power, antenna gains, range, and required signal-to-noise ratio.

Subroutine PATH computes eight values of geographic path information for a given propagation path. The method assumes a spherical earth with a radius of 6371 km. Required input for this routine is the transmitter and receiver latitude and west longitude in radians. The following information, in radians, is returned by this subroutine: distance between transmitter and receiver, midpoint latitude, midpoint west longitude, latitude of a point 1000 km from the transmitter and receiver, west longitude of a point 1000 km from the transmitter and receiver, and azimuth from receiver to transmitter. Subroutines GCRAZ and RAZGC are called by PATH.

Subroutine GCRAZ computes the great circle range and azimuth between two points on the earth's surface in radians. Latitude and west longitude of the two points, in radians, are required as input. This computation assumes a spherical earth and recognizes the degenerate cases of a point at the North or South Pole and coincident points.

Subroutine RAZGC computes the latitude and west longitude in radians of a point at a specified distance and azimuth from a given point on the earth's surface. Latitude, west longitude, distance, and azimuth in radians of the given point to the new point are required as inputs. This computation assumes a spherical earth and recognizes the degenerate cases of the given point being at the North or South Pole and the case when distance is zero.

Subroutine SUBSOL computes the coordinates, in radians, of the sun's subsolar point for a given time. The required input to this routine is a six-element integer array containing month, day, hour (UT), minute (UT), Julian day, and year. The latitude and longitude of the subsolar point are stored in common SUN.

The function COSLAW performs the law of cosines for spherical triangles with arc lengths specified in radians.

Subroutine QLOF calculates the LUF for undisturbed ionospheric conditions, when the 1–8 Å X-ray flux is less than 5.0 X 10^{-3} erg/cm^2 • s. The required input is the first seven values of geographic path information calculated by the PATH subroutine and time information. This subroutine returns the calculated value of the LUF in megahertz. Subroutine ABSORB is called by subroutine QLOF.

Subroutine ABSORB calculates the ionospheric absorption index and solar zenith angle in radians at a specified position, given the values of latitude and longitude in radians at that position, and date, and time and the coordinates of the subsolar point at that time. Function CH is used by subroutine ABSORB.

The function CH calculates Chapman's grazing incidence integral for a parameter related to the atmospheric density and scale height and the solar zenith angle. The function CHPINT, which calculates the integrand of the Chapman integral, is used by the function CH.

Subroutine DLOF calculates the LUF for disturbed ionospheric conditions, when the 1–8 Å X-ray flux is greater than or equal 5.0×10^{-3} erg/cm^2 • s. The required input for this subroutine is the latitude and longitude of the transmission path end points in radians, 1–8 Å X-ray flux in erg/cm^2 • s and geographic path information. The subroutine returns the calculated value of the LUF in megahertz. Subroutines MINPT and NEWTON are called by subroutine DLOF.

Subroutine MINPT calculates both the minimum solar zenith angle over a specified propagation path and coordinates of the point along that path at which the minimum zenith angle occurs. The required input for this subroutine is the path length and receiver latitude and longitude in radians. The subroutine returns the minimum zenith angle and its location in radians. Subroutines GCRAZ and RAZGC are used by subroutine MINPT.

Subroutine NEWTON performs the Newton iteration function for a nonlinear equation. The values of the solar zenith angle in radians and the solar 1–8 Å X-ray flux in erg/cm^2 • s are input to this subroutine and the calculated value of the LUF in megahertz is returned.

Subroutine ADJUST adjusts the calculated LUF for transmitter power, transmitter and receiver antenna gains, range, and required signal-to-noise ratio. Input values for this subroutine are latitude and longitude and distance between the transmitter and receiver in radians. The transmitter and receiver antenna types, receiver antenna azimuth to the transmitter in degrees, the transmitter power in watts, and receiver signal-to-noise ratio are also used by this subroutine and are stored in common SYSDAT. This subroutine returns the adjusted value of the LUF in megahertz. Subroutines GCRAZ, GTABLE, and the function ANTFAC are called by ADJUST.

The function ANTFAC calculates the antenna correction factor, given the difference between the transmission path bearing and the azimuth at which the antenna is currently pointing and the antenna type. The values input to this function are the difference between the path bearing and antenna azimuth in radians and the antenna type. The function IDANT is used by ANTFAC. This integer function returns the index to the antenna table, given the antenna type.

Subroutine GTABLE calculates the mainlobe antenna gain in decibels, given the antenna type, range in radians, and frequency in megahertz. Values input to the subroutine are antenna type, range, and frequency, and antenna gain is returned. This routine also uses the values of antenna name, polarization, azimuth pattern, and short-range or long-range antenna pattern numeric identifier, which are stored in common ANTNAM.

4.0 LUF MODEL ALGORITHMS

For path lengths less than or equal to 3300 km, the expression for the LUF in the QLOF 2.0 model, used for undisturbed solar conditions (X-ray flux less than 5×10^{-3} erg/cm^2 • s), is given by

$$\mathrm{LUF} = K A_i^{1/2} \quad (1)$$

where K contains the transmission path length effects and A_i is the absorption index which contains the seasonal, latitudinal, and diurnal effects.

For path lengths greater than 3300 km, the LUF for the given propagation path is calculated by using

$$\mathrm{LUF} = \frac{K}{2}[2A_{i1} + A_{i2} + A_{i3}]^{1/2} \quad (2)$$

where A_{i1} is equal to the absorption index at the path midpoint, A_{i2} is equal to the absorption index 1000 km from the transmitter, and A_{i3} is equal to the absorption index 1000 km from the receiver. When undisturbed propagation conditions exist, this calculated value of the LUF is then adjusted for transmitter power, antenna gains, range and required signal-to-noise ratio.

For path lengths less than or equal to 2000 km, the equation for the K factor is given by

$$K = [E/40]^{1/2} \quad (3)$$

where

$$E = [1 - 0.9784/\{1 + [(C - 0.985)/B]^2\}]^{-1/2} \quad (4)$$

and

$$B = \sin\,[D/(2R_e)] \quad (5)$$

$$C = \cos\,[D/(2R_e)] \quad (6)$$

with D equal to the path length in km and R_e equal to the radius of the earth (6371 km).

For path lengths greater than 2000 km and less than or equal to 3300 km, the equation for K is

$$K = 0.045\,[4 + (1.875 \times 10^{-3})D] \quad (7)$$

For path lengths greater than 3300 km and less than or equal to 6600 km, the equation for K is

$$K = 0.045\,[7.5 + 0.001D] \quad (8)$$

For path lengths greater than 6600 km, the equation for K is

$$K = 0.045\,[7.5 + 0.001D] \bullet [1 - 0.3768(D_r - 1.0361)] \quad (9)$$

where D_r is the path length in radians.

The absorption index, A_i, is calculated as a function of solar zenith angle, χ, at each control point. For night time propagation conditions, the absorption index is set to a value of 1.0×10^{-11} dB • MHz. The equations for the solar zenith angle are

$$\chi_0 = \sin(\lambda_c) \sin(\lambda_s) + \cos(\lambda_c) \cos(\lambda_s) \cos(\omega_c - \omega_s) \tag{10}$$

for the interval $-1 \leq \chi_0 \leq 1$

and

$$\chi = \cos^{-1}(\chi_0) \tag{11}$$

where λ_c and ω_c are the latitude and longitude of the control point in radians, and λ_s and ω_s are the latitude and longitude of the subsolar point.

The solar zenith angle at noon at the control point is given by

$$\chi_n = \mathrm{ABS}(\lambda_s - \lambda_c) \tag{12}$$

The absorption index is set to 1×10^{-13} dB • MHz when the noontime solar zenith angle exceeds 89.95 degrees. For values less than 89.95 degrees, the absorption index is

$$A_i = A[\mathrm{CH}(921.0, \chi)/\mathrm{CH}(921.0, \chi_n)]^{-2m} \tag{13}$$

where

$$A = 286[1 + 0.5\lambda_c] W \cos^n(\chi_n) \tag{14}$$

and the CH is the Chapman function shown in general form in Eq. 24. The value $286\ (1 + 0.50\lambda_c)$ represents the latitudinal variation of the absorption index, W represents the winter anomaly effect, and $\cos^n \chi_n$ represents the seasonal variation with a latitudinal effect. The equation for the coefficient W is

$$W = 1.0 + 0.275\ [30.0 - \mathrm{ABS}(60.0 - \lambda_c) \tag{15}$$

The values of n used to calculate the seasonal variation factor are given for the following intervals by

$$n = 1.4 - 2.44\ \mathrm{ABS}(\lambda_c) \text{ for } \mathrm{ABS}(\lambda_c) < 0.45 \text{ radian} \tag{16}$$

$$n = 0.3 \quad \text{for } 0.45 \leq \mathrm{ABS}(\lambda_c) < 1.0875 \text{ radians} \tag{17}$$

$$n = -1.07\ [\mathrm{ABS}(\lambda_c) - 1.0875] + 0.3 \text{ for } 1.0875 \leq \mathrm{ABS}(\lambda_c) < 1.367 \text{ radians} \tag{18}$$

$$n = 0.0 \quad \text{for } \mathrm{ABS}(\lambda_c) \geq 1.367 \text{ radians} \tag{19}$$

In Eq. 14 the minimum value for A is limited to 1×10^{-11} in the QLOF 2.0 model. For a solar zenith angle greater than 103.1 degrees, the absorption index is

$$A_i = 0.01A \tag{20}$$

When the solar zenith angle is less than or equal to 103.1 degrees, the second parameter in Eq. 13 is used. This ratio of Chapman functions is evaluated for values related to the atmospheric density, scale height, solar zenith angle, and noontime solar zenith angle. Values for the factor m in Eq. 13 are calculated for the following intervals by

$$m = 0.5[0.58 + 0.08\lambda_d/18] \quad \text{for } 0 \text{ deg} < \lambda_d \leq 18 \text{ deg} \tag{21}$$

$$m = 0.5[0.66 + 0.22(\lambda_d - 18)/6] \quad \text{for } 18 \text{ deg} < \lambda_d \leq 24 \text{ deg} \tag{22}$$

$$m = 0.44 \quad \text{for } \lambda_d > 24 \text{ deg} \tag{23}$$

where λ_d is equal to the absolute value of the latitude of the control point in degrees.

The general form for the Chapman function, which is difficult to evaluate for some values of X and χ, is

$$\mathrm{CH}(X,\chi) = X\sin(\chi)\int_0^{\chi} \exp\,[X - X\sin(\chi)/\sin(\lambda)]\,\mathrm{cosec}^2(\lambda)\,d\lambda \tag{24}$$

where λ is equal to the latitude of the control point in radians. In the QLOF 2.0 model, four different approximations are used to evaluate the Chapman function, depending upon the value of χ. All approximations are accurate to better than 0.1 percent. The value of X is set to 921.0 in these approximations. The first approximation is

$$\mathrm{CH}(921,\chi) = \sec(\chi) \tag{25}$$

which is used in the interval $\chi < 41.8$ deg.

The second approximation uses a two-point Gaussian-Laguerre integral, which can be evaluated by first defining a function

$$\mathrm{CI}(Z) = \exp\,[2X\sin(Q)\cos(\chi + Q)/U + Z]/U^2 \tag{26}$$

where $Q = Z \cdot G/2$ (27)

$$U = \sin(Q + \chi) \tag{28}$$

and $G = (G_1 - \chi)/20$ (29)

where $G_1 = \tan^{-1}[G_0/(1 - G_0^2)^{1/2}]$ (30)

with $G_0 = X\sin(\chi)/[X + \ln(X) + 20]$ (31)

This function can then be used to approximate the Chapman integral using the equation

$$\mathrm{CH}(X,\chi) = X\sin(\chi) \cdot G \cdot [0.1464466\ \mathrm{CI}(3.414214) + 0.8535534\ \mathrm{CI}(0.5857864)] \tag{32}$$

which is used in the interval $41.8 \text{ deg} \leq \chi \leq 85.5 \text{ deg}$.

The third approximation uses a four-point Gaussian-Laguerre integral, which can be expressed as

$$\begin{aligned} CH(X,\chi) = X \sin(\chi) \bullet G \bullet [&0.5392947 \times 10^{-3} CI\,(9.395071) \\ &+ 0.03888791\ CI\,(4.536620) \\ &+ 0.3574187\ CI\,(1.745761) \\ &+ 0.6031541\ CI\,(0.3225477)] \end{aligned} \tag{33}$$

which is used in the interval $85.5\,\deg < \chi \leq 90\,\deg$.

The fourth approximation uses a truncated ten-point Gaussian-Laguerre integral, which approximates the Chapman integral using the equation

$$\begin{aligned} CH(X,\chi) = X \sin(\chi) \bullet G \bullet [&0.4249314 \times 10^{-6}\ CI\,(16.27926) \\ &+ 0.[illegible]25923 \times 10^{-4}\ CI\,(11.84379) \\ &+ 0.7530084 \times 10^{-3}\ CI\,(8.330153) \\ &+ 0.009501517\ CI\,(5.552496) \\ &+ 0.06208746\ CI\,(3.401434) \\ &+ 0.2180683\ CI\,(1.808343) \\ &+ 0.4011199\ CI\,(0.7294545) \\ &+ 0.3084411\ CI\,(0.1377935)] \end{aligned} \tag{34}$$

which is used in the interval $90\,\deg < \chi \leq 103.1\,\deg$.

For disturbed solar conditions (X-ray flux greater than $5 \times 10^{-3}\ \mathrm{erg/cm^2 \bullet s}$), the DLOF model is used to calculate the LUF. The equations used for transmission path lengths less than 3500 km are

$$LUF_1 = [F_X \cos^3(\chi_0)/1.03856 \times 10^{-6}]^{1/4} \tag{35}$$

where F_X is equal to the 1–8 Å solar X-ray flux, χ_0 is the minimum solar zenith angle over the propagation path, and

$$LUF = LUF_1[0.5368/\sin(\gamma_1)]^{1/2} \tag{36}$$

where

$$\gamma_1 = \tan^{-1}(\gamma) \tag{37}$$

$$\gamma = [\cos(\theta) - 0.96224]/\sin(\theta) \tag{38}$$

$$\theta = D_r/2 \tag{39}$$

and D_r is equal to the transmission path length in radians.

For transmission path lengths equal to or greater than 3500 km, the relationship between the LUF and solar X-ray flux was determined empirically. Curve-fitting techniques were used to determine the relationship, to a first order of approximation, to be

$$F_X - 0.01038(F_1 - 15.0) + 0.003 \sin[0.849(F_1 - 15.6)] = 0 \tag{40}$$

where

$$F_1 = LUF[1 + \sec^2(\chi)/10] \tag{41}$$

To find the LUF, Eq. 40 and 41 are solved by using Newton's method and the given values of χ and F_χ. Newton's method refines an initial guess, X_0, of a solution of a general nonlinear equation $f(x) = 0$ and takes the following form:

$$X_i + 1 = X_i \frac{-f(X_i)}{\frac{\partial}{\partial X} f(X)} \qquad (i = 0, 1, 2, \ldots) \tag{42}$$

where

$$X_0 = [F_\chi/0.1038 + 150]/[10 + \sec^2(\chi)] \tag{43}$$

In Eq. 42 if X = LUF and if Eq. 40 and 41 are used to evaluate the partial derivative of $f(X)$, then

$$\frac{\partial}{\partial X} f(X) = \frac{F_1\{0.01038 - 0.0025473 \cos[0.8491(F_1 - 15.6)]\}}{\mathrm{LUF}} \tag{44}$$

If Newton's method fails to converge after 20 iterations, the LUF is set to a maximum value of 50 MHz.

The LUF calculated for quiet solar conditions is adjusted for transmitter power, antenna gains, transmission path length, and required signal-to-noise ratio. This is done because the QLOF routine was calibrated by using oblique sounder system data, and the calculations must be adjusted for other systems. A ratio of signal loss margins is used to accomplish this adjustment. The signal loss margin (SLM) is the difference in decibels between the minimum usable signal at the receiver and the signal level expected at the same terminal under conditions of no ionospheric absorption. The expression for the adjusted LUF is

$$\mathrm{LUF}_2 = [\mathrm{SLM}_1/\mathrm{SLM}_2]^{1/2}\, \mathrm{LUF}_1 \tag{45}$$

where

LUF_1 = unadjusted LUF (MHz)

SLM_1 = signal loss margin for the calibration path (dB)

SLM_2 = signal loss margin for the new path (dB)

LUF_2 = adjusted LUF for the new path (MHz)

The equation for the signal loss margin for the calibration path is

$$\mathrm{SLM}_1 = 37.0 - 20 \log_{10}(d/4287) - 8.28 + 27.5 \log_{10}(f) \tag{46}$$

where d is equal to the transmission path distance in kilometers and f is set equal to the unadjusted LUF.

The equation for the signal loss margin for the new path is

$$\mathrm{SLM}_2 = 10\log_{10}(P_t) + G_t + G_r + 7.5 \log_{10}(f) - 20\log_{10}(d) + 111.55 - R_1 \tag{47}$$

where P_t is the transmitter power in watts, G_t is the transmitter antenna gain in decibels, G_r is the receiver antenna gain in decibels, and R_1 is the required signal-to-noise ratio for the communications circuit.

Receiver and transmitter antenna gains are determined from tables of antenna gains in subroutine GTABLE. The tables present antenna gains as a function of frequency and elevation angle (21 frequencies and six elevation angles per table). Elevation angles are calculated by using

$$\Delta = \tan^{-1} [\cos (D/2NR_e) - R_e(R_e + H)]/\sin (D/2NR_e) \tag{48}$$

where

H = reflectivity layer height (320 km)

N = number of hops

Δ = elevation angle (minimum 3.5 deg)

D = ground range of one hop

R = earth radius (6371 km)

Gain values are interpolated from the antenna tables by using a third-order Lagrangian interpolation formula:

$$\begin{aligned} G_{1,2} = {} & (\Delta - X_1)(\Delta - X_2)/[(X_0 - X_1)(X_0 - X_2)]g_0 \\ & + (\Delta - X_0)(\Delta - X_2)/[(X_1 - X_0)(X_1 - X_2)]g_1 \\ & + (\Delta - X_0)(\Delta - X_1)/[(X_2 - X_0)(X_2 - X_1)]g_2 \end{aligned} \tag{49}$$

where the parameters X_0, X_1, and X_2 are the three closest elevation angles in the antenna table and g_0, g_1, and g_2 are the corresponding antenna gains at X_0, X_1, and X_2. Gain values are interpolated at a frequency below the unadjusted LUF, G_1, and above the unadjusted LUF, G_2. Then a linear interpolation is done to determine the antenna gain at the unadjusted LUF:

$$G = [(F - F_0)/(F_1 - F_0)](G_1 - G_2) + G_1 \tag{50}$$

where F_0 is the frequency below the unadjusted LUF, and F_1 is the frequency above the unadjusted LUF.

5.0 LUF MODEL TEST CASES

The following tables of test case results are provided as an aid in determining the proper operation of the LUF model algorithms. Table 5-1 lists the results of exercising the LUF model for the range of season values. Additional parameter values used for these tests were 1–8 Å solar X-ray flux equal to 0.001 erg/cm² • s, required signal-to-noise ratio equal to 20 dB, transmitter power equal to 5000 watts, transmitter latitude equal to 33 degrees (0.57596 radian), transmitter longitude equal to 117 degrees (2.04204 radians), receiver latitude equal to 30 degrees (0.52360 radian), receiver longitude equal to 90 degrees (1.5708 radians), transmitter antenna number 101 (Quarter Wave Vertical), and receiver antenna number 144 (Horizontal LPA), and the transmitter and receiver antennas were directed toward each other (default condition) with antenna bearings of 90.1 and 284.4 degrees, respectively.

Table 5-1. LUF model season results (MHz).

Time (UT)	Season (Month Number) Winter (1)	Spring (4)	Summer (7)	Fall (10)
0000	2.00	4.09	4.66	2.38
0400	2.00	2.00	2.00	2.00
0800	2.00	2.00	2.00	2.00
1200	2.00	2.00	2.00	2.00
1600	5.35	6.19	6.30	5.95
2000	6.67	6.93	7.03	6.63

Table 5-2 lists the results of exercising the LUF model over the range of allowable 1–8 Å X-ray flux levels. The date used for these tests was 15 July 1989; all other parameters were the same as those used to produce Table 5-1.

Table 5-2. LUF model 1–8 Å X-ray flux results (MHz).

Time (UT)	1–8 Å X-Ray Flux (erg/cm² • s) 1×10^{-6} to 4.9×10^{-3}	5×10^{-3}	1×10^{-2}	1×10^{-1}	1.0
0000	4.66	12.77	15.18	27.00	48.00
0400	2.00	3.19	3.79	6.75	12.00
0800	2.00	2.00	2.00	2.00	2.00
1200	2.00	2.00	2.00	2.00	2.00
1600	6.30	10.18	12.10	21.52	38.27
2000	7.03	14.56	17.32	30.79	48.00

Table 5-3 lists the results of exercising the LUF model over the range of required signal-to-noise ratio values. The date used was 15 July 1989; all other parameters were the same as those used to produce Table 5-1.

Table 5-3. LUF model signal-to-noise ratio results (MHz).

Time (UT)	Required Signal-to-Noise Ratio (dB)						
	-30	-20	-10	0	10	20	30
0000	3.48	3.64	3.83	4.06	4.33	4.66	5.08
0400	2.00	2.00	2.00	2.00	2.00	2.00	2.00
0800	2.00	2.00	2.00	2.00	2.00	2.00	2.00
1200	2.00	2.00	2.00	2.00	2.00	2.00	2.00
1600	4.72	4.95	5.20	5.50	5.86	6.30	6.86
2000	5.28	5.53	5.81	6.15	6.54	7.03	7.64

Tables 5-4 through 5-8 list the results of exercising the LUF model for various locations of the transmitter and receiver. The date used to produce these values was 15 July 1989 at 0000 UT; all other parameters were the same as those used to produce Table 5-1.

Table 5-4. LUF model results (MHz) for transmitter 75 degrees north latitude (1.3090 radians) and 150 degrees west longitude (2.6180 radians).

Receiver, North Latitude		Receiver, West Longitude, deg (radians)					
		0	60	120	180	240	300
deg	radians	(0.0000)	(1.0472)	(2.0944)	(3.1416)	(4.1888)	(5.2360)
75	1.3090	9.22	6.84	5.70	5.97	7.67	9.91
35	0.6109	6.59	7.75	9.20	10.31	11.65	8.60
0	0.0000	5.93	6.19	10.46	11.97	10.52	6.98
-35	-0.6109	4.76	4.88	9.30	10.87	8.83	5.14
-75	-1.3090	2.73	4.69	5.95	7.67	5.77	4.47

Table 5-5. LUF model results (MHz) for transmitter 35 degrees north latitude (0.6109 radian) and 150 degrees west longitude (2.6180 radians).

Receiver, North Latitude		Receiver, West Longitude, deg (radians)					
		0	60	120	180	240	300
deg	radians	(0.0000)	(1.0472)	(2.0944)	(3.1416)	(4.1888)	(5.2360)
75	1.3090	11.32	11.25	9.77	10.14	12.45	12.11
35	0.6109	7.86	8.26	6.74	7.44	11.47	9.84
0	0.0000	4.29	6.85	8.63	9.53	10.34	7.15
-35	-0.6109	3.44	6.77	9.26	10.47	9.31	5.48
-75	-1.3090	6.76	6.92	8.09	10.06	7.91	7.47

Table 5-6. LUF model results (MHz) for transmitter 0 degrees north latitude (0.0 radians) and 150 degrees west longitude (2.6180 radians).

Receiver, North Latitude		Receiver, West Longitude, deg (radians)					
		0	60	120	180	240	300
deg	radians	(0.0000)	(1.0472)	(2.0944)	(3.1416)	(4.1888)	(5.2360)
75	1.3090	10.37	10.19	11.22	11.74	11.36	11.15
35	0.6109	5.92	7.20	8.67	9.52	10.38	7.83
0	0.0000	3.11	6.29	6.20	6.79	9.09	5.40
-35	-0.6109	3.58	5.79	7.66	8.80	9.52	7.73
-75	-1.3090	8.34	8.01	8.90	10.96	9.55	9.64

Table 5-7. LUF model results (MHz) for transmitter -35 degrees north latitude (-0.6109 radian) and 150 degrees west longitude (2.6180 radians).

Receiver, North Latitude		Receiver, West Longitude, deg (radians)					
		0	60	120	180	240	300
deg	radians	(0.0000)	(1.0472)	(2.0944)	(3.1416)	(4.1888)	(5.2360)
75	1.3090	8.08	8.49	10.14	10.64	9.81	8.79
35	0.6109	3.08	6.25	9.40	10.44	9.49	5.35
0	0.0000	4.06	4.83	7.68	8.78	9.70	4.72
-35	-0.6109	4.98	4.89	5.90	7.28	11.26	5.38
-75	-1.3090	5.32	6.71	7.67	9.76	10.58	5.46

Table 5-8. LUF model results (MHz) for transmitter -75 degrees north latitude (-1.3090 radians) and 150 degrees west longitude (2.6180 radians).

Receiver, North Latitude		Receiver, West Longitude, deg (radians)					
		0	60	120	180	240	300
deg	radians	(0.0000)	(1.0472)	(2.0944)	(3.1416)	(4.1888)	(5.2360)
75	1.3090	2.00	3.95	6.18	7.33	6.57	3.90
35	0.6109	2.00	2.00	7.26	10.02	6.57	2.00
0	0.0000	2.00	2.00	7.27	11.16	7.23	2.00
-35	-0.6109	2.00	2.00	4.74	10.32	5.53	2.00
-75	-1.3090	2.00	2.00	2.00	2.00	2.00	2.00

Table 5-9 lists the results of exercising the LUF model over the range of transmitter and receiver antenna types. The date used to produce these values was 15 July 1989 at 0000 UT; all other parameters were the same as those used to produce Table 5-1.

Table 5-9. LUF model results (MHz) for transmitter and receiver antenna types.

Receiver Antenna	Transmitter Antenna								
	000	101	102	121	122	141	142	144	161
000	4.56	4.65	4.65	4.64	4.89	4.81	4.83	4.58	4.36
001	4.10	4.16	4.17	4.16	4.34	4.28	4.30	4.11	3.95
041	4.49	4.57	4.57	4.57	4.80	4.72	4.75	4.51	4.29
101	4.65	4.73	4.74	4.73	4.99	4.90	4.93	4.66	4.43
102	4.65	4.74	4.74	4.73	5.00	4.91	4.94	4.67	4.43
121	4.64	4.73	4.73	4.73	4.99	4.90	4.93	4.66	4.43
122	4.89	4.99	5.00	4.99	5.31	5.19	5.23	4.91	4.64
141	4.81	4.90	4.91	4.90	5.19	5.09	5.12	4.82	4.57
142	4.83	4.93	4.94	4.93	5.23	5.12	5.16	4.85	4.59
144	4.58	4.66	4.67	4.66	4.91	4.82	4.85	4.59	4.37
161	4.36	4.43	4.43	4.43	4.64	4.57	4.59	4.37	4.18

Table 5-10 lists the results of exercising the LUF model over a range of transmitter and receiver antenna bearings. The date used to produce these values was 15 July 1989 at 0000 UT, transmitter antenna was 122, and receiver antenna was 141; all other parameters were the same as those used to produce Table 5-1.

For this test, the azimuth from the transmitter to the receiver was 90.1 degrees, and from the receiver to the transmitter the azimuth was 284.4 degrees. Antenna bearing values were varied in 10-degree increments around this optimum orientation.

Table 5-10. LUF model results (MHz) for transmitter and receiver antenna bearings.

Receiver, Antenna Bearing (deg)	Transmitter, Antenna Bearing (deg)						
	60	70	80	90	100	110	120
250	13.25	11.24	10.43	10.20	10.42	11.21	13.16
260	7.12	6.75	6.56	6.50	6.56	6.75	7.11
270	5.92	5.70	5.59	5.55	5.58	5.70	5.91
280	5.53	5.35	5.26	5.22	5.25	5.35	5.52
290	5.55	5.37	5.27	5.24	5.27	5.37	5.54
300	6.00	5.77	5.65	5.61	5.65	5.77	5.99
310	7.37	6.96	6.76	6.69	6.75	6.95	7.36

6.0 REFERENCES

1. Bleiweiss, M. P., "Solar Influences on HF Absorption and the Resulting Hawaii to California Lowest Observed Frequency," Naval Weapons Center, China Lake, Calif., TP 4911, May 1970.

2. ———, "A Prediction Scheme for the Lowest Observed Frequency (LOF) of the Guam-North West Cape HF Propagation Path and Eight Other Pacific Paths," Naval Electronics Laboratory Center, San Diego, Calif., NELC TR 1851, December 1972.

3. Argo, P. E., J. R. Hill, "Lowest Observable Frequency (LOF) Model: SOLRAD Application," Naval Electronics Laboratory Center, San Diego, Calif., NELC TN 3304, January 1977.

4. Argo, P. E., D. B. Sailors, "Quiet Time Lowest Observable Frequency (QLOF), Calculation Program," Naval Ocean Systems Center, San Diego, Calif., NOSC TD 231, April 1979.

5. Sailors, D. B., W. K. Moision, "Quiet Time Lowest Observable Frequency Model Uncertainty Assessment," Naval Ocean Systems Center, San Diego, Calif., NOSC TR 1189, August 1987.

6. Rose, R. B., J. R. Hill, M. P. Bleiweiss, "Sudden Ionospheric Disturbance Grid," Naval Electronics Laboratory Center, San Diego, Calif., NELC TR 1938, December 1974.

7. Roy, T. N., D. B. Sailors, "Sudden Ionospheric Disturbance Model Uncertainty Assessment," Naval Ocean Systems Center, San Diego, Calif., NOSC TR 1140, July 1986.

8. LaBahn, R. W., "Operational Users Manual for Advanced PROPHET on MS-DOS-Based Microcomputer Systems," Naval Ocean Systems Center, San Diego, Calif., NOSC TD 848, August 1985.

9. Sailors, D. B., "Operational User's Manual for Tactical Decisions Aids for HF Communication," Naval Ocean Systems Center, San Diego, Calif., NOSC TD 1065, July 1987.

Appendix

FORTRAN 77 PROGRAM AND SUBROUTINE LISTINGS FOR LUF MODEL

The LUF calculation program and subroutines that follow are written in FORTRAN 77. The parameters passed to the subroutines and those returned are described in the comments portion of the routines.

```
      program luf
cp*************************************************************************
c     This program calculates the lowest usable frequency LUF given
c     the values specified.
c     The daylight LUF decision between subroutines QLOF (quiet solar
c     conditions) and DLOF (disturbed solar conditions) is based on
c     a latched peak x-ray vice the instantaneous x-ray level.  This
c     latch is set when the x-ray flux exceeds 5.0e-3, and stays set
c     for the remainder of the day, eliminating the 'step' when
c     the x-ray level fluctuates between disturbed and non-disturbed.
c     The latch is reset when the path become totally dark (over-
c     night), but may be set immediately at sunrise if x-ray levels
c     continue to exceed the threshold.
c
c     Update 9/23/86  now includes call to ADJUST after call to QLOF
c
c     Receiver and transmitter azimuth, stabrg(1) and stabrg(2), are
c     set so the antennas point toward each other as default values.
c     To allow manual entry of antenna azimuth, change antdef data
c     statement to false.
c
c     Antenna table:
c        001: FRD-10 CDAA  (RECEIVE ONLY)
c        041: OE316/TSC99 HERMES LP  (RECEIVE ONLY)
c        101: QUARTER WAVE VERTICAL
c        102: LOADED WHIP (SHORT)
c        121: HALF WAVE HORIZONTAL DIPOLE
c        122: INVERTED L
c        141: TERMINATED RHOMBIC
c        142: TERMINATED SLOPING VEE
c        144: HORIZONTAL LPA  (54 DEGREES)
c        161: VERTICAL LOG PERIODIC
c        000: ISOTROPIC OR (UNKNOWN)
c
c     Parameters input:
c        itime: month,day,hour (UT),minute (UT),julian day,year
c        xflux: current value of x-ray flux level (1.0e-6 to 1.0)
c        tlat:  transmitter latitude in radians, + north, - south
c        tlon:  transmitter longitude in radians, + west, - east
c        rlat:  receiver latitude in radians, + north, - south
c        rlon:  receiver longitude in radians, + west, - east
c        signse: required signal to noise ratio (-30 dB minimum)
c        staant(1): receiver antenna number
c        staant(2): transmitter antenna number
c        stabrg(1): receiver antenna azimuth to transmitter (degrees)
c        stabrg(2): transmitter antenna azimuth to receiver (degrees)
c        stapwr(2): transmitter power (watts)
c
c     Parameters returned:
c        pluf:  power dependent luf in MHz
c        cpnt:  geographic path information in radians
c
c     Subroutines and functions used:  path
c                                      dlof
c                                      qlof
c                                      adjust
c                                      coslaw
c                                      subsol
```

```
c                                          gtable
c                                          gcraz
c
c     common blocks:  sun
c                     sysdat
cz*********************************************************************
      logical active,antdef
      dimension cpnt(8)
      integer itime(6),staant
      real lof,pluf,xflux,tlat,tlon,rlat,rlon,chi,xxx,xlof
      common /sun/  slat,slon
      common /sysdat/  staant(2),stabrg(2),stapwr(2),signse
      data models/5/,ten/10./,rnd/0.05/,slm/40.0/
      data halfpi/1.5707963/, dtr/0.0174532925/
      data rtd/57.295780/
      data active / .false. /
c
c     Antenna azimuth - default: true
c     true:  Receiver and transmitter antennas point toward each other,
c              input list values for antenna bearings are not used
c     false:  Receiver and transmitter antenna bearings are taken from
c              input list
c
      data antdef / .true. /
c
c     Initialize antenna table
c
      call gtable(0,0.0,0.0,gain0)
c
c     Initialize LUF and geographic path information variables
c     prior to subroutine calls
c
      pluf=0.0
      do i = 1,8
         cpnt(i) = 0.0
      continue
c
c     Enter LUF calculation parameters
c
      itime(1)=1
      itime(2)=1
      itime(3)=0
      itime(4)=0
      itime(5)=1
      itime(6)=89
      xflux=1.0e-3
      tlat=0.37385
      tlon=2.76024
      rlat=0.57125
      rlon=2.0450
      signse=20.0
      stapwr(2)=5000.0
      staant(1)=144
      staant(2)=101
      stabrg(1)=263.2
      stabrg(2)=63.8
c
c     antenna azimuth default calculation
```

```
c
      if(.not. antdef) go to 120
      call gcraz(rlat,rlon,tlat,tlon,range,azim)
      stabrg(1)=azim*rtd
      call gcraz(tlat,tlon,rlat,rlon,range,azim)
      stabrg(2)=azim*rtd
120   continue
c
c     Calculate control point information
c
      call path(tlat,tlon,rlat,rlon,cpnt)
c
c     Set the sub-solar point for the time given
c
      call subsol( itime )
      if( active ) go to 150
c
c     Determine what part of path is sunlit - if any
c
      chi = coslaw( halfpi - slat, halfpi - cpnt(2), slon - cpnt(3) )
      xxx = chi
      if ( cpnt(4) .eq. 0.0 )  go to 130
      chi = coslaw( halfpi - slat, halfpi - cpnt(4), slon - cpnt(5) )
      if ( chi .lt. xxx )  xxx = chi
      chi = coslaw( halfpi - slat, halfpi - cpnt(6), slon - cpnt(7) )
      if ( chi .lt. xxx )  xxx = chi
130   if ( xxx .lt. halfpi )  go to 140
c
c     Night time over all the path
c
      call qlof( cpnt, itime, lof )
c
c     Update 9/23/86 - call to ADJUST
c
      call adjust(lof,cpnt,tlat,tlon,rlat,rlon)
      pluf = lof
      go to 260
c
c     Daytime over some part of the path
c
140   continue
      if ( xflux .ge. 5.0e-3 ) active = .true.
      if( .not. active ) go to 200
c
c     Disturbed sun LUF calculation
c
150   call dlof( rlat, rlon, tlat, tlon, xflux, cpnt, xlof )
      pluf = amin1(amax1(xlof,2.0),48.0)
      go to 260
200   continue
c
c     Quite sun LUF calculation
c
      call qlof( cpnt, itime, lof )
c
c     Update 9/23/86 - call to ADJUST
c
      call adjust(lof,cpnt,tlat,tlon,rlat,rlon)
```

```
      pluf  =lof
c
c     Skywave calculations complete
c
260   continue
      end
```

```
      subroutine path (tlat, tlon, rlat, rlon, cpnt)
cp*********************************************************************
c       subroutine path
c
c       call path(tlat,tlon,rlat,rlon,cpnt)
c
c       This routine computes the range, azimuth and control point
c       coordinates for a given propagation path.  The method assumes
c       a spherical earth with a radius of 6371 km.  The required
c       input for this module is:
c               tlat: transmitter latitude in radians
c               tlon: transmitter west (positive) longitude in radians
c               rlat: receiver latitude in radians
c               rlon: receiver west (positive) longitude in radians
c
c     This subroutine returns the following information in an 8 word
c     real array (cpnt):
c       cpnt(1): distance between the receiver and transmitter in radians
c       cpnt(2): latitude of midpoint in radians
c       cnpt(3): west longitude in radians
c       cpnt(4): latitude of point 1000 km from the receiver in radians
c       cpnt(5): west longitude of point 1000 km from receiver in radians
c       cpnt(6): latitude of point 1000 km from transmitter in radians
c       cpnt(7): west longitude of point 1000 km from transmitter
c                 in radians
c       cpnt(8): azimuth from receiver to transmitter in radians
c
c      * cpnt(4) through cpnt(7) will not be computed for paths less
c                 than 1000 km (0.15696 radians) in length.
c
c
c          Parameters input: tlat, tlon, rlat, rlon
c
c          Parameters returned: cpnt
c
c          Subroutines and functions used:  gcraz
c                                                razgc
c
c          Common blocks:  none
c
cz*********************************************************************
c
      real cpnt(8)
      real rlat
      real rlon
      real tlat
      real tlon
      real pl
c
c     Get range and azimuth between points 1 and 2
c
      call gcraz( rlat, rlon, tlat, tlon, cpnt(1), cpnt(8) )
c
c     Get mid-point coordinates
c
      pl = cpnt(1)/2.0
      call razgc( rlat, rlon, pl, cpnt(8), cpnt(2), cpnt(3) )
c
```

```
c     Path length >= 1000 km?
c
      if ( cpnt(1) .ge. 0.156961231 ) then
c
c     Get coordinates of 1000 km points
c
         pl = 0.156961231
         call razgc( rlat, rlon, pl, cpnt(8), cpnt(4), cpnt(5) )
         pl = cpnt(1) - 0.156961231
         call razgc( rlat, rlon, pl, cpnt(8), cpnt(6), cpnt(7) )
         return
      else
         return
      end if
      end
```

```
      subroutine gcraz( lat1, lon1, lat2, lon2, range, azim )
cp*********************************************************************
c      subroutine gcraz
c
c         call gcraz (lat1,lon1,lat2,lon2,range,azim)
c
c         This routine computes the great circle range and azimuth
c         between two points on the earth's surface.  lat1 and lon1
c         are the coordinates of point 1 , lat2 and lon2 are the
c         coordinates of point 2.  Both longitudes are west longitudes.
c         West longitudes are positive throughout the Muf85 algorithm.
c         Latitudes are positive if north and negative if south.
c         The output is range, the distance between the two points
c         in radians and azim, the azimuth from one point to the other
c         in radians.  This method assumes a spherical earth and
c         recognizes the degenerate cases of point 1 at the north
c         or south pole or points 1 and 2 coincident.  All coordinates
c         are in radians.
c
c         Parameters input: lat1, lon1, lat2, lon2
c
c         Parameters returned: range, azim
c
c         Subroutines and functions used:   none
c
c         Common blocks:   none
c
c         Method: Uses law of cosines for sides on spherical triangle
c                 defined by (lat1,lon1),(lat2,lon2) and north pole.
c
cz*********************************************************************
c
      real lat1
      real lon1
      real lat2
      real lon2
      real range
      real azim
      real s1
      real c1
      real s2
      real c2
      real cr
      real ca
c
      data pi/3.141592654/,twopi/6.283185308/,halfpi/1.570796327/,
     &     dtr/0.017453293/,rtd/57.29577951/
c
c     Test for degenerate cases
c
c        1)Point 1 at north or south pole:
c
      if ( abs( lat1 - halfpi ) .le. 1.0e-5 ) then
c
c     Point 1 is at the north pole
c
         range = halfpi - lat2
         azim = pi
```

```
           return
        else
           if ( abs( lat1 + halfpi ) .le. 1.0e-5 ) then
c
c       Point 1 is at the south pole
c
              range = halfpi + lat2
              azim = 0.0
              return
           end if
        end if
c
c          2)Coincident points:
c
        if ( abs( lat1 - lat2 ) .le. 1.0e-5 .and.
     &       abs( lon1 - lon2 ) .le. 1.0e-5 ) then
c
c       Points 1 and 2 are coincident
c
           range = 1.0e-8
           azim = 0.0
           return
        end if
c
c       General case
c
        s1 = sin( lat1 )
        c1 = cos( lat1 )
        s2 = sin( lat2 )
        c2 = cos( lat2 )
        cr = s1*s2 + c1*c2*cos( lon1 - lon2 )
        cr = amin1( amax1( cr, -1.0 ), +1.0 )
        range = acos( cr )
        ca = ( s2 - s1*cr )/( c1*sin( range ) )
        ca = amin1( amax1( ca, -1.0 ), +1.0 )
        azim = acos( ca )
        if ( sin( lon1 - lon2 ) .lt. 0.0 )  azim = twopi - azim
        return
        end
```

```
      subroutine razgc( lat1, lon1, range, azim, lat2, lon2 )
cp***********************************************************************
c       subroutine razgc
c
c          call razgc(lat1,lon1,range,azim,lat2,lon2)
c
c          This routine computes the latitude and west(positive) longi-
c          tude (lat2, lon2) of a point a specified range from a given
c          point on the earth's surface.  Also required for input
c          is the azimuth (azim) to the new point in radians.  This
c          method assumes a spherical earth (6371.0 km) and recognizes
c          the degenerate cases of the given point being at the north
c          or south pole.  For the degenerate cases, azim should be 0
c          or pi and lon2 is undefined.  However, azim is not checked,
c          and lon2 is arbitrarily set equal to lon1.  This routine
c          recognizes the degenerate case when range is set to zero.
c          All coordinates are in radians.
c
c          Parameters input: lat1, lon1, range, azim
c
c          Parameters returned: lat2, lon2
c
c          Subroutines and functions used:  none
c
c          Common blocks:  none
c
c          Method: Uses law of cosines for sides on spherical triangle
c                  defined by (lat1,lon1), north pole and point defined
c                  by azim and range.
cz***********************************************************************
c
      real lat1
      real lon1
      real lat2
      real lon2
      real s1
      real c1
      real cr
      real ca
      real cg
      real a
      real g
      real sa
c
      data  pi/3.141592654/,twopi/6.283185308/,halfpi/1.570796327/
     &        rtd/57.29577951/,dtr/0.017453293/
c
c     Test for degenerate cases
c
c        1)Given point is north or south pole:
c
      if ( abs( lat1 - halfpi ) .le. 1.0e-5 ) then
c
c     The given point is the north pole
c
         lat2 = halfpi - range
         lon2 = lon1
         return
```

```
      else
         if ( abs( lat1 + halfpi ) .le. 1.0e-5 ) then
c
c     The given point is the south pole
c
            lat2 = range - halfpi
            lon2 = lon1
            return
         end if
      end if
c
c        2)Coincident points:
c
      if ( range .eq. 0.0 ) then
c
c     Point 2 coincident with point 1
c
         lat2 = lat1
         lon2 = lon1
         return
      end if
c
c     General case
c
      s1 = sin( lat1 )
      c1 = cos( lat1 )
      cr = cos( range )
      ca = s1*cr + c1*sin( range )*cos( azim )
      ca = amin1( amax1( ca, -1.0 ), +1.0 )
      a = acos( ca )
c
c     Test if destination ends up on the poles
c
      if( abs(a).le.1.0e-5 ) then
         lat2 = halfpi
         lon2 = lon1
         return
      else
         if( abs(a-pi) .le. 1.0e-5 ) then
            lat2 = -halfpi
            lon2 = lon1
            return
         end if
      end if
c
c     Get destination coordinates
c
      cg = ( cr - s1*ca )/( c1*sin( a ) )
      cg = amin1( amax1( cg, -1.0 ), +1.0 )
      g = acos( cg )
      lat2 = halfpi - a
      sa = sin( azim )
      if ( sa .ge. 0.0 )  lon2 = amod( lon1 - g, twopi )
      if ( sa .lt. 0.0 )  lon2 = amod( lon1 + g, twopi )
      return
      end
```

```
      subroutine subsol ( itime )
cp***********************************************************************
c     subroutine subsol
c
c     call subsol(itime)
c
c     This routine computes the coordinates of the sub-solar point for
c     a given time.  Itime is a six element integer array containing
c     month, day, hour, minute, julian day, and year.  The coordinates
c     computed are in radians and are stored in slat and slon in
c     common sun.
c
c     Parameters input: itime
c
c     Parameters returned: slat, slon
c
c     Subroutines and functions used:  none
c
c     Common blocks:  sun
c
cz***********************************************************************
c
      integer  itime(6),year,ihour,minute,jday,k
      real     day,date,x,sx,cx,twocx,c2x,s2x,c3x,s3x,c4x,s4x,c5x,s5x
      real     dec,eqnt
c
      common   /sun/      slat, slon
c
      data   a0/ 0.3798/,a1/-23.0009/
      data   a2,a3,a4,a5,a6/-0.3802, -0.1550, -0.0076, -0.0025, -0.0004/
      data   b1,b2,b3,b4,b5/ 3.5354,  0.0302,  0.0728,  0.0032,  0.0020/
      data   c1,c2,c3,c4,c5 /0.5965, -2.9502, -0.0653, -0.1248, -0.0103/
      data   d1,d2,d3,d4,d5/-7.3435, -9.4847, -0.3083, -0.1747, -0.0159/
      data   one, two / 1.0,2.0/
      data   twopi /6.28319/, dtr /0.0174533/
c
      ihour=itime(3)
      minute=itime(4)
      jday=itime(5)
      year=itime(6)
      day = jday + ihour/24.0 + minute/1440.0
c
      k = mod (year, 4)
      date = 365.0*float( k ) + 0.0078*( float( year ) - 68.0 )
      if ( k .ne. 0 )  date = date + 1.0
      date = date + day
      x = date*twopi/365.25
      sx =  sin( x )
      cx =  cos( x )
      twocx = two*cx
c
c     Recursive calculation for cos(n*x) and sin(n*x),n=2,6
c
      c2x = twocx*cx - one
      s2x = twocx*sx
      c3x = twocx*c2x - cx
      s3x = twocx*s2x - sx
      c4x = twocx*c3x - c2x
```

```
      s4x = twocx*s3x - s2x
      c5x = twocx*c4x - c3x
      s5x = twocx*s4x - s3x
      c6x = twocx*c5x - c4x
c
c     Fourier expansion for solar declination and equation of time
c
      dec =  a0 + a1*cx + a2*c2x + a3*c3x + a4*c4x + a5*c5x + a6*c6x
     &           + b1*sx + b2*s2x + b3*s3x + b4*s4x + b5*s5x
      eqnt = c1*cx + c2*c2x + c3*c3x + c4*c4x + c5*c5x
     &   +   d1*sx + d2*s2x + d3*s3x + d4*s4x + d5*s5x
c
      gha = ihour - ( 12.0 - eqnt/60.0 ) + minute/60.0
      slat = dec*dtr
      slon = 15.0*gha*dtr
      if ( slon .lt. 0.0 )  slon = slon + twopi
c
      return
      end
```

```
      function coslaw ( a, b, c )
cp**********************************************************************
c       real function coslaw
c
c       x=coslaw(a,b,c)
c
c       This routine performs the law of cosines for spherical
c       triangles of radians a, b, and c.
c
c       Parameters input: a, b, c
c
c       Subroutines and functions used:  none
c
c       Common blocks:  none
c
cz**********************************************************************
      coslaw = acos(cos(a)*cos(b)+sin(a)*sin(b)*cos(c))
      return
      end
```

```
      subroutine qlof (cpnt, itime, lof)
cp**********************************************************************
c       subroutine qlof
c
c       call qlof(cpnt,itime,luf)
c
c       This routine computes the LUF for solar quiet conditions (when
c       x-ray flux < 5.0e-3).  The required input is cpnt which is
c       defined in subroutine path and itime which is a six element
c       integer array containing month,day,hour,minute,julian day,and
c       year.  This routine returns LUF.
c
c       Parameters input: cpnt, itime
c
c       Parameters returned: lof
c
c       Subroutines and functions used:  absorb
c
c       Common blocks:  none
c
c       Method:  Uses algorithms described in NOSC TD-231 (Quiet Time
c                Lowest Observable Frequency (QLOF), Calculation
c                Program, P.E. Argo and D.B. Sailors, 1979)
c
cz**********************************************************************
      dimension cpnt(8)
      real            lof,a,b,c,d,k,k3
      integer         itime(6)
c
c     Path length calculation in km
c
      d = cpnt(1)*6371.0
      lof = 0.5
c
c     Calculate luf for path length <= 2000 km
c
      if(d.gt.2000.) go to 200
      a=cpnt(1)/2.
      b=sin(a)
      c=cos(a)
      call absorb(itime,cpnt(2),cpnt(3),ai,chi)
      lof=sqrt(ai/40./sqrt(1.-0.9784/(1.+((c-0.985)/b)**2)))
      go to 1000
c
c     Calculate luf for 2000 < path length <= 3300 km
c
200   if ( d .gt. 3300.0 )  go to 400
      k = ( 4.0 + 1.875e-3*d )*0.045
      call absorb( itime, cpnt(2), cpnt(3), ai, chi )
      lof = k*sqrt( ai )
      go to 1000
c
c
c     Calculate luf for 3300 < path length <= 6600 km
c
400   k = ( 7.5 + 0.001*d )*0.045
c
c     Calculate luf for path length > 6600 km
```

```
c
      if (d .le. 6600.0) go to 500
      k3 = 1-0.3768*(cpnt(1)-1.0361)
      k= k3*k
c
c     Calculate absorption index at path midpoint and 1000 km
c     control points
c
500   call absorb( itime, cpnt(2), cpnt(3), ab1, chi )
      call absorb( itime, cpnt(4), cpnt(5), ab2, chi )
      call absorb( itime, cpnt(6), cpnt(7), ab3, chi )
      lof = k*sqrt( ( ab1*2.0 + ab2 + ab3 )/4.0 )
1000  continue
c
c     QLOF minimum value = 0.5 MHz, maximum value = 50.0 MHz
c
      lof = amin1( amax1( lof, 0.5 ) , 50.0 )
      return
      end
```

```
      subroutine absorb ( itime, lat, lon, ai, chi )
cp*********************************************************************
c       subroutine absorb
c
c       call absorb (itime,lat,lon,ai,chi)
c
c       This routine computes the ionospheric absorption index at
c       position lat and lon at itime.  Itime is a 6 element integer
c       array for month,day,hour,minute,julian day, and year.  Lat
c       and lon are in radians.  The results of this routine are
c       stored in ai (absorption index) and chi (solar zenith angle in
c       radians). Lat and lon are in radians and west longitude.
c
c       Parameters input: itime, lat, lon
c
c       Parameters returned: ai, chi
c
c       Subroutines and functions used:  ch
c
c       Common blocks:     sun
c
cz*********************************************************************
      real            lat,lon,ai,chi,m,n,lad,w,alat,csxn,absp
      integer         itime(6)
c
      common          /sun/     slat,slon
      data            rtd /57.29577951/
c
c     Compute solar zenith angle
c
      w = 1.0
      chi=sin(lat)*sin(slat)+cos(lat)*cos(slat)*cos(lon-slon)
      chi=amax1(chi,-1.)
      chi=amin1(chi,1.)
      chi=acos(chi)
c
c     Calculation of solar zenith angle at noon
c
      chinon = abs( slat - lat )
      if ( chinon .lt. 1.57 ) go to 80
c
c     Absorption index at noon
c
      ai = 1.0e-13
      return
80    lad = abs(lat)*rtd
c
c     Test if high latitude winter correction needed
c
      if ( lad .lt. 30.0 )  go to 100
      if( (itime(1).eq.1 .or. itime(1).eq.12) .and. lat.gt.0.0 )
     &    go to 90
      if( (itime(1).eq.6 .or. itime(1).eq.7) .and. lat.lt.0.0 )
     &    go to 90
      go to 100
90    w = 1.0 + 0.0275*( 30.0 - abs( 60.0 - lad ) )
c
c     Compute absorption index based on latitude
```

```
c
100    continue
       alat = abs(lat)
       if(alat .lt. 0.45)n=1.4-alat*2.44
       if(alat .ge. 0.45  .and. alat .lt. 1.0875)n=0.3
       if(alat .ge. 1.0875  .and. alat .lt. 1.367)
      &            n=-(alat-1.0875)*1.07 + 0.3
       if(alat .ge. 1.367)n=0.
c
c      Calculation of diurnal variation of absorption index
c
       csxn = cos( chinon )**n
       absp = 286.0*w*( 1.0 + 0.5*alat )*csxn
       if ( absp .lt. 1.0e-11 )  absp = 1.0e-11
       if ( lad .gt. 18.0 )  go to 201
       m = 0.5*( 0.58 + ( lad/18.0 )*0.08 )
       go to 300
201    continue
       if ( lad .gt. 24.0 )  go to 202
       m = 0.5*( 0.66 + 0.22*( lad - 18.0 )/6.0 )
       go to 300
202    continue
       m = 0.44
c
c      Adjust according to solar zenith angle if daylit
c
300    continue
c
c      Absorption index for chi > 103 degrees
c
       ai = absp*0.01
       if ( chi .gt. 1.8 )  return
       ai = absp*( ch(921.0,chi )/ch( 921.0, chinon ) ) **(-2.0*m)
       return
       end
```

```
      function ch ( x, y )
cp*********************************************************************
c       real function ch
c
c       x=ch(x,y)
c
c       This function computes the chapman's grazing incidence integral
c       where x is the parameter related to the atmospheric density and
c       scale height and y is the solar zenith angle in radians. This
c       routine is accurate to 0.1% when x*(u-sin(i))<10 or cos(y)>0.
c
c       Parameters input: x, y
c
c       Subroutines and functions used:  chpint
c
c       Common blocks:  chpmn
c
c       Method:  Uses four different approximations for the Chapman
c                 function, depending upon the value of the solar zenith
c                 angle.  All are accurate to better than 0.1 percent.
c                 The four approximations are 1) secant(y), 2) 2-point
c                 Gaussian-Laguerre integral, 3) 4-point Gaussian-
c                 Laguerre integral and 4) truncated 10-point Gaussian-
c                 Laguerre integral.
cz*********************************************************************
      common         /chpmn/   xc, g, yc
      real x,y,xc,yc,cy,cy1,ch,g
c
      xc = x
      yc = y
      cy = cos(y)
      cy1 = cy - 0.0174533
      if(350.*y .gt. x*cy1**4) go to 10
c
c     Chapman function for solar zenith angle < 41.8 degrees
c
      ch = 1./cy
      return
10    g = x*sin( y )/( x + alog( x ) + 20.0 )
      g = atan( g/sqrt( 1.0 - g*g ) )
      g = ( g - y )/20.0
      if(cy1 .lt. 0.0) go to 30
      if(x*cy1 .lt. 40.*y) go to 20
c
c     Chapman function for 41.8 <= solar zenith angle <= 85.5 degrees
c
      ch=-x*sin(y)*g*(.1464466*chpint(3.414214)
     &                + .8535534*chpint(.5857864))
      return
c
c     Chapman function for 85.5 < solar zenith angle <= 90 degrees
c
20    ch=-x*sin(y)*g*(.5392947e-3*chpint(9.395071)
     &                + .03888791*chpint(4.536620)
     &                + .3574187*chpint(1.745761)
     &                + .6031541*chpint(.3225477) )
      return
c
```

```
c       Chapman function for 90 < solar zenith angle <= 103 degrees
c
30      ch=-x*sin(y)*g*(.4249314e-6*chpint(16.27926)
     &                 + .2825923e-4*chpint(11.84379)
     &                 + .7530084e-3*chpint(8.330153)
     &                 + .009501517*chpint(5.552496)
     &                 + .06208746*chpint(3.401434)
     &                 + .2180683*chpint(1.808343)
     &                 + .4011199*chpint(.7294545)
     &                 + .3084411*chpint(.1377935) )
        return
        end
```

```
      function chpint ( z )
cp***********************************************************************
c       real function chpint
c
c       x=chpint(z)
c
c       This routine computes the integrand of the chapman integral.
c
c       Parameters input: z
c
c       Subroutines and functions used:   none
c
c       Common blocks:   chpmn
c
cz***********************************************************************
c
      common     /chpmn/    x, g, y
      real        q,z,g,u
c
      q = z*g
      u = sin( q + y )
      q = q/2.0
      chpint = exp( 2.0*x*sin( q )*cos( y + q )/u + z )/u/u
      return
      end
```

```
      subroutine dlof ( tlat, tlon, rlat, rlon, xflux, cpnt, lof )
cp***********************************************************************
c       subroutine dlof
c
c       call dlof(tlat,tlon,rlat,rlon,xflux,cpnt,lof)
c
c       This routine computes the LUF for solar disturbed conditions
c       (x-ray flux => 5.0e-3). Tlat/tlon & rlat/rlon are the end point
c       coordinates in radians.  Xflux is the current x-ray flux.
c       Cpnt is an eight element array containing the path length and
c       coordinates of control points along the path (subroutine PATH
c       has a complete description).
c       This routine returns lof which is the LUF in MHz.
c       This routine assumes west longitudes.
c
c       Parameters input: tlat, tlon, rlat, rlon, xflux, cpnt
c
c       Parameters returned: lof
c
c       Subroutines and functions used:  minpt
c                                           newton
c
c       Common blocks:  sun
c
c       Method:  Uses algorithms described in NOSC TR-1938, (Sudden
c                Ionospheric Disturbance Grid, R.B. Rose, J.R. Hill
c                and M.P. Bleiweiss, 1974).
cz***********************************************************************
      real            cpnt(8),lof,dist,flux,theta,alfa,x
      common          /sun/      slat, slon
c
      lof = 0.5
c
c     Calculate minimum solar zenith angle
c
      call minpt( rlat, rlon, cpnt, clat, clon, chi )
      dist = cpnt(1)*6371.0
c
c     IF minimum solar zenith angle > 89.95 degrees - set lof to
c     minimum value
c
      if ( chi .gt. 1.57 )  go to 100
      flux = xflux
c
      if ( dist .lt. 3500.0 )  go to 11
c
c     Long range lof calculation
c
      call newton( chi, flux, lof )
      go to 100
c
c     Short range lof calculation
c
11    x = sqrt( ( flux*cos( chi )**3 )/1.03856e-6 )
20    lof = sqrt( x )
      theta = ( dist/6371.0 )*0.5
      alfa = ( cos( theta ) - 0.96224 )/sin( theta )
      alfa = atan( alfa )
```

```
      alfa = acos ( 0.9891*cos( alfa ) )
      lof = lof*sqrt( 0.5368/sin( alfa ) )
100   continue
c
c     DLOF minimum value = 0.5 MHz, maximum value = 50.0 MHz
c
      lof = amin1( amax1( lof, 0.5 ) , 50.0 )
      return
      end
```

```
      subroutine minpt(rlat,rlon,cpnt,clat,clon,chi)
cp*************************************************************************
c       subroutine minpt
c
c       call minpt(rlat,rlon,cpnt,clat,clon,chi)
c
c       This routine computes the minimum solar zenith angle over a
c       specified propagation path and coordinates of the point in the
c       path at which the minimum zenith angle occurs.  Required input
c       is cnpt(1), the path length in radians, and rlat and rlon are
c       the receiver location in radians.  Returned are clat and clon,
c       coordinates of the minimum zenith angle in radians and chi the
c       solar zenth angle at clat, clon in radians.
c
c       Parameters input: rlat, rlon, cpnt
c
c       Parameters returned: clat, clon, chi
c
c       Subroutines and functions used:  gcraz
c                                          razgc
c
c       Common blocks:  sun
c
cz*************************************************************************
      real            cpnt(8),clat,clon,pl,azim,range,chi
c
      common /sun/ slat,slon
c
      call gcraz(rlat,rlon,slat,slon,chi,azim)
      clat=rlat
      clon=rlon
      pl=cpnt(1)/10.0
      azim=cpnt(8)
c
      do i=1,10,1
        range=float(i-1)*pl
        call razgc(rlat,rlon,range,azim,xlat,xlon)
        call gcraz(xlat,xlon,slat,slon,dist,dummy)
        if( dist.lt.chi) then
          chi=dist
          clat=xlat
          clon=xlon
        end if
      end do
      return
      end
```

```
      subroutine newton( chi, flux, lof )
cp**********************************************************************
c     subroutine newton
c
c     call newton( chi, flux, lof )
c
c     This routine performs the newton iteration scheme for a
c     non-linear equation.
c
c     Parameters input:
c          chi:  solar zenith angle in radians
c          flux: solar 1-8 angstrom x-ray flux
c
c     Parameters returned:
c          lof: LUF in megahertz
c
c     Subroutines and functions used:  none
c
c     Common blocks:   none
cz**********************************************************************
      real    lof,sechi,x,eps,tolf,f1,f156,fflux,dflux,dx,a,tol
c
      lof = 50.0
      sechi = 1.0/(cos(chi)**2)
      x = (flux/0.1038 + 150.0)/(10.0 + sechi)
      eps = 0.0001
      tolf= 0.01
c
c     Iterate until  a) convergence occurs
c                    b) zero derivitive is produced
c                    c) max 20 iterations occurs
c
      do i=1,20,1
        f1 = ( 1.0 + sechi/10.0 )*x
        f156 = 0.8491*(f1 - 15.6)
        fflux = 0.01038*( f1 - 15.0 ) -0.003*sin(f156) - flux
        dflux = ( 0.01038 - 0.0025473*cos(f156) )*f1/x
c
c     Test for zero derivitive
c
        if( abs(dflux) .lt. eps ) go to 9999
c
c     Ok, continue on
c
        dx = fflux/dflux
        x = x-dx
        a = abs(x)
        if( a .gt. 1.0 ) then
          tol = eps
        else
          tol = eps*a
        end if
c
c     Test for convergence
c
        if( abs(dx) .lt. tol .and. abs(fflux) .le. tolf ) go to 120
      end do
c
```

```
c      Failed to converge after 20 iterations
c
       go to 9999
c
c      Convergence occurred, set lof to last trial
c
120    lof = x
9999   return
       end
```

```
      subroutine adjust (luf, cpnt, tlat, tlon, rlat, rlon)
cp**********************************************************************
c     subroutine adjust
c
c     call adjust(luf,cpnt,tlat,tlon,rlat,rlon)
c
c     This routine added 9/23/86 adjusts the calculated LUF as a
c     function of transmitter power, antenna gains, range and signal
c     to noise ratio.
c     Cpnt is an eight element array set by subroutine path. Only the
c     first element of cpnt(range) is used.
c     kxmt is the pointer to transmitter values in common sysdat.
c     krcv is the pointer to receiver values in common sysdat.
c
c     Parameters input:
c        cpnt: path control point information in radians
c        tlat: transmitter latitude in radians
c        tlon: transmitter west longitude in radians
c        rlat: receiver latitude in radians
c        rlon: receiver west longitude in radians
c
c     Parameters returned:
c        luf: adjusted LUF
c
c     Subroutines and functions used: antfac
c                                     gcraz
c                                     gtable
c
c     Common blocks:  sysdat
cz**********************************************************************
      real luf,cpnt(8),trange,brng,raddif,ctgain,crgain,slm1,slm2,ratslm
      integer kxmt,krcv
      common /sysdat/ staant(2),stabrg(2),stapwr(2),signse
      data dtr /0.0174533/
c
      kxmt = 2
      krcv = 1
      ctgain = 0.
      crgain = 0.
      trange = cpnt(1)*6371.
c
c     If frequency is less than 2 MHz or greater than 48 MHz,
c     do not adjust
c
      if (luf .le. 2.0) j=1
      if (luf .ge. 48.0) j=2
      j=3
      go to (400,500,1000),j
c
c     Assign LUF to 2 MHz and exit
c
400   luf = 2.0
      go to 9000
c
c     Assign LUF to 48 MHz and exit
c
500   luf = 48.0
      go to 9000
```

```
c
c       Calculate transmitter antenna gain
c
1000    if (kxmt.eq.0)go to 2000
        call gtable(staant(kxmt),trange,luf,ctgain)
c
c       Compute azimuth orientation for the transmitter
c
        brng = stabrg(kxmt)*dtr
        call gcraz(tlat,tlon,rlat,rlon,pl,az)
c
c       Calculate difference between the path bearing and the azimuth
c       at which the antenna is pointing
c
        raddif = az-brng
        if(brng .lt. 0.0)raddif = 0.0
        b = antfac(raddif,staant(kxmt))
        ctgain = ctgain-b
c
c       Calculate receiver antenna gain
c
2000    if(krcv.eq.0)go to 3000
        call gtable(staant(krcv),trange,luf,crgain)
c
c       Compute azimuth orientation for the receiver
c
        brng = stabrg(krcv)*dtr
        call gcraz(rlat,rlon,tlat,tlon,pl,az)
        raddif = az-brng
        if(brng .lt. 0.0) raddif = 0.0
        b = antfac(raddif,staant(krcv))
        crgain = crgain - b
c
c       LUF corrections based on antenna gains, transmitter power,
c       signal to noise ratio and range
c       If both kxmt and krcv are zero - do not adjust
c
3000    if (kxmt .eq. 0 .and. krcv .eq. 0) go to 9000
        slm1 = 37.0-20.0*alog10(trange/4287.0)+51.72+27.5*alog10(luf)-60.0
        slm2 = 10.0*alog10(stapwr(kxmt))+ctgain+crgain+7.5*alog10(luf)-
       &20.0*alog10(trange)-signse+111.55
c
c       Keep slm2 positive and the ratio of slm1/slm2 to the maximum of 15
c
        if (slm2 .le. 0.0) then
          ratslm = 15.0
        else
          ratslm = slm1/slm2
          if (ratslm .gt. 15.0) ratslm = 15.0
        end if
c
        luf = luf*sqrt(ratslm)
c
c       Keep the LUF between 2 MHz and 48 MHz
c
        if(luf .le. 2.0) luf=2.0
        if(luf .ge. 48.0) luf=48.0
9000    return
```

end

```
      real function antfac (raddif,itype)
cp***************************************************************************
c     real function antfac
c
c     x=antfac(raddif,itype)
c
c     This real function returns an antenna correction factor
c     given the difference between the path bearing and the
c     azimuth at which the antenna is currently pointing, and
c     the antenna type.  Raddif is in radians, while itype is
c     an integer number corresponding to the antenna type.
c     example:  fields = fields-antfac(raddif,itype)
c
c     Parameters input:
c        raddif: difference between the path bearing and antenna
c                azimuth
c        itype:  antenna type
c
c     Subroutines and functions used:  idant
c
c     Common blocks:  antnam
cz***************************************************************************
      character*1  po(24)
      character*32 aname(21)
      integer      antabl(21)
      real         shape(21),bakmax,v
      common         /antnam/ aname,po,shape,antabl
c
c     If itype is not found in the current table, no shape data will be
c     available.  Set antfac to zero (omni) and return
c
1000  v=0.0
c
c     Get the pointer to the antenna table entry
c
      index=idant(itype)
      if(index.lt.1.or.index.gt.20) go to 1500
c
c     If shape is zero, antenna pattern is omnidirectional
c     If shape is positive, antenna has bidirectional pattern
c     If shape is negative, antenna has unidirectional pattern
c
c     Maximum front/back ratio = value in variable 'bakmax'
c
      bakmax = (abs(shape(index)/2.)) + 15.0
      if(shape(index).lt.0.0) go to 1200
c
c     Omnidirectional pattern section
c
      if(shape(index).eq.0.0) go to 1500
c
c     Bidirectional pattern section
c
1100  if(abs(cos(raddif)).lt.1.e-5) go to 1400
      v = -abs(shape(index))*10.*alog10(abs(cos(raddif)))
      if(v.gt.bakmax) v = bakmax
      if(v.lt.1.e-5) v = 0.0
      go to 1500
```

```
c
c       Unidirectional pattern section
c
1200    if(cos(raddif).lt.1.e-5) go to 1400
        v = -abs(shape(index))*10.*alog10(abs(cos(raddif)))
        if(v.gt.bakmax) v = bakmax
        go to 1500
1400    v = bakmax
1500    antfac = v
        return
        end
```

```
      integer function idant( itype )
cp***************************************************************************
c     integer function idant
c
c     i = idant(itype)
c
c     This integer function returns the index to the antenna table
c     for antenna type, itype.  In the event antenna itype is not
c     found in the antenna table, the index, idant, will be returned
c     as 21 (isotropic).  Isotropic will always be available in
c     the table and may be purposely selected as itype '000'.
c     Idant may then be used by the calling routine to index other
c     antenna attributes such as name, polarization, shape, or
c     transmit capability (identified as 0 or a number greater than
c     100 for the entry in antabl).
c
c     Parameters input:
c        itype:  antenna type
c
c     Subroutines and functions used:   none
c
c     Common blocks:     antnam
cz***************************************************************************
      character*1  po(24)
      character*32 aname(21)
      integer      antabl(21),index
      real         shape(21)
      common       /antnam/ aname,po,shape,antabl
c
c     Find the first entry matching the type.  If two are
c     in the table, one is long range and should be first,
c     while the second is short range.  Other than the gain
c     array and the "antabl" sign, both entries are identical.
c
      do i=1,21,1
        index=i
        if( abs(itype).gt.255 ) go to 100
        if(abs(antabl(index)).eq.itype) go to 200
100     continue
      end do
200   idant=index
      return
      end
```

```
      subroutine gtable(itype,r,f,gain)
cp**********************************************************************
c     subroutine gtable
c
c     call gtable(itype,r,f,gain)
c
c     This subroutine is called by subroutine adjust.f
c     Added 9/23/86 (modfied 9/7/88)
c     Antenna types 001,041,101,102,121,122,141,142,144,and 161
c     now have both the upper and lower antenna tables.
c
c     This routine computes the mainlobe antenna gain when given
c     the antenna type, range, and frequency.  The antenna types
c     supported and gains are listed in the data statements.
c
c     The antenna name is stored in aname, polarization in po, azimuth
c     pattern in shape, and the numeric identifier in antabl which is
c     used to signal short or long range pattern capability. A call
c     with a frequency of zero must be included before gtable is used.
c     e.g.  call gtable( 0, 0.0, 0.0, gain0)
c
c     Note:  it is important when building the tables that the
c     numeric identifier (antabl) matches the antenna name (aname).
c     Recieve only antennas are flagged by an identifier whose
c     absolute value is less than or equal to 100.  The short range
c     pattern is identified by a negative antabl, while long range is
c     a positive antabl.
c     If the antenna is not in the table the gain will be set to
c     0.0 (isotropic).
c
c     Note:  a Lagrange polynomial of degree two is used in this
c     subroutine to interpolate the launch angles.
c
c     Parameters input:
c         itype:  antenna type
c             r:  range
c             f:  frequency
c
c     Parameters output:
c          gain:  antenna gain
c
c     Subroutines and functions used:  none
c
c     Common blocks: antnam
c
cz**********************************************************************
      dimension      gains(2), itmp(21,6)
      integer        freset(21),ii,m,ifirst,isecond,ithird
      logical        found
      dimension      igain(21,3,21), idb000(21,3)
      integer        antabl(21),tdata(21),klong,kshort,index
      real           shape(21),sdata(21),gain,eradis,radian,dlt,x,y
      real           x1,x2,x3,y1,y2,y3,temp1,temp2,temp3
      character*1    po(24),pdata(24)
      character*32   aname(21),adata(21),adata1(10),adata2(10),adata3
      equivalence( adata(1), adata1(1) )
      equivalence( adata(11),adata2(1) )
      equivalence( adata(21),adata3 )
```

```
      common /antnam/ aname,po,shape,antabl
c
c     Define each antenna gain table by number from default list
c
      dimension idb001(21,3), idb041(21,3), idb101(21,3), idb102(21,3),
     &          idb121(21,3), idb122(21,3), idb141(21,3), idb142(21,3),
     &          idb144(21,3), idb161(21,3)
      dimension ndb001(21,3), ndb041(21,3), ndb101(21,3), ndb102(21,3),
     &          ndb121(21,3), ndb122(21,3), ndb141(21,3), ndb142(21,3),
     &          ndb144(21,3), ndb161(21,3)
c
c     Establish order for the antennas
c
      equivalence (igain(1,1,1), idb001(1,1))
      equivalence (igain(1,1,2), ndb001(1,1))
      equivalence (igain(1,1,3), idb041(1,1))
      equivalence (igain(1,1,4), ndb041(1,1))
      equivalence (igain(1,1,5), idb101(1,1))
      equivalence (igain(1,1,6), ndb101(1,1))
      equivalence (igain(1,1,7), idb102(1,1))
      equivalence (igain(1,1,8), ndb102(1,1))
      equivalence (igain(1,1,9), idb121(1,1))
      equivalence (igain(1,1,10),ndb121(1,1))
      equivalence (igain(1,1,11),idb122(1,1))
      equivalence (igain(1,1,12),ndb122(1,1))
      equivalence (igain(1,1,13),idb141(1,1))
      equivalence (igain(1,1,14),ndb141(1,1))
      equivalence (igain(1,1,15),idb142(1,1))
      equivalence (igain(1,1,16),ndb142(1,1))
      equivalence (igain(1,1,17),idb144(1,1))
      equivalence (igain(1,1,18),ndb144(1,1))
      equivalence (igain(1,1,19),idb161(1,1))
      equivalence (igain(1,1,20),ndb161(1,1))
c
c     21st entry - isotropic - do not change this entry
c
      equivalence (igain(1,1,21),idb000(1,1))
c
      data adata1/' 001 - frd-10 cdaa            [r]',
     &           ' 001 - frd-10 cdaa            [r]',
     &           ' 041 - oe316/tsc99 hermes lp [r]',
     &           ' 041 - oe316/tsc99 hermes lp [r]',
     &           ' 101 - quarter wave vertical     ',
     &           ' 101 - quarter wave vertical     ',
     &           ' 102 - loaded whip (short)       ',
     &           ' 102 - loaded whip (short)       ',
     &           ' 121 - half wave horiz dipole    ',
     &           ' 121 - half wave horiz dipole    '/
      data adata2/' 122 - inverted l                ',
     &           ' 122 - inverted l                ',
     &           ' 141 - terminated rhombic        ',
     &           ' 141 - terminated rhombic        ',
     &           ' 142 - terminated sloping vee    ',
     &           ' 142 - terminated sloping vee    ',
     &           ' 144 - horizontal lpa  (54 deg) ',
     &           ' 144 - horizontal lpa  (54 deg) ',
     &           ' 161 - vertical log periodic     ',
     &           ' 161 - vertical log periodic     '/
```

```
      data adata3/' 000 - isotropic (or unknown)   '/
c
c     Frequencies used in gain tables
c
      data freset/ 2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,25,
     &             30/
c
c     Polarization data for the 21 antennas
c
      data pdata/ 'v','v','h','h','v',
     &            'v','v','v','h','h',
     &            'h','h','h','h','h',
     &            'h','h','h','v','v',
     &            'v',' ',' ',' '/
c
c     Azimuth pattern data for the 21 antennas
c     0 = omindirectional, + = bidirectional, - = unidirectional
c
      data sdata/ 0.0,  0.0,  0.0,  0.0,  0.0,
     &            0.0,  0.0,  0.0,  2.8,  2.8,
     &           -8.5,-11.0,-45.0,-45.0,-32.0,
     &          -32.0,-3.5, -3.5, -1.0, -1.0,
     &            0.0/
c
      data tdata/ 001,-001, 041,-041, 101,-101, 102, -102, 121,-121,
     &     122,-122, 141,-141, 142,-142, 144,-144, 161,-161, 000/
c
      data idb001/  9, 11, 13, 15, 17, 18, 20,  7,  8,  9, 10, 12, 14,
     l             17, 19, 21, 23, 21, 22, 20, 16,
     m             11, 13, 14, 16, 16, 17, 17, 10, 10, 10, 10, 10, 10,
     m              9,  9,  9,  9, 11, 12, 18, 15,
     s              9,  9,  9,  9, 12, 14, 16, 11, -1,-12,-24,-27,-29,
     s            -32,-34,-36,-40,-36,-31,-15,-23/
      data ndb001/  7,  7,  8,  8, 11, 15, 18,  9, -7,-24,-20,-20,-20,
     &            -20,-20,-20,-20,-20,-20,-20,-20,
     &            -20,-20,-20,-20,-20,-20,-20,-20,-20,-20,-20,-20,-20,
     &            -20,-20,-20,-20,-20,-20,-20,-20,
     &            -20,-20,-20,-20,-20,-20,-20,-20,-20,-20,-20,-20,-20,
     &            -20,-20,-20,-20,-20,-20,-20,-20/
      data idb041/ -1,  0,  1,  1,  2,  3,  3,  4,  4,  5,  5,  5,  6,
     &              6,  7,  7,  9,  8, 10,  9, 11,
     &              4,  5,  6,  7,  8,  8,  9,  9, 10, 10, 11, 11, 12,
     &             12, 12, 13, 13, 13, 14, 15, 16,
     &              5,  6,  7,  8,  9, 10, 10, 11, 11, 11, 11, 11, 12,
     &             12, 12, 12, 11, 11, 11,  9,  6/
      data ndb041/  5,  7,  8,  8,  9,  9, 10, 10, 10, 10, 10,  9,  9,
     &              9,  8,  7,  7,  5,  4,-15, -2,
     &              5,  6,  7,  7,  7,  7,  6,  5,  4,  2, -2, -7,-20,
     &            -13, -5, -2,  1,  2,  3, -2, -9,
     &              5,  6,  6,  5,  3,  1, -4,-15,-11, -3, -1,  1,  2,
     &              1,  0, -5,-11, -8, -6,  0, -1/
      data idb101/ -2, -2, -3, -3, -3, -4, -3, -3, -3, -3, -3, -3, -3,
     l             -3, -2, -2, -2, -2, -2,-10,-10,
     m              1,  1,  1,  1,  1,  2,  2  2,  2,  3,  3,  3,  3,
     m              4,  4,  4,  4,  4,  4,-10,-10,
     s              0,  0,  1,  1,  1,  2,  2,  2,  1,  1,  1,  1,  0,
     s              0,  0,  0,  0,  0,  0,  1,  0/
      data ndb101/ -2, -1, -1,  0,  0, -1, -1, -2, -2, -2, -2, -1, -1,
```

```
&                 -1,  0, -1, -1, -1, -1, -1, -1,
&                 -7, -5, -6, -9, -8, -7, -7, -5, -7, -9, -8, -7, -7,
&                 -7, -7, -7, -8, -7, -6, -7, -7,
&                -10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,
&                -10,-10,-10,-10,-10,-10,-10,-10/
 data idb102/-10, -9, -5, -3, -1,  0,  0,  0,  0,  0, -1, -1, -2,
l                -2, -3, -3, -4, -4, -4, -6, -8,
m               -10, -8, -4, -2,  0,  0,  1,  1,  1,  1,  1,  0,  0,
m                 0, -1, -1, -2, -2, -3,  0, -7,
s               -10,-10,-10, -8, -6, -5, -5, -5, -5, -5, -5, -6, -6,
s                -6, -7, -7, -8, -8, -9,-10,-10/
 data ndb102/-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,
&                -10,-10,-10,-10,-10,-10,-10,-10,
&                -10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,
&                -10,-10,-10,-10,-10,-10,-10,-10,
&                -10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,
&                -10,-10,-10,-10,-10,-10,-10,-10/
 data idb121/ -1, -2, -3, -3, -4, -4, -4, -4, -5, -5, -5, -5, -5,
l                -5, -5, -5, -5, -5, -5, -5, -5,
m                 2,  2,  2,  2,  2,  1,  1,  1,  1,  1,  1,  1,  1,
m                 1,  1,  1,  1,  1,  1,  1,  1,
s                 3,  3,  3,  3,  3,  3,  3,  3,  3,  3,  3,  3,  3,
s                 3,  3,  3,  3,  3,  3,  3,  3/
 data ndb121/  3,  3,  3,  3,  3,  3,  3,  3,  3,  3,  3,  3,  3,
l                 3,  3,  3,  3,  3,  3,  3,  3,
m                 3,  3,  3,  3,  3,  3,  3,  3,  3,  3,  3,  3,  3,
m                 3,  3,  3,  3,  3,  3,  3,  3,
s                 3,  3,  3,  3,  3,  3,  3,  3,  3,  3,  3,  3,  3,
s                 3,  3,  3,  3,  3,  3,  3,  3/
 data idb122/-10, -9, -7,-10,-10, -7, -5, -6, -9,-10,-10,-10, -8,
l               -10, -7, -3, -3, -6,-10,-10,-10,
m               -10, -4, -2, -1,  0,  1,  2,  1, -2, -7, -8, -5, -4,
m                -8,  0,  3,  3,  0, -8, -4, -6,
s               -10, -4,  1,  3,  4,  4,  4,  2, -1, -5, -7, -6, -7,
s                -8, -1, -4,  0, -2, -7,  0, -7/
 data ndb122/-17, -7, -1,  3,  4,  4,  4,  3, 0, -5,-10, -9, -8,
&                 -9,-10, -6, -2, -1,  0, -1, -2,
&               -19, -7,  0,  4,  5,  5,  4,  2, -1, -5,-11,-14,-15,
&               -16,-17,-14,-10,-10,-10, -4, -9,
&               -20, -7,  0,  4,  5,  5,  4,  2, -1, -6,-13,-20,-20,
&               -20,-20,-19,-16,-12, -8, -4,-15/
 data idb141/-10,-10,-10, -9, -4, -1,  2,  5,  7,  9, 11, 13, 14,
l                15, 16, 17, 18, 18, 19, 19, 17,
m               -10, -8, -1,  4,  8, 11, 13, 15, 16, 17, 18, 18, 18,
m                17, 16, 14, 12,  9,  6,-10,  0,
s                -7,  2,  7, 10, 11, 10,  7,  1, -7,-10,-10,-10,-10,
s                -3,  4,  7,  9,  7,  6, -7,  4/
 data ndb141/ -3,  4,  7,  7,  3, -5,-10,-10,-10,-10, -5,  0,  1,
&                 4,-10, -8, -6, -1,  4,-10, 6,
&                -1,  1, -6,-10, -6, -1, -6,-10,-10,-10,  0,  1, -8,
&                -8, -7, -8, -9, -9,-10, -9,-10,
&                -3,-10, -10, 0, -9,-10,-10,-10,-10, -9, -1,-10,-10,
&               -10,-10,-10,-10,-10,-10,-10,-10/
 data idb142/-10,-10,-10, -9, -6, -4, -2,  0,  1,  2,  3,  4,  5,
l                 5,  6,  6,  7,  7,  7,  5,  7,
m               -10, -7, -3,  0,  2,  4,  6,  7,  7,  8,  8,  8,  7,
m                 6,  6,  4,  2,  0, -1,  1,  0,
s               -10, -4, -1,  0,  0, -1, -3, -5, -4, -3, -2, -3, -6,
```

```
     s               -5, -5, -3, -2, -5, -7,  1, -6/
      data ndb142/ -9, -5, -3, -4, -7, -6, -4, -3, -6,-10,-10,-10,-10,
     &               -9, -8, -5, -1, -4, -6, -6,-10,
     &              -10,-10,-10, -7, -8,-10,-10,-10,-10,-10,-10,-10,-10,
     &               -8,-10, -7, -7, -8,-10, -7, -8,
     &              -10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,
     &              -10,-10,-10,-10,-10,-10,-10,-10/
      data idb144/-10,-7, -4, -2, -1, -1, -1, -1, -1,  0,  0,  0,  1,
     l                1,  1,  2,  2,  1,  1,  2,  4,
     m               -1,  3,  5,  7,  7,  8,  8,  8,  8,  8,  9,  8, 10,
     m                9,  9, 10, 10, 10, 10,  9, 10,
     s                4,  7,  8,  9,  9,  9,  9,  9,  9, 10,  9,  9, 10,
     s                9,  8,  9,  9,  8,  8,  6,  5/
      data ndb144/  5,  7,  8,  9,  9,  9,  8,  8,  8,  8,  8,  7,  8,
     &                7,  6,  6,  6,  5,  4,  0, -5,
     &                6,  7,  7,  7,  6,  5,  4,  3,  2,  2,  1,  0,  1,
     &               -4,  2, -4, -4, -7,-10, -5,  0,
     &                6,  7,  6,  5,  4,  3,  2,  3,  2,  2,  1, -1,  0,
     &               -1, -3, -2, -2, -3, -4, -2, -1/
      data idb161/-10,  7,  6,  6,  5,  5,  4,  4,  4,  4,  4,  4,  3,
     l                3,  3,  3,  3,  3,  3,  3,  3,
     m              -10,  7,  7,  6,  6,  6,  6,  5,  5,  5,  5,  5,  5,
     m                5,  5,  5,  5,  5,  5,  5,  5,
     s              -10, -7, -7, -8, -8, -8, -8, -8, -8, -8, -8, -8, -7,
     s               -7, -7, -7, -7, -7, -7, -7, -7/
      data ndb161/-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,
     &              -10,-10,-10,-10,-10,-10,-10,-10,
     &              -10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,
     &              -10,-10,-10,-10,-10,-10,-10,-10,
     &              -10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,-10,
     &              -10,-10,-10,-10,-10,-10,-10,-10/
c
c     Isotropic gain table - never change or delete this entry
c
      data idb000/ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
     l                0,  0,  0,  0,  0,  0,  0,  0,
     m                0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
     m                0,  0,  0,  0,  0,  0,  0,  0,
     s                0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
     s                0,  0,  0,  0,  0,  0,  0,  0/
c
c     Lagrange interpolation method, where
c       deg : launch angle
c       a,b,c : interpolation angles
c       x : antenna gain of angle a
c
      fn(deg,x,a,b,c) = (deg-b)*(deg-c)/((a-b)*(a-c))*x
c
c     If frequency equals zero, this must be a table definition call
c     Set up all elements, and return with a gain of zero
c
      if( f .ne. 0.0 ) go to 200
c
c     Initializaton of the variables
c
        do i=1,21,1
          aname(i) = adata(i)
          po(i)    = pdata(i)
```

```
            shape(i) = sdata(i)
            antabl(i)= tdata(i)
          end do
c
        gain=0.0
        go to 9999
c
c       Preset gain to 0.0 (isotropic)
c
200     gain=0.0
c
c       Insure antenna type exists
c
        klong = 0
        kshort= 0
500     do n=1,21,1
          if( iabs(itype) .gt. 255 .or. itype .eq. 0 ) return
          if( itype .eq. antabl(n) ) klong = n
          if( itype .eq.-antabl(n) ) kshort= n
        end do
c
1500    if (klong .eq. 0 .or. kshort .eq. 0) return
c
c       Combines upper and lower antenna tables into new array itmp
c
        index = klong
        do k=1,6,1
          do j=1,21,1
            if (k .ge. 4) then
              itmp(j,k) = igain(j,k-3,index+1)
            else
              itmp(j,k) = igain(j,k,index)
            end if
          end do
        end do
c
c       Compute mainlobe antenna gain based on
c       launch angle and frequency for this antenna
c
c       Compute the frequency index
c
        ii=int(f)
        i=ii
        if(ii.lt.2) i=2
        if(ii.gt.20.and.ii.lt.25) i=20
        if(ii.ge.25.and.ii.lt.30) i=21
        if(ii.ge.30) i=22
        i=i-1
c
c       Compute the elevation angle
c
        eradis = 6371.0
        h = 320.0
        do n = 1,5,1
          x = r / (2 * n * eradis)
          y = eradis / (eradis + h)
          radian = atan((cos(x) - y) / sin(x))
          dlt = radian * 57.2958
```

```
         if (dlt .ge. 3.5) go to 4000
      end do
c
c     Determine which three angles to interpolate
c
4000  if (klong .ne. 0 .or. kshort .ne. 0) then
         m=1
         if (dlt .gt. 40.0) m=2
         if (dlt .gt. 50.0) m=3
         if (dlt .gt. 70.0) m=4
         go to (4100,4200,4300,4400),m
c
c     If launch angle <= 40 deg, use the angles 6, 20, 40 to interpolate
c
4100     x1 = 6.0
         x2 = 20.0
         x3 = 40.0
         ifirst = 1
         iscond = 2
         ithird = 3
         go to 4500
c
c     If 40 < launch angle <= 50, use the angles 20, 40, 50 to
c     interpolate
c
4200     x1 = 20.0
         x2 = 40.0
         x3 = 50.0
         ifirst = 2
         iscond = 3
         ithird = 4
         go to 4500
c
c     If 50 < launch angle <= 70, use the angles 40, 50, 70 to
c     interpolate
c
4300     x1 = 40.0
         x2 = 50.0
         x3 = 70.0
         ifirst = 3
         iscond = 4
         ithird = 5
         go to 4500
c
c     If launch angle > 70 deg, use the angles 50, 70, 90 to
c     interpolate
c
4400     x1 = 50.0
         x2 = 70.0
         x3 = 90.0
         ifirst = 4
         iscond = 5
         ithird = 6
c
c     Three point Lagrange interpolation - one above and one below
c     the frequency
c
4500     do icount=1,2,1
```

```
          templ = float(itmp(i,ifirst))
          temp2 = float(itmp(i,iscond))
          temp3 = float(itmp(i,ithird))
          yl = fn(dlt,templ,xl,x2,x3)
          y2 = fn(dlt,temp2,x2,xl,x3)
          y3 = fn(dlt,temp3,x3,xl,x2)
          gains(icount) = yl + y2 + y3
          i = i+1
          if (i .eq. 22) i=21
        end do
       if ( f .ge. 25.0 )  i=22
c
c     Compute the gain for input frequency (LUF)
c     Gain(1) is the gain below the input frequency (LUF)
c     Gain(2) is the gain above the input frequency (LUF)
c
        gain =((-freset(i-2)+f) / (freset(i-1)-freset(i-2))) *
     &           (gains(2) - gains(1)) + gains(1)
      else
        gain = 0.0
      end if
c
c     Keep the minimum antenna gain to -10.0 dB
c
      if(gain .le. -10.0) gain=-10.0
9999  return
      end
```

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 1990	Specification Document: Jun 1989—Sep 1989

4. TITLE AND SUBTITLE

SPECIFICATION FOR HF LOWEST USABLE FREQUENCY (LUF) MODEL

5. FUNDING NUMBERS

54-MP10
DN309 030
OMN

6. AUTHOR(S)

T. N. Roy

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Ocean Systems Center
San Diego, CA 92152-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

TD 1753

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Naval Oceanographic Office
NSTL Station
Bay St. Louis, MS 39522

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This report documents a high frequency (HF) lowest usable frequency (LUF) model called LUF. Developed at the Naval Ocean Systems Center, this semiempirical model permits simplified HF propagation predictions to be performed in near-real-time. The model predicts the LUF for arbitrary input values of receiver/transmitter position, date, time (UT), X-ray flux level, transmitter power, transmitter and receiver antenna type, transmitter and receiver antenna bearing, and required signal-to-noise ratio. Several algorithms are contained in the model to describe the dependence of the LUF on these input parameters. An appendix contains the computer FORTRAN 77 code required to implement the fundamental features of this model.

14. SUBJECT TERMS

lowest usable frequency
high frequency
electromagnetic propagation
communications modeling
ionospheric propagation

15. NUMBER OF PAGES

61

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAME AS REPORT