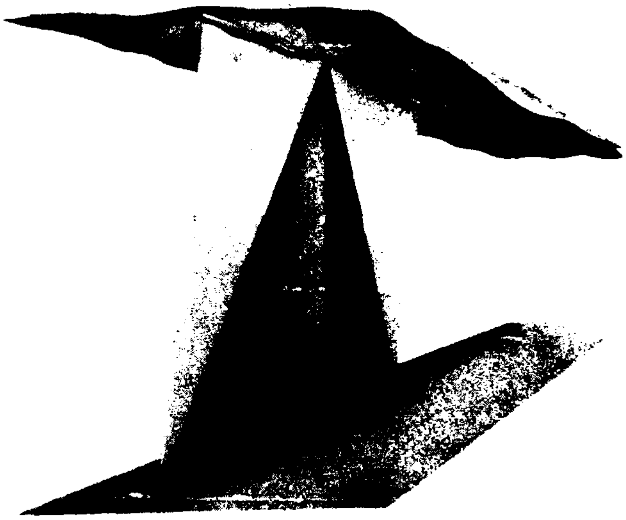
4

AD-A210 516

**Technical Document 1499**
April 1989

# "TWIRE:" A Test Program for VLF Airborne Transmissions

R. A. Pappert
L. R. Hitney



**Approved for public release; distribution is unlimited.**

89 [illegible]

# NAVAL OCEAN SYSTEMS CENTER

San Diego, California 92152-5000

---

| | |
|---|---|
| E. G. SCHWEIZER, CAPT, USN | R. M. HILLYER |
| **Commander** | **Technical Director** |

## ADMINISTRATIVE INFORMATION

This work was performed by Code 542 of Naval Ocean Systems Center and funded by the Defense Nuclear Agency.

| | |
|---|---|
| Released by<br>J. A. Ferguson, Head<br>Modeling Branch | Under authority of<br>J. H. Richter, Head<br>Ocean and Atmospheric<br>Sciences Division |

# REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION
UNCLASSIFIED

1b. RESTRICTIVE MARKINGS

2a. SECURITY CLASSIFICATION AUTHORITY

2b. DECLASSIFICATION/DOWNGRADING SCHEDULE

3. DISTRIBUTION/AVAILABILITY OF REPORT
Approved for public release; distribution is unlimited.

4. PERFORMING ORGANIZATION REPORT NUMBER(S)
NOSC TD 1499

5. MONITORING ORGANIZATION REPORT NUMBER(S)

6a. NAME OF PERFORMING ORGANIZATION
Naval Ocean Systems Center

6b. OFFICE SYMBOL (if applicable)
Code 542

7a. NAME OF MONITORING ORGANIZATION

6c. ADDRESS (City, State and ZIP Code)
San Diego, CA 92152-5000

7b. ADDRESS (City, State and ZIP Code)

8a. NAME OF FUNDING/SPONSORING ORGANIZATION
Defense Nuclear Agency

8b. OFFICE SYMBOL (if applicable)
DNA

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

8c. ADDRESS (City, State and ZIP Code)
Washington, DC 20305

10. SOURCE OF FUNDING NUMBERS

| PROGRAM ELEMENT NO. | PROJECT NO. | TASK NO. | AGENCY ACCESSION NO. |
|---|---|---|---|
| PE62715H | 599QMXBB | 542MP20 | DN651 524 |

11. TITLE (include Security Classification)
"TWIRE:" A TEST PROGRAM FOR VLF AIRBORNE TRANSMISSIONS

12. PERSONAL AUTHOR(S)
R. A. Pappert and L. R. Hitney

13a. TYPE OF REPORT
Interim

13b. TIME COVERED
FROM May 1987 TO Jan 1988

14. DATE OF REPORT (Year, Month, Day)
April 1989

15. PAGE COUNT
102

16. SUPPLEMENTARY NOTATION

17. COSATI CODES

| FIELD | GROUP | SUB-GROUP |
|---|---|---|
| | | |
| | | |
| | | |

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)

VLF airborne transmissions
point dipole
counterpoise
towline configurations
drogue radius
modal eigenangle
waveguide calculations

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

A computer program called TWIRE, which combines a TACAMO configuration code, a radiation resistance code and a VLF propagation code, has been assembled for the purpose of conducting case studies for airborne VLF transmissions and for the purpose of pointing out possible future areas of improvement for such transmissions. A major deficiency of the program is that it is not fully automated because the configuration code requires initial inputs which often have to be determined by trial and error. Nevertheless, the program is useful for case studies and provides a tool for assessing the adequacy for field calculation purposes of point dipoles as approximations to rationally determined TACAMO antenna configurations. Sample results of such comparisons are included.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT
☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT ☐ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED

22a. NAME OF RESPONSIBLE PERSON
R. A. Pappert

22b. TELEPHONE (include Area Code)
(619) 553-3069

22c. OFFICE SYMBOL
Code 542

DD FORM 1473, 84 JAN

83 APR EDITION MAY BE USED UNTIL EXHAUSTED
ALL OTHER EDITIONS ARE OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

DD FORM 1473, 84 JAN

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

# CONTENTS

## APPENDIX

## FIGURES

## FIGURES (continued)

## FIGURES (continued)

# 1. INTRODUCTION

Many years ago a computer program was developed at NADC (Huang 1969) which computes the steady state shape of the TACAMO towline assuming an aircraft orbiting in a circular orbit at constant altitude and speed. Although the towline configuration was never checked experimentally, the vertical separation between towplane and conical drogue at the bottom of the towline did compare favorably with a limited number of measurements. Bickel et al (1971) using this code along with a VLF propagation program (Pappert and Shockey 1971) presented some calculations for the vertical electric field at the ground produced by a TACAMO antenna and compared those results with fields generated by an elevated rotating dipole. To the authors' knowledge those are the only long range propagation comparisons made with a rationally determined TACAMO antenna.

The VLF propagation code mentioned above was developed for calculating VLF fields produced by antennas of arbitrary length, shape and elevation. The program was based on simple segmentation of the antenna with each segment acting as a short dipole. The field strength at any distance from the antenna was then calculated as the phasor sum of the contributions from each segment. To perform the calculations presented by Bickel et al (1971), the positions and orientations of the antenna segments had to be manually input to the propagation code along with the current moment for each segment. The latter required an assumed current amplitude and current distribution along the antenna. In particular, no rationale was used to relate current amplitude to radiated power. The latter is generally the quantity assumed in systems calculations.

The purpose of the present report is to describe a computer program called TWIRE which combines several programs and facilitates calculations of the type made by Bickel et al (1971) and which uses as one of its fundamental inputs the power radiated by the antenna. Specifically, TWIRE combines the configuration code of NADC with a simple radiation resistance program (Pappert 1986) and a fast mode conversion propagation code (Ferguson & Snyder, 1980). TWIRE also makes allowance for the counterpoise which is typically about an eighteenth of the rf wavelength (Fern 1986).

As mentioned above, the NADC code calculates the steady state antenna configuration by assuming an aircraft executing a circular orbit at constant altitude and speed. The radiation resistance code assumes a sinusoidal current distribution along the antenna and is strictly valid for a flat infinitely conducting ground. Nevertheless, it is believed to be a reasonable approximation for high conductivity ground (e.g., seawater). Towline plus counterpoise is assumed to be one-half of the rf wavelength. Propagation in either laterally homogeneous or inhomogeneous environments is allowed for by the fast-mode conversion program.

The NADC program determines steady state configurations of the towline from force equations by an iteration procedure which uses as starting conditions the radius and altitude of the conical drogue at the bottom of the towline. To achieve convergence it is generally necessary to begin with good drogue starting conditions. That requirement, unfortunately, prevents full automation of TWIRE. An effective operating procedure is to determine starting conditions by running the NADC code separately for a set of operating conditions, and to then employ those starting conditions in TWIRE.

A brief description of the NADC model is given in the following section. The radiation resistance formula is summarized in section 3 and formulas for mode sums are given in section 4. A brief program description is presented in section 5 and sample input and output is discussed in section 6. Sample results are given in section 7 and a summary including areas of improvement concludes the study in section 8. A listing of TWIRE is given in appendix A and for the purpose of running separately to obtain good drogue starting conditions, the NADC program is listed in appendix B.

## 2. BRIEF DESCRIPTION OF THE NADC MODEL

Several of the assumptions used in constructing the NADC model are (Huang 1969):

1. Towline is inextensible but perfectly flexible.
2. Towplane travels in a perfectly circular path at a constant altitude and speed with no towline pay-out or reel-in.
3. No wind or no wind shear present.
4. Air density varies with altitude as in a standard atmosphere.

Working with a cylindrical coordinate system Huang (1969) derived the steady state configuration of the towline by summing up the forces acting on an element of the towline $\Delta S$ and equating the sum to zero. His vector force equation is:

$$\frac{\partial \vec{T}}{\partial s} + C_D \rho \frac{d}{2} |\vec{V}_\perp| \vec{V}_\perp + \pi C_f \rho \frac{d}{2} |\vec{V}_{Rel}| \vec{V}_{Rel} - \mu g \vec{k} - \mu \frac{\partial^2 \vec{r}}{\partial t^2} = 0 , \qquad (1)$$

where

$s$ = distance along towline measured from its bottom.

$\vec{T}$ = towline tension at s.

$d$ = diameter of towline.

$\rho$ = air density.

$\vec{V}_\perp$ = component of the relative air velocity perpendicular to the towline.

$\vec{V}_{Rel}$ = relative air velocity.

$C_D$ = drag coefficient (dimensionless).

$C_f$ = skin friction coefficient (dimensionless).

$\mu$ = towline mass/unit length.

$g$ = gravitational acceleration.

$\vec{k}$ = unit vector in z (altitude) direction.

$\vec{r}$ = $r \vec{u}_r$ = radius vector from axis of rotation to point on towline.

$\vec{u}_r$ = unit vector from axis of rotation to point on towline.

$r$ = towline radius measured from axis of rotation.

When reduced to component form equation (1) becomes (Huang 1969)

$$(TR')' - Tr(\Theta')^2 + C_D \frac{d}{2} \rho(1 - (r\Theta')^2)^{1/2} r^3 r' \omega^2 \Theta' + \mu r \omega^2 = 0 \, , \tag{2}$$

$$(Tr\Theta')' + Tr'\Theta' - \pi C_f \frac{d}{2} \rho(r\omega)^2 - C_D \frac{d}{2} \rho(r\omega)^2 (1 - (r\Theta')^2)^{3/2} = 0 \, , \tag{3}$$

$$(Tz')' + C_D \frac{d}{2} \rho r^3 \omega^2 \Theta' z' (1 - (r\Theta')^2)^{1/2} - \mu g = 0 \, . \tag{4}$$

A derivative with respect to s is indicated by the prime, $\theta$ is the polar angle, z the altitude of the towline at position s and $\omega$ is the angular speed of the orbiting towplane. Huang (1969) solved these equations by an iteration procedure beginning at the drogue end. As mentioned in the introduction, this generally requires good drogue starting conditions. To complete the description of the problem, force equations and parameters appropriate to the drogue are also required. Unfortunately, the NADC report does not make it clear how the program could be altered to accommodate other drogues, nor is there sufficient discussion to know where many of the drogue parameters came from (presumably some were determined experimentally). Whether or not the equations and parameters used in the NADC report would be suitable, for example, to airborne command post towlines and drogues, is not known.

Under some operating conditions, it is known that two steady state solutions to equations (2) through (4) exist (Huang 1969 and Fern 1986). For the conditions used as examples in the results section of this report, it is believed that only a single solution exists. An often observed operational behavior is a yo-yo effect associated with an orbiting aircraft. Fern (1986) gives an insightful discussion of the effect's origin, pointing out how the lift on the towline varies from upwind to downwind portions of the aircraft orbit and how quite large excursions of the towline could occur if the operating conditions are such that two steady state configurations exist simultaneously. Fern (1986), using the electromagnetic code NEC (Burke and Poggio 1977), also studied electrical properties of the TACAMO antenna. Although superior to the radiation resistance calculation used in this report, NEC would probably be too cumbersome to use in combination with the fast mode conversion program. Perhaps, though, MININEC (Rockway and Logan 1986) could be incorporated in the future.

## 3. RADIATION RESISTANCE

This section contains the formula for the radiation resistance used in this report for thin antennas of arbitrary elevation and orientation over perfectly conducting ground. The formula is based on segmentation of the antenna and figure 1 shows the configuration for the $n^{th}$ dipole of current moment $\vec{M}_n$. The dipole is located at $(x_n, y_n, z_n)$ with orientation $\phi_n$ and $\gamma_n$ relative to the x and z axes, respectively. For the special case of a perfectly conducting ground the time averaged radiated power is (see reference 8):

$$Pw = 30\, k^2 \sqrt{\frac{\pi}{2}} \sum_{n,m} M_n M_m$$

$$\cdot \left\{ \left[ \cos\gamma_n \cos\gamma_m + \frac{1}{2} \sin\gamma_n \sin\gamma_m \cos(\phi_n - \phi_m) \right] (\omega_{nm})^{1/2} J_{1/2}(\omega_{nm}) \right.$$

$$
\begin{aligned}
&+ \left[\cos\gamma_n \cos\gamma_m - \frac{1}{2}\sin\gamma_n \sin\gamma_m \cos(\phi_n - \phi_m)\right](\omega^+_{nm})^{1/2} J_{1/2}(\omega^+_{nm}) \\
&+ \left[-\cos\gamma_n \cos\gamma_m + \frac{1}{2}\sin\gamma_n \sin\gamma_m \cos(\phi_n - \phi_m)\right] \\
&\cdot (\omega_{nm})^{3/2}\left[J_{3/2}(\omega_{nm}) - (z_{nm})^2(\omega_{nm})^{-1}J_{5/2}(\omega_{nm})\right] \\
&- \left[\cos\gamma_n \cos\gamma_m + \frac{1}{2}\sin\gamma_n \sin\gamma_m \cos(\phi_n - \phi_m)\right] \\
&\cdot (\omega^+_{nm})^{3/2}\left[J_{3/2}(\omega^+_{nm}) - (z^+_{nm})^2(\omega^+_{nm})^{-1} J_{5/2}(\omega^+_{nm})\right] \\
&+ \left[\cos\gamma_n \sin\gamma_m \cos(\Phi_{nm} - \phi_m) z_{nm} + \cos\gamma_m \sin\gamma_n \cos(\Phi_{nm} - \phi_n) z_{nm}\right. \\
&\left. + \frac{1}{2}\sin\gamma_n \sin\gamma_m \cos(2\Phi_{nm} - \phi_n - \phi_m) r_{nm}\right] \cdot r_{nm}(\omega_{nm})^{-5/2} J_{5/2}(\omega_{nm}) \\
&+ \left[-\cos\gamma_n \sin\gamma_m \cos(\Phi_{nm} - \phi_m) z^+_{nm} + \cos\gamma_m \sin\gamma_n \cos(\Phi_{nm} - \phi_n) z^+_{nm}\right. \\
&\left. - \frac{1}{2}\sin\gamma_n \sin\gamma_m \cos(2\Phi_{nm} - \phi_n - \phi_m) r_{nm}\right] r_{nm}(\omega^+_{nm})^{5/2} J_{5/2}(\omega^+_{nm}) \Big\} \, , \qquad (5)
\end{aligned}
$$

where

$$
r_{nm} = k\left((x_n - x_m)^2 + (y_n - y_m)^2\right)^{1/2}, \; z^{\pm}_{nm} = k(z_n \pm z_m) \, , \qquad (6)
$$

$$
\Phi_{nm} = \tan^{-1} \frac{y_n - y_m}{x_n - x_m} \, , \; \omega^{\pm}_{nm} = \left((z^{\pm}_{nm})^2 + r^2_{nm}\right)^{1/2} . \qquad (7)
$$

Also, k is the free space wavenumber and the J's are half integer order Bessel functions of the first kind which are expressible in terms of sines and cosines and, therefore, easily calculated. In the case of a single dipole, equation (5) is reduced to the result (the dipole subscript is omitted in (8) and (9) so that $\gamma$ is the dipole orientation relative to the z axis):

$$
Pw = 20\, k^2\, M^2\, f(z^+, \gamma) \, , \qquad (8)
$$

where $z^+ = 2kz$ with z the dipole height above ground and where

$$
\begin{aligned}
f(z^+, \gamma) = &\left[1 + \frac{3}{(z^+)^3}(\sin(z^+) - z^+\cos z^+)\right]\cos^2\gamma \\
&+ \left[1 + \frac{3}{2(z^+)^3}((1 - (z^+)^2)\sin z^+ - z^+\cos z^+)\right]\sin^2\gamma \, . \qquad (9)
\end{aligned}
$$

To complete the description between radiated power and antenna current, the current distribution is assumed sinusoidal, a reasonable assumption for thin antennas. In particular, as mentioned in the introduction, the towline length plus counterpoise is taken to be one-half of the free space rf wavelength. That distribution coupled with equation (5) then suffices to determine the current amplitude in terms of the radiated power.

## 4. MODE SUMS

In the following (x, z) is the plane of propagation with x the range coordinate and positive z directed towards the ionosphere with z = 0 the ground. The x coordinate of the towplane is taken to be x = 0. For a slab mode conversion model with $x_n$ the beginning of the n$^{th}$ slab and with the first slab described by the region $x < x_2$ the mode sums for a W segmented antenna may be written as

$$E_n^{(1)} = Q \sum_m f_{nm}^{(1)} (z_R) \exp(-ikS_m^{(1)}x) \sum_{w=1}^{W} \frac{M_w \exp(ikS_m^{(1)}\bar{x}_w)}{\left[\sin\left(\dfrac{x-\bar{x}_w}{a}\right)\right]^{1/2}} \Big[\lambda_{1nm}^{(1)} f_{1m}^{(1)} (\bar{z}_w)\cos\gamma_w$$

$$+ \lambda_{2nm}^{(1)} f_{2m}^{(1)} (\bar{z}_w)\sin\gamma_w \sin\phi_w + \lambda_{3nm}^{(1)} f_{3m}^{(1)} (\bar{z}_w)\sin\gamma_w\cos\phi_w\Big] , \qquad (10)$$

$$E_n^{(p)} = Q \sum_j \sum_m a_{jm}^{(p)} \left[\delta_{1n} \frac{S_j^{(p)}}{S_m^{(1)}} + (1-\delta_{1n})\right] f_{nj}^{(p)} (z_R)\exp(-ik(S_m^{(1)}x_2 + S_j^{(p)} (x - x_p))$$

$$\cdot \sum_{w=1}^{W} \frac{M_w \exp(ikS_m^{(1)}\bar{x}_w)}{\left[\sin\dfrac{(x-\bar{x}_w)}{a}\right]^{1/2}} \Big[\lambda_{1nm}^{(1)} f_{1m}^{(1)} (\bar{z}_w) \cos\gamma_w + \lambda_{2nm}^{(1)} f_{2m}^{(1)} (\bar{z}_w) \sin\gamma_w\sin\phi_w$$

$$+ \lambda_{3nm}^{(1)} f_{3m}^{(1)} (\bar{z}_w) \sin\gamma_w\cos\phi_w\Big] , \quad p \neq 1 , \qquad (11)$$

where j and m are mode indices and w is the segment index. The barred quantities represent midpoint values for the w$^{th}$ segment. The superscripts denote slab number and the subscript, n, denotes the electric field component at the receiver with the convention, n = 1 → EZ, n = 2 → EY, n = 3 → EX. The λ's and f's are excitation factors and height gains, respectively. The first subscripted index on λ pertains to the orientation of the transmitter with $\lambda_1$ being the vertical dipole excitation factor, $\lambda_2$ the broadside excitation factor and $\lambda_3$ the end-on excitation factor. Similarly, $f_1$ is the modal height gain for EZ, $f_2$ the modal height gain for EY and $f_3$ the modal height gain for EX. With allowance for a different indexing convention and height gain normalization, formulas for the λ's and the f's will be found in reference 9. The $a_{jm}^{(p)}$ are cumulative mode conversion coefficients and physically represent the accumulative conversion from a unit amplitude wave in mode m in the transmitter region to mode j in the p$^{th}$ slab. Reference 11 shows how the cumulative conversion coefficients are calculated by the "Fast MC" method. Other quantities appearing in equations (10) and (11) are the dipole moment, M; the earth's radius, a; the sine of the modal eigenangle, S; the free space wavenumber, k; the dipole segment orientation factors, (γ, ϕ), defined in the previous section; the receiver altitude, $z_R$ and $i = \sqrt{-1}$. The excitation factors and eigenangle inputs are obtained from the computer code MODESRCH developed by Morfitt and Shellman (1976). With the dipole moment expressed in units of amp-meters and the electric field strength E in microvolts/m, the constant Q is

$$Q = 2.853 \times 10^{-3} f^{3/2} \qquad (12)$$

where f is the frequency in kilohertz.

Equation (10) is used for laterally homogeneous waveguide calculations and equation (11) for laterally inhomogeneous waveguide calculations. Sample results are given in a later section. An additional point is that off axis antenna effects have been ignored in equations (10) and (11). That should be a good approximation for ranges of several hundred kilometers or greater.

## 5. BRIEF PROGRAM DESCRIPTION

Appendix A contains a listing of TWIRE. The nucleus of this code is the FASTMC code (Ferguson and Snyder 1980) and in this section only those program elements of FASTMC which have been altered or program elements which have been added to FASTMC will be discussed.

MAIN controls the program flow. Data is supplied to MAIN via namelist/datum/. The latter is identical to its counterpart in FASTMC. However, the quantities al, incl, theta and talt which appear in namelist are not used in TWIRE. Main calls TACAMO, COORD and RPOWER which are not program elements of FASTMC.

TACAMO is the NADC program for calculating the steady state towline configuration for an aircraft orbiting in a circle at constant altitude and speed. Data is supplied to TACAMO via namelist /tacin/. The namelist inputs are:

v = aircraft speed in knots.

rpl = aircraft radius in ft.

zpl = aircraft altitude in ft.

rzd = drogue starting radius in ft. If not sufficiently close to final iterate the program will abort.

zzd = drogue starting altitude in ft. If not sufficiently close to final iterate the program will abort.

amg = drogue weight in lbs (typically ≈ 100 lbs).

psi = rotation angle (degrees) discussed subsequently in connection with the subroutine coord.

iclock = 0 if aircraft executes a counterclockwise orbit, ≠ 0 otherwise.

itrset = number of times Huang's original iteration scheme is used before changing to a two dimensional Newton iteration which has been added to the original NADC program. Generally, itrset has been set to zero in carrying out the sample calculations in this study.

cpl = counterpoise length in ft.

chicpl = inclination of counterpoise in degrees from horizontal ($90^\circ \leq \text{chicpl} \leq 0^\circ$ with 0° horizontal and 90° vertical).

Aerodynamic characteristics of the conical drogue, determined by parameters not listed above, are fixed in data statements and were apparently derived from data presented in reference 7. Towline drag and mass coefficients are also fixed in data statements. Lengths of the towline segments in ft. are given in the data statement dstore for segments 1 through 17 and are given by dds for segments greater than 17. These values can be altered to accommodate individual needs.

The Newton iteration mentioned above is developed from the definitions

$$f_1 = z_l(z_d, r_d) - z_{pl} \,, \tag{13}$$

$$f_2 = r_l(z_d, r_d) - r_{pl} \,, \tag{14}$$

where $z_l$ is the iterated plane altitude and $r_l$ the iterated plane radius corresponding to the starting drogue conditions $z_d$ and $r_d$. Also, $z_{pl}$ and $r_{pl}$ are the true plane altitude and radius respectively. Expanding $f_1$ and $f_2$ about $z_d$ and $r_d$ yields

$$f_1(z_d + \delta z_d,\, r_d + \delta r_d) \approx f_1(z_d, r_d) + \frac{\partial z_l}{\partial z_d}\,\delta z_d + \frac{\partial z_l}{\partial r_d}\,\delta r_d \approx 0 \,, \tag{15}$$

$$f_2(z_d + \delta z_d,\, r_d + \delta r_d) \approx f_2(z_d, r_d) + \frac{\partial r_l}{\partial z_d}\,\delta z_d + \frac{\partial r_l}{\partial r_d}\,\delta r_d \approx 0 \,. \tag{16}$$

Solution of equations (15) and (16) gives

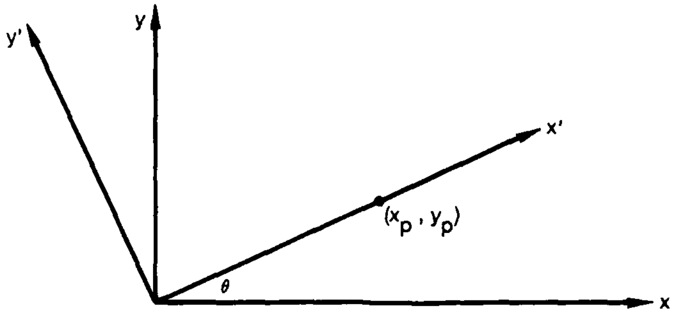
$$\delta z_d = \left( f_2 \frac{\partial z_l}{\partial r_d} - f_1 \frac{\partial r_l}{\partial r_d} \right) / \Delta \,,\; \delta r_d = \left( f_1 \frac{\partial r_l}{\partial z_d} - f_2 \frac{\partial z_l}{\partial z_d} \right) / \Delta \,, \tag{17}$$

$$\Delta = \frac{\partial z_l}{\partial z_d}\,\frac{\partial r_l}{\partial r_d} - \frac{\partial r_l}{\partial z_d}\,\frac{\partial z_l}{\partial r_d} \,. \tag{18}$$

TACAMO is also included as a separate entry in Appendix B. As mentioned previously, it is best when working with new operating conditions to run TACAMO by itself to get starting conditions for TWIRE. A possible step to fuller automation would be to develop a multidimensional grid of starting values for a cross section of operating conditions and to use interpolation to determine drogue starting values for intermediate operating conditions. In that connection, operating frequencies range from about 17 kHz to 30 kHz and TACAMO aircraft generally operate at air speeds between about 170 and 250 knots at altitudes between about 20,000 and 40,000 ft. with radii > 4,000 ft. (Fern 1986).

COORD performs the following coordinate transformations. In the schematic below, the towplane's (x,y) coordinates are denoted by $(x_p, y_p)$. The axis are first rotated by the angle $\theta = \tan^{-1}(y_p/x_p)$, so that in the x', y' system $y'_p = 0$.

The y' axis is the direction of the towplane for counterclockwise rotation (i.e., iclock = 0). Coordinates of the point (x,y) in the unprimed system become in the prime system



$$x' = x\cos\theta + y\sin\theta\ , \tag{19}$$

$$y' = -x\sin\theta + y\cos\theta\ . \tag{20}$$

The prime coordinate system is next translated so that the towplane is at the origin. That is

$$x_T' = x' - x_p'\ , \tag{21}$$

$$y_T' = y'\ . \tag{22}$$

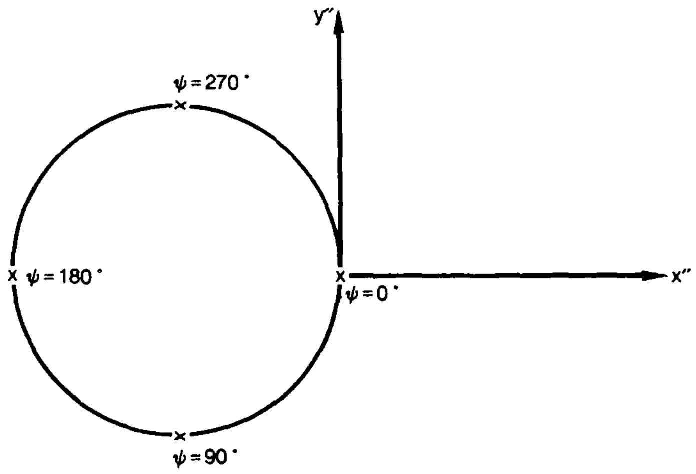
If the airplane is moving in a clockwise rotation (iclock = 1), the y′ coordinates are reversed in sign. Let $\psi$ (psi) be the direction between the x′ direction (i.e., direction from center of orbit to towplane) and the direction of propagation, denoted by x″.

Then,

$$x'' = x_T'\cos\psi + y_T'\sin\psi\ , \tag{23}$$

$$y'' = -x_T'\cos\psi + y_T'\cos\psi\ . \tag{24}$$

Values of $\psi$ at cardinal points on the orbit with x″ directed to the right are indicated below.



The counterpoise coordinates are also calculated in COORD subject to the assumption that the counterpoise is in the direction $-\vec{v}/|\vec{v}|$, where $\vec{v}$ is the towplane's velocity and droops at an angle chicp with the horizontal.

RPOWER is the program element where equation (5) is implemented subject to the assumption that the antenna length plus counterpoise is one-half the rf wavelength and that the current is sinusoidally distributed. Outputs of the subroutine are the current moments (idl) of the antenna segments for a given radiated power input, pin.

MCFLDS is an original program element of the program FASTMC which in TWIRE has been modified to allow for antenna segmentation. IF iopt = 1 (range option flag), MCFLDS is called from MAIN for each range between dmin and dmax at deltad intervals and the electric field components EZ or EY are computed according to equations (10) or (11). Field values in complex form are contained in SUMOUT and field amplitudes in Amp. If iopt = 2 (height gain option), MCFLDS is used as an intermediate step in calculating the electric field components. Final field calculations in this instance are performed in MAIN. Again, field values in complex form are contained in SUMOUT and field amplitudes in Amp.

## 6. SAMPLE INPUT AND OUTPUT

Three files are required for input to TWIRE. The three files are called twzy.in, twzy.com and cards.30. They represent input for a range calculation of the vertical electric field component EZ and the horizontal electric field component EY for a laterally homogeneous guide. The file twzy.in first calls for the file twzy.com and then for data contained in namelist tacin, which is used for the towline configuration calculation. Listed in order in namelist tacin are:

v = towplane velocity in knots.

rpl = towplane radius in feet.

zpl = towplane altitude in feet.

rzd = drogue starting radius in feet.

zzd = drogue starting altitude in feet.

amg = drogue weight in pounds.

psi = angle in degrees between radius vector from center of towplane's orbit to towplane and the direction of rf propagation.

iclock = flag for clockwise (1) or counterclockwise (0) towplane rotation.

chicp = droop angle of the counterpoise measured in degrees from the horizontal.

cpl = counterpoise length in feet.

The namelist tacin is then repeated a second time as the configuration calculation is repeated for each component calculated.

The file twzy.com begins with the character string "name". Its remaining contents in this instance are:

rcomp = electric field component calculated (1 for EZ, 2 for EY).

nprint = printout flag

= 0 none

= 1 slab identification data and field amplitude and phase

= 3,4 intermediate mode conversion data used for diagnostics only.

naplot = 0 amplitude plot flag (0 for no plot, 1 for plot).

dmin = minimum range in Mm.

dmax = maximum range in Mm.

nrptsl = number of ranges calculated at equal increments between dmin and dmax.

ampmax = upper limit of field amplitude plot dB/$\mu$V/m.

ampmin = lower limit of field amplitude plot in dB/$\mu$V/m.

ralt = receiver altitude in km.

cards.30 = waveguide mode constant data (see e.g., reference 6).

The calculations are repeated for rcomp = 2 (i.e., for EY calculations).

The waveguide input file, cards.30, obtained from reference 6 begins with DATA followed by a descriptor title. The next line identifies the slab properties as follows:

R = range location of slab in km.

F = frequency in kHz.

A = propagation azimuth from magnetic north (degrees).

C = geomagnetic field codip (degrees).

M = geomagnetic field strength ($W/m^2$).

S = ground conductivity (S/m).

E = ground dielectric constant.

T = Top height used for waveguide calculations (km).

# INPUT FILE: TWZY.IN

```
twzy.com
 &tacin
 v=200.,
 rpl=4000.,
 zpl=20000.,
 rzd=700.0,
 zzd=9000.,
 amg=100.,
 psi=180.0,
 iclock=0,
 chicp=0.,
 cpl=1800.,
 &end
y
y
 &tacin
 v=200.,
 rpl=4000.,
 zpl=20000.,
 rzd=700.0,
 zzd=9000.,
 amg=100.,
 psi=180.0,
 iclock=0,
 chicp=0.,
 cpl=1800.,
 &end
```

## INPUT FILE: TWZY.COM

```
name
 &datum rcomp=1,nprint=1,naplot=0,
        dmin=0.000,dmax=5.000,
        nrptsl=11,
        ampmax=80.,ampmin=0.,
        ralt=9.144, &end
input cards.30
name
 &datum rcomp=2,&end
start
quit
```

## INPUT FILE: CARDS.30

```
DATA
towline test
R   .000 F 30.0000 A  90.000 C  40.000 M  .500E-04 S 4.640E+00 E 81.0 T 87.0
1 89.97300 -6.722872 1.14900804E-05-9.69784160E-05-5.66454815E-13 8.91278878E-16
2 89.97300 -6.722872 4.94460073E-09 5.55613688E-09 1.00000072E+00-7.01791265E-08
1 89.79732 -6.138471 1.39372627E-04-4.28004772E-04-9.34471576E-13-3.41234067E-13
2 89.79732 -6.138471-9.26029919E-09-1.90273877E-08 1.00000119E+00 9.79737678E-08
1 89.91204 -3.159701-5.67914452E-04-1.12545611E-02-4.18753747E-12-1.40717722E-12
2 89.91204 -3.159701 1.34759489E-07 1.77823267E-C7 1.00000048E+00-4.46208873E-07
1 88.82600 -1.083422 2.33682222E-03-5.44904545E-03-2.69803867E-11-6.00914040E-14
2 88.82600 -1.083422-2.19763109E-07-3.34162451E-07 1.00000072E+00 7.85546206E-09
1 84.71127  -.222291-2.45107268E-03-1.99295953E-02-1.35681544E-11-9.35714700E-12
2 84.71127  -.222291 2.98093312E-07 4.92028562E-07 1.00000048E+00 2.00596716E-07
1 82.74648  -.273032 2.91138096E-03-4.90130810E-03-1.01852499E-10 9.41601918E-12
2 82.74648  -.273032-4.07881686E-07-6.45559965E-07 1.00000012E+00-5.70226248E-08
1 80.40218  -.234551-3.23274592E-03-1.51210632E-02-5.33708217E-11-3.51053041E-11
2 80.40218  -.234551 5.61018396E-07 8.20390710E-07 1.00000036E+00 1.91906668E-08
1 78.70553  -.293832 3.81615874E-03-6.45386567E-03-1.83607657E-10 3.90313511E-11
2 78.70553  -.293832-6.92323852E-07-9.63369075E-07 1.00000048E+00 2.79036698E-07
1 76.58138  -.261791-4.16685035E-03-1.15461908E-02-1.42259260E-10-8.68795105E-11
2 76.58138  -.261791 9.04359183E-07 1.10827352E-06 9.99999583E-01 3.27973055E-07
1 74.96912  -.345282 4.87595052E-03-8.77621677E-03-2.45324872E-10 9.70813221E-11
2 74.96912  -.345282-1.07347228E-06-1.22331073E-06 9.99999821E-01 4.46039110E-07
1 72.92796  -.296672-5.04938047E-03-8.35422613E-03-3.03403247E-10-1.68299430E-10
2 72.92796  -.296672 1.32530784E-06 1.27687861E-06 1.00000048E+00 9.59014642E-07
1 71.32272  -.409471 5.76619152E-03-1.14735551E-02-2.64180205E-10 1.84621096E-10
2 71.32272  -.409471-1.53844314E-06-1.33109688E-06 1.00000048E+00 9.05900492E-07
1 69.33108  -.339462-5.64039499E-03-5.37072308E-03-5.54849555E-10-2.73569306E-10
2 69.33108  -.339462 1.79977269E-06 1.25654230E-06 9.99999106E-01 1.31630159E-07
1 67.68570  -.479001 6.25931984E-03-1.42788989E-02-2.23180446E-10 2.91375785E-10
2 67.68570  -.479001-2.05986134E-06-1.21624373E-06 9.99997795E-01 2.21355563E-07
1 65.73211  -.392672-5.82792237E-03-2.63220887E-03-9.07945108E-10-3.89095423E-10
2 65.73211  -.392672 2.30049409E-06 1.01230273E-06 9.99995053E-01-3.48421486E-06
1 64.01385  -.546571 6.22112770E-03-1.70211233E-02-1.11896700E-10 3.96509547E-10
2 64.01385  -.546571-2.59987064E-06-8.40697567E-07 1.00000191E+00-1.35956998E-05
1 62.08826  -.459332-5.56169311E-03-1.49533094E-04-1.37232026E-09-4.94906449E-10
2 62.08826  -.459332 2.79967230E-06 5.28230146E-07 1.00002074E+00 2.41414864E-06

r 40.0
```

ramax = maximum receiver altitude in km (used with iopt = 2).

ratic = tic mark spacing on receiver altitude scale (used with iopt = 2).

dist = range locations in Mm at which height gains are calculated (used with iopt = 2).

nrd = numbers of dist (used with iopt = 2).

nrpts2 = number of height gain points calculated between ramin and ramax (used with iopt = 2).

Following the quantities contained in the datum namelist, comes the slab identification data described previously and the tacin namelist data also described previously (except for itrset which controls the iterations as described in section 5 — default is 0). Apart from input quantities discussed in connection with namelist tacin, the following appear:

abase = base area of drogue in sq. ft.

cddrog = drogue drag coefficient.

amug = towline weight lbs/ft.

d = towline diameter in ft.

al = towline length in ft.

dds = a spacing increment along the towline in ft.

picf = towline skin friction coefficient.

cd = towline drag coefficient.

All of the above, except al, are set in data statements in the subroutine TACAMO. The quantity al is calculated subject to the condition that the towline length plus counterpoise is one-half a rf wavelength.

Following the slab identification card is the mode data. Each eigenangle in degrees appears twice along with four complex quantities from which all excitation factors are calculated. The cards.30 file is terminated with the r = 40.0 character string. If more slabs were used in the modeling (i.e., lateral inhomogeneity allowed for) then slab identification and data for each additional slab would follow in order with a space separating each new slab entry beginning with the range information (R etc.) record.

Output corresponding to the preceding input is shown on pages 16 through 21. The output first echoes the namelist datum. Quantities not specified in the datum entry in twzy.com appear in the output along with their default values. In order, quantities not discussed in connection with the input are (also see reference 3):

iopt = range or height gain option flat (1 for range, 2 for height gain).

al, incl, theta, talt, TOPHT = quantities not used as input in TWIRE.

npplot = phase plot flag (0 no plot, 1 plot).

npdiff = not used in present program.

nrcurv = number of curves on a plot (maximum of 4).

sizex = size of x axis in inches.

sizey = size of y axis in inches.

amptic = tic mark spacing on amplitude axis.

phsmax = phase axis maximum in degrees.

phsmin = phase axis minimum in degrees.

phstic = tic mark spacing on phase axis.

dtic = tic mark spacing in Mm on range axis.

tlong, tlat, rbear, totape = not used in TWIRE.

power = radiated power in kw.

ramin = minimum receiver altitude in km (used with iopt = 2).

The towline configuration follows and is described by:

s = distance in feet along towline as measured from its bottom.

r = radius of towline in feet at s.

z = altitude of towline in feet at s.

th = azimuth of towline in radians measured in counterclockwise direction at s.

The remaining quantities t, rp, zp and rthp are of no interest in the present application. In the output it will be seen that the program iterated twice, there being three sets of s, r, z and th values given at the bottom and top of the towline. Final values calculated from the bottom to the top follow. Next comes imid which is the towline segment where the current is a maximum. The midpoint coordinates are listed for each segment after the coordinate transformations discussed in section 5 have been made. Also listed is gamma and phi for each segment where gamma is the inclination in degrees of the segment from vertical and phi is its azimuth in degrees. Following the midpoint information is a printout of the electric field component being calculated and the receiver altitude. The principal output then follows showing dist in Mm, field amplitude in dB/$\mu$V/m and its phase in degrees. The output then repeats itself for the second field component, EY.

OUTPUT FILE

```
name
$DATUM
IOPT =  1,
ROOMP =  1,
AL =  0.000000E+00,
INCL =  0.000000E+00,
THETA =  0.000000E+00,
TALT =  0.000000E+00,
RALT =  9.14400005,
TOPHT =  90.0,
NAPLOT =  0,
NPPLOT =  0,
NPDIFF =  0,
NPRINT =  1,
NRCURV =  1,
SIZEX =  5.0,
SIZEY =  6.0,
AMPMAX =  80.0,
AMPMIN =  0.000000E+00,
AMPTIC =  10.0,
PHSMAX =  360.0,
PHSMIN =  -360.0,
PHSTIC =  90.0,
DMAX =  5.0,
DMIN =  0.000000E+00,
DTIC =  1.0,
TLONG =  0.000000E+00,
TLAT =  0.000000E+00,
RBEAR =  0.000000E+00,
POWER =  1.0,
TOTAPE =  0,
NRPTS1 =  11,
RAMIN =  0.000000E+00,
RAMAX =  50.0,
RATIC =  5.0,
DIST =  0.000000E+00, 0.000000E+00, 0.000000E+00, 0.000000E+00,
0.000000E+00, 0.000000E+00, 0.000000E+00, 0.000000E+00, 0.000000E+00,
0.000000E+00, 0.000000E+00, 0.000000E+00, 0.000000E+00, 0.000000E+00,
0.000000E+00, 0.000000E+00, 0.000000E+00, 0.000000E+00, 0.000000E+00,
0.000000E+00,
NRD =  0,
NRPTS2 =  51
$end
input cards.30
DATA
towline test
Slab  1 R   .000 F 30.0000 A  90.000 C  40.000 M  .500 S 4.640E+00 E 81.0 T 87.0
Insufficient number of slabs, horizontally homogeneous only
$TACIN
V =  200.0,
RPL =  4000.0,
ZPL =  20000.0,
RZD =  700.0,
ZZD =  9000.0,
```

# OUTPUT FILE (CONTINUED)

```
AMG =  100.0,
PSI =  180.0,
ICLOCK =  0,
ITRSET =  0,
CPL =  1800.0,
CHICP =  0.00000000000000E+000
$end
v(knot)=   .2000E+03  rpl=   .4000E+04  zpl=   .2000E+05  rzd=   .7000E+03  zzd=   .9000E+04
amg=   .1000E+03  abase=   .5600E+01  cddrog=   .6000E+00
amug=   .1095E+00  d=   .1750E-01  al=   .1459E+05  dds=   .1000E+04  picf=   .2200E-01  cd=   .1030E+01
     s          r          z          th          t          rp          zp          rthp
 .0000E+00  .7000E+03  .9000E+04  .0000E+00  .9561E+02  -.1381E+00  .9885E+00  .6113E-01
 .1459E+05  .3712E+04  .2080E+05  .4635E+01  .1291E+04   .3618E+00  .3811E+00  .8508E+00
     s          r          z          th          t          rp          zp          rthp
 .0000E+00  .7402E+03  .8893E+04  .0000E+00  .9539E+02  -.1453E+00  .9877E+00  .5688E-01
 .1459E+05  .3960E+04  .2002E+05  .4715E+01  .1214E+04   .3296E+00  .3132E+00  .8907E+00
     s          r          z          th          t          rp          zp          rthp
 .0000E+00  .7508E+03  .9002E+04  .0000E+00  .9536E+02  -.1472E+00  .9875E+00  .5578E-01
 .1459E+05  .4000E+04  .2000E+05  .4724E+01  .1201E+04   .3232E+00  .3044E+00  .8960E+00
     s          r          z          th          t          rp          zp          rthp
 .0000E+00  .7508E+03  .9002E+04  .0000E+00  .9536E+02  -.1472E+00  .9875E+00  .5578E-01
 .2000E+02  .7479E+03  .9022E+04  .1657E-02  .9758E+02  -.1474E+00  .9867E+00  .6837E-01
 .7000E+02  .7405E+03  .9071E+04  .7221E-02  .1031E+03  -.1474E+00  .9843E+00  .9712E-01
 .2000E+03  .7215E+03  .9198E+04  .2988E-01  .1174E+03  -.1451E+00  .9769E+00  .1569E+00
 .4000E+03  .6934E+03  .9393E+04  .8329E-01  .1393E+03  -.1356E+00  .9661E+00  .2196E+00
 .6000E+03  .6678E+03  .9585E+04  .1540E+00  .1608E+03  -.1202E+00  .9580E+00  .2605E+00
 .8000E+03  .6458E+03  .9776E+04  .2376E+00  .1821E+03  -.1004E+00  .9524E+00  .2879E+00
 .1000E+04  .6280E+03  .9966E+04  .3309E+00  .2033E+03  -.7758E-01  .9488E+00  .3063E+00
 .1300E+04  .6102E+03  .1025E+05  .4832E+00  .2348E+03  -.4072E-01  .9459E+00  .3220E+00
 .1600E+04  .6037E+03  .1053E+05  .6438E+00  .2660E+03  -.3129E-02  .9448E+00  .3277E+00
 .1900E+04  .6081E+03  .1082E+05  .8058E+00  .2971E+03   .3278E-01  .9445E+00  .3267E+00
 .2200E+04  .6227E+03  .1110E+05  .9639E+00  .3280E+03   .6455E-01  .9447E+00  .3216E+00
 .2600E+04  .6554E+03  .1148E+05  .1163E+01  .3690E+03   .9915E-01  .9447E+00  .3125E+00
 .3000E+04  .7003E+03  .1186E+05  .1345E+01  .4098E+03   .1250E+00  .9443E+00  .3043E+00
 .3400E+04  .7541E+03  .1223E+05  .1511E+01  .4503E+03   .1440E+00  .9431E+00  .2997E+00
 .3800E+04  .8145E+03  .1261E+05  .1664E+01  .4906E+03   .1581E+00  .9408E+00  .2997E+00
 .4400E+04  .9141E+03  .1317E+05  .1876E+01  .5504E+03   .1739E+00  .9352E+00  .3086E+00
 .5000E+04  .1023E+04  .1373E+05  .2073E+01  .6095E+03   .1875E+00  .9262E+00  .3270E+00
 .6000E+04  .1223E+04  .1465E+05  .2386E+01  .7051E+03   .2138E+00  .9028E+00  .3731E+00
 .7000E+04  .1453E+04  .1553E+05  .2688E+01  .7962E+03   .2452E+00  .8665E+00  .4348E+00
 .8000E+04  .1716E+04  .1637E+05  .2985E+01  .8808E+03   .2819E+00  .8147E+00  .5068E+00
 .9000E+04  .2017E+04  .1715E+05  .3277E+01  .9568E+03   .3196E+00  .7463E+00  .5839E+00
 .1000E+05  .2352E+04  .1786E+05  .3563E+01  .1022E+04   .3512E+00  .6633E+00  .6609E+00
 .1100E+05  .2713E+04  .1847E+05  .3838E+01  .1077E+04   .3709E+00  .5722E+00  .7315E+00
 .1200E+05  .3086E+04  .1900E+05  .4101E+01  .1120E+04   .3737E+00  .4818E+00  .7926E+00
 .1300E+05  .3453E+04  .1944E+05  .4351E+01  .1155E+04   .3613E+00  .4010E+00  .8418E+00
 .1400E+05  .3803E+04  .1981E+05  .4589E+01  .1185E+04   .3388E+00  .3352E+00  .8791E+00
 .1459E+05  .4000E+04  .2000E+05  .4724E+01  .1201E+04   .3232E+00  .3044E+00  .8960E+00
imid= 21
 AFTER  TRANSFORMATION
  segment      xmid       ymid        zmid        gamma      phi
     1       3991.607   -749.322    9011.749      9.200     66.535
     2       3994.340   -744.185    9046.258      9.736     60.267
     3       4004.651   -730.949    9134.634     11.189     49.348
```

OUTPUT FILE (CONTINUED)

```
         4         4031.104    -706.514     9295.523       13.616       39.187
         5         4071.887    -676.366     9488.878       15.806       34.128
         6         4119.392    -645.224     9680.599       17.204       32.436
         7         4170.580    -612.793     9871.172       18.074       32.282
         8         4236.866    -569.909    10108.325       18.640       33.301
         9         4316.654    -515.276    10392.220       18.987       35.481
        10         4394.909    -456.552    10675.718       19.104       38.281
        11         4470.218    -393.580    10959.109       19.123       41.521
        12         4552.580    -314.074    11289.741       19.100       45.842
        13         4639.201    -215.955    11667.590       19.151       51.274
        14         4716.023    -109.182    12045.240       19.305       57.235
        15         4781.598       6.547    12422.377       19.607       63.643
        16         4842.999     165.626    12892.170       20.260       72.285
        17         4888.013     373.282    13452.779       21.420       82.969
        18         4875.182     681.105    14189.255       23.711       97.507
        19         4750.215    1089.204    15088.847       27.589      115.286
        20         4469.080    1495.129    15951.461       32.566      132.807
        21         4018.064    1846.740    16761.982       38.460      150.065
        22         3406.158    2079.605    17504.593       44.976      167.169
        23         2673.062    2129.689    18165.823       51.681     -175.876
        24         1888.200    1950.678    18738.163       58.075     -159.094
        25         1138.123    1528.149    19222.350       63.723     -142.580
        26          506.998     884.130    19627.093       68.347     -126.555
        27          115.445     255.867    19905.950       71.337     -114.284
        28             .001     900.000    20000.764       90.000       90.000
 z COMP    RALT =    9.144
  DIST   AMPLITUDE  PHASE          DIST   AMPLITUDE  PHASE          DIST   AMPLITUDE  PHASE          DIST   AMPLITUDE  PHASE
   .000   99.9900     .0000        1.500   49.1092  410.5222        3.000   45.2930  622.4276        4.500   38.9580  561.7921
   .500   54.3566  151.4190        2.000   52.4274  414.5638        3.500   45.3529  661.7356        5.000   45.4528  552.6127
  1.000   29.7413  297.9408        2.500   35.2352  444.8034        4.000   41.8110  667.8203
1
 name
 $DATUM
 IOPT =  1,
 RCOMP =  2,
 AL =  0.000000E+00,
 INCL =  0.000000E+00,
 THETA =  0.000000E+00,
 TALT =  0.000000E+00,
 RALT =  9.14400005,
 TOPHT =  90.0,
 NAPLOT =  0,
 NPPLOT =  0,
 NPDIFF =  0,
 NPRINT =  1,
 NRCURV =  1,
 SIZEX =  5.0,
 SIZEY =  6.0,
 AMPMAX =  80.0,
 AMPMIN =  0.000000E+00,
 AMPTIC =  10.0,
 PHSMAX =  360.0,
 PHSMIN =  -360.0,
 PHSTIC =  90.0,
```

OUTPUT FILE (CONTINUED)

```
DMAX =  5.0,
DMIN =  0.000000E+00,
DTIC =  1.0,
TLONG =  0.000000E+00,
TLAT =  0.000000E+00,
RBEAR =  720.0,
POWER =  1.0,
TOTAPE =  0,
NRPTS1 =  11,
RAMIN =  0.000000E+00,
RAMAX =  50.0,
RATIC =  5.0,
DIST =  0.000000E+00, 0.000000E+00, 0.000000E+00, 0.000000E+00,
0.000000E+00, 0.000000E+00, 0.000000E+00, 0.000000E+00, 0.000000E+00,
0.000000E+00, 0.000000E+00, 0.000000E+00, 0.000000E+00, 0.000000E+00,
0.000000E+00, 0.000000E+00, 0.000000E+00, 0.000000E+00, 0.000000E+00,
0.000000E+00,
NRD =  0,
NRPTS2 =  51
$end
start
$TACIN
V =  200.0,
RPL =  4000.0,
ZPL =  20000.0,
RZD =  700.0,
ZZD =  9000.0,
AMG =  100.0,
PSI =  180.0,
ICLOCK =  0,
ITRSET =  0,
CPL =  1800.0,
CHICP =  0.0000000000000000E+000
$end
v(knot)=   .2000E+03  rpl=   .4000E+04  zpl=   .2000E+05  rzd=   .7000E+03  zzd=   .9000E+04
amg=   .1000E+03  abase=   .5600E+01  cddrog=   .6000E+00
amug=   .1095E+00  d=   .1750E-01  al=   .1459E+05  dds=   .1000E+04  picf=   .2200E-01  cd=   .1030E+01
      s           r           z           th          t           rp          zp          rthp
  .0000E+00   .7000E+03   .9000E+04   .0000E+00   .9561E+02  -.1381E+00   .9885E+00   .6113E-01
  .1459E+05   .3712E+04   .2080E+05   .4635E+01   .1291E+04   .3618E+00   .3811E+00   .8508E+00
      s           r           z           th          t           rp          zp          rthp
  .0000E+00   .7402E+03   .8893E+04   .0000E+00   .9539E+02  -.1453E+00   .9877E+00   .5688E-01
  .1459E+05   .3960E+04   .2002E+05   .4715E+01   .1214E+04   .3296E+00   .3132E+00   .8907E+00
      s           r           z           th          t           rp          zp          rthp
  .0000E+00   .7508E+03   .9002E+04   .0000E+00   .9536E+02  -.1472E+00   .9875E+00   .5578E-01
  .1459E+05   .4000E+04   .2000E+05   .4724E+01   .1201E+04   .3232E+00   .3044E+00   .8960E+00
      s           r           z           th          t           rp          zp          rthp
  .0000E+00   .7508E+03   .9002E+04   .0000E+00   .9536E+02  -.1472E+00   .9875E+00   .5578E-01
  .2000E+02   .7479E+03   .9022E+04   .1657E-02   .9758E+02  -.1474E+00   .9867E+00   .6837E-01
  .7000E+02   .7405E+03   .9071E+04   .7221E-02   .1031E+03  -.1474E+00   .9843E+00   .9712E-01
  .2000E+03   .7215E+03   .9198E+04   .2988E-01   .1174E+03  -.1451E+00   .9769E+00   .1569E+00
  .4000E+03   .6934E+03   .9393E+04   .8329E-01   .1393E+03  -.1356E+00   .9661E+00   .2196E+00
  .6000E+03   .6678E+03   .9585E+04   .1540E+00   .1608E+03  -.1202E+00   .9580E+00   .2605E+00
  .8000E+03   .6458E+03   .9776E+04   .2376E+00   .1821E+03  -.1[illegible]E+00   .9524E+00   .2879E+00
```

OUTPUT FILE (CONTINUED)

```
 .1000E+04  .6280E+03  .9966E+04  .3309E+00  .2033E+03 -.7758E-01  .9488E+00  .3063E+00
 .1300E+04  .6102E+03  .1025E+05  .4832E+00  .2348E+03 -.4072E-01  .9459E+00  .3220E+00
 .1600E+04  .6037E+03  .1053E+05  .6438E+00  .2660E+03 -.3129E-02  .9448E+00  .3277E+00
 .1900E+04  .6081E+03  .1082E+05  .8058E+00  .2971E+03  .3278E-01  .9445E+00  .3267E+00
 .2200E+04  .6227E+03  .1110E+05  .9639E+00  .3280E+03  .6455E-01  .9447E+00  .3216E+00
 .2600E+04  .6554E+03  .1148E+05  .1163E+01  .3690E+03  .9915E-01  .9447E+00  .3125E+00
 .3000E+04  .7003E+03  .1186E+05  .1345E+01  .4098E+03  .1250E+00  .9443E+00  .3043E+00
 .3400E+04  .7541E+03  .1223E+05  .1511E+01  .4503E+03  .1440E+00  .9431E+00  .2997E+00
 .3800E+04  .8145E+03  .1261E+05  .1664E+01  .4906E+03  .1581E+00  .9408E+00  .2997E+00
 .4400E+04  .9141E+03  .1317E+05  .1876E+01  .5504E+03  .1739E+00  .9352E+00  .3086E+00
 .5000E+04  .1023E+04  .1373E+05  .2073E+01  .6095E+03  .1875E+00  .9262E+00  .3270E+00
 .6000E+04  .1223E+04  .1465E+05  .2386E+01  .7051E+03  .2138E+00  .9028E+00  .3731E+00
 .7000E+04  .1453E+04  .1553E+05  .2688E+01  .7962E+03  .2452E+00  .8665E+00  .4348E+00
 .8000E+04  .1716E+04  .1637E+05  .2985E+01  .8808E+03  .2819E+00  .8147E+00  .5068E+00
 .9000E+04  .2017E+04  .1715E+05  .3277E+01  .9568E+03  .3196E+00  .7463E+00  .5839E+00
 .1000E+05  .2352E+04  .1786E+05  .3563E+01  .1022E+04  .3512E+00  .6633E+00  .6609E+00
 .1100E+05  .2713E+04  .1847E+05  .3838E+01  .1077E+04  .3709E+00  .5722E+00  .7315E+00
 .1200E+05  .3086E+04  .1900E+05  .4101E+01  .1120E+04  .3737E+00  .4818E+00  .7926E+00
 .1300E+05  .3453E+04  .1944E+05  .4351E+01  .1155E+04  .3613E+00  .4010E+00  .8418E+00
 .1400E+05  .3803E+04  .1981E+05  .4589E+01  .1185E+04  .3388E+00  .3352E+00  .8791E+00
 .1459E+05  .4000E+04  .2000E+05  .4724E+01  .1201E+04  .3232E+00  .3044E+00  .8960E+00
imid= 21
 AFTER  TRANSFORMATION
  segment        xmid          ymid         zmid         gamma        phi
      1       3991.607      -749.322     9011.749       9.200      66.535
      2       3994.340      -744.185     9046.258       9.736      60.267
      3       4004.651      -730.949     9134.634      11.189      49.348
      4       4031.104      -706.514     9295.523      13.616      39.187
      5       4071.887      -676.366     9488.878      15.806      34.128
      6       4119.392      -645.224     9680.599      17.204      32.436
      7       4170.580      -612.793     9871.172      18.074      32.282
      8       4236.866      -569.909    10108.325      18.640      33.301
      9       4316.654      -515.276    10392.220      18.987      35.481
     10       4394.909      -456.552    10675.718      19.104      38.281
     11       4470.218      -393.580    10959.109      19.123      41.521
     12       4552.580      -314.074    11289.741      19.100      45.842
     13       4639.201      -215.955    11667.590      19.151      51.274
     14       4716.023      -109.182    12045.240      19.305      57.235
     15       4781.598         6.547    12422.377      19.607      63.643
     16       4842.999       165.626    12892.170      20.260      72.285
     17       4888.013       373.282    13452.779      21.420      82.969
     18       4875.182       681.105    14189.255      23.711      97.507
     19       4750.215      1089.204    15088.847      27.589     115.286
     20       4469.080      1495.129    15951.461      32.566     132.807
     21       4018.064      1846.740    16761.982      38.460     150.065
     22       3406.158      2079.605    17504.593      44.976     167.169
     23       2673.062      2129.689    18165.823      51.681    -175.876
     24       1888.200      1950.678    18738.163      58.075    -159.094
     25       1138.123      1528.149    19222.350      63.723    -142.580
     26        506.998       884.130    19627.093      68.347    -126.555
     27        115.445       255.867    19905.950      71.337    -114.284
     28           .001       900.000    20000.764      90.000      90.000
y COMP    RALT =    9.144
 DIST   AMPLITUDE  PHASE          DIST   AMPLITUDE  PHASE          DIST   AMPLITUDE  PHASE          DIST   AMPLITUDE  PHASE
```

OUTPUT FILE (CONTINUED)

```
    .000   99.9900     .0000       1.500   43.8222    5.9773       3.000   38.5422 -102.1602       4.500   31.2166 -135.0235
    .500   55.1470 -107.5491       2.000   38.6708   47.5314       3.500   38.9922  -47.7427       5.000   30.8122 -128.2652
   1.000   52.0568  -41.7453       2.500   36.1082  -38.3315       4.000   36.0473   -3.8035
 1
  quit
  End of job,  0 curves plotted in  0 graphs
```

## 7. SAMPLE RESULTS

Shown in figures 2 through 5 are drogue altitude and drogue radius for towline lengths ranging between 14,000 and 30,000 ft. The figures apply to a towplane altitude of 30,000 ft and speed of 200 knots. Figure 2 is for a towplane radius, $r_d$, of 4000 ft., figure 3 for $r_d$ = 6000 ft., figure 4 for $r_d$ = 8000 ft. and figure 5 for $r_d$ = 30,000 ft. which approximates straight flight. These curves can be used to obtain starting conditions for other operating conditions by gradually evolving from the conditions of the figures to the desired operating conditions (the procedure can be very tedious in some instances). As mentioned previously, a possible step to fuller automation of the present program would be to develop a multidimensional grid of starting values for a variety of operating conditions, of which figures 2 through 5 are a sample, and to use interpolation to determine drogue starting conditions for intermediate operating conditions.

It will be seen that there is a rapid change in drogue radius and altitude which occurs for a towline length of about 16,500 ft. in figure 3 and for a towline length of about 21,500 ft. in figure 4. On the basis of the examination made in this study it does not appear that there are two solutions in the neighborhood of those points and the physical origin of the rapid changes is not known.

It is known that a simple exponential $\beta$ = 0.5 km $^{-1}$, h′ = 87 km (notation of Wait & Spies, 1964) profile does an excellent job of predicting nocturnal VLF propagation to the east (Pappert and Hitney 1988). Figures 6 through 13 show results for the vertical electric field at the ground generated by a TACAMO antenna under several operating conditions. The results are for easterly propagation (azimuth = 90°) and the $\beta$ = 0.5 km$^{-1}$, h′ = 87 km profile. For all of the figures a seawater path is assumed (i.e., conductivity = 4.64 s/m, relative permittivity = 81) and the geomagnetic field strength is taken to be 0.5 Gauss with a dip of 50°. Also, for all figures the towplane's speed is taken to be 200 knots. Figures 6 through 9 are for 20 kHz whereas figures 10 through 13 are for 30 kHz. Results are given for towplane altitudes (Aalt) of 20,000 and 30,000 ft. and for towplane radii (Rpl) of 4000 and 8000 ft. The counterpoise length has been taken to be 1800 ft. at 30 kHz and 2800 ft. at 20 kHz. All of the figures apply to counterclockwise rotation (Iclock = 0) with $\psi$ = 0° (see section 5) and a radiated power (Pwr) of 1 kW. Also shown on the curves are results for point dipole calculations with altitude and angular orientation factors for the point dipole determined by the towline segment which contains the current maximum (i.e., imid discussed in section 6). In calculating the point dipole results, allowance has been made for its radiation resistance dependence on height and tilt relative to the z axis as expressed by equation (9). Except in rare instances the point dipole results agree with the TACAMO antenna calculation to better than 2 dB and the results are almost indistinguishable in figures 6 and 8. Interestingly these are the cases of high verticality or low $\gamma$'s.

Although probably not as much of interest as the air-to-ground transmission of EZ just discussed, figures 14 through 21 show results for the transverse electric (TE) field component EY at 30,000 ft (i.e., Ralt = 30.00). Other than the receiver altitude difference, the results are for the same set of operating conditions which applied to figures 6 through 13. Shown again are results for the point dipole. It will be seen that except in rare instances the shapes of the mode sum plots for the TACAMO antenna and point dipole calculations are very similar. Agreement in terms of absolute dB levels is not quite as good in this instance as it was for the EZ component and there are occurrences where the signal levels differ by 10 dB or more.

Although the $\beta$ = 0.5 km $^{-1}$, h′ = 87 km profile does not accurately predict westerly propagation (Pappert and Hitney 1988), figures 22 through 25 are presented as illustrating, for that direction (azimuth = 270°) and profile, comparisons between the TACAMO antenna and point dipole calculations. Figures 22 and 23 show results for Ez at the ground at 20 kHz with the towplane altitude 20,000 ft. Figure 22 gives results for the towplane radius of 4000 ft. and figure 23 gives results for the towplane radius of 8000 ft. The TACAMO and point dipole results for the high verticality (low $\gamma$) case are in very good agreement. The TACAMO and point dipole results for the low verticality case (high $\gamma$) shown in figure 23 do not agree as well in this instance as they did for the corresponding easterly propagation case (figure 7). It is known that the VLF fields penetrate to higher altitudes for westerly propagation than they do for easterly propagation. Thus, the modes are more hybrid for westerly propagation and that may be partly responsible for the larger differences noted for the low verticality westerly propagation.

Figures 24 and 25 show results for EY at 30,000 ft. at 20 kHz with the towplane altitude 20,000 ft. Figure 24 gives results for the towplane radius of 4000 ft. and figure 25 gives results for the towplane radius of 8000 ft. Again, the TACAMO and point dipole calculations for the high verticality (low $\gamma$) case are in very good agreement. The differences between the TACAMO and point dipole calculations shown in figure 25 for the low verticality (high $\gamma$) case appear to be comparable to those for the corresponding easterly propagation case (figure 15).

Figures 26 and 27 compare signal levels for easterly propagation for $\psi = 90°$ and 270°. The results apply to a frequency of 30 kHz with the towplane at 20,000 ft. and in an orbit of radius 4000 ft. Figure 26 is for EZ at the ground and figure 27 is for EY at 30,000 ft. Differences between the curves are measures of the orbital modulation expected as the towplane executes its orbit. The modulation expected is one maximum and one minimum associated with each rotation of the towplane. In rare instances, the minimum field occurs for $\psi$'s close to 0° and 180°. Based on curves not shown such a circumstance for the EY component occurs in the present example at a range of 1.425 Mm. The expected modulation in this case is twomaxima and two minima during the course of one rotation of the towplane. Examples of each type of modulation are shown in figure 28. The amplitude excursions at 2.6 Mm for Ez and at 2 Mm for EY are consistent with expectations based on figures 26 and 27 respectively.

The preceding results have all been for laterally uniform waveguides. Results for a laterally nonuniform waveguide are shown in figures 29 and 30. The guide is characterized by waveguide modes corresponding to an azimuth of 90° out to 2 Mm, between 2 Mm and 4 Mm the waveguide modes are taken to be those corresponding to an azimuth of 270° and beyond 4 Mm the waveguide modes are again taken to be those characterized by an azimuth of 90°. The electron density in each slab is described by the $\beta = .5\text{km}^{-1}$, $h' = 87$ km profile used previously. Also, the geomagnetic field magnitude and dip as well as the ground parameters are identical to those used previously. Although such a waveguide is physically unrealistic it can be used as a check on the performance of the program for a laterally inhomogeneous environment. Figure 29 provides results at 30 kHz for EZ at 30,000 ft. The towplane's altitude is 30,000 ft., its radius is 4000 ft. and its speed is 200 knots. Figure 30 gives results at 30 kHz for the EY component at 30,000 ft. for the same towplane conditions. Also, shown on the plots are the point dipole results. The good agreement between the TACAMO antenna results and the point dipole results serves as a check on the performance of TWIRE in a laterally inhomogeneous environment.

# 8. SUMMARY AND AREAS OF IMPROVEMENT

A computer program called TWIRE, which combines a TACAMO antenna configuration code (Huang 1969), a radiation resistance code (Pappert 1986) and a VLF propagation code (Ferguson and Snyder 1980), has been assembled for the purpose of conducting case studies for VLF airborne transmissions and for the purpose of pointing out possible areas of improvement for calculating such transmissions. In particular, by combining the three codes TWIRE provides a tool for conducting VLF airborne transmission studies based on a rationally determined antenna configuration. A major deficiency of the program is that it is not fully automated because the configuration code requires initial inputs which often have to be determined in advance. A possible scheme for overcoming that deficiency would be to provide a multidimensional grid of starting conditions for different operating conditions (i.e., towline length and towplane's altitude, speed and radius) and to use an interpolation scheme for intermediate operating conditions. One problem that would have to be resolved in connection with this approach would be the resolution of how to handle operating conditions when more than one stable towline configuration exists. Other areas of improvement relate to a fuller treatment of the dynamics of the towline. The possibility of extending the steady state NADC model to include transient and wind effects is such an area. Also, the NADC report does not fully describe the origin of the drogue parameters and formulas which provide crucial starting conditions for the integrations necessary to determine the towline configuration. It is not clear, therefore, how to adapt the NADC program to airborne systems other than TACAMO. Better documentation of the NADC code would, therefore, be of value.

Because of simplifications used in the development of the radiation resistance calculations, TWIRE is strictly valid for infinite ground conductivity. This is probably the easiest element to improve. It is quite likely that approximate allowance for finite ground conductivity could be made without increasing cpu time. It is even possible that MININEC could be incorporated in place of the program element RPOWER.

In the present study, the principle objective has been to illustrate the feasibility of performing VLF airborne transmissions for a rationally determined antenna configuration without specific regard to cpu time. It is quite possible that TWIRE can be speeded up by simply rearranging do loops involving sums over modes and antenna segments.

Perhaps the greatest utility of TWIRE will be to simply assess the adequacy, for field calculation purposes, of point dipoles as approximations to TACAMO configurations. For example, limited results given in section 7 would indicate that a sensibly chosen point dipole would suffice for many, if not most, systems applications.

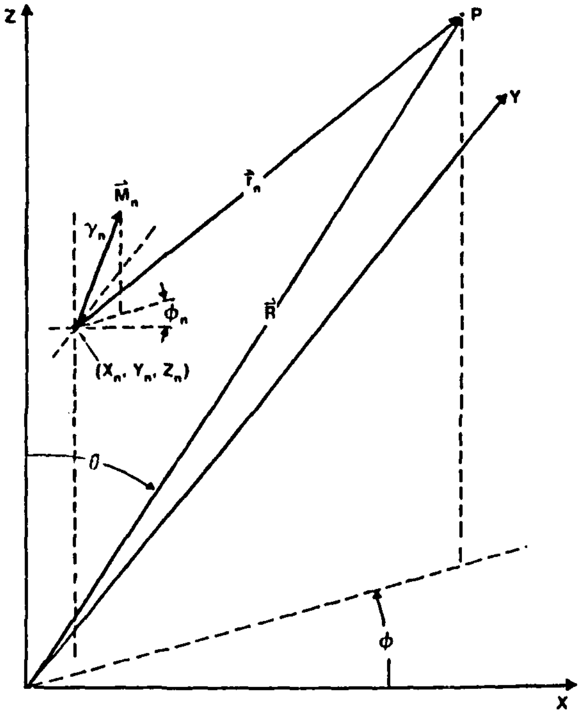


**Figure 1.** Dipole geometry.

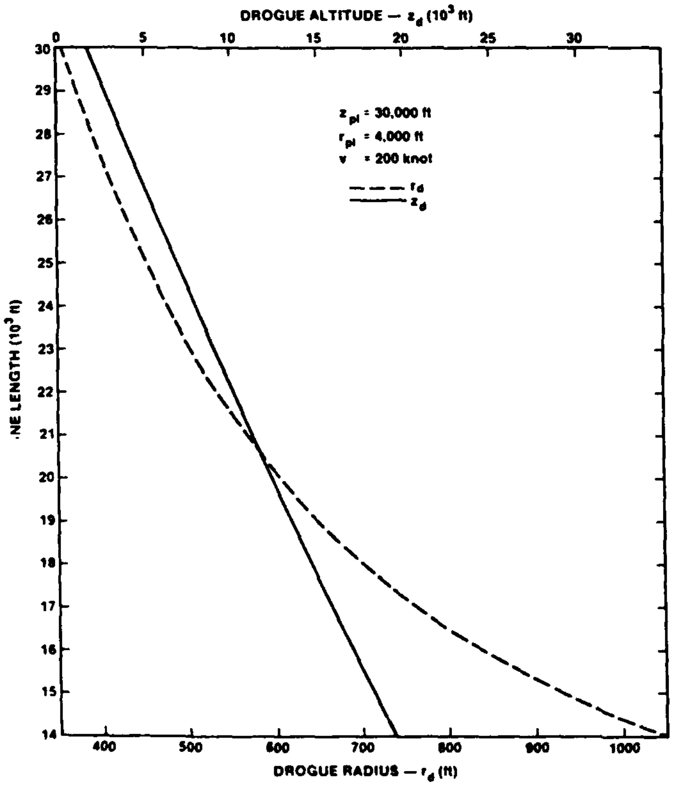


**Figure 2.** Towline length vs. drogue starting conditions.

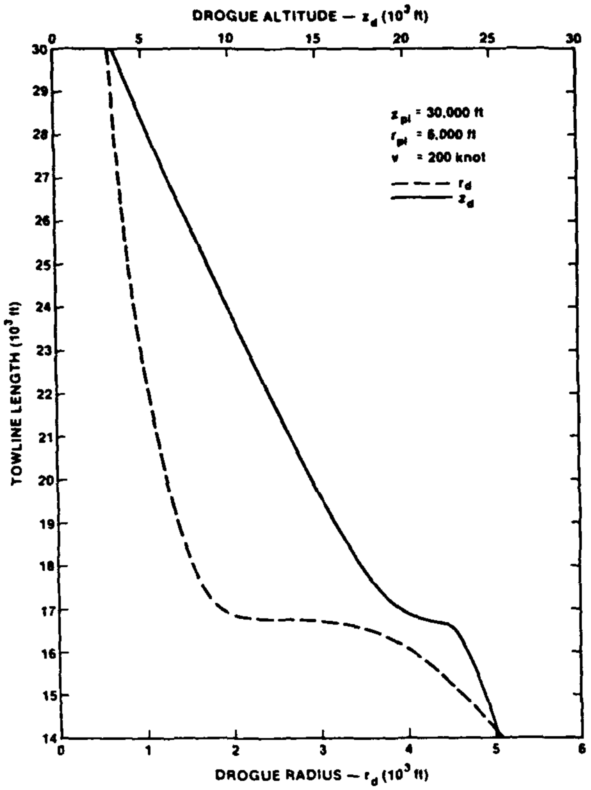


**Figure 3.** Towline length vs. drogue starting conditions.

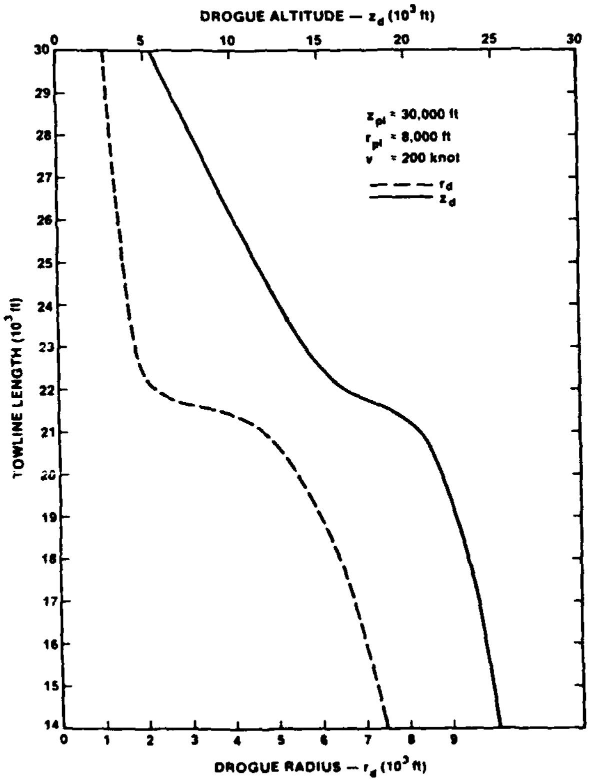


**Figure 4.** Towline length vs. drogue starting conditions.

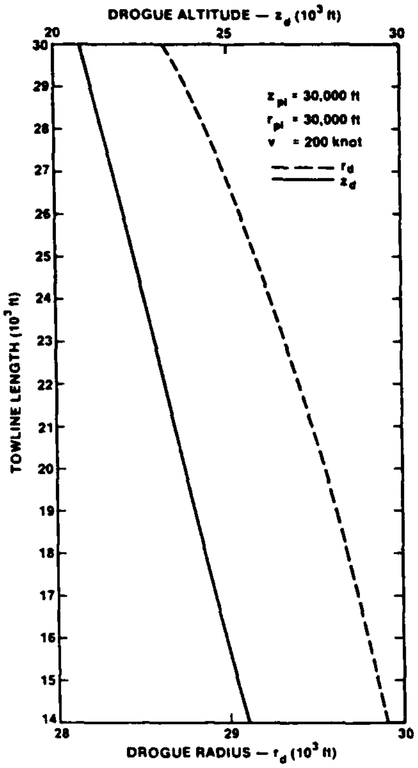


**Figure 5.** Towline length vs. drogue starting conditions.

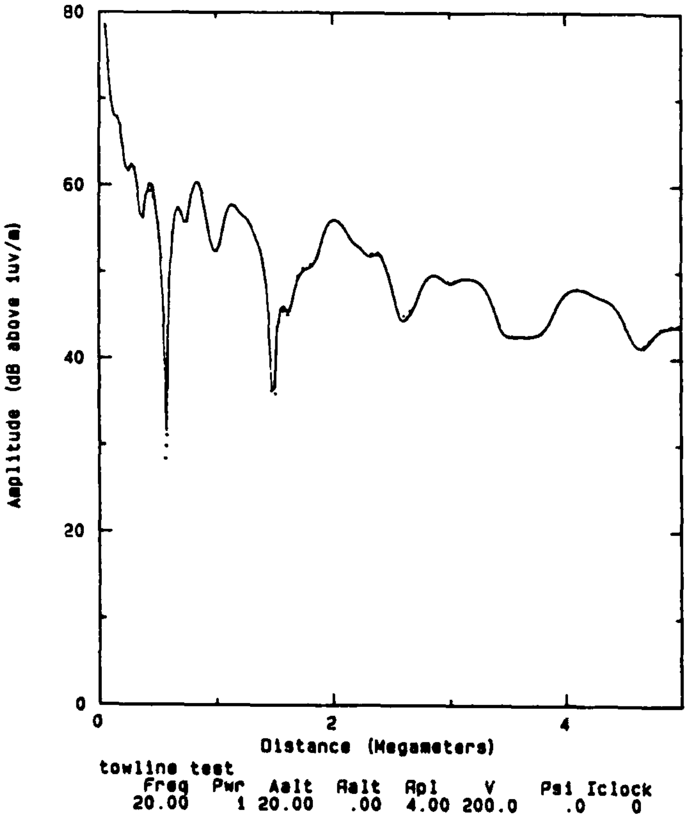


**Figure 6.** EZ field strength comparisons between "TWIRE" ( ——— ) and point dipole (· · · ·) results for an azimuth of 90°. Point dipole orientation and altitude given by $\gamma$ = 17.84°, $\phi$ = -58.69° and z = 12,945 ft.

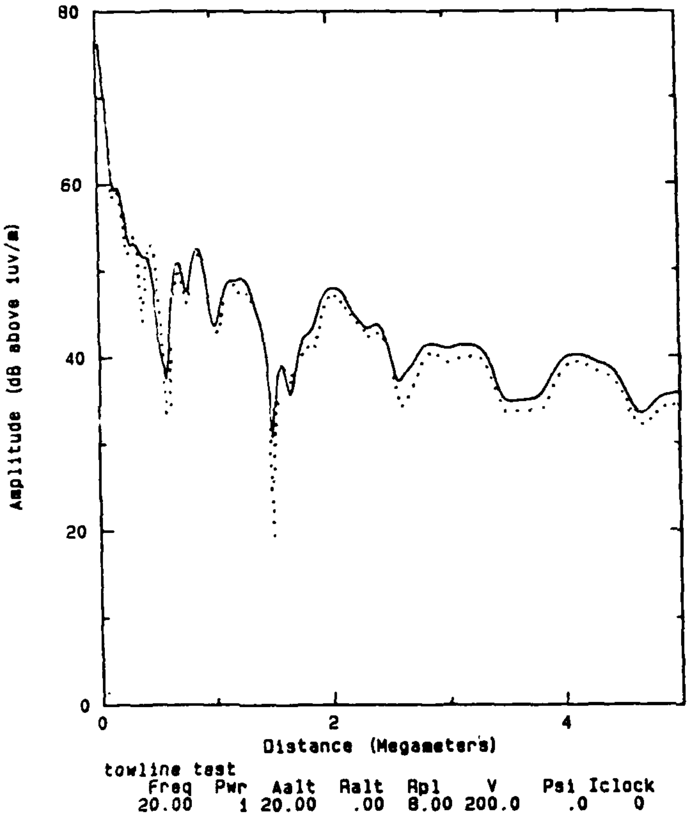


**Figure 7.** EZ field strength comparisons between "TWIRE" ( ——— ) and point dipole (· · · ·) results for an azimuth of 90°. Point dipole orientation and altitude given by $\gamma$ = 70.33°, $\phi$ = 7.23° and z = 17,250 ft.

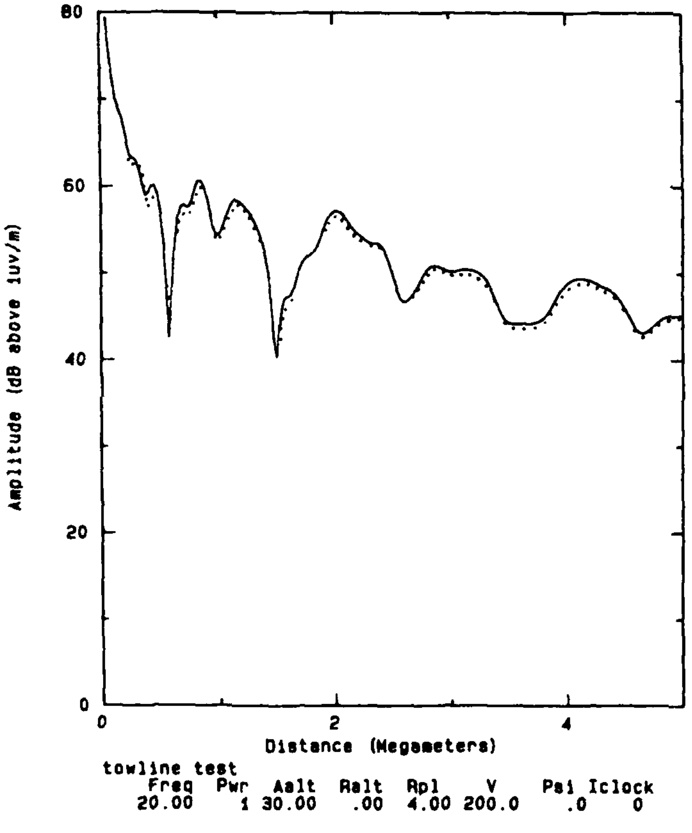


**Figure 8.** EZ field strength comparisons between "TWIRE" ( ——— ) and point dipole (· · · ·) results for an azimuth of 90°. Point dipole orientation and altitude given by $\gamma$ = 19.00°, $\phi$ = -47.18° and z = 22.627 ft.

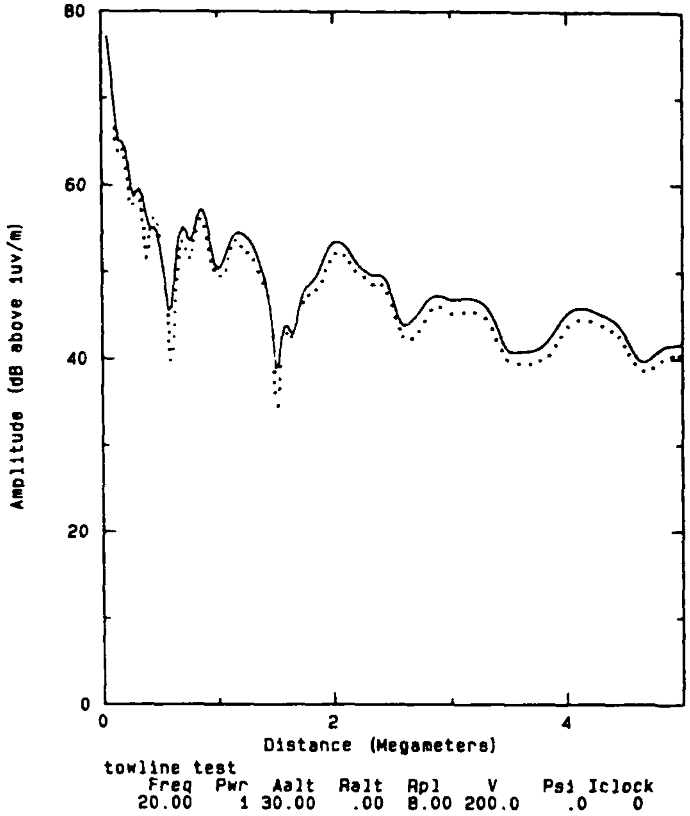


**Figure 9.** EZ field strength comparisons between "TWIRE" ( ——— ) and point dipole (· · · ·) results for an azimuth of 90°. Point dipole orientation and altitude given by $\gamma$ = 55.89°, $\phi$ = -3.63° and z = 26,037 ft.

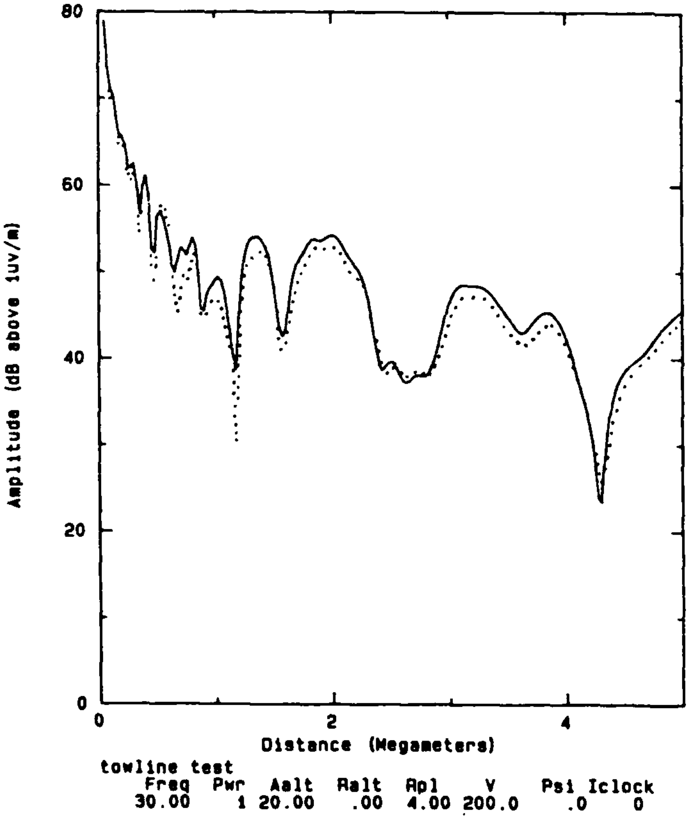


**Figure 10.** EZ field strength comparisons between "TWIRE" ( ——— ) and point dipole (· · · ·) results for an azimuth of 90°. Point dipole orientation and altitude given by $\gamma$ = 38.38°, $\phi$ = -29.96° and z = 16,757 ft.

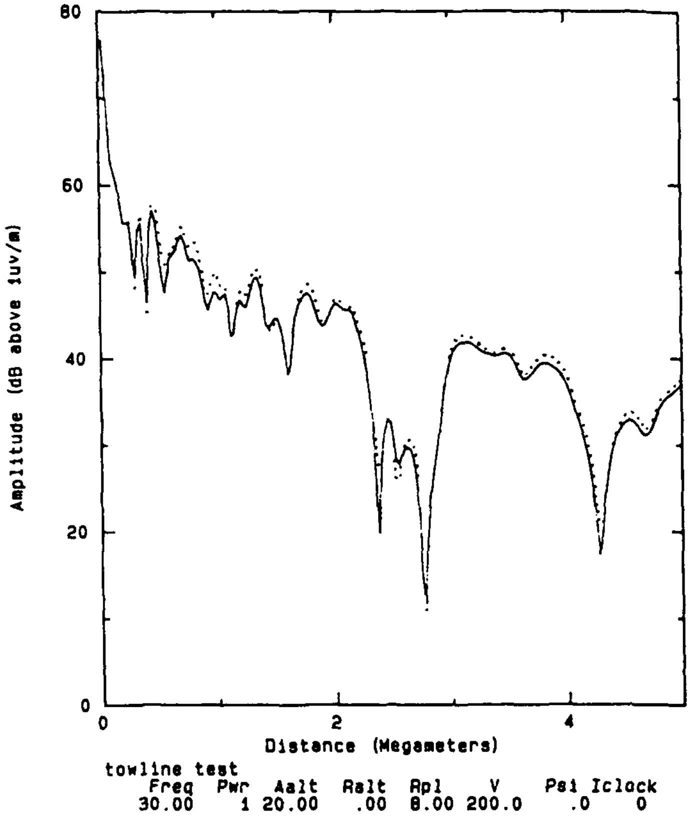


**Figure 11.** EZ field strength comparisons between "TWIRE" ( ——— ) and point dipole (· · · ·) results for an azimuth of 90°. Point dipole orientation and altitude given by $\gamma$ = 73.08°, $\phi$ = 40.75° and z = 18,287 ft.

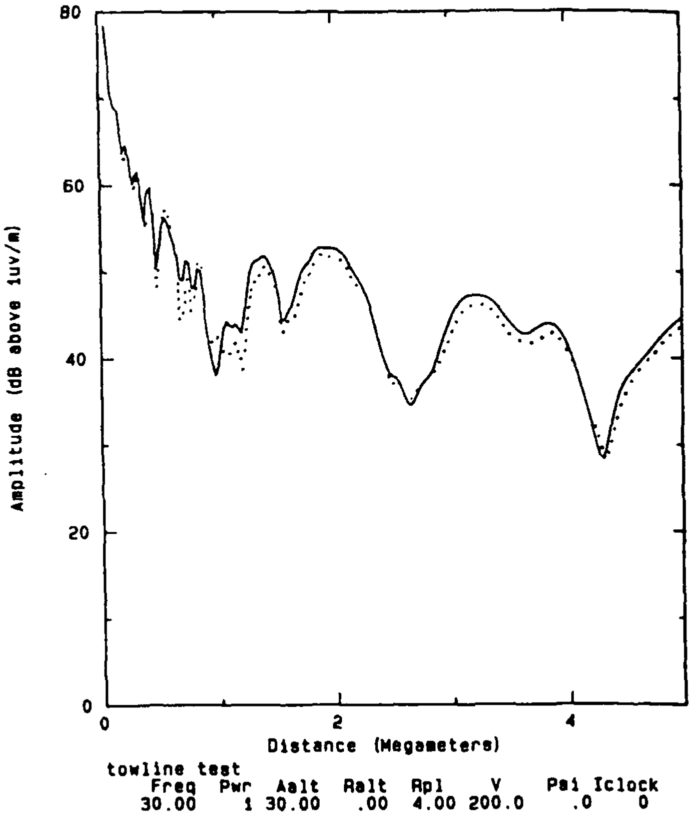
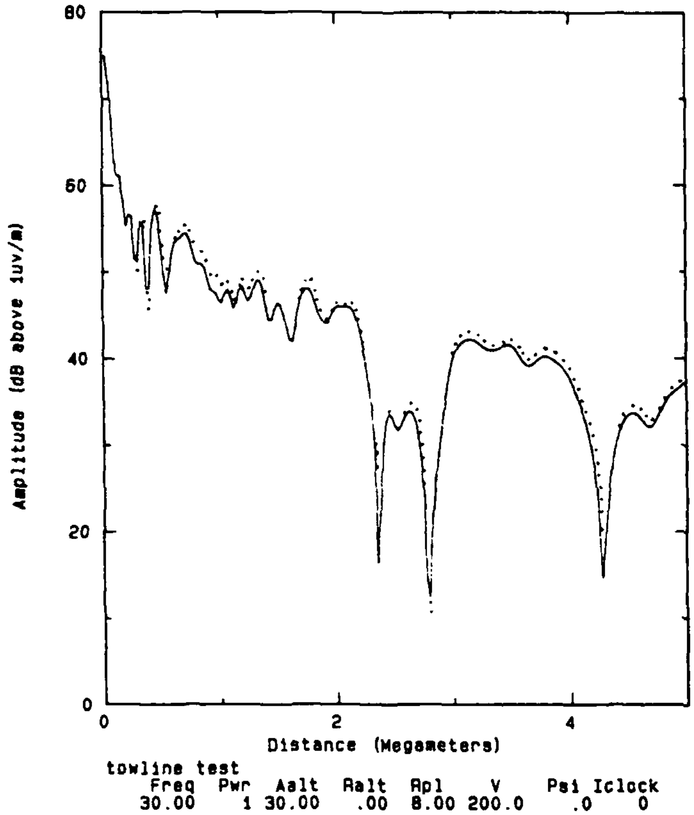
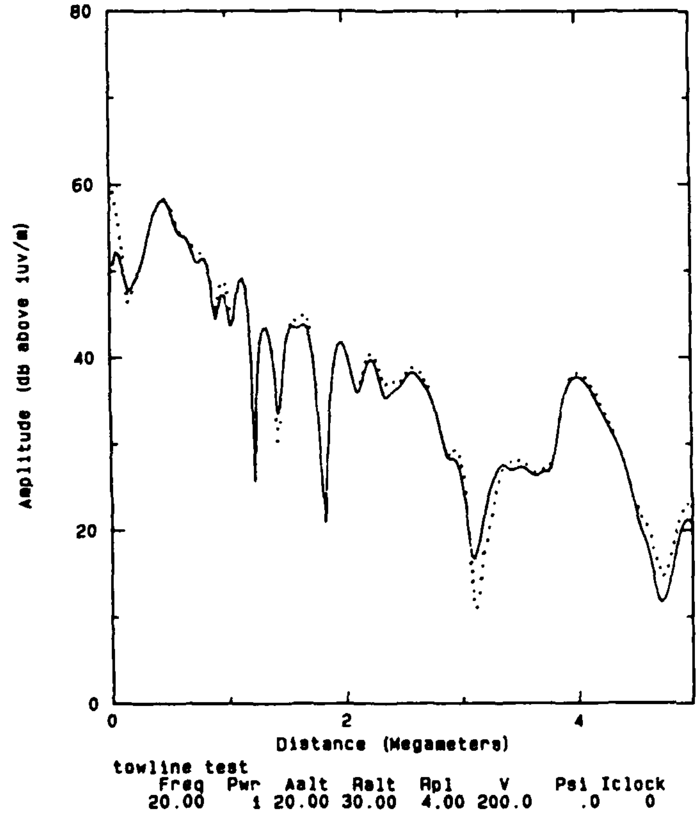


**Figure 12.** EZ field strength comparisons between "TWIRE" ( ——— ) and point dipole (· · · ·) results for an azimuth of 90°. Point dipole orientation and altitude given by $\gamma$ = 36.40°, $\phi$ = -21.89° and z = 26,294 ft.



```
towline test
     Freq  Pwr  Aalt   Ralt  Rpl     V    Psi Iclock
    30.00    1 30.00    .00  8.00 200.0    .0    0
```

**Figure 13.** EZ field strength comparisons between "TWIRE" ( ——— ) and point dipole (· · · ·) results for an azimuth of 90°. Point dipole orientation and altitude given by $\gamma$ = 69.96°, $\phi$ = 42.40° and z = 27,959 ft.



```
towline test
     Freq  Pwr  Aalt  Ralt   Rpl     V    Psi Iclock
    20.00    1 20.00 30.00  4.00 200.0    .0    0
```

**Figure 14.** EY field strength comparisons between "TWIRE" ( ——— ) and point dipole (· · · ·) results for an azimuth of 90°. Point dipole orientation and altitude given by $\gamma$ = 17.84°, $\phi$ = -58.69° and z = 12,495 ft.

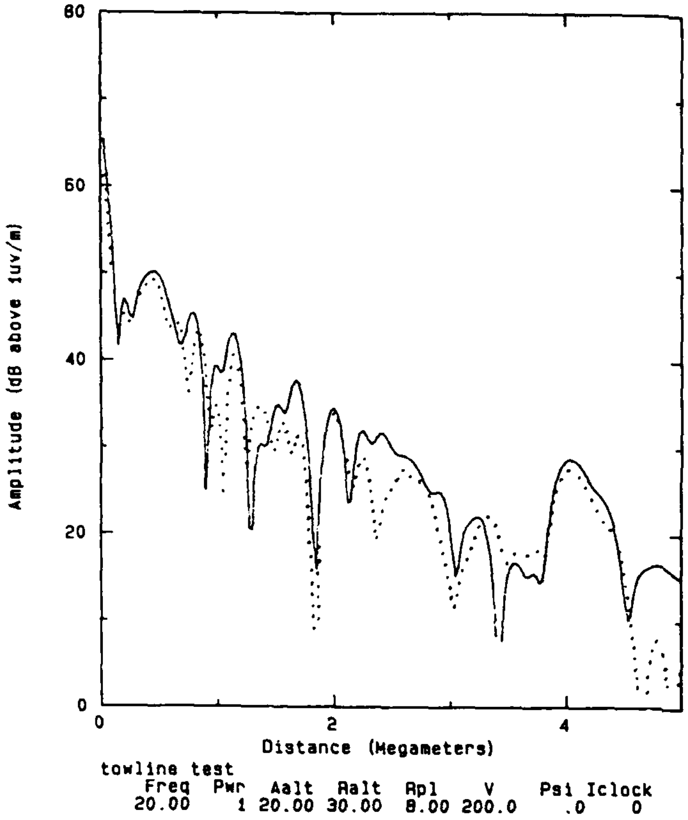


**Figure 15.** EY field strength comparisons between "TWIRE" ( ——— ) and point dipole (· · · ·) results for an azimuth of 90°. Point dipole orientation and altitude given by $\gamma$ = 70.33°, $\phi$ = 7.23° and z = 17,250 ft.

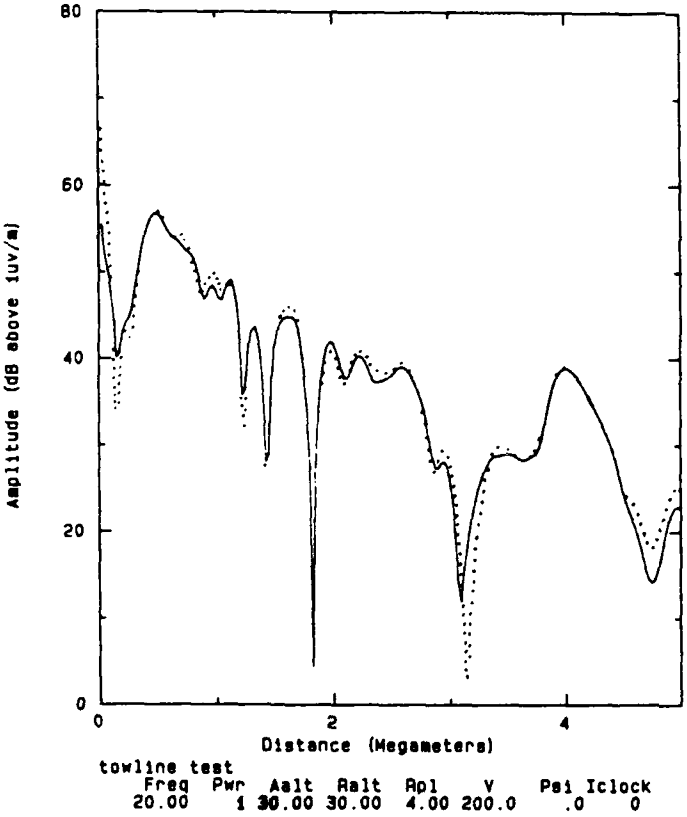


**Figure 16.** EY field strength comparisons between "TWIRE" ( ——— ) and point dipole (· · · ·) results for an azimuth of 90°. Point dipole orientation and altitude given by $\gamma$ = 19.00°, $\phi$ = -47.18° and z = 22.627 ft.

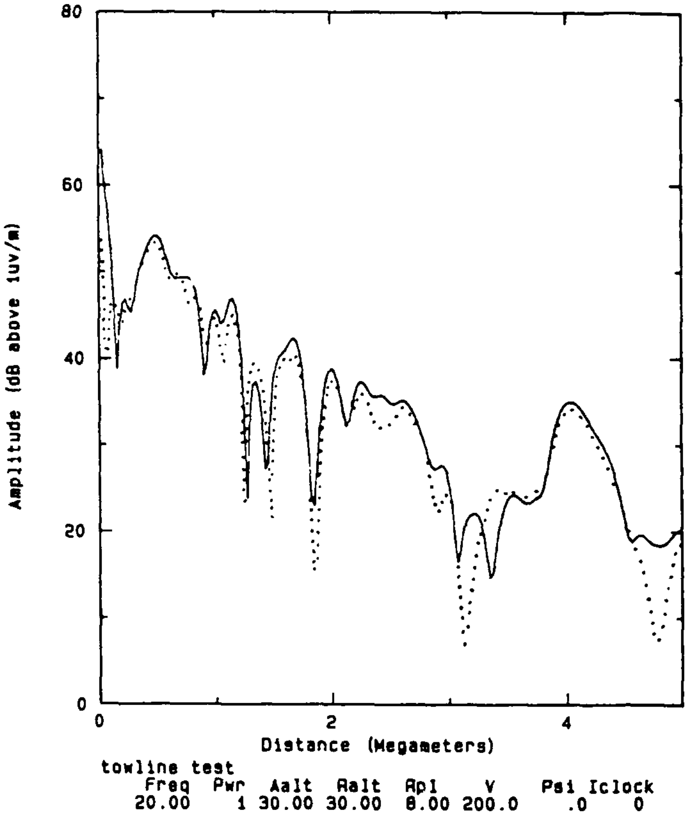


**Figure 17.** EY field strength comparisons between "TWIRE" ( ——— ) and point dipole (· · · ·) results for an azimuth of 90°. Point dipole orientation and altitude given by $\gamma$ = 55.89°, $\phi$ = -3.63° and z = 26,037 ft.

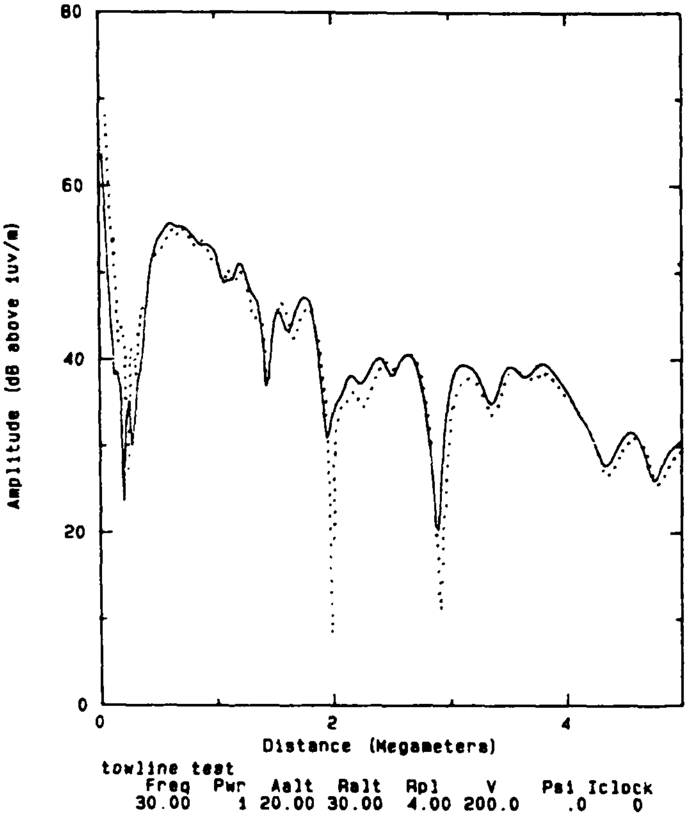


**Figure 18.** EY field strength comparisons between "TWIRE" ( ——— ) and point dipole (· · · ·) results for an azimuth of 90°. Point dipole orientation and altitude given by $\gamma$ = 38.38°, $\phi$ = -29.96° and z = 16,757 ft.

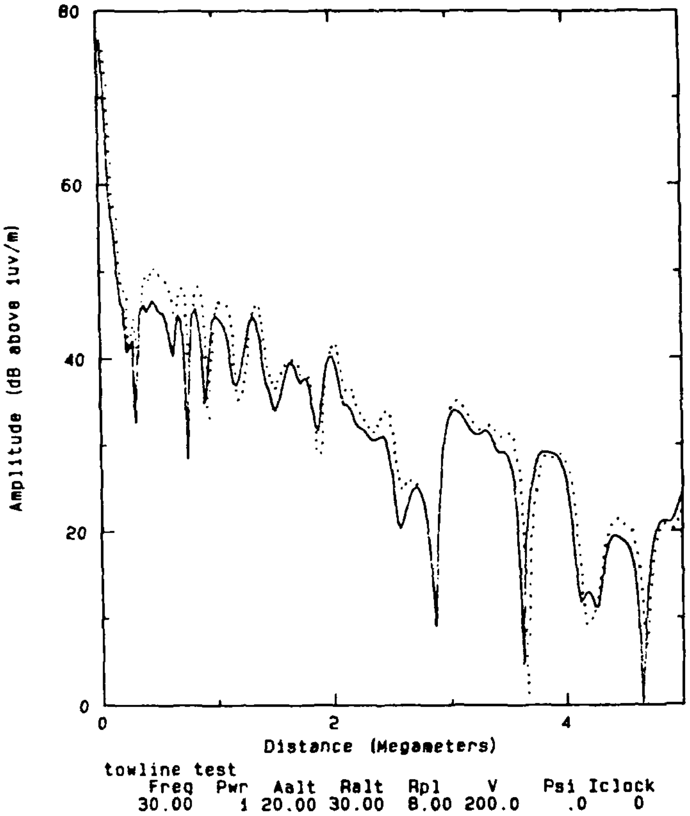


**Figure 19.** EY field strength comparisons between "TWIRE" ( ——— ) and point dipole (· · · ·) results for an azimuth of 90°. Point dipole orientation and altitude given by $\gamma$ = 73.08°, $\phi$ = 40.75° and z = 18,287 ft.

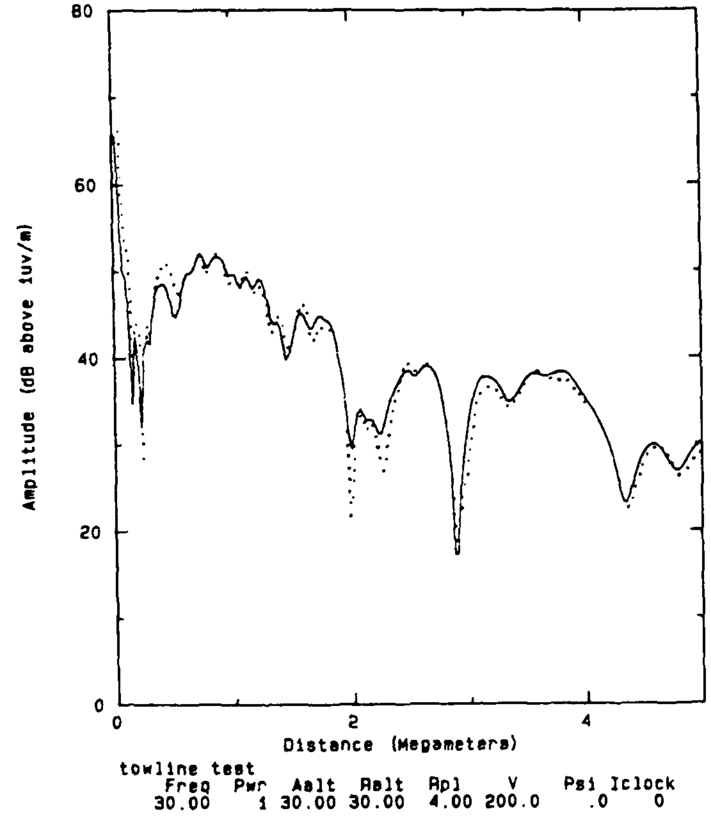


**Figure 20.** EY field strength comparisons between "TWIRE" ( ——— ) and point dipole (· · · ·) results for an azimuth of 90°. Point dipole orientation and altitude given by $\gamma$ = 36.40°, $\phi$ = -21.89° and z = 26,294 ft.

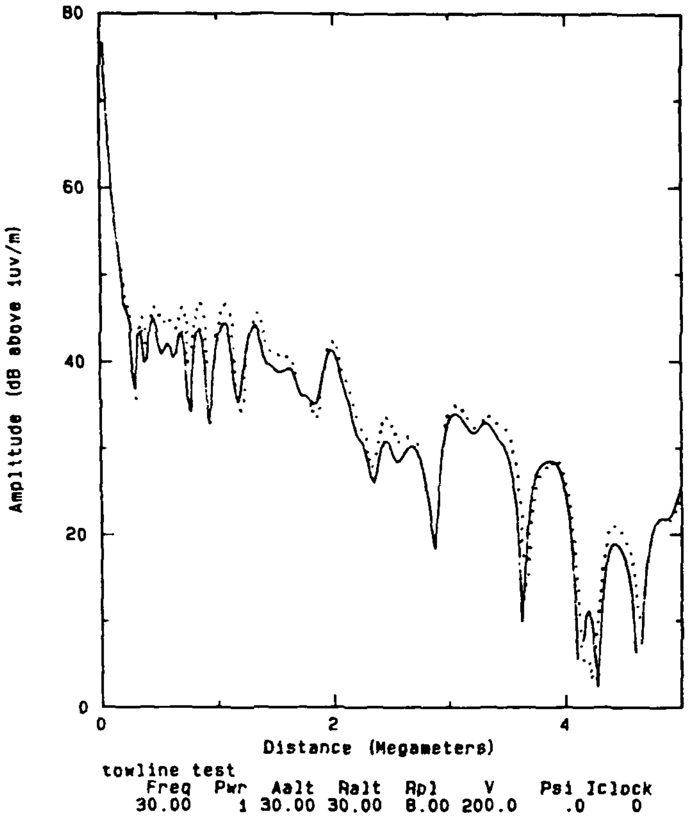


**Figure 21.** EY field strength comparisons between "TWIRE" ( ——— ) and point dipole (· · · ·) results for an azimuth of 90°. Point dipole orientation and altitude given by $\gamma$ = 69.96°, $\phi$ = 42.40° and z = 27,959 ft.

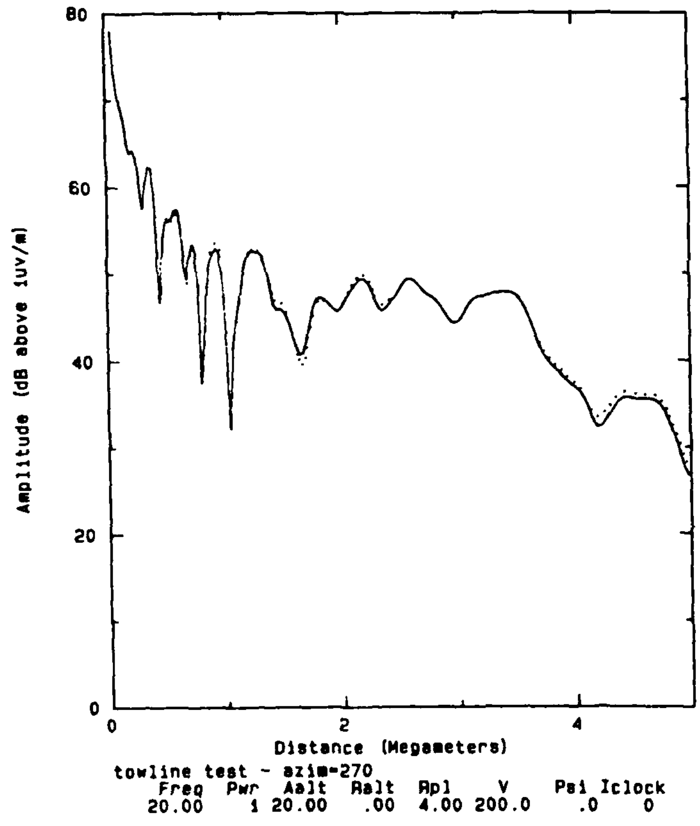


**Figure 22.** EZ field strength comparisons between "TWIRE" ( ——— ) and point dipole (· · · ·) results for an azimuth of 270°. Point dipole orientation and altitude given by $\gamma$ = 17.84°, $\phi$ = -58.69° and z = 12,945 ft.

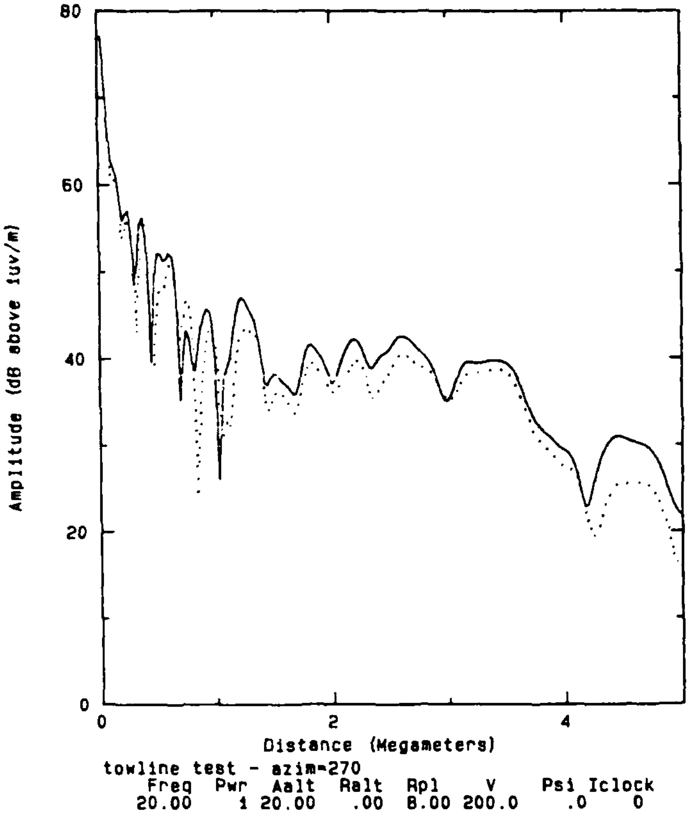


**Figure 23.** EZ field strength comparisons between "TWIRE" ( ——— ) and point dipole (· · · ·) results for an azimuth of 270°. Point dipole orientation and altitude given by $\gamma$ = 70.33°, $\phi$ = 7.23° and z = 17,250 ft.

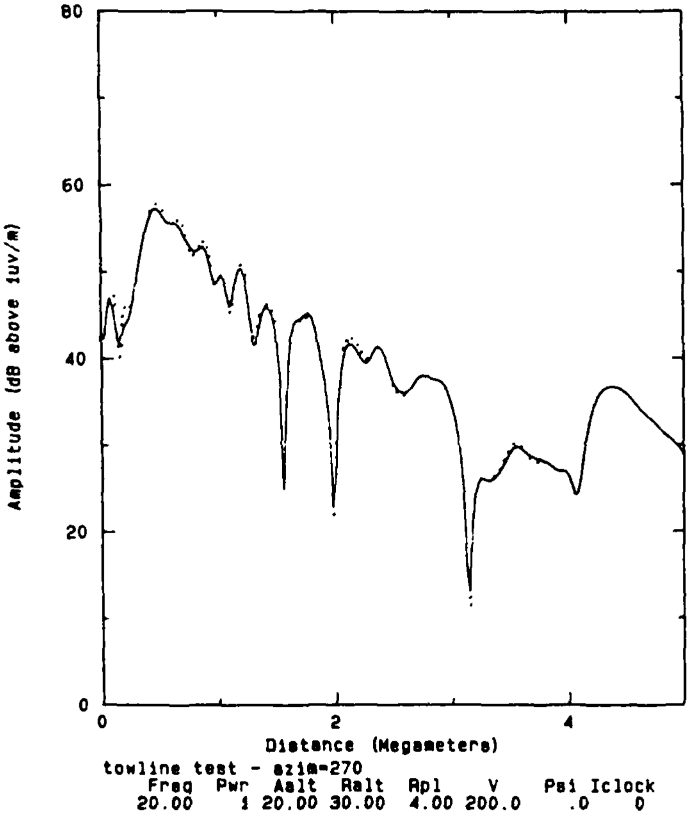


**Figure 24.** EY field strength comparisons between "TWIRE" ( ——— ) and point dipole (· · · ·) results for an azimuth of 270°. Point dipole orientation and altitude given by $\gamma$ = 17.84°, $\phi$ = -58.69° and z = 12,945 ft.

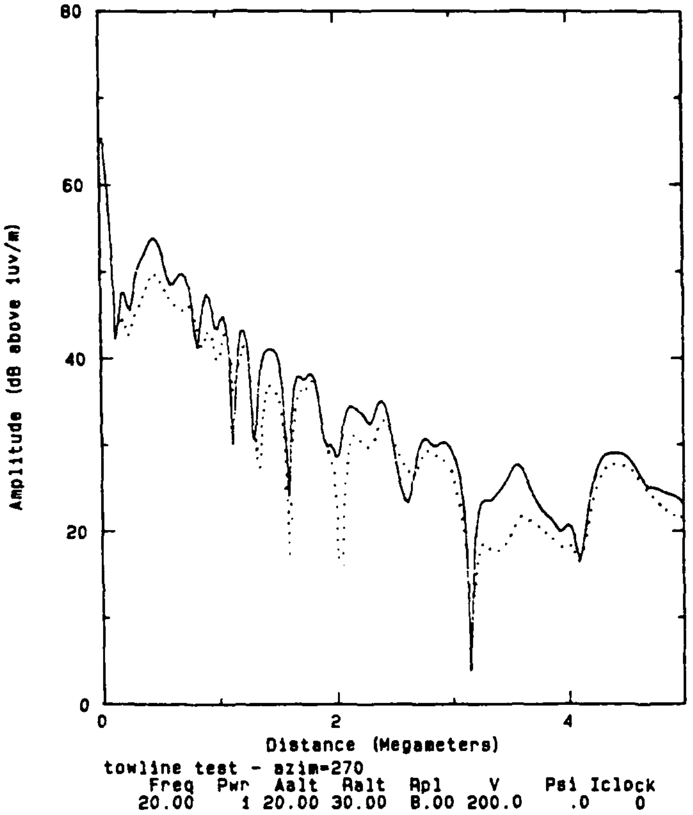


**Figure 25.** EY field strength comparisons between "TWIRE" ( ——— ) and point dipole (· · · ·) results for an azimuth of 270°. Point dipole orientation and altitude given by $\gamma$ = 70.33°, $\phi$ = 7.23° and z = 17,250 ft.

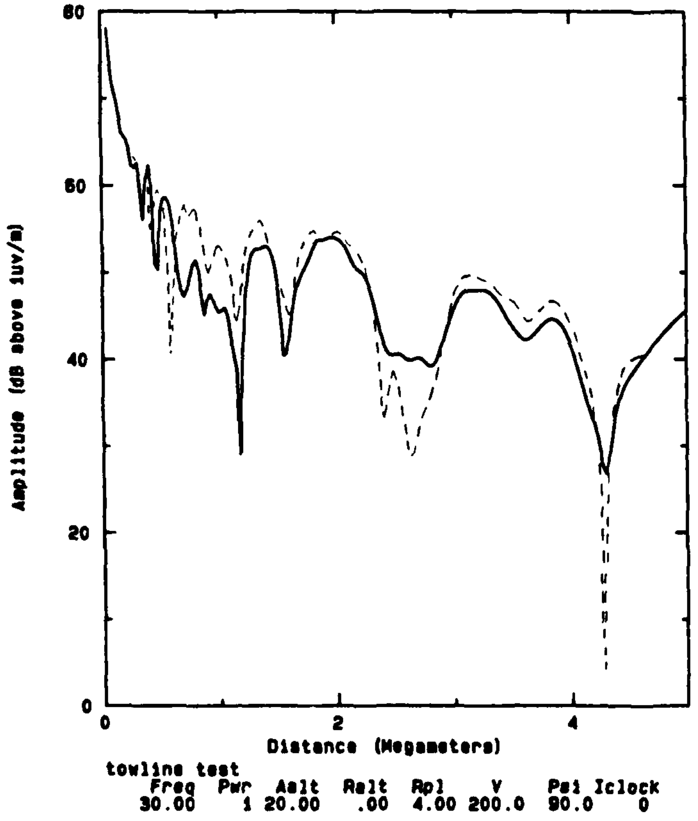


**Figure 26.** EZ field strengths calculated using "TWIRE" for azimuths of 90° ( ——— ) and 270° (- - - -).

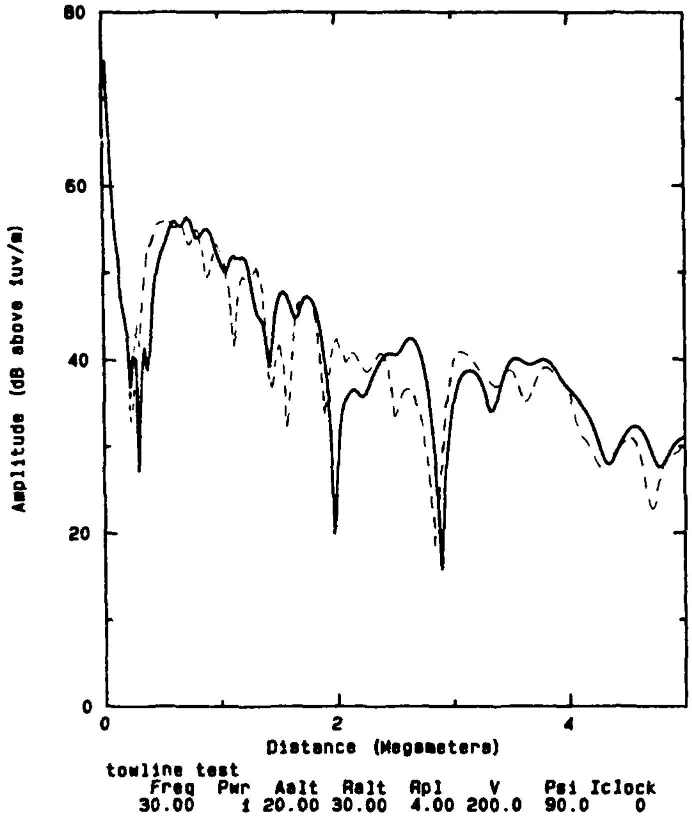


**Figure 27.** EY field strengths calculated using "TWIRE" for azimuths of 90° ( ——— ) and 270°(- - - -).

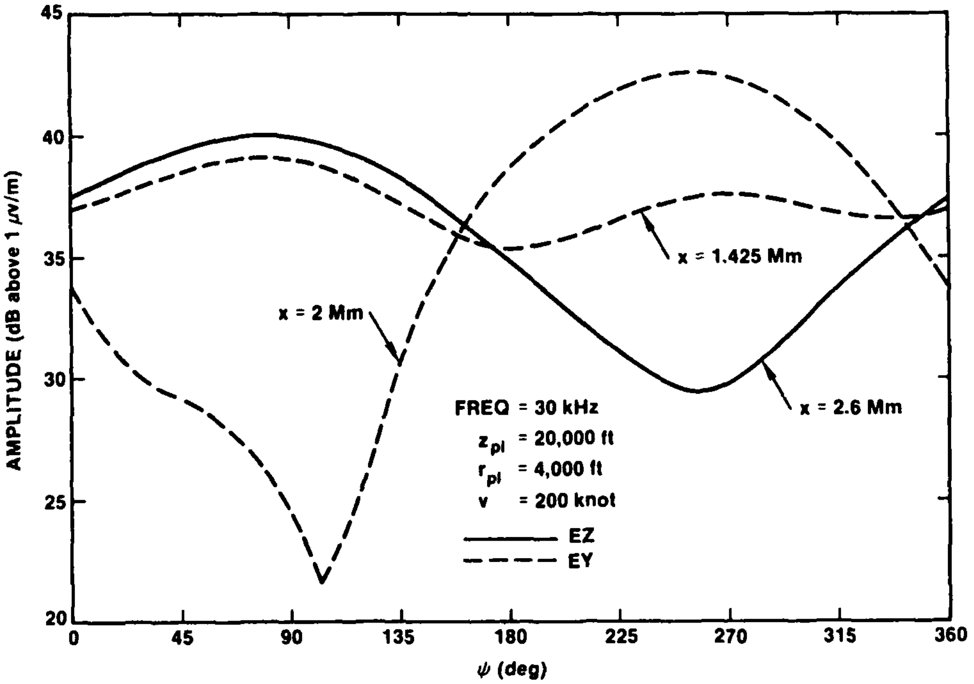


**Figure 28.** Rotational dependence of EZ and EY field strengths calculated using "TWIRE".

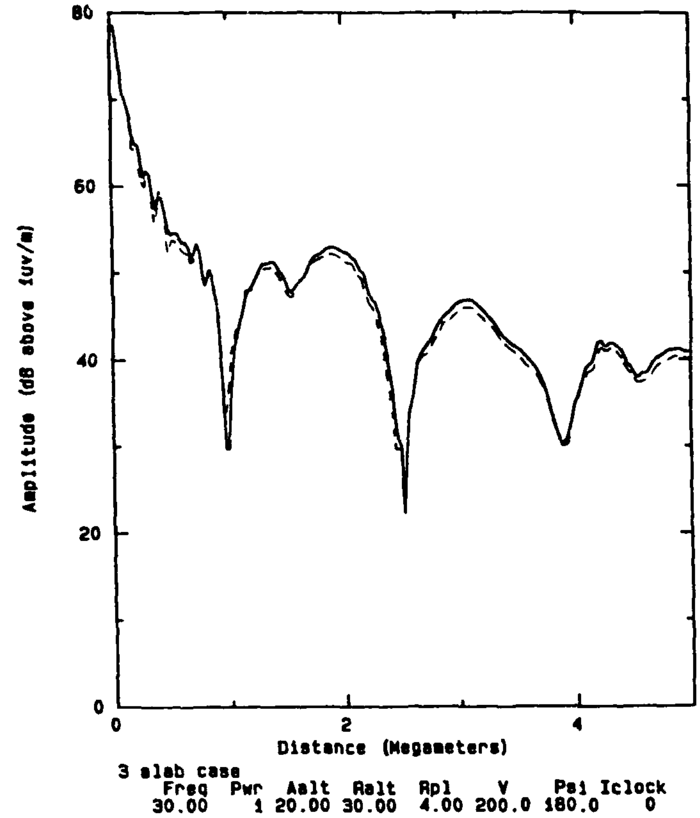


**Figure 29.** EZ field strength comparisons between "TWIRE" ( ——— ) and point dipole (- - - -) results for a laterally inhomogeneous guide.

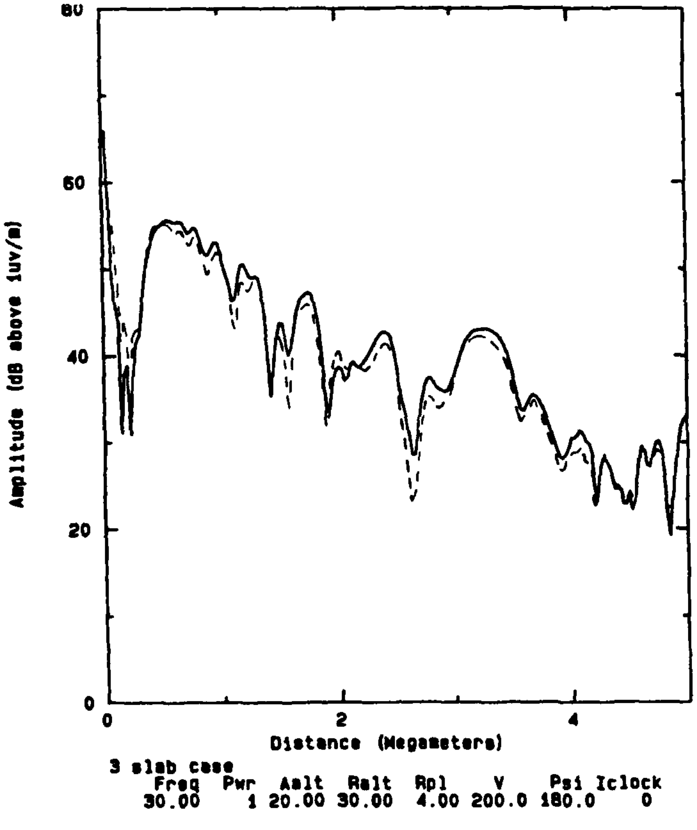


**Figure 30.** EY field strength comparisons between "TWIRE" ( ——— ) and point dipole (- - - -) results for a laterally inhomogeneous guide.

## REFERENCES

1. Bickel, J.E., D.G. Morfitt, I.J. Rothmuller and W.F. Moler, "Propagation Analysis of Diversity for VLF Communication Systems," Naval Electronics Laboratory Center Technical Document No. 139, September 1971.
2. Burke, G.J. and A.J. Poggio, "Numerical Electromagnetic Code (NEC) -Method of Moments," Naval Ocean Systems Center Technical Document 116, prepared by Lawrence Livermore Laboratory for the Naval Ocean Systems Center and Air Force Weapons Laboratory, July 1977.
3. Ferguson, J.A. and F.P. Snyder, "Approximate VLF/LF Waveguide Mode Conversion Model: Computer applications: FASTMC and BUMP," Naval Ocean Systems Center Technical Document 400, November 1980.
4. Fern, D.G., "Electrical and Dynamical Analysis of the TACAMO Dual Trailing Wire Antenna System," Naval Ocean Systems Center Technical Note 1477, December 1986.
5. Huang, S.L., "Mathematical Model for Long Cable Towed by Orbiting Aircraft," Naval Air Development Center Report No. NADC-AM-6849, June 1969.
6. Morfitt, D.G. and C.H. Shellman, "MODESRCH, An Improved Computer Program for Obtaining ELF/VLF/LF Mode Constants in an Earth-Ionosphere Waveguide," Naval Electronics Laboratory Center Interim Report No. 77T, Prepared for Defense Nuclear Agency, October 1976.
7. NAVAIRDEVCEN Report, "Characteristics of Long Towlines in Non-Maneuvering and Orbiting Flight," No. NADC-ED-6340, June 1963.
8. Pappert, R.A., "Radiation Resistance of Thin Antennas of Arbitrary Elevation and Configuration Over Perfectly Conducting Ground," Naval Ocean Systems Center Technical Report No. 1112, June 1986.
9. Pappert, R.A. and J.A. Ferguson, "VLF/LF Mode Conversion Model Calculations for Air To Air Transmissions in the Earth Ionosphere Waveguide," Radio Sci, Vol 21 (4), pp 551-558, 1986.
10. Pappert, R.A. and L.R. Hitney, "Empirical Modeling of Nighttime Easterly and Westerly VLF Propagation in the Earth-Ionosphere Waveguide." Radio Sci., Vol 23 (4), pp 599-611, July 1988.
11. Pappert, R.A. and D.G. Morfitt, "Theoretical and Experimental Sunrise Mode Conversion Results at VLF," Radio Sci., Vol. 10 (5), pp 537-546, 1975.
12. Pappert, R.A. and L.R. Shockey, "WKB Mode Summing Program for VLF/ELF Antennas of Arbitrary Length, Shape and Elevation," Naval Electronics Laboratory Center Interim Report No. 713, Prepared for Defense Atomic Support Agency, June 1971.
13. Rockway, J.W. and J.C. Logan, "The new MININEC (Version III): A Mini-Numerical Electromagnetic Code," Naval Ocean Systems Center Technical Document 938, September 1986.
14. Wait, J.R. and K.P. Spies, "Characteristics of the Earth-Ionosphere Waveguide for VLF Radio Waves," National Bureau of Standards Technical Note 300, Gaithersburg, MD, 1964.

## APPENDIX A—PROGRAM LISTING OF “TWIRE”

```
c     twire: a program assembled from fastmc(Ferguson and Snyder-NOSC TD
c     400,Nov. 1980),from a towline configuration program(Huang-NADC AM
c     6849,June 1969) and from a radiation resistance program(Pappert-
c     NOSC TR 1112,June 1986)
c
      parameter (maxmds=30,nstore=(maxmds*(maxmds+3))*2+4)
      parameter (maxsgs=50)
c
      implicit real*8 (a-h,o-z)
      common/mc inpt/xtra(3,maxmds),temp(maxmds),freq,omega,waveno,mik,
     $       alpha,ak,ak13,ka13,ka23,ngsq,nthsq,sum0,aconst,econst
      common/mc stor/a(maxmds,maxmds),fofr(maxmds),s(maxmds),tp(maxmds),
     $       xval,sigma,epsr,nrmode,nrslab
      common/mc plot/dmin,dmax,dtic,ramin,ramax,ratic,
     $       ampmin,ampmax,amptic,phsmin,phsmax,phstic,
     $       plotid,bumpid,iopt,frq,power,rcomp,
     $       incl,theta,dst,
     $       talt,al,aalt,ralt,ident,nrpts,sizex,sizey,cdate(3),
     $       ctime(2),nrcrvs,nrplts,nrcurv,naplot,npplot,npdiff,nprint
      common sumout(501),xy(501),amp(501),phs(501),
     $       hgt(3,maxmds),hgr(3,maxmds)
      dimension store(nstore),dist(20),idummy(8)
      complex *16 solsav
      common/com01/antlen(maxsgs),antrad(maxsgs),antalt(maxsgs),
     $             antang(maxsgs),npts
      common/com08/solsav(maxmds,maxsgs)
      equivalence (store,a)
      integer rcomp,totape,ctime,cdate
      character* 4 label,ident
      character*40 fname,profil,blank
      character*80 bcd,plotid,bumpid
      real*4 xy,amp,phs,dmin,dmax,dtic,ramin,ramax,ratic,dist,dst,
     $       ampmin,ampmax,amptic,phsmin,phsmax,phstic,
     $       sizex,sizey,frq,al,talt,aalt,ralt,incl,theta,
     $       tlong,tlat,rbear,power
      real*8 ka13,ka23,nthsq
      complex* 8 sumout
      complex*16 tp,a,s,fofr,xtra,temp,hgt,hgr,mi,
     $           mik,ngsq,tmp1,tmp2,tmp3,tmp4,ratio(4),xtra0(maxmds)
c
c rcomp                   received component, =1 is z, =2 is y
c power                   radiated power in kw
c talt                    platform altitude in km
c al                      antenna length as percentage of wavelength
c incl,theta              orientation of transmitter antenna
c topht                   height of the bottom of the ionosphere in km
c naplot                  amplitude plot flag
c ampmin,ampmax,ampinc    amplitude plot range and tic interval in db
c npplot                  phase plot flag
c phsmin,phsmax,phsinc    phase plot range and tic interval in degrees
c sizex,sizey             axis lengths in inches
c nrcurv                  number of curves per graph, max of 4
c nprint                  printout flag
c       =0                none
c       =1                amp and phs only
```

```
c        -2                   rfacmset cards and mode constants
c        -3                   conversion coefficients and relative excitation
c        -4                   slab integrals
c
c iopt=1                      fields vs distance
c
c   ralt                        receiver altitude in km
c   npdiff                      phase difference plot flag
c   nrptsl                      number of points to output, max of 501
c   dmin,dmax,dtic              distance range and tic interval in mm
c   totape                      output flag, =1 writes to logical unit 2
c   tlong,tlat                  transmitter location in degrees west,north
c   rbear                       path bearing in degrees east of north
c
c   note:  tlong,tlat,rbear are for totape=1 output only
c
c iopt=2                      fields vs altitude
c
c   ramin,ramax,ratic           receiver altitude range in km
c   dist,nrd                    receiver distances in mm, number of distances
c
c
      namelist/datum/iopt,rcomp,al,incl,theta,talt,ralt,topht,
     $         naplot,npplot,npdiff,nprint,nrcurv,sizex,sizey,
     $         ampmax,ampmin,amptic,phsmax,phsmin,phstic,
     $         dmax,dmin,dtic,tlong,tlat,rbear,power,totape,nrptsl,
     $         ramin,ramax,ratic,dist,nrd,nrpts2
c
      data label/'zyx '/,nflag/0/,idummy/8*0/,
     $     epsln0/8.85434d-12/,dtr/.01745329252d0/,
     $     topht/90.d0/,
     $     tlong,tlat,rbear/0.,0.,0./,totape/0/,
     $     dist/20*0./,nrd/0/,
     $     nrptsl/501/,nrpts2/51/,
     $     blank/'                                        '/
      data mi/(0.d0,-1.d0)/
c
c     unit        use
c     1           primary input
c     2           totape=1 output
c     3           temporary storage of mc data
c     4           alternate input of data for a single radial
c
c
      bumpid(1:4)='    '
c
c     print 2000
      read 2001,fname
      lunitl=1
      open(unit=1,file=fname,status='old')
      open(unit=3,status='scratch',form='unformatted')

c     get date and time for plot identification
c     call date(cdate)
c     call time(ctime)
```

```
c       print 1004,cdate,ctime
c
c      get control card
10     continue
c      print 2002
       read(lunit1,1002,end=998) bcd
       print 1003,bcd
       lunit4=lunit1
c
       if(bcd(1:4) .eq. 'name' .or. bcd(1:4) .eq. 'NAME') then
c        print 2003
c        read namelist
         read(lunit1,datum)
         print datum
         idummy(3)=int(a1+.5)
         if(naplot+npplot+npdiff .eq. 3) then
           print *,'NAPLOT,NPPLOT,NPDIFF can not all be 1'
           go to 10
         endif
         if(phsmin .eq. 0. .or. phsmax .eq. 0.) then
           print *,'PHSMIN or PHSMAX is zero'
           go to 10
         end if
         if(nrcurv .gt. 4) then
           print *,'NRCURV .gt. 4'
           go to 10
         end if
         if(sizey .gt. 9.) then
           print *,'SIZEY .gt. 9'
           go to 10
         end if
         if(iopt .eq. 1) then
           if(nrpts1 .gt. 501) then
             print *,'NRPTS1 .gt. 501'
             go to 10
           end if
           nrpts=nrpts1
         else
           nrpts=nrpts2
         end if
         go to 10
       end if
c
       if(bcd(1:4) .eq. 'bump' .or. bcd(1:4) .eq. 'BUMP') then
c        store bump id card
         bumpid=bcd
         go to 10
       end if
c
       if(bcd(1:4) .eq. 'outp' .or. bcd(1:4) .eq. 'OUTP') then
         j=7
         do while (bcd(j:j) .eq. ' ')
           if(j .eq. 80) then
             print *,'Output file name not found'
             go to 10
```

```
          end if
          j=j+1
        end do
        fname=bcd(j:)
        open(unit=2,file=fname,status='new',form='unformatted')
        go to 10
      end if
c
      if(bcd(1:4) .eq. 'inpu' .or. bcd(1:4) .eq. 'INPU') then
        j=7
        do while (bcd(j:j) .eq. ' ')
          if(j .eq. 80) then
            print *,'Input file name not found'
            go to 10
          end if
          j=j+1
        end do
        fname=bcd(j:)
        open(unit=4,file=fname,status='old')
        lunit4=4
        read(lunit4,1002) bcd
        print 1003,bcd
        go to 19
      end if
c
      if(bcd(1:4) .eq. 'clos' .or. bcd(1:4) .eq. 'CLOS') then
        if(bcd(7:7) .eq. 'o' .or. bcd(7:7) .eq. 'O') then
          close(unit=2)
        else
          close(unit=4)
          lunit4=lunit1
        end if
        go to 10
      end if
c
19    if(bcd(1:4) .eq. 'sw  ' .or. bcd(1:4) .eq. 'SW  ') then
c       get xmtr and profile parameters
        read(bcd,1018) tlong,tlat,rbear,profil
        go to 20
      end if
c
      if(bcd(1:4) .eq. 'ipsq' .or. bcd(1:4) .eq. 'IPSQ') then
c       get xmtr and profile parameters
        read(bcd,1018) tlong,tlat,rbear,profil
        go to 20
      end if
c
      if(bcd(1:4) .eq. 'data' .or. bcd(1:4) .eq. 'DATA') then
        profil=blank
        rbear=720.
        go to 20
      end if
c
      if(bcd(1:4) .eq. 'star' .or. bcd(1:4) .eq. 'STAR') go to 30
c
```

```
      if(bcd(1:4) .eq. 'quit' .or. bcd(1:4) .eq. 'QUIT') go to 998
c
        print *,'Error in control card'
        go to 10
c
c     read propagation path data
20    read(lunit4,1002) plotid
      if(rbear .ne. 720.) then
        j=1
        do while (plotid(j:j+12) .ne. '             ')
          if(j .eq. 67) then
            print *,'Plot identification is too long to append bearing'
            go to 22
          end if
          j=j+1
        end do
        write(plotid(j:j+12),1021) rbear
      end if
22    print 1003,plotid
      nrmode=maxmds
      nrslab=0
      rho=-1.
      th=0.
      nthsq=1.+alpha*topht
      rewind 3
23    read(lunit4,1002) bcd
      read(bcd,1020) rr,ff,aa,cc,bb,ss,ee
      if(rr .ne. 40. .and. ss .eq. 0.) go to 23
      xval=rr*1000.
      if(nrslab .gt. 0) write(3) store
c
      if(rr .lt. 40.) then
        nrslab=nrslab+1
        nm=0
        bb=bb*10000.
        if(bcd(71:71) .eq. 't' .or. bcd(71:71) .eq. 'T')
     $    read(bcd,1028) th
        if(nprint .ge. 1) print 1022,nrslab,rr,ff,aa,cc,bb,ss,ee,th
        if(nprint .ge. 4) print 1024
        if(rr .eq. 0.) then
c         if(nprint .le. 1) print 1022,nrslab,rr,ff,aa,cc,bb,ss,ee,th
          frq=ff
          freq=ff
          const=682.2408*dsqrt(freq)
          omega=6.283185308d3*freq
          waveno=20.958445d-3*freq
          mik=dcmplx(0.d0,-waveno)
          aconst=-8.686d3*waveno
          econst=20.*dlog10(35.*waveno)
          ak=alpha/waveno
          ak13=dexp(dlog(ak)/3.d0)
          ka13=1.d0/ak13
          ka23=ka13**2
        end if
        if(rho .gt. rr) then
```

```
          print *,'Rhos are out of order'
          go to 10
        end if
        rho=rr
        sigma=ss
        epsr=ee
        ngsq=dcmplx(epsr,-sigma/(omega*epsln0))
        if(th .gt. 0.) nthsq=1.+alpha*th
24      read(lunit4,1023) indx1,tr1,ti1,itrm1,tmp1,tmp2
c
        if(indx1 .gt. 0) then
          read(lunit4,1023) indx2,tr2,ti2,itrm2,tmp3,tmp4
          if(nm .eq. maxmds) go to 24
          if(zabs(tmp1) .eq. 0.) go to 24
          if(tr1 .ne. tr2 .or. ti1 .ne. ti2 .or. itrm1 .ne. itrm2) then
            print *,'Modes are out of order'
            go to 10
          end if
          if(itrm1 .eq. 0) then
            print *,'Mode conversion flag is missing'
            go to 10
          end if
          nm=nm+1
          if(nprint .ge. 4) then
            print 1025,nm,indx1,tr1,ti1,itrm1,tmp1,tmp2,
     $                    indx2,tr2,ti2,itrm2,tmp3,tmp4
          end if
          tp(nm)=dcmplx(tr1,ti1)
          s(nm)=zsin(tp(nm)*dtr)
          ratio(2*indx1-1)=tmp1
          ratio(2*indx1  )=tmp2
          ratio(2*indx2-1)=tmp3
          ratio(2*indx2  )=tmp4
          temp(nm)=ratio(4)
c         get vertical excitation factor for printout
          xtra0(nm)=(-2.124292961d0,0.d0)*ratio(1)*s(nm)**2
c         get ey/hy
          if(itrm1 .eq. 1) then
            fofr(nm)=ratio(3)/ratio(1)
          else
            fofr(nm)=ratio(2)/(ratio(3)*ratio(4))
          end if
          if(nrslab .eq. 1) then
c           get hy excitation factor at the transmitter
            xtra(1,nm)=-ratio(1)*s(nm)
            xtra(2,nm)= ratio(3)*ratio(4)
            xtra(3,nm)= ratio(1)
          end if
          go to 24
        else
          if(nprint .gt. 1) print 1027,nm
          nrmode=nm
        end if
c
        if(nprint .gt. 1) then
```

```
c         print mode constants table
          print 1031
          do 26 m=1,nrmode
          sr=dble(s(m))
          si=dimag(s(m))
          atten=aconst*si
          voverc=1.d0/sr
          wr=dble(xtra0(m))
          wi=dimag(xtra0(m))
          wml=econst+10.d0*dlog10(dmax1(wr**2+wi**2,1.d-20))
          wal=datan2(wi,wr)-1.5707963267949d0
          print 1032,m,atten,voverc,wml,wal
26        continue
        end if
c
c       get conversion coefficients
        call mcstep
        go to 23
c
      else
c
        if(nrslab .le. 1) then
          print *,'Insufficient number of slabs, ',
     $             'horizontally homogeneous only'
        end if
      end if
c
c     calculate mode sum
30    ident=label(rcomp:rcomp)//'   '
      sum0=const*sqrt(power)
      call tacamo
      call coord
      call rpower
c
      if(iopt .eq. 1) then
c
        deltad=(dmax-dmin)/float(nrpts-1)
        dst=dmin
        do 31 i=1,nrpts
        xy(i)=dst
        call mcflds(i,dst,sumout(i),amp(i),phs(i))
31      dst=dst+deltad
c
        if(totape .gt. 0)
c         save mode sum
     $    write(2) sumout,frq,tlong,tlat,rbear,power,rcomp,incl,theta,
     $              talt,ralt,dmin,dmax,idummy,profil
c
c       mode sum output
        call mcplts
c
      else
c
        deltar=(ramax-ramin)/float(nrpts-1)
        do 43 i=1,nrd
```

```
        dst=dist(i)
        ralt=0.
        call mcflds(i,dst,sumout(1),amp(1),phs(1))
        phs1=phs(1)
        cycle=0.
c
        ralt=ramin
        do 42 j=1,nrpts
        call htgain(2,freq,sigma,epsr,alpha,nrmode,tp,dble(ralt),hgr)
        tmp1=(0.d0,0.d0)
        do 41 m=1,nrmode
        do 41 jj=1,npts
41      tmp1=tmp1+solsav(m,jj)*hgr(rcomp,m)
        tar=tmp1
        tai=tmp1*mi
        phs2=datan2(tai,tar)*57.2957795d0
        if(dabs(phs1-phs2) .ge. 180.d0) then
          if(phs1 .le. phs2) then
            cycle=cycle-360.d0
          else
            cycle=cycle+360.d0
          end if
        end if
        phs1=phs2
        xy(j)=ralt
        amp(j)=10.*dlog10(tar**2+tai**2)
        phs(j)=phs2+cycle
        sumout(j)=tmp1
42      ralt=ralt+deltar
43      call mcplts
        ralt=0.
        call htgain(1,freq,sigma,epsr,alpha,nrmode,tp,dble(ralt),hgr)
c
      end if
      if(nprint.gt.0) print 1000
      if(lunit4 .eq. 4) then
        read(lunit4,1002,end=49) bcd
        print 1003,bcd
        go to 19
49      close(unit=4)
      end if
      go to 10
c
998   print 1998,nrcrvs,nrplts
      stop
c
1000  format('1')
1002  format(a80)
1003  format(1x,a80)
c1004  format('Additional plot identification: ',3a4,1x,2a4)
1018  format(9x,f7.0,2f6.0,6x,a40)
1020  format(1x,f7.0,3(2x,f8.0),2(2x,e10.0),2(2x,e5.0))
1021  format(', rbear=',f5.1)
1022  format(' Slab ',i2,' R',f7.3,' F',f8.4,' A',f8.3,' C',f8.3,' M',
     $  f6.3,' S',1pe10.3,' E',0pf5.1,' T',f5.1)
```

```
1023  format(i1,2f9.0,i1,4e15.0)
1024  format(/11x,' M  ID   THETA''')
1025  format(11x,i2,3x,i1,0p2f10.5,i2,2(1x,1p2e16.8)/
     $       16x,      i1,0p2f10.5,i2,2(1x,1p2e16.8))
1026  format(16x,      i1,0p2f10.5,i2,2(1x,1p2e16.8)/
     $       16x,      i1,0p2f10.5,i2,2(1x,1p2e16.8))
1027  format('+',80x,' Modes',i3)
1028  format(71x,f5.0)
1031  format(' MODE  ATTEN   V/C         WAIT''S EXC')
1032  format(i4,f8.3,f9.5,2(f10.3,f7.3))
1998  format(' End of job, ',i2,' curves plotted in ',i2,' graphs')
2000  format(' Enter .com file name:  ')
2001  format(a40)
2002  format(' Control card:  [name input output start quit close] '$)
2003  format(' Enter DATUM:  iopt[1] rcomp[1] power[1]'/
     $       '               incl[0] theta[0] talt[0] ralt[0]'/
     $       '               dmin[0] dmax[10] dtic[1]'/
     $       '               tlong[0] tlat[0] rbear[0] totape[0]'/
     $       '               ampmin[-50] ampmax[70] amptic[10]'/
     $       '               phsmin[-360] phsmax[360] phstic[90]'/
     $       '               sizex[5] sizey[6]'/
     $       '               nrd[0] dist[20*0]'/
     $       '               ramin[0] ramax[50] ratic[5]'/
     $       '               nrpts[501] nrpts1[501] nrpts2[51]'/
     $       '               nrcurv[1] naplot[1] npplot[0] nprint[0]')
      end
```

```
      block data
c
      parameter (maxmds=30,nstore=(maxmds*(maxmds+3))*2+4)
c
      implicit real*8 (a-h,o-z)
      common/mc inpt/xtra(3,maxmds),temp(maxmds),freq,omega,waveno,mik,
     $       alpha,ak,ak13,ka13,ka23,ngsq,nthsq,sum0,aconst,econst
      common/mc plot/dmin,dmax,dtic,ramin,ramax,ratic,
     $       ampmin,ampmax,amptic,phsmin,phsmax,phstic,
     $       plotid,bumpid,iopt,frq,power,rcomp,
     $       incl,theta,dst,
     $       talt,al,aalt,ralt,ident,nrpts,sizex,sizey,cdate(3),
     $       ctime(2),nrcrvs,nrplts,nrcurv,naplot,npplot,npdiff,nprint
      integer rcomp,ctime,cdate
      character* 4 ident
      character*80 plotid,bumpid
      real*8 ka13,ka23,nthsq
      complex*16 xtra,temp,mik,ngsq
      real*4 dmin,dmax,dtic,ramin,ramax,ratic,dst,
     $       ampmin,ampmax,amptic,phsmin,phsmax,phstic,
     $       sizex,sizey,frq,al,talt,aalt,ralt,incl,theta,
     $       power
c
      data iopt/1/,
     $     alpha/3.14d-04/,
     $     rcomp/1/,incl,theta/0.,0./,talt,ralt/0.,0./,
     $     power/1./,al/0./,
     $     dmin,dmax,dtic/0.,10.,1./,
     $     ramin,ramax,ratic/0.,50.,5./,
     $     ampmin,ampmax,amptic/-50.,70.,10./,
     $     phsmin,phsmax,phstic/-360.,360.,90./,
     $     sizex,sizey/5.,6./,
     $     nrpts/501/,
     $     nrcurv/1/,naplot/1/,npplot/0/,npdiff/0/,nprint/0/,
     $     nrcrvs,nrplts/0,0/
c
      end
```

```
      subroutine coord
c
      implicit real*8 (a-h,o-z)
c
      parameter (maxsgs=50)
c
      real*4 gamma,phi
      common/com01/antlen(maxsgs),antrad(maxsgs),antalt(maxsgs),
     $             antang(maxsgs),npts
      common/com02/cpl,psi,iclock,chicp
      common/com03/x(maxsgs),y(maxsgs),z(maxsgs)
      common/com04/xmid(maxsgs),ymid(maxsgs),zmid(maxsgs)
      common/com05/gamma(maxsgs),phi(maxsgs)
c
      nsegs=npts-1
c
c     calculate coordinates of ends of segments of the antenna
900   format(2x,' segment',9x,'x',11x,'y',11x,'z')
      do i=1,nsegs+1
        x(i)=antrad(i)*cos(antang(i))
        y(i)=antrad(i)*sin(antang(i))
        z(i)=antalt(i)
      enddo
c
c     Transformation
      capphi=datan2(y(npts),x(npts))
      do i=1,nsegs+1
        xprime= x(i)*cos(capphi)+y(i)*sin(capphi)
        yprime=-x(i)*sin(capphi)+y(i)*cos(capphi)
        x(i)=xprime
        y(i)=yprime
      enddo
c
      do i=1,nsegs+1
        x(i)=x(i)-x(npts)
        y(i)=y(i)-y(npts)
      enddo

c     if airplane is moving in a clockwise orbit (iclock=1) then change y
      if(iclock .eq. 1) then
        do i=1,nsegs+1
          y(i)=-y(i)
        enddo
      endif
c
      cpsi=cos(psi*0.0174533)
      spsi=sin(psi*0.0174533)
      do i=1,nsegs+1
        xpp= x(i)*cpsi+y(i)*spsi
        ypp=-x(i)*spsi+y(i)*cpsi
        x(i)=xpp
        y(i)=ypp
      enddo
c
      z(nsegs+2)=z(nsegs+1)-cpl*sin(chicp*.0174533)
```

```
      x(nsegs+2)= -cpl*spsi*cos(chicp*.0174533)
      y(nsegs+2)=-cpl*cpsi*cos(chicp*.0174533)
      if(iclock .eq. 1) then
        x(nsegs+2)=-x(nsegs+2)
        y(nsegs+2)=-y(nsegs+2)
      endif

      do i=1,nsegs+1
        if(antlen(i) .le. (antlen(nsegs+1)+cpl)/2. .and.
     $     antlen(i+1) .gt. (antlen(nsegs+1)+cpl)/2.)then
           imid=i
           go to 915
        end if
      end do
  915 print *, 'imid=',imid
c     calculate midpoints of segments of the antenna
      do i=1,nsegs+1
        xmid(i)=(x(i+1)+x(i))/2.0
        ymid(i)=(y(i+1)+y(i))/2.0
        zmid(i)=(z(i+1)+z(i))/2.0
      end do

c     calculate gamma and phi
c     gamma=inclination angle of antenna from the vertical
c     phi=azimuth
      print *,' AFTER  TRANSFORMATION'
      print 910
910   format(2x,' segment',6x,'xmid',8x,'ymid',8x,'zmid',10x,'gamma',
     $       7x,'phi')
      do i=1,nsegs+1
        delx=x(i+1)-x(i)
        dely=y(i+1)-y(i)
        delz=z(i+1)-z(i)
        gamma(i)=atan2(sqrt(delx**2+dely**2),delz)
        phi(i)=atan2(dely,delx)
        print 905,i,xmid(i),ymid(i),zmid(i),
     $            gamma(i)*57.29578,phi(i)*57.29578
905     format(' ',4x,i3,6x,5(f10.3,2x))
      enddo
      return
      end
```

```
      subroutine mcflds(index,dst,sumout,amp,phs)
c
c     calculate mode sum
c
      parameter (maxmds=30,nstore=(maxmds*(maxmds+3))*2+4)
      parameter (maxsgs=50)
c
      implicit real*8 (a-h,o-z)
      real*8 idl
      real*4 gamma,phi
c
      common/com01/antlen(maxsgs),antrad(maxsgs),antalt(maxsgs),
     $             antang(maxsgs),npts
      common/com04/xmid(maxsgs),ymid(maxsgs),zmid(maxsgs)
      common/com05/gamma(maxsgs),phi(maxsgs)
      common/com06/idl(maxsgs)
      common/mc inpt/xtra(3,maxmds),temp(maxmds),freq,omega,waveno,mik,
     $       alpha,skip(7),const0,aconst,econst
      common/mc stor/a(maxmds,maxmds),fofr(maxmds),s(maxmds),tp(maxmds),
     $       xtwo,sigma,epsr,nrmode,nrslab
      common/mc plot/iskip1(12),plotid(20),iskip2(23),rcomp,
     $       incl,theta,iskip3,talt,al,aalt,ralt,
     $       ident,iskip4(14),nprint
      common iskip5(2505),hgt(3,maxmds),hgr(3,maxmds),
     $       soln a(maxmds,maxsgs),soln b(maxmds,maxsgs)
      dimension store(nstore),t(3)
      complex*16 ssav
      dimension ssav(maxmds)
      equivalence (store,a)
      complex*16 a,s,tp,fofr,xtra,temp,solna,solnb,hgt,hgr,
     $           ta,tb,mi,one,mik,mikx
      complex *16 solsav
      common/com08/solsav(maxmds,maxsgs)
      real*4 amp,phs,incl,theta,talt,al,aalt,ralt,dst
      integer rcomp,tcomp,pnmode
      complex* 8 sumout
      complex*16 sum
      character*4 plotid,ident
      data mi/(0.d0,-1.d0)/,one/(1.d0,0.d0)/
c
      const1=2.853d-3*freq**1.5
      sum=0.
      rho=dst*1000.
      numseg=npts-1
900   format(' i,dst,ant len,rad,alt,ang=',i3,5e12.5)
905   format(' i,dst,x,y,zmid,gamma,phi,idl=',i3,7e12.5)
c
      if(index .eq. 1) then
        rewind 3
        read(3) store
        do m=1,nrmode
          ssav(m)=s(m)
        enddo
        xone=0.d0
        phs1=0.
```

```
        cycle=0.
        mikx=mik*xtwo
        call htgain(1,freq,sigma,epsr,alpha,nrmode,tp,dble(ralt),hgr)
      end if
      do i=1,numseg+1
        if(index .eq. 1) then
c
c       antenna orientation factors
        t(1)=cos(gamma(i))
        t(2)=sin(gamma(i))*sin(phi(i))
        t(3)=sin(gamma(i))*cos(phi(i))
c
c       xone is distance of previous slab from the transmitter
c       xtwo is distance of next slab
c
c       get data for first slab
        call htgain(1,freq,sigma,epsr,alpha,nrmode,tp,antalt(i),hgt)
        do 689 m=1,nrmode
c       hy excitation factor
        ta=(0.d0,0.d0)
        do 686 tcomp=1,3
686     ta=ta+xtra(tcomp,m)*hgt(tcomp,m)*t(tcomp)
        soln a(m,i)=ta
        if(rcomp .eq. 1) then
c         ez excitation factor
          ta=-s(m)*ta
        else
c         ey excitation factor
          ta=fofr(m)*ta
        end if
        soln b(m,i)=ta*hgr(rcomp,m)
689     continue
c
        endif
        enddo
c
        if(dst .eq. 0.) then
          amp=100.d0
          phs=0.
          sumout=amp
          return
        end if
c
720   if(rho .gt. xtwo) then
c
c       end of current slab
        mikx=mik*(xtwo-xone)
        pnmode=nrmode
      do i=1,numseg+1
        do 735 m=1,pnmode
c       get excitation factors at end of slab
        soln a(m,i)=soln a(m,i)*zexp(mikx*(s(m)-one))
735     solsav(m,i)=soln a(m,i)
        enddo
        xone=xtwo
```

```
c
c       get data for next slab
        read(3) store
        mikx=mik*xtwo
        call htgain(1,freq,sigma,epsr,alpha,nrmode,tp,dble(ralt),hgr)
      do i=1,numseg+1
        do 740 m2=1,nrmode
c       hy excitation factor
        ta=(0.d0,0.d0)
        do 737 m1=1,pnmode
737     ta=ta+solsav(m1,i)*a(m1,m2)
        soln a(m2,i)=ta
        if(rcomp .eq. 1) then
c         ez excitation factor
          ta=-s(m2)*ta
        else
c         ey excitation factor
          ta=fofr(m2)*ta
        end if
        soln b(m2,i)=ta*hgr(rcomp,m2)
740     continue
c
        enddo
        go to 720
c
      else
c
c       get sum of modes
        mikx=mik*(rho-xone)
      do i=1,numseg+1
        ta=0.d0
        factor=idl(i)*constl/dsqrt(dabs(dsin((rho-xmid(i))/6.366d3)))
        do 730 m=1,nrmode
        tb=solnb(m,i)*zexp(mikx*(s(m)-one))*zexp(-mik*xmid(i)*ssav(m))
     $     *factor
        solsav(m,i)=tb
730     ta=ta+tb
        sum=sum+ta
        enddo
      end if
        sumr=sum
        sumi=mi*sum
        phs2=datan2(sumi,sumr)*57.2957795d0
        if(dabs(phs1-phs2) .ge. 180.d0) then
          if(phs1 .le. phs2) then
            cycle=cycle-360.d0
          else
            cycle=cycle+360.d0
          end if
        end if
        phs1=phs2
        sumout=sum
        amp=10.*dlog10(dmax1(1.d-30,sumr**2+sumi**2))
        phs=phs2+cycle
        return
```

```
c
c
1030  format('1',20a4/1x,a1,' component      INCL =',f4.0,
     $        '      THETA =',f5.0,'      PALT =',f5.1,'      AALT =',f5.1,
     $        '      RALT =',f5.1)
1031  format('0SLAB   RHO   MODE  ATTEN   V/C         REL EXC 1'/
     $        1x,i3,f9.3)
1032  format(13x,i3,f8.3,f9.5,2(f10.3,f7.3))
      end
```

```
      subroutine mcplts
c
c     plot mode sum amplitude and relative phase
c
      real*8 cpl,psi,v,rpl,zpl,zplane,radius,chicp
      common/com02/cpl,psi,iclock,chicp
      common/com09/v,rpl,zpl
      common/mc plot/dmin,dmax,dtic,ramin,ramax,ratic,
     $       ampmin,ampmax,amptic,phsmin,phsmax,phstic,
     $       id(20),bumpid(20),iopt,freq,power,rcomp,
     $       ang1,ang2,dst,talt,
     $       a1,aalt,ralt,ident,nrpts,sizex,sizey,cdate(3),ctime(2),
     $       nrcrvs,nrplts,nrcurv,naplot,npplot,npdiff,nprint
      common skip1(1002),xy(501),out1(501),out2(501),phi1(501),up(501)
      dimension x1(2),y1(2),u1(2),label(7,5),iplbl1(13),iplbl2(13)
      integer bumpid,cdate,ctime,rcomp,headng(13)
      logical up,ul,nframe,pflag
      character* 4 ident
      character*28 hlabel(5)
      character*52 hhead,blank,plbl1,plbl2
      character*80 plotid,saveid
      character*80 bumpch
      equivalence (label,hlabel),(headng,hhead),(id,plotid),
     $            (iplbl1,plbl1),(iplbl2,plbl2)
      equivalence (bumpch,bumpid)
      data nframe/.true./
      data jopt/0/
      data x1/-1.1,-0.1/,ul/2*.false./
      data hhead/'    Freq  Pwr  Aalt  Ralt  Rpl    V    Psi Iclock   '/
      data blank/'                                                    '/
      data hlabel/
     $    ' Amplitude (dB above 1uv/m) ',' Relative phase (Degrees)  ',
     $    '   Phase-phase1 (Degrees)   ','  Distance (Megameters)    ',
     $    '   Altitude (kilometers)    '/
c
c     convert plane altitude and plane radius from ft to km
      zplane=zpl*1.0d-3
      radius=rpl*1.0d-3
c     convert receiver altitude from km to kft
      ral=ralt/0.3048
      if(nprint .gt. 0) then
        if(iopt .eq. 1) then
          print 1001,ident,ralt
        else
          print 1003,ident,dst
        end if
        if(out1(1) .ge. 100.) out1(1)=99.99
        nl=nrpts/4+1
        do 19 il=1,nl
19      print 1011,(xy(i),out1(i),out2(i),i=il,nrpts,nl)
      end if
c
      if(naplot+npplot+npdiff .eq. 0) return
c
      if(jopt .ne. iopt) then
```

```
        jopt=iopt
        ndata=0
        line=0
      end if
      ndata=ndata+1
      line=line+1
      if(ndata .eq. 1) then
        if(.not. nframe) then
          call pltend
          line=0
        end if
        nframe=.true.
        do 1 i=1,nrpts
1       phi1(i)=out2(i)
      end if
      if(iopt .eq. 1) then
        xmin=dmin
        xmax=dmax
        xtic=dtic
      else
        ymin=ramin
        ymax=ramax
        ytic=ratic
      end if
c
      if(naplot .eq. 1) then
        nplot=1
        amax=ampmax
        amin=ampmin
        atic=amptic
        do 29 i=1,nrpts
        if(out1(i) .le. amax) then
          if(out1(i) .ge. amin) then
            up(i)=.false.
          else
            up(i)=.true.
            out1(i)=amin
          end if
        else
          up(i)=.true.
          out1(i)=amax
        end if
29      continue
        if(iopt .eq. 1) then
          ymax=amax
          ymin=amin
          ytic=atic
        else
          xmax=amax
          xmin=amin
          xtic=atic
        end if
        go to 100
      end if
c
```

```
30    if(npplot .eq. 1) then
        nplot=2
        fk=0.
        do 39 i=1,nrpts
        out1(i)=out2(i)+fk
        up(i)=.false.
35      if(out1(i) .ge. phsmin) then
          if(out1(i) .le. phsmax) go to 39
          fk=fk-phsmax
          out1(i)=out1(i)-phsmax
        else
          fk=fk-phsmin
          out1(i)=out1(i)-phsmin
        end if
        up(i)=.true.
        go to 35
39      continue
        if(iopt .eq. 1) then
          ymax=phsmax
          ymin=phsmin
          ytic=phstic
        else
          xmax=phsmax
          xmin=phsmin
          xtic=phstic
        end if
        go to 100
      end if
c
40    if(npdiff .eq. 1 .and. iopt .eq. 1 .and. ndata .gt. 1) then
        nplot=3
        fk=0.
        do 49 i=1,nrpts
        out1(i)=out2(i)-phi1(i)+fk
        up(i)=.false.
45      if(out1(i) .ge. phsmin) then
          if(out1(i) .le. phsmax) go to 49
          fk=fk-phsmax
          out1(i)=out1(i)-phsmax
        else
          fk=fk-phsmin
          out1(i)=out1(i)-phsmin
        end if
        up(i)=.true.
        go to 45
49      continue
        if(iopt .eq. 1) then
          ymax=phsmax
          ymin=phsmin
          ytic=phstic
        else
          xmax=phsmax
          xmin=phsmin
          xtic=phstic
        end if
```

```
          go to 100
        end if
c
50      if(line .eq. nrcurv) then
          call pltend
          nframe=.true.
          line=0
        else
          call pltoff
          saveid=plotid
          yp0=yp
        end if
        return
c
100     ipwr=power
        iang1=ang1
        iang2=ang2
        if(line .eq. 1) then
          if(nframe) then
            call pltbgn
            nrplts=nrplts+1
            xp=0.
            yp=0.
c           call symbol(xp,yp,.1,'FASTMC',0.,6)
            xp=xp+0.7
            call number(xp,yp,.1,float(nrplts),0.,-1)
            xp=xp+1.0
            call symbol(xp,yp,.1,cdate,0.,9)
            xp=xp+1.0
            call symbol(xp,yp,.1,ctime,0.,8)
            call plot(1.1,0.8+0.45*nrcurv,-3)
            nframe=.false.
          end if
          call bordr2(sizex,xmin,xmax,xtic,2.*xtic,-1,
       $              sizey,ymin,ymax,ytic,2.*ytic,-1)
          if(iopt .eq. 1) then
            xp=-.7
            yp= .5*(sizey-2.8)
            call symbol(xp,yp,.1,label(1,nplot),90.,28)
            xp= .5*(sizex-2.8)
            yp=-.4
            call symbol(xp,yp,.1,label(  1,4  ), 0.,28)
          else
            xp=-.7
            yp= .5*(sizey-2.8)
            call symbol(xp,yp,.1,label(  1,5  ),90.,28)
            xp= .5*(sizex-2.8)
            yp=-.4
            call symbol(xp,yp,.1,label(1,nplot), 0.,28)
          end if
          xp=0.
          yp=-.6
          if(bumpch(1:4) .ne. '    ') then
            call symbol(xp,yp,.1,bumpid(1),0.,4)
            xp=xp+0.5
```

```
          call symbol(xp,yp,.1,bumpid(2),0.,68)
          xp=0.
          yp=yp-.15
        end if
        call symbol(xp,yp,.1,id,0.,68)
        saveid=plotid
        yp=yp-.15
        call symbol(xp,yp,.1,headng,0.,52)
        yp=yp-.15
        dummy=9999.
        write(plbl1,2000)ident,freq,ipwr,zplane,ral,radius,v,psi,iclock
2000  format(a1,1x,f6.2,1x,i4,3(1x,f5.2),1x,f5.1,1x,f5.1,1x,i4)
        call symbol(xp,yp,.1,iplbl1,0.,52)
        xp=0.
        yp0=yp
      else
        call plton
        yp=yp0-.15
        write(plbl2,2000)ident,freq,ipwr,zplane,ral,radius,v,psi,iclock
c
c       check for common data in plot label
        if(plbl2( 1:1 ) .eq. plbl1( 1:1 ))
     $     plbl2( 1:1 )=' '
        if(plbl2( 3:8 ) .eq. plbl1( 3:8 ))
     $     plbl2( 3:8 )='       '
        if(plbl2(10:13) .eq. plbl1(10:13))
     $     plbl2(10:13)='    '
        if(plbl2(15:19) .eq. plbl1(15:19))
     $     plbl2(15:19)='    '
        if(plbl2(21:25) .eq. plbl1(21:25))
     $     plbl2(21:25)='    '
        if(plbl2(27:31) .eq. plbl1(27:31))
     $     plbl2(27:31)='    '
        if(plbl2(33:37) .eq. plbl1(33:37))
     $     plbl2(33:37)='  '
        if(plbl2(39:43) .eq. plbl1(39:43))
     $     plbl2(39:43)='   '
        if(plbl2(45:48) .eq. plbl1(45:48))
     $     plbl2(45:48)='    '
c
        if(plotid .eq. saveid) then
          pflag=.false.
        else
          call symbol(xp,yp,.1,id,0.,68)
          pflag=.true.
        end if
        if(plbl2 .ne. plbl1 .and.
     $     plbl2 .ne. blank) then
          if(pflag) then
            yp=yp-.15
            call symbol(xp,yp,.1,headng,0.,52)
            yp=yp-.15
          end if
          call symbol(xp,yp,.1,iplbl2,0.,52)
        end if
```

```
      end if
c
      yl(1)=yp+.05
      yl(2)=yp+.05
      call curve(xl,yl,ul,2,0.,0.,1.,1.,line)
c
c     draw curves
      nrcrvs=nrcrvs+1
      if(iopt .eq. 1) then
        call curve(xy,outl,up,nrpts,xmin,ymin,(xmax-xmin)/sizex,
     $       (ymax-ymin)/sizey,line)
      else
        call curve(outl,xy,up,nrpts,xmin,ymin,(xmax-xmin)/sizex,
     $       (ymax-ymin)/sizey,line)
      end if
c
      if(nplot-2) 30,40,50
c
1001  format(' ',a1,' COMP   RALT = ',f7.3/
     $       4('  DIST   AMPLITUDE  PHASE',7x))
1003  format(' ',a1,' COMP   DIST = ',f6.3/
     $       4('  RALT   AMPLITUDE  PHASE',7x))
1011  format(4(f7.3,2f10.4,5x))
      end
```

```
      subroutine mcstep
c
c     calculate mode conversion coefficients
c
      parameter (maxmds=30)
c
      implicit real*8 (a-h,o-z)
      common/mc inpt/xtra(3,maxmds),temp(maxmds),freq.omega,waveno,mik,
     $       alpha,ak,ak13,ka13,ka23,ngsq,nthsq,skip1(3)
      common/mc stor/a(maxmds,maxmds),fofr(maxmds),s(maxmds),tp(maxmds),
     $       skip2(3),nrmode,nrslab
      common/mc plot/iskip1(78),nprint
      common norm(maxmds,maxmds),capi(maxmds),ans(maxmds),ps(maxmds),
     $        ex0(maxmds), ey0(maxmds), hx0(maxmds), tau(maxmds),
     $       pex0(maxmds),pey0(maxmds),phx0(maxmds),ptau(maxmds),
     $        ext(maxmds), eyt(maxmds), hxt(maxmds), hyt(maxmds),
     $       pext(maxmds),peyt(maxmds),phxt(maxmds),phyt(maxmds)
      complex*16 thetap,fofr,xtra,temp,ans,tp,a,s,ssq,sj,sm,tj,tm,ps,
     $           q0,h10,h20,h1prm0,h2prm0,qt,h1t,h2t,h1prmt,h2prmt,
     $           a1,a2,a3,a4,mik,ey0j,ey0m,
     $           fprp,dfprp,fprl,dfprl,mult,argt,arg0,argg,norm,capi,
     $            ex0, ext, ey0, eyt, hx0, hxt, hyt, ngsq, tau,
     $           pex0,pext,pey0,peyt,phx0,phxt,phyt,pngsq,ptau,
     $           one,zero,w
      real*8 ka13,ka23,nthsq
      integer pnmode
      data      one/(1.d0,0.d0)/,zero/(0.d0,0.d0)/
      data      w/(0.d0,-1.45749544d0)/
c
      do 90 m=1,nrmode
      thetap=m
      ssq=s(m)**2
      tm=zsqrt(ngsq-ssq)
      tau(m)=tm
      a2=dcmplx(0.d0,ka13)*tm
      a1=a2/ngsq
      q0=ka23*(one-ssq)
      call mdhnkl(q0,h10,h20,h1prm0,h2prm0,thetap,'MC0 ')
      ex0(m)=-tm/ngsq
      ey0(m)=fofr(m)
      hx0(m)=tm*fofr(m)
c     hy0   =one
      qt=ka23*(nthsq-ssq)
      call mdhnkl(qt,h1t,h2t,h1prmt,h2prmt,thetap,'MCT ')
      a3=h1t   *h20   -h10   *h2t
      a4=h1t   *h2prm0-h1prm0*h2t
      fprl=a4-a1*a3
      fprp=a4-a2*a3
      a3=h1prmt*h20   -h10   *h2prmt
      a4=h1prmt*h2prm0-h1prm0*h2prmt
      dfprl=a4-a1*a3
      dfprp=a4-a2*a3
      ext(m)=dcmplx(0.d0,ak13)*dfprl/w
      eyt(m)=ey0(m)*fprp/w
      hxt(m)=dcmplx(0.d0,-ak13)*ey0(m)*dfprp/w
```

```
      hyt(m)=fprl/w
90    continue
c
      if(nrslab .eq. 1) then
c
c       first slab
        do 32 m=1,nrmode
        do 32 j=1,nrmode
        if(j .eq. m) then
          a(j,m)=one
        else
          a(j,m)=zero
        end if
32      continue
c
      else
c
c       integrals in current slab
        if(nprint .gt. 3) print 906,nrslab
        do 140 m=1,nrmode
        sm=s(m)
        tm=tau(m)
        ey0m=ey0(m)
        if(nprint .gt. 3) print 902
        do 140 j=1,nrmode
        if(j .eq. m) then
c         modes are equal
          mult=sm*(2.d0/alpha)
          ssq=sm**2
          q0=one-ssq
          qt=nthsq-ssq
          argt=temp(m)*(qt*eyt(m)**2-hxt(m)**2)+(qt*hyt(m)**2-ext(m)**2)
          arg0=temp(m)*(q0*ey0(m)**2-hx0(m)**2)+(q0          -ex0(m)**2)
          argg=(one+ey0m**2)*sm/(mik*tm)
        else
          sj=s(j)
          tj=tau(j)
          ey0j=ey0(j)
          mult=one/(mik*(sj-sm))
          argt=temp(m)*(eyt(m)*hxt(j)-eyt(j)*hxt(m))
     $         -        (hyt(m)*ext(j)-hyt(j)*ext(m))
          arg0=temp(m)*(ey0(m)*hx0(j)-ey0(j)*hx0(m))
     $         -        (       ex0(j)-       ex0(m))
          argg=(one+ey0j*ey0m)*(sj+sm)/(mik*(tj+tm))
        end if
        norm(j,m)=mult*(argt-arg0)-argg
        if(nprint .gt. 3) print 908,m,j,norm(j,m)
140     continue
c
        if(nthsq .ne. pnthsq) then
c         previous slab had different topht, must recompute fields
          do 200 j=1,pnmode
          thetap=j
          ssq=ps(j)**2
          a2=dcmplx(0.d0,kal3)*ptau(j)
```

```
          a1=a2/pngsq
          q0=ka23*(one-ssq)
          call mdhnkl(q0,h10,h20,h1prm0,h2prm0,thetap,'MCP0')
          qt=ka23*(nthsq-ssq)
          call mdhnkl(qt,h1t,h2t,h1prmt,h2prmt,thetap,'MCPT')
          a3=h1t   *h20   -h10   *h2t
          a4=h1t   *h2prm0-h1prm0*h2t
          fprl=a4-a1*a3
          fprp=a4-a2*a3
          a3=h1prmt*h20   -h10   *h2prmt
          a4=h1prmt*h2prm0-h1prm0*h2prmt
          dfprl=a4-a1*a3
          dfprp=a4-a2*a3
          pext(j)=dcmplx(0.d0,ak13)*dfprl/w
          peyt(j)=pey0(j)*fprp/w
          phxt(j)=dcmplx(0.d0,-ak13)*pey0(j)*dfprp/w
          phyt(j)=fprl/w
200       continue
        end if
c
c       integrals across slab boundary
        init=0
        if(nprint .gt. 2) print 900,nrslab
c
        do 500 j=1,pnmode
        sj=ps(j)
        sjr=sj
        sji=sj*(0.d0,-1.d0)
        tj=ptau(j)
        ey0j=pey0(j)
        if(nprint .gt. 2) print 902
c
        do 400 m=1,nrmode
        sm=s(m)
        smr=sm
        smi=sm*(0.d0,-1.d0)
        tm=tau(m)
        ey0m=ey0(m)
        if(smr .eq. sjr .and. smi .eq. sji) then
c         modes are equal
          mult=sm*(2.d0/alpha)
          ssq=sm**2
          q0=one-ssq
          qt=nthsq-ssq
          argt=temp(m)*(qt*eyt(m)**2-hxt(m)**2)+(qt*hyt(m)**2-ext(m)**2)
          arg0=temp(m)*(q0*ey0(m)**2-hx0(m)**2)+(q0          -ex0(m)**2)
          argg=(one+ey0m**2)*sm/(mik*tm)
        else
          mult=one/(mik*(sj-sm))
          argt=temp(m)*(eyt(m)*phxt(j)-peyt(j)*hxt(m))
     $         -        (hyt(m)*pext(j)-phyt(j)*ext(m))
          arg0=temp(m)*(ey0(m)*phx0(j)-pey0(j)*hx0(m))
     $         -        (       pex0(j)-        ex0(m))
          argg=(one+ey0j*ey0m)*(sj+sm)/(mik*(tj+tm))
        end if
```

```
       capi(m)=mult*(argt-arg0)-argg
       if(nprint .gt. 3) print 910,m,j,capi(m)
400    continue
c
c      calculate conversion coefficients for hy
       call clineq(norm,capi,ans,nrmode,maxmds,init,err)
       init=1
       if(nprint .gt. 3) print 902
       do 430 m=1,nrmode
       a(j,m)=ans(m)
       if(nprint .gt. 2) then
        ar=ans(m)
        ai=ans(m)*(0.d0,-1.d0)
        db=10.*dlog10(ar*ar+ai*ai)
        ang=datan2(ai,ar)*57.2957795d0
        print 903,m,j,ar,ai,db,ang
       end if
430    continue
c
500    continue
      end if
c
c     save data for next slab
      pngsq=ngsq
      pnthsq=nthsq
      pnmode=nrmode
      do 60 j=1,pnmode
      ps(j)=s(j)
      ptau(j)=tau(j)
      pex0(j)=ex0(j)
      pey0(j)=ey0(j)
      phx0(j)=hx0(j)
      pext(j)=ext(j)
      peyt(j)=eyt(j)
      phxt(j)=hxt(j)
60    phyt(j)=hyt(j)
      return
c
900   format('0conversion coefficients for slab ',i2)
901   format(' A(',i2,',',i2,')=',1p2e15.5,5x,
     $       '20*log10(A)=',0p2f10.3)
902   format(' ')
903   format(' T(',i2,',',i2,')=',1p2e15.5,5x,
     $       '20*log10(T)=',0p2f10.3)
906   format('0Integrals in slab',i3)
908   format(' Norm(',       i2,',',i2,') =',1p2e15.6)
910   format(' Capi(',       i2,',',i2,') =',1p2e15.6)
      end
```

```
      subroutine rpower
c     subroutine to calculate the radiated power - pw
c
      parameter (maxsgs=50)
      implicit real*8 (a-h,o-z)
c
      real*8 j12,j32,j52
      real*4 pin
      real*4 gamma,phi
      real*8 i0,idl,idlsav,moment,ms
      dimension moment(maxsgs),ms(maxsgs),sbar(maxsgs),r(maxsgs,maxsgs)
c
      common/com01/antlen(maxsgs),antrad(maxsgs),antalt(maxsgs),
     $             antang(maxsgs),npts
      common/com02/cpl,psi,iclock,chicp
      common/com03/x(maxsgs),y(maxsgs),z(maxsgs)
      common/com04/xmid(maxsgs),ymid(maxsgs),zmid(maxsgs)
      common/com05/gamma(maxsgs),phi(maxsgs)
      common/com06/idl(maxsgs)
      common/mc inpt/skip1(242),waveno,skip2(13)
      common/mc plot/skip3(27),pin,iskip4(24)
c
      pi=3.1415926536
      nsegs=npts-1
c
      antlen(npts+1)=antlen(npts)+cpl
      antalt(npts+1)=antalt(npts)
c
c     convert lengths to kilometers
      do i=1,npts+1
         antlen(i)=antlen(i)*0.3048d-3
         antalt(i)=antalt(i)*0.3048d-3
      enddo
      do i=1,nsegs+1
         xmid(i)=xmid(i)*0.3048d-3
         ymid(i)=ymid(i)*0.3048d-3
         zmid(i)=zmid(i)*0.3048d-3
      enddo
c
      do i=1,nsegs+1
        sbar(i)=antlen(i+1)-antlen(i)
        moment(i)=sin(waveno*((antlen(i)+antlen(i+1))/2.0))
        ms(i)=moment(i)*sbar(i)
      enddo
c
      sumeq=0.0
      sum=0.0
      do n=1,nsegs+1
      do m=1,n
      zp=waveno*(zmid(n)+zmid(m))
      zpsq=zp*zp
      cosgnm=cos(gamma(n))*cos(gamma(m))
      ssqgnm=sin(gamma(n))*sin(gamma(m))
      cospnm=cos(phi(n)-phi(m))
      singnm=0.5*sin(gamma(n))*sin(gamma(m))*cospnm
```

```
 if(n .eq. m) then
   czp=cos(zp)
   szp=sin(zp)
   zpcu=zpsq*zp
   term=(1.0+3.0/zpcu*(szp-zp*czp))*cosgnm
$      +(1.0+1.5/zpcu*((1.0-zpsq)*szp-zp*czp))*ssqgnm
   sumeq=sumeq+ms(n)*ms(m)*term
 else
 zm=waveno*(zmid(n)-zmid(m))
 zmsq=zm*zm
 r(n,m)=waveno*sqrt((xmid(n)-xmid(m))**2+(ymid(n)-ymid(m))**2)
 rsq=r(n,m)*r(n,m)
 wm=sqrt(zmsq+rsq)
 wmm1=1.0/wm
 wmm12=sqrt(wmm1)
 wmm32=wmm1*wmm12
 wmm52=wmm1*wmm32
 wp=sqrt(zpsq+rsq)
 call jfunct(wm,j12,j32,j52)
 term1=( cosgnm+singnm)*wmm12*j12
 term3=(-cosgnm+singnm)*wmm32*(j32-zmsq*wmm1*j52)
 if(xmid(n) .eq. xmid(m)) then
   if(ymid(n) .eq. ymid(m)) then
     phinm=0.0
   else
     if(ymid(n) .gt. ymid(m)) then
       phinm=pi/2.0
     else
       phinm=-pi/2.0
     endif
   endif
 else
   phinm=atan2(ymid(n)-ymid(m),xmid(n)-xmid(m))
 endif
 cospm=cos(phinm-phi(m))
 cospn=cos(phinm-phi(n))
 cos2p=cos(2.0*phinm-phi(n)-phi(m))
 cscnm=cos(gamma(n))*sin(gamma(m))*cospm
 cscmn=cos(gamma(m))*sin(gamma(n))*cospn
 ssc=0.5*sin(gamma(n))*sin(gamma(m))*cos2p
 term5=(cscnm*zm+cscmn*zm+ssc*r(n,m))*r(n,m)*wmm52*j52
 call jfunct(wp,j12,j32,j52)
 wpm1=1.0/wp
 wpm12=sqrt(wpm1)
 wpm32=wpm1*wpm12
 wpm52=wpm1*wpm32
 term2=( cosgnm-singnm)*wpm12*j12
 term4=( cosgnm+singnm)*wpm32*(j32-zpsq*wpm1*j52)
 term6=(-cscnm*zp+cscmn*zp-ssc*r(n,m))*r(n,m)*wpm52*j52
 sum=sum+ms(n)*ms(m)*(term1+term2+term3-term4+term5+term6)
 endif
 enddo
 enddo
 pw=20.0*waveno*waveno*sumeq
$  +60.0*waveno*waveno*sum
```

```
i0=sqrt(pin*1.0e3/pw)
do i=1,nsegs+1
  idl(i)=i0*ms(i)*1.0e3
enddo
return
end
```

```
      SUBROUTINE JFUNCT(Q,J12,J32,J52)
c
      implicit real*8 (a-h,o-z)
      REAL*8 J12,J32,J52

      SINQ=SIN(Q)
      COSQ=COS(Q)
      RTQ=SQRT(Q)
      J12=SINQ/RTQ
      J32=(-COSQ+J12/RTQ)/RTQ
      J52=3.*J32/Q-J12
      RETURN
      END
```

```
      subroutine tacamo
c
      implicit real*8 (a-h,o-z)
      parameter (maxsgs=50,maxmds=30)
c     TOWPLANE STEADY STATE CONFIGURATION...towplane in circular orbit
c
c     v       towplane true airspeed,knots
c     rpl     towplane turn radius(feet) for straight and level flight
c     zpl     towplane density altitude ft.
c     amg     drogue weight,lbs.
c     abase   drogue base area,sq. ft.
c     cddrog  drogue drag coefficient
c     amug    towline weight,lb/ft
c     d       towline diameter,ft.
c     al      towline length,ft.
c     picf    towline skin friction coefficient
c     cd      towline drag coefficient
c     x       distance from towpoint to drogue center of gravity,ft.
c     xcp     distance from towpoint to ddrogue center of pressure,ft.
c     alpha   pitch angle of drogue centerline(radians),positive is nose up
c     cappal  side slip angle in radians,towpoint left positive
c
c
c
      dimension dsstor(17)
      common/com01/antlen(maxsgs),antrad(maxsgs),antalt(maxsgs),
     $             antang(maxsgs),npts
      common/com02/cpl,psi,iclock,chicp
      common/com09/v,rpl,zpl
      common/mc inpt/skip1(8,maxmds),freq,skip2(15)
c
      namelist/tacin/v,rpl,zpl,rzd,zzd,amg,psi,iclock,itrset,cpl,chicp
      data g/32.17/,pi/3.1416/,acl/.53/,acm/-.68/,x/1.34/,xcp/2.31/,
     $     abase/5.6/,cddrog/.6/,amug/1.095e-1/,d/1.75e-2/,dds/1.e3/,
     $     picf/2.2e-2/,cd/1.03/,delr/5.0/,delz/50.0/
      data dsstor/50.,130.,4*200.,4*300.,4*400.,2*600.,1000./
c
c
      read tacin
      print tacin
      al=4.9179d+5/freq-cpl
      iter=1
      print 1101,v,rpl,zpl,rzd,zzd
      print 1102,amg,abase,cddrog
      print 1103,amug,d,al,dds,picf,cd
      ddrog=sqrt(4.*abase/pi)
      if(cd .le. .1) stop
      k=1
      istop=0
      index=0
    5 iter=iter+1
      if(iter .gt. 30) then
         print 1000
         stop
      endif
```

```
16    continue
      zz=zzd
      rz=rzd
      i=1
      ds=20.
      omega=v*1.69/rpl
      m=1
      arg=1.-.006875*zz/1000.
      if(arg .gt. 0.)then
        rhoz=0.002378*arg**4.256
      else
        print *, 'does not converge'
        stop
      end if
      omegsq=omega*omega
      s=0.
      thz=0.
      qs=.5*rhoz*rz*rz*omegsq*abase
      a=0.5*amg*x/qs+acl*x*.04
      b=(acl+cddrog)*x-ddrog*acm
      c=-0.04*acl*x-amg*x/qs
      alpha=(-b+sqrt(b*b-4.*a*c))*.5/a
      if(alpha .ge. .8) then
         a2=.85*qs*xcp
         b2=-amg*x-.85*pi*qs*xcp
         c2=amg*x*pi*.5-.85*qs*xcp
         alpha=(-b2-sqrt(b2*b2-4.*a2*c2))*.5/a2
         m=2
         am=amg/g
      endif
      al=-amg/g*rz*omegsq-qs*acl*.04
      bl=qs*(ddrog*acm/(x*cos(alpha))-acl-cddrog)
      cappal=al/bl
      if(cappal .le. .04) then
         trpz=-amg/g*rz*omegsq
      else
         trpz=-amg/g*rz*omegsq+acl*(cappal-.04)*qs
      endif
      if(alpha .le. .04) then
         tzpz=amg
         trthpz=0.5*rhoz*abase*cddrog*rz*rz*omegsq
      else
         if(m .eq. 1) then
            tzpz=amg-acl*(alpha-.04)*qs
            trthpz=0.5*rhoz*abase*cddrog*rz*rz*omegsq
         else
            if(m .ne. 2) stop
            tzpz=amg-.85*qs*sin(alpha)**2.*cos(alpha)
            trthpz=.85*qs*sin(alpha)**3.
         endif
      endif
      tz=sqrt(trpz*trpz+trthpz*trthpz+tzpz*tzpz)
      rpz=trpz/tz
      thpz=trthpz/(tz*rz)
      zpz=tzpz/tz
```

```
      rthpz=thpz*rz
      npts=1
      if(mod(index,3) .eq. 0 .or. k .eq. 4) then
        print 1
        print 21,s,rz,zz,thz,tz,rpz,zpz,rthpz
      endif
      if(k .eq. 4) then
        antlen(npts)=s
        antrad(npts)=rz
        antalt(npts)=zz
        antang(npts)=thz
      endif
    8 s=s+ds
      npts=npts+1
      if(npts .gt. maxsgs-2) then
        print *,'number of segments too big'
        stop
      endif
      z=zz
      t=tz
      r=rz
      rp=rpz
      thp=thpz
      zp=zpz
      kk=1
   10 za=.5*(z+zz)
      ta=.5*(t+tz)
      ra=.5*(r+rz)
      thpa=.5*(thp+thpz)
      zpa=.5*(zp+zpz)
      rpa=.5*(rp+rpz)
      arg=1.-.006875*za/1000.
      if(arg .gt. 0.)then
        rho=.002378*arg**4.256
      else
        print *, 'does not converge'
        stop
      end if
      thpasq=thpa*thpa
      rasq=ra*ra
      qd=.5*rho*d*rasq*omegsq*cd
      strth=sqrt(1.-rasq*thpasq)
      trp=(ta*ra*thpasq-amug/g*ra*omegsq-qd*ra*rpa*thpa*strth)*ds+trpz
      trthp=(-ta*rpa*thpa+qd*strth**3.+.5*rho*d*picf*rasq*omegsq)*ds+
     $      trthpz
      tzp=(amug-qd*ra*thpa*zpa*strth)*ds+tzpz
      tl=sqrt(trp*trp+trthp*trthp+tzp*tzp)
      rpl=trp/tl
      rl=rz+(rpl+rpz)*.5*ds
      thpl=trthp/(tl*rl)
      zpl=tzp/tl
      zl=zz+.5*(zpl+zpz)*ds
      rthpl=thpl*rl
      thl=thz+(thpz+thpl)*.5*ds
      if(abs((zl-z)/z) .gt. .001) go to 100
```

```
      if(abs((rl-r)/r) .gt. .001) go to 100
      if(abs((tl-t)/t) .gt. .001) go to 100
   32 if(1 .le. i .and. i .le. 17) then
         ds=dsstor(i)
      else
         if(i .eq. 18) then
            if(s-al+1.0e-4 .eq. 0.0) then
               ds=dds
            else
               if(s-al+1.0e-4 .lt. 0.0) then
                  go to 50
               else
                  go to 190
               endif
            endif
         else
            print *,' i is not between 1 and 18'
            stop
         endif
      endif
      i=i+1
   50 zz=zl
      rz=rl
      tz=tl
      rpz=rpl
      thpz=thpl
      zpz=zpl
      thz=thl
      trpz=trp
      trthpz=trthp
      tzpz=tzp
      if(al-s .lt. 1000.)ds=al-s
      if(k .eq. 4) then
        print 21,s,rl,zl,thl,tl,rpl,zpl,rthpl
        antlen(npts)=s
        antrad(npts)=rl
        antalt(npts)=zl
        antang(npts)=thl
      endif
      go to 8
  100 if(kk .gt. 20) then
         print 3
         go to 32
      endif
      z=zl
      r=rl
      t=tl
      rp=rpl
      thp=thpl
      zp=zpl
      rthp=rthpl
      kk=kk+1
      go to 10
  190 continue
      if(mod(index,3) .eq. 0 .or. k .eq. 4)
```

```
     $  print 21,s,rl,zl,thl,tl,rpl,zpl,rthpl
      if(k .eq. 4) then
        antlen(npts)=s
        antrad(npts)=rl
        antalt(npts)=zl
        antang(npts)=thl
      endif
      if(iter .gt. itrset) go to 18
      go to(200,203,18,999),k
  200 if(abs((zl-zpl)/zpl)-.010)202,210,210
  202 go to(204,240),k
  203 if(abs(zl-zpl)-2000.)204,204,210
  204 if(abs((rl-rpl)/rpl)-.003)206,220,220
  206 go to(240,200),k
  210 k=1
      zzd=zzd+.6*(zpl-zl)
      go to 5
  220 if(k .eq. 1) then
         drzd=.16*(rpl-rl)
      else
         if(k .eq. 2) then
            if(abs((rl-rstore)/rpl) .le. .002) drzd=.2*(rpl-rl)
            drzd=(rpl-rl)*drzd*.6/(rl-rstore)
            if(abs(drzd) .gt. 150.) then
               drzd=150.*drzd/abs(drzd)
            else
               if(abs(drzd) .le. 5.) drzd=5.*drzd/abs(drzd)
            endif
         else
            print *,' k is not 1 or 2'
            stop
         endif
      endif
      rzd=rzd+drzd
      k=2
      rstore=rl
      go to 5
  240 k=3
18    continue
      if(index .gt. 30)go to 1199
      if(istop .eq. 500)go to 999
      if(mod(index,3) .eq. 0)then
        if(abs((zl-zpl)/zpl) .lt. .001 .and. abs((rl-rpl)/rpl) .lt.
     $      .001)go to 17
        f1=zl-zpl
        f2=rl-rpl
        zlsav=zl
        rlsav=rl
        zzdsav=zzd
        rzdsav=rzd
        zzd=zzd+delz
        index=index+1
        go to 16
      else
        if(mod(index,3) .eq. 1)then
```

```
          dz1dzd=(z1-z1sav)/delz
          dr1dzd=(r1-r1sav)/delz
          zzd=zzdsav
          rzd=rzdsav+delr
          index=index+1
          go to 16
        else
          dz1drd=(z1-z1sav)/delr
          dr1drd=(r1-r1sav)/delr
          den=dz1dzd*dr1drd-dr1dzd*dz1drd
          delzd=(f2*dz1drd-f1*dr1drd)/den
          delrd=(f1*dr1dzd-f2*dz1dzd)/den
          zzd=zzdsav+delzd
          rzd=rzdsav+delrd
          index=index+1
          go to 16
        end if
      end if
17    k=4
      istop=500
      go to 16
999   return
1199  print *,'does not converge'
      stop
c
    1 format(1h ,5x,1hs,11x,1hr,11x,1hz,11x,2hth,10x,1ht,10x,2hrp,10x,
     $       2hzp,9x,4hrthp)
    2 format(8e12.4)
   21 format(8e12.4)
    3 format(13h not converge)
 1000 format(1h1,14hr1not converge)
 1101 format(1h ,8hv(knot)=,e12.4,2x,4hrp1=,e12.4,2x,4hzp1=,e12.4,2x,
     $       4hrzd=,e12.4,2x,4hzzd=,e12.4)
 1102 format(1h ,4hamg=,e12.4,2x,6habase=,e12.4,2x,7hcddrog=,e12.4)
 1103 format(1h ,5hamug=,e12.4,2x,2hd=,e12.4,2x,3ha1=,e12.4,2x,4hdds=,
     $       e12.4,2x,5hpicf=,e12.4,2x,3hcd=,e12.4)
 1203 format(1h ,3x,6halpha=,e12.4,5x,7hcappa1=,e12.4)
c
      end
```

```
      SUBROUTINE HTGAIN(IOPT,FREQ,SIGMA,EPSR,ALPHA,NRMODE,TP,Z,HG)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMPLEX*16 TP(1),HG(3,1),HG0,
     $           C,SSQ,NGSQ,SQROOT,RATIO,A1,A2,A3,A4,MI,ONE
     $           P0,H10,H20,H1PRM0,H2PRM0,H1Z,H2Z,H1PRMZ,H2PRMZ,EXPZ
      REAL*8 K,KA13,KA23
      DATA MI/(0.D0,-1.D0)/,ONE/(1.D0,0.D0)/,HG0/(0.D0,1.45749544D0)/
      DATA DTR/1.745329252D-02/
C
      NGSQ=DCMPLX(EPSR,-SIGMA/(5.5633459D-8*FREQ))
      K=2.0958426D-02*FREQ
      IF(ALPHA .EQ. 0.D0) GO TO 5
      AK=ALPHA/K
      AK13=DEXP(DLOG(AK)/3.D0)
      AK23=AK13**2
      KA13=1.D0/AK13
      KA23=KA13**2
      P1=KA23*ALPHA*Z
      EXPZ=DEXP(.5D0*ALPHA*Z)
 5    DO 20 M=1,NRMODE
      SSQ=ZSIN(TP(M)*DTR)**2
      CSQ=ONE-SSQ
      SQROOT=ZSQRT(NGSQ-SSQ)
      IF(DIMAG(TP(M)) .LE. -10.D0 .OR. ALPHA .EQ. 0.D0) GO TO 10
      P0=KA23*(ONE-SSQ)
      CALL MDHNKL(P0    ,H10,H20,H1PRM0,H2PRM0,TP(M),'HG 1')
      CALL MDHNKL(P0+P1,H1Z,H2Z,H1PRMZ,H2PRMZ,TP(M),'HG 2')
      A1=H10    *H2Z    -H1Z    *H20
      A2=H1PRM0*H2Z    -H1Z    *H2PRM0
      A3=H10    *H2PRMZ-H1PRMZ*H20
      A4=H1PRM0*H2PRMZ-H1PRMZ*H2PRM0
      RATIO=SQROOT/NGSQ
      C=.5D0*AK23+KA13*MI*RATIO
      HG(1,M)=EXPZ*(C*A1+A2)
      HG(2,M)=KA13*MI*SQROOT*A1+A2
      HG(3,M)=.5D0*AK*MI*HG(1,M)+AK13*MI*EXPZ*(C*A3+A4)
      IF(IOPT .EQ. 1) GO TO 20
      HG(1,M)=HG(1,M)/HG0
      HG(2,M)=HG(2,M)/HG0
      HG(3,M)=HG(3,M)/(RATIO*HG0)
      GO TO 20
 10   C=ZSQRT(ONE-SSQ)
      EXPZ=ZEXP(DCMPLX(0.D0,K*Z)*C)
      A1=(NGSQ*C-SQROOT)/(NGSQ*C+SQROOT)
      A2=(C-SQROOT)/(C+SQROOT)
      HG(1,M)=EXPZ+A1/EXPZ
      HG(2,M)=EXPZ+A2/EXPZ
      HG(3,M)=(EXPZ-A1/EXPZ)*C
      IF(IOPT .EQ. 1) GO TO 20
      HG(1,M)=HG(1,M)/(ONE+A1)
      HG(2,M)=HG(2,M)/(ONE+A2)
      HG(3,M)=HG(3,M)/((ONE-A1)*C)
 20   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE MDHNKL (Z,H1,H2,H1PRME,H2PRME,THETA,IDBG)
      IMPLICIT COMPLEX*16 (A-H,O-Z)
      COMPLEX*16 I,MPOWER,MTERM
      REAL*8 A,B,C,D,CAP,PART1,PART2,ZMAG
      CHARACTER*4 IDBG
      DIMENSION A(30), B(30), C(30), D(30), CAP(30), PART1(2), PART2(2)
      EQUIVALENCE (PART1,TERM4), (PART2,SUM4)
      DATA A   /  9.3043671692922944819D-01,  3.1014557230974314911D+01,
     $            2.0676371487316209897D+02,  5.7434365242545027449D+02,
     $            8.7021765519007617234D+02,  8.2877871922864397320D+02,
     $            5.4168543740434246542D+02,  2.5794544638302022111D+02,
     $            9.3458495066311674231D+01,  2.6626351870744066662D+01,
     $            6.1210004300561072794D+00,  1.1592803844803233472D+00,
     $            1.8401275944132116616D-01,  2.4833030963741048003D-02,
     $            2.8842080097260218300D-03,  2.9133414239656786138D-04,
     $            2.5827494893312753646D-05,  2.0256858739853140063D-06,
     $            1.4155736366074870734D-07,  8.8695090013000443124D-09,
     $            5.0110220346327933889D-10,  2.5658074934115685526D-11,
     $            1.1961806496091228666D-12,  5.0988092481207283185D-14,
     $            1.9948392989517716388D-15,  7.1886100863126905797D-17,
     $            2.3938095525516785112D-18,  7.3883010881224645255D-20,
     $            2.1194208514407528762D-21,  5.6653858632471341093D-23/
      DATA B   /  6.7829872514427588456D-01,  1.1304978752404598033D+01,
     $            5.3833232154307609704D+01,  1.1962940478735024376D+02,
     $            1.5337103177865415841D+02,  1.2780919314887846509D+02,
     $            7.4742218215718400631D+01,  3.2355938621523117060D+01,
     $            1.0785312873841039006D+01,  2.8532573740320209005D+00,
     $            6.1360373635097223595D-01,  1.0937678009821251966D-01,
     $            1.6422939954686564465D-02,  2.1055051223957133911D-03,
     $            2.3316778764072130571D-04,  2.2528288660939256561D-05,
     $            1.9156708045016374595D-06,  1.4446989475879618839D-07,
     $            9.7286124416697769730D-09,  5.8854279743918795891D-10,
     $            3.2160808603234314644D-11,  1.5952782045255116351D-12,
     $            7.2151886229105003778D-14,  2.9876557444763976717D-15,
     $            1.1368553061173507104D-16,  3.9889659863766691603D-18,
     $            1.2946984700995355913D-19,  3.8985199340546088228D-21,
     $            1.0920223904914870636D-22,  2.8527230681595795812D-24/
      DATA C   /  4.6521835846461472410D-01,  6.2029114461948629822D+00,
     $            2.5845464359145262382D+01,  5.2213059311404570392D+01,
     $            6.2158403942148298012D+01,  4.8751689366390821897D+01,
     $            2.7084271870217123228D+01,  1.1215019407957400909D+01,
     $            3.5945575025504490022D+00,  9.1815006450841609147D-01,
     $            1.9128126343925335199D-01,  3.3122296699437809740D-02,
     $            4.8424410379295043444D-03,  6.0568368204246458321D-04,
     $            6.5550182039227768583D-05,  6.1985987743950608612D-06,
     $            5.1654989786625507119D-07,  3.8220488188402150986D-08,
     $            2.5278100653705126277D-09,  1.5033066103898380141D-10,
     $            8.0822936042464409157D-12,  3.9473961437101054471D-13,
     $            1.7590891906016512675D-14,  7.1814214762263778920D-16,
     $            2.6957287823672589641D-17,  9.3358572549515461865D-19,
     $            2.9922619406895981315D-20,  8.9015675760511620701D-22,
     $            2.4644428505125033375D-23,  6.3656020935361057409D-25/
      DATA D   /  6.7829872514427588456D-01,  4.5219915009618392131D+01,
     $            3.7683262508015326776D+02,  1.1962940478735024344D+03,
     $            1.9938234131225040548D+03,  2.0449470903820554375D+03,
```

```
     $            1.4201021460986496090D+03,  7.1183064967350857463D+02,
     $            2.6963282184602597492D+02,  7.9891206472896585111D+01,
     $            1.9021715826880139294D+01,  3.7188105233392256682D+00,
     $            6.0764877832340288572D-01,  8.4220204895828535644D-02,
     $            1.0026214868551016149D-02,  1.0363012784032058021D-03,
     $            9.3867869420580235442D-05,  7.5124345274574017960D-06,
     $            5.3507368429183773360D-07,  3.4135482251472901638D-08,
     $            1.9618093247972931935D-09,  1.0209780508963274472D-10,
     $            4.8341763773500352579D-12,  2.0913590211334783723D-13,
     $            8.2990437346566602039D-15,  3.0316141496462685641D-16,
     $            1.0228117913786331176D-17,  3.1967863459247792364D-19,
     $            9.2821903191776400453D-21,  2.5103962999804300309D-22/
      DATA CAP /  1.0416666666666666663D-01,  8.3550347222222222116D-02,
     $            1.2822657455632716019D-01,  2.9184902646414046315D-01,
     $            8.8162726744375764874D-01,  3.3214082818627675264D+00,
     $            1.4995762986862554546D+01,  7.8923013011586517530D+01,
     $            4.7445153886826431887D+02,  3.2074900908906619004D+03,
     $            2.4086549640874004605D+04,  1.9892311916950979121D+05,
     $            1.7919020077753438063D+06,  1.7484377180034121023D+07,
     $            1.8370737967633072978D+08,  2.0679040329451551508D+09,
     $            2.4827519375935888472D+10,  3.1669454981734887315D+11,
     $            4.2771126865134715582D+12,  6.0971132411392560749D+13,
     $            9.1486942234356396792D+14,  1.4413525170009350101D+16,
     $            2.3788844395175757942D+17,  4.1046081600946921885D+18,
     $            7.3900049415704853993D+19,  1.3859220004603943141D+21,
     $            2.7030825930275761623D+22,  5.4747478619645573335D+23,
     $            1.1498937014386333524D+25,  2.5014180692753603969D+26/
      DATA I/(0.D0,1.D0)/
      DATA ONE/(1.D0,0.D0)/,TWO/(2.D0,0.D0)/,ZERO/(0.D0,0.D0)/
      DATA ROOT3/(1.73205080756888D0,0.D0)/
      DATA ALPHA/(8.53667218838951D-1,0.D0)/
      DATA CONST1/( 2.58819045102522D-01,-9.65925826289067D-01)/
      DATA CONST2/( 2.58819045102522D-01, 9.65925826289067D-01)/
      DATA CONST3/(-9.65925826289067D-01, 2.58819045102522D-01)/
      DATA CONST4/(-9.65925826289067D-01,-2.58819045102522D-01)/
C
      ZPOWER=ONE
      SUM3=ZERO
      SUM4=ZERO
      ZMAG=ZABS(Z)
      IF(ZMAG .GT. 6.1D0) GO TO 70
      SUM1=ZERO
      SUM2=ZERO
      ZTERM=-Z**3/(200.D0,0.D0)
      DO 50 M=1,30
      SUM1=SUM1+DCMPLX(A(M),0.D0)*ZPOWER
      SUM2=SUM2+DCMPLX(B(M),0.D0)*ZPOWER
      SUM3=SUM3+DCMPLX(C(M),0.D0)*ZPOWER
      TERM4=DCMPLX(D(M),0.D0)*ZPOWER
      SUM4=SUM4+TERM4
      IF(DABS(PART1(1)/PART2(1)) .LE. 1.D-17 .AND.
     $   DABS(PART1(2)/PART2(2)) .LE. 1.D-17) GO TO 60
50    ZPOWER=ZPOWER*ZTERM
60    GM2F=I*(Z*SUM2-TWO*SUM1)/ROOT3
      GPMFP=I*(SUM4+TWO*Z*Z*SUM3)/ROOT3
```

```
      H1=Z*SUM2+GM2F
      H2=H1-TWO*GM2F
      H1PRME=SUM4+GPMFP
      H2PRME=H1PRME-TWO*GPMFP
      GO TO 999
70    MPOWER=ONE
      SUM1=ONE
      SUM2=ONE
c     RTZ=CDSQRT(Z)
      RTZ=ZSQRT(Z)
      SQRTZB=RTZ*Z
      ZTERM=I/SQRTZB
      MTERM=-ZTERM
      DM=ZERO
      TERM3=ONE
      DO 80 M=1,30
      ZPOWER=ZPOWER*ZTERM
      MPOWER=MPOWER*MTERM
      DM=DM+ONE
      TERM1=DCMPLX(CAP(M),0.D0)*ZPOWER
      TERM2=DCMPLX(CAP(M),0.D0)*MPOWER
      IF(ZABS(TERM2/TERM3) .GE. 1.D0) GO TO 81
      SUM1=SUM1+TERM1
      SUM2=SUM2+TERM2
      SUM3=SUM3+DM*TERM1
      TERM4=DM*TERM2
      SUM4=SUM4+TERM4
      IF(DABS(PART1(1)/PART2(1)) .LE. 1.D-17 .AND.
     $   DABS(PART1(2)/PART2(2)) .LE. 1.D-17) GO TO 81
80    TERM3=TERM2
81    ZTERM=(-1.5D0,0.D0)/Z
      SUM3=SUM3*ZTERM
      SUM4=SUM4*ZTERM
      TERM1=((-0.25D0,0.D0)-I*SQRTZB)/Z
      TERM2=((-0.25D0,0.D0)+I*SQRTZB)/Z
c     EXP1=CDEXP((0.D0,0.66666666666666667D0)*SQRTZB)
      EXP1=ZEXP((0.D0,0.66666666666666667D0)*SQRTZB)
      EXP2=CONST1*EXP1
      EXP3=CONST2/EXP1
      EXP4=CONST3*EXP1
      EXP5=CONST4/EXP1
c     ZTERM=ALPHA/CDSQRT(RTZ)
      ZTERM=ALPHA/ZSQRT(RTZ)
      TERM4=Z
      IF(PART1(1) .GE. 0.D0 .OR. PART1(2) .GE. 0.D0) GO TO 90
      H1=ZTERM*(EXP2*SUM2+EXP5*SUM1)
      H1PRME=ZTERM*(EXP2*(SUM2*TERM2+SUM4)+EXP5*(SUM1*TERM1+SUM3))
      GO TO 110
90    H1=ZTERM*EXP2*SUM2
      H1PRME=ZTERM*EXP2*(SUM2*TERM2+SUM4)
110   IF(PART1(1) .GE. 0.D0 .OR. PART1(2) .LT. 0.D0) GO TO 120
      H2=ZTERM*(EXP3*SUM1+EXP4*SUM2)
      H2PRME=ZTERM*(EXP3*(SUM1*TERM1+SUM3)+EXP4*(SUM2*TERM2+SUM4))
      GO TO 999
120   H2=ZTERM*EXP3*SUM1
```

```
      H2PRME=ZTERM*EXP3*(SUM1*TERM1+SUM3)
C CALCULATE WRONSKIAN AS PARTIAL CHECK ON VALIDITY
999   SUM4=H1*H2PRME-H1PRME*H2
      IF(DABS(PART2(1)) .LE. 1.D-8 .AND.
     $   DABS(PART2(2)+1.457495441040461D0) .LE. 1.D-8) GO TO 1000
      PRINT 1001,SUM4,THETA,IDBG
1000  RETURN
1001  FORMAT(' ***** POSSIBLE ERROR IN MDHNKL:  W = ',1P2E15.6,
     $       ' FOR THETA = ',0P2F10.4,' AT ',A4)
      END
```

```
      SUBROUTINE CLIN EQ (A, B, X, N, N DIM, IFLAG, ERR)
C
C     CLIN EQ USES L-U DECOMPOSITION TO FIND THE TRIANGULAR MATRICES L
C     AND U SUCH THAT L*U=A.  THE MATRICES L AND U ARE STORED IN A.
C     THIS FORM IS USED WITH BACK SUBSTITUTION TO FIND THE SOLUTION X OF
C     A*X=L*U*X=B.  N IS THE NUMBER OF EQUATIONS AND NDIM IS
C     THE DIMENSION OF ALL ARRAYS.  ERR IS THE ESTIMATED RELATIVE ERROR
C     OF THE SOLUTION VECTOR.
C
C     IF IFLAG=0, THEN L, U AND X ARE COMPUTED.  OTHERWISE, IT IS
C     ASSUMED THAT L AND U HAVE BEEN COMPUTED IN A PREVIOUS CALL AND ARE
C     STILL STORED IN A.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMPLEX*16 A, B, X, T
      DIMENSION  A(N DIM, N DIM), B(NDIM), X(N DIM)
      DIMENSION  IROW(51), Q(51)
      DATA  EPS /1.D-15/
C
      IF (N .GT. 51) THEN
        PRINT 999
        STOP
      END IF
      IF (N .EQ. 1) THEN
        X(1)=B(1)/A(1,1)
        RETURN
      END IF
      IF (IFLAG .EQ. 0) THEN
        DO 50  I=1,N
        Q(I)=0.D0
        DO 40  J=1,N
        QQ=ZABS(A(I,J))
   40   IF (Q(I) .LT. QQ)  Q(I)=QQ
        IF (Q(I) .EQ. 0.D0) THEN
          LOOP=40
          GO TO 900
        END IF
   50   CONTINUE
        ERR=EPS
        PPIV=0.D0
        DO 100  I=1,N
  100   IROW(I)=I
C
        DO 500  L=1,N
        PIVOT=0.D0
        K=L-1
        DO 240  I=L,N
        IF (K .GE. 1) THEN
          DO 220  J=1,K
  220     A(I,L)=A(I,L)-A(J,L)*A(I,J)
        END IF
        F=ZABS(A(I,L))/Q(I)
        IF (PIVOT .LE. F) THEN
          PIVOT=F
          NPIVOT=I
```

```
        END IF
  240   CONTINUE
        IF (PIVOT .EQ. 0.D0) THEN
          LOOP=240
          GO TO 900
        END IF
        IF (PPIV .GT. PIVOT) ERR=ERR*PPIV/PIVOT
        PPIV=PIVOT
        IF (NPIVOT .NE. L) THEN
          Q(NPIVOT)=Q(L)
          J=IROW(L)
          IROW(L)=IROW(NPIVOT)
          IROW(NPIVOT)=J
          DO 260  I=1,N
          T=A(L,I)
          A(L,I)=A(NPIVOT,I)
          A(NPIVOT,I)=T
  260     CONTINUE
        END IF
        IF (L .NE. N) THEN
          T=(1.D0,0.D0)/A(L,L)
          K=L+1
          M=L-1
          DO 450  I=K,N
          IF (M .GE. 1) THEN
            DO 350  J=1,M
  350       A(L,I)=A(L,I)-A(L,J)*A(J,I)
          END IF
          A(L,I)=T*A(L,I)
  450     CONTINUE
        END IF
  500   CONTINUE
        IF (ERR .GT. 1.D-5) PRINT 998,ERR
      END IF
C
      DO 620  I=2,N
  620 X(I)=(0.D0,0.D0)
      J=IROW(1)
      X(1)=B(J)/A(1,1)
      DO 700  I=2,N
      J=IROW(I)
      K=I-1
      DO 650  L=1,K
  650 X(I)=X(I)+A(I,L)*X(L)
      X(I)=(B(J)-X(I))/A(I,I)
  700 CONTINUE
      K=N-1
      DO 800  I=1,K
      J=N-I
      M=J+1
      DO 800  L=M,N
      X(J)=X(J)-X(L)*A(J,L)
  800 CONTINUE
      RETURN
  900 PRINT 997,LOOP
```

```
      STOP
  997 FORMAT('0******ERROR IN CLIN EQ, MATRIX IS SINGULAR, AT LOOP ',I3)
  998 FORMAT('0******CAUTION-',
     $       'CLIN EQ HAS DECOMPOSED AN ILL-CONDITIONED MATRIX.'/15X,
     $       'RESULTS WILL HAVE RELATIVE ERROR =',1PE12.2)
  999 FORMAT('0******ERROR IN CLIN EQ, MATRIX SIZE GREATER THAN 51')
      END
```

```
      subroutine pltbgn
      character*1 answr
      logical first,autopl
      dimension ia(8),ibuff(2)
      data ia/82,79,57,48,73,87,73,80/
c ASCII        R  O  9  0  I  W  I  P
      data first/.true./,autopl/.false./
      if(first) then
        open(unit=52,file='/dev/plt7550a')
        print *,'If this is the hp 7550 plotter and you want auto feed,'
        print *,'then set up the plotter, load a sheet and answer y:'
        print *,'Do you want auto feed?'
        read 1, answr
1       format(a1)
        if(answr .eq. 'y' .or. answr .eq. 'Y') then
          autopl=.true.
        else
          autopl=.false.
        end if
      end if
      if(.not.autopl .or. first) then
        print *,'Set up plotter, enter rotation (y/n) when ready'
        read 1, answr
        first=.false.
      end if
      call hpinit(2,0,0,0,52)
      if(answr .eq. 'y' .or. answr .eq. 'Y')
     $  call buff(1,ia,xbuff,8)
      call newpen(1)
      return
      end
      subroutine plton
      dimension ia(3)
      data ia/27,46,89/
c ASCII      esc  .  Y
      call buff(1,ia,xbuff,-3)
      call newpen(1)
      return
      end
      subroutine pltend
      call newpen(0)
      entry pltoff
      call plot(0.0,0.0,999)
      return
      end
```

```
      subroutine bordr2(xlng,xmin,xmax,xticl,xtic2,ndx,
     $                  ylng,ymin,ymax,yticl,ytic2,ndy)
c
      xscale=xlng/(xmax-xmin)
      yscale=ylng/(ymax-ymin)
      if(xticl*xscale .le. 0. .or. xtic2*xscale .le. 0.) go to 999
      if(yticl*yscale .le. 0. .or. ytic2*yscale .le. 0.) go to 999
c
      if(iabs(ndx) .gt. 9) then
        sx=.15
        nx=ndx-(ndx/10)*10
      else
        sx=.1
        nx=ndx
      end if
      xo=.5*sx
      if(iabs(ndy) .gt. 9) then
        sy=.15
        ny=ndy-(ndy/10)*10
      else
        sy=.1
        ny=ndy
      end if
      yo=.5*sy
c
      xres=abs(xticl)/10.
      yres=abs(yticl)/10.
c
      tl=.05
      t2=.10
      yval=ymin
      ytc2=ymin
      xp=0.
      yp=0.
      go to 115
112   yval=yval+yticl
      if(abs(yval-ymax) .le. yres) then
        call plot(0.,ylng,2)
        if(abs(yval-ytc2) .le. yres) then
          xln=-sy*(3+ny)
          yln=ylng-yo
          if(abs(yval) .gt. yres .or. abs(yval) .ge.  10.)
     $      xln=xln-sy*aint(alog10(abs(yval)))
          if(abs(yval) .lt. yres) yval=0.
          if(yval .lt. 0.) xln=xln-sy
          call plot(xln,yln,3)
          call number(xln,yln,sy,yval,0.,ny)
        end if
        call plot(0.,ylng,3)
        go to 120
      end if
      yp=(yval-ymin)*yscale
      call plot(xp,yp,2)
      if(abs(yval-ytc2) .gt. yres) go to 118
        call plot(t2,yp,2)
```

```
115       xln=-sy*(3+ny)
          yln=yp-yo
          if(abs(yval) .gt. yres .or. abs(yval) .ge.  10.)
     $      xln=xln-sy*aint(alog10(abs(yval)))
          if(abs(yval) .lt. yres) yval=0.
          if(yval .lt. 0.) xln=xln-sy
          call plot(xln,yln,3)
          call number(xln,yln,sy,yval,0.,ny)
          ytc2=ytc2+ytic2
          go to 119
118     call plot(tl,yp,2)
119     call plot(xp,yp,3)
        if(abs(yval-ymax) .gt. yres) go to 112
c
120     yp=ylng
        tl=ylng-.05
        t2=ylng-.10
        xval=xmin
        xtc2=xmin+xtic2
122     xval=xval+xticl
        if(abs(xval-xmax) .gt. xres) go to 123
          call plot(xlng,ylng,2)
          if(abs(xval-xtc2) .le. xres) xtc2=xtc2+xtic2
          go to 130
123     xp=(xval-xmin)*xscale
        call plot(xp,yp,2)
        if(abs(xval-xtc2) .gt. xres) go to 128
          call plot(xp,t2,2)
          xtc2=xtc2+xtic2
          go to 129
128     call plot(xp,tl,2)
129     call plot(xp,yp,2)
        if(abs(xval-xmax) .gt. xres) go to 122
c
130     xp=xlng
        tl=xlng-.05
        t2=xlng-.10
        ytc2=ytc2-ytic2
        if(abs(ytc2-ymax) .le. yres) go to 135
132     yval=yval-yticl
        if(abs(yval-ymin) .gt. yres) go to 133
          call plot(xlng,0.,2)
          go to 140
133     yp=(yval-ymin)*yscale
        call plot(xp,yp,2)
        if(abs(yval-ytc2) .gt. yres) go to 138
          call plot(t2,yp,2)
135       ytc2=ytc2-ytic2
          go to 139
138     call plot(tl,yp,2)
139     call plot(xp,yp,2)
        if(abs(yval-ymin) .gt. yres) go to 132
c
140     yp=0.
        tl=.05
```

```
      t2=.10
      yln=-2.*sx
      xtc2=xtc2-xtic2
      if(abs(xtc2-xmax) .le. xres) go to 145
142   xval=xval-xtic1
      if(abs(xval-xmin) .le. xres) then
        call plot(0.,0.,2)
        if(abs(xval-xtc2) .le. xres) then
          xln=-xo*(2+nx)
          if(abs(xval) .gt. xres .or. abs(xval) .ge.  10.)
     $      xln=xln-xo*aint(alog10(abs(xval)))
          if(abs(xval) .lt. xres) xval=0.
          if(xval .lt. 0.) xln=xln-xo
          call plot(xln,yln,3)
          call number(xln,yln,sx,xval,0.,nx)
        end if
        call plot(0.,0.,3)
        return
      end if
      xp=(xval-xmin)*xscale
      call plot(xp,yp,2)
      if(abs(xval-xtc2) .gt. xres) go to 148
        call plot(xp,t2,2)
145     xln=xp-xo*(2+nx)
        if(abs(xval) .gt. xres .or. abs(xval) .ge.  10.)
     $    xln=xln-xo*aint(alog10(abs(xval)))
        if(abs(xval) .lt. xres) xval=0.
        if(xval .lt. 0.) xln=xln-xo
        call plot(xln,yln,3)
        call number(xln,yln,sx,xval,0.,nx)
        xtc2=xtc2-xtic2
        go to 149
148   call plot(xp,t1,2)
149   call plot(xp,yp,3)
      if(abs(xval-xmin) .gt. xres) go to 142
c
999   print 100,xlng,xmin,xmax,xtic1,xtic2,ndx,
     $          ylng,ymin,ymax,ytic1,ytic2,ndy
100   format('0Error in BORDR2:'/
     $       '0xlng, xmin, xmax, xtic1, xtic2, ndx = ',1p5e11.3,i5/
     $       '0ylng, ymin, ymax, ytic1, ytic2, ndy = ',1p5e11.3,i5)
      call pltend
      stop
      end
```

```
      subroutine curve(x,y,up,nrpts,xmin,ymin,xinc,yinc,line)
c
c x,y,up must be dimensioned at least nrpts
c xmin,ymin are x,y origin in user units
c xinc,yinc are x,y scales in user units per inch
c
c line=1:  solid
c      2:  long dash
c      3:  medium dash
c      4:  short dash
c      5:  dotted
c      6:  short + long dash
c      7:  short + short + long dash
c
      logical up,up1,up2
      dimension ipen(8),joc(7),x(nrpts),y(nrpts),up(nrpts)
      data ipen/2,2,2,3,2,3,2,3/,joc/18, 11, 14, 23, 32, 41, 16/
      data delr/.1/
c
      if(nrpts .le. 1) go to 99
c
      if(line) 1,2,3
1     kk=mod(line,7)+7
      go to 4
2     kk=0
      go to 4
3     kk=mod(line,7)
4     kk=kk+1
      jo=joc(kk)/10
      jc=joc(kk)-10*jo
      ip=ipen(jo)
c
      j=0
      dr=0.
      rho1=0.
      rho2=delr
      px1=(x(1)-xmin)/xinc
      py1=(y(1)-ymin)/yinc
      up1=up(1)
      if(.not. up1) then
c
c go to first position with pen up
        call plot(px1,py1,3)
        if(kk .eq. 6) then
          px2=(x(2)-xmin)/xinc
          py2=(y(2)-ymin)/yinc
          delx=px2-px1
          dely=py2-py1
          rho=sqrt(delx**2+dely**2)
          if(rho .eq. 0.) then
            dx 6=delx*.1
            dy 6=dely*.1
          else
            dx 6=delx*delr/rho*.1
            dy 6=dely*delr/rho*.1
```

```
        end if
        call plot(px1+dx6,py1+dy6,2)
      end if
    end if
c
    do 40 i=2,nrpts
    px2=(x(i)-xmin)/xinc
    py2=(y(i)-ymin)/yinc
    up2=up(i)
    if(up2) then
      dr=0.
      rho1=0.
      rho2=delr
      go to 39
    end if
    if(up1) then
c pen has been up, prepare to lower pen
      call plot(px2,py2,3)
      go to 39
    end if
    if(kk .eq. 2) go to 38
    delx=px2-px1
    dely=py2-py1
    rho=sqrt(delx**2+dely**2)
    rho1=rho1+rho
    if(rho2 .gt. rho1) go to 38
    delx=delx*delr/rho
    dely=dely*delr/rho
    dx 6=delx*.1
    dy 6=dely*.1
    if(dr .eq. 0.) go to 20
    dx=delx*dr/delr
    dy=dely*dr/delr
    px1=px1+dx
    py1=py1+dy
    go to 21
20  if(rho2 .gt. rho1) go to 38
    px1=px1+delx
    py1=py1+dely
21  call plot(px1,py1,ip)
    if(kk .eq. 6) call plot(px1+dx6,py1+dy6,2)
    j=j+1
    ip=ipen(jo+mod(j,jc))
    rho2=rho2+delr
    go to 20
38  call plot(px2,py2,ip)
    dr=rho2-rho1
39  px1=px2
    py1=py2
    up1=up2
40  continue
99  return
    end
```

# APPENDIX B—PROGRAM LISTING OF "TACAMO"

```
c     TACAMO: TOWLINE CONFIGURATION CODE FOR TOWPLANE IN STEADY STATE
c             CONFIGURATION...towplane in circular orbit...(HUANG,NADC-
c             AM-6849,June 1969)
c
c     v       towplane true airspeed,knots
c     rpl     towplane turn radius(feet) for straight and level flight
c     zpl     towplane density altitude ft.
c     amg     drogue weight,lbs.
c     abase   drogue base area,sq. ft.
c     cddrog  drogue drag coefficient
c     amug    towline weight,lb/ft
c     d       towline diameter,ft.
c     al      towline length,ft.
c     picf    towline skin friction coefficient
c     cd      towline drag coefficient
c     x       distance from towpoint to drogue center of gravity,ft.
c     xcp     distance from towpoint to ddrogue center of pressure,ft.
c     alpha   pitch angle of drogue centerline(radians),positive is nose up
c     cappal  side slip angle in radians,towpoint left positive
c
c
      namelist/datum/v,rpl,zpl,al,rzd,zzd,amg,itrset
c
      dimension dsstor(17)
c
      data g/32.17/,pi/3.1416/,acl/.53/,acm/-.68/,x/1.34/,xcp/2.31/,
     $     abase/5.6/,cddrog/.6/,amug/1.095e-1/,d/1.75e-2/,dds/1.e3/,
     $     picf/2.2e-2/,cd/1.03/,delr/5./,delz/50./
      data dsstor/50.,130.,4*200.,4*300.,4*400.,2*600.,1000./
c
      read datum
      iter=1
      print 1101,v,rpl,zpl,rzd,zzd
      print 1102,amg,abase,cddrog
      print 1103,amug,d,al,dds,picf,cd
      ddrog=sqrt(4.*abase/pi)
      if(cd .le. .1)stop
      k=1
      index=0
    5 iter=iter+1
      if(iter .gt. 30) then
         print 1000
         stop
      endif
16    continue
      zz=zzd
      rz=rzd
      i=1
      ds=20.
      omega=v*1.69/rpl
      m=1
      arg=1.-.006875*zz/1000.
      if(arg .gt. 0.)then
        rhoz=0.002378*arg**4.256
      else
```

```
          print *, 'does not converge'
          stop
        end if
        omegsq=omega*omega
        s=0.
        thz=0.
        qs=.5*rhoz*rz*rz*omegsq*abase
        a=0.5*amg*x/qs+acl*x*.04
        b=(acl+cddrog)*x-ddrog*acm
        c=-0.04*acl*x-amg*x/qs
        alpha=(-b+sqrt(b*b-4.*a*c))*.5/a
        if(alpha .ge. .8) then
           a2=.85*qs*xcp
           b2=-amg*x-.85*pi*qs*xcp
           c2=amg*x*pi*.5-.85*qs*xcp
           alpha=(-b2-sqrt(b2*b2-4.*a2*c2))*.5/a2
           m=2
           am=amg/g
        endif
        al=-amg/g*rz*omegsq-qs*acl*.04
        bl=qs*(ddrog*acm/(x*cos(alpha))-acl-cddrog)
        cappal=al/bl
        if(cappal .le. .04) then
           trpz=-amg/g*rz*omegsq
        else
           trpz=-amg/g*rz*omegsq+acl*(cappal-.04)*qs
        endif
        if(alpha .le. .04) then
           tzpz=amg
           trthpz=0.5*rhoz*abase*cddrog*rz*rz*omegsq
        else
           if(m .eq. 1) then
              tzpz=amg-acl*(alpha-.04)*qs
              trthpz=0.5*rhoz*abase*cddrog*rz*rz*omegsq
           else
              if(m .ne. 2) stop
              tzpz=amg-.85*qs*sin(alpha)**2.*cos(alpha)
              trthpz=.85*qs*sin(alpha)**3.
           endif
        endif
        tz=sqrt(trpz*trpz+trthpz*trthpz+tzpz*tzpz)
        rpz=trpz/tz
        thpz=trthpz/(tz*rz)
        zpz=tzpz/tz
        rthpz=thpz*rz
        if(mod(index,3) .eq. 0 .or. k .eq. 4)then
        print 1
        print 21,s,rz,zz,thz,tz,rpz,zpz,rthpz
        end if
      8 s=s+ds
        z=zz
        t=tz
        r=rz
        rp=rpz
        thp=thpz
```

```
      zp=zpz
      kk=1
   10 za=.5*(z+zz)
      ta=.5*(t+tz)
      ra=.5*(r+rz)
      thpa=.5*(thp+thpz)
      zpa=.5*(zp+zpz)
      rpa=.5*(rp+rpz)
      arg=1.-.006875*za/1000.
      if(arg .gt. 0.)then
        rho=.002378*(1.-.006875*za/1000.)**4.256
      else
        print *, 'does not converge'
        stop
      end if
      thpasq=thpa*thpa
      rasq=ra*ra
      qd=.5*rho*d*rasq*omegsq*cd
      strth=sqrt(1.-rasq*thpasq)
      trp=(ta*ra*thpasq-amug/g*ra*omegsq-qd*ra*rpa*thpa*strth)*ds+trpz
      trthp=(-ta*rpa*thpa+qd*strth**3.+.5*rho*d*picf*rasq*omegsq)*ds+
     $      trthpz
      tzp=(amug-qd*ra*thpa*zpa*strth)*ds+tzpz
      tl=sqrt(trp*trp+trthp*trthp+tzp*tzp)
      rpl=trp/tl
      rl=rz+(rpl+rpz)*.5*ds
      thpl=trthp/(tl*rl)
      zpl=tzp/tl
      zl=zz+.5*(zpl+zpz)*ds
      rthpl=thpl*rl
      thl=thz+(thpz+thpl)*.5*ds
      if(abs((zl-z)/z) .gt. .001) go to 100
      if(abs((rl-r)/r) .gt. .001) go to 100
      if(abs((tl-t)/t) .gt. .001) go to 100
   32 if(1 .le. i .and. i .le. 17) then
         ds=dsstor(i)
      else
         if(i .eq. 18) then
            if(s-al+1.0e-4 .eq. 0.0) then
               ds=dds
            else
               if(s-al+1.0e-4 .lt. 0.0) then
                  go to 50
               else
                  go to 190
               endif
            endif
         else
            print *,' i is not between 1 and 18'
            stop
         endif
      endif
      i=i+1
   50 zz=zl
      rz=rl
```

```
      tz=t1
      rpz=rp1
      thpz=thp1
      zpz=zp1
      thz=th1
      trpz=trp
      trthpz=trthp
      tzpz=tzp
      if(al-s .lt. 1000.)ds=al-s
      if(k .eq. 4)print 21,s,r1,z1,th1,t1,rp1,zp1,rthp1
      go to 8
  100 if(kk .gt. 20) then
         print 3
         go to 32
      endif
      z=z1
      r=r1
      t=t1
      rp=rp1
      thp=thp1
      zp=zp1
      rthp=rthp1
      kk=kk+1
      go to 10
190   continue
      if(mod(index,3) .eq. 0 .or. k .eq. 4)then
      print 21,s,r1,z1,th1,t1,rp1,zp1,rthp1
      end if
      if(iter .gt. itrset)go to 18
      go to(200,203,18,1199),k
  200 if(abs((z1-zp1)/zp1)-.010)202,210,210
  202 go to(204,240),k
  203 if(abs(z1-zp1)-2000.)204,204,210
  204 if(abs((r1-rp1)/rp1)-.003)206,220,220
  206 go to(240,200),k
  210 k=1
      zzd=zzd+.6*(zp1-z1)
      go to 5
  220 if(k .eq. 1) then
         drzd=.16*(rp1-r1)
      else
         if(k .eq. 2) then
            if(abs((r1-rstore)/rp1) .le. .002) drzd=.2*(rp1-r1)
            drzd=(rp1-r1)*drzd*.6/(r1-rstore)
            if(abs(drzd) .gt. 150.) then
               drzd=150.*drzd/abs(drzd)
            else
               if(abs(drzd) .le. 5.) drzd=5.*drzd/abs(drzd)
            endif
         else
            print *,' k is not 1 or 2'
            stop
         endif
      endif
      rzd=rzd+drzd
```

```
        k=2
        rstore=rl
        go to 5
    240 k=3
  18    continue
        if(index .gt. 30)go to 999
        if(istop .eq. 500)go to 1199
        if(mod(index,3) .eq. 0)then
          if(abs((zl-zpl)/zpl) .lt. .001 .and. abs((rl-rpl)/rpl) .lt.
       $      .001)go to 17
          fl=zl-zpl
          f2=rl-rpl
          print *,'fl=',fl
          print *,'f2=',f2
          zlsav=zl
          rlsav=rl
          zzdsav=zzd
          rzdsav=rzd
          zzd=zzd+delz
          index=index+1
          go to 16
        else
          if(mod(index,3) .eq. 1)then
            dzldzd=(zl-zlsav)/delz
            drldzd=(rl-rlsav)/delz
            zzd=zzdsav
            rzd=rzdsav+delr
            index=index+1
            go to 16
          else
            dzldrd=(zl-zlsav)/delr
            drldrd=(rl-rlsav)/delr
            den=dzldzd*drldrd-drldzd*dzldrd
            delzd=(f2*dzldrd-fl*drldrd)/den
            delrd=(fl*drldzd-f2*dzldzd)/den
            zzd=zzdsav+delzd
            rzd=rzdsav+delrd
            index=index+1
            go to 16
          end if
        end if
  17    k=4
        print 1301
        print 1101,v,rpl,zpl,rzd,zzd
        print 1102,amg,abase,cddrog
        print 1103,amug,d,al,dds,picf,cd
        print 1203,alpha,cappal
        istop=500
        go to 16
  1199  stop
  999   print *, 'does not converge'
        stop
  c
      1 format(lh ,5x,lhs,llx,lhr,llx,lhz,llx,2hth,10x,lht,10x,2hrp,10x,
       $       2hzp,9x,4hrthp)
```

```
    2 format(8e12.4)
   21 format(8e12.4)
    3 format(13h not converge)
 1000 format(1h1,14hr1not converge)
 1101 format(1h ,8hv(knot)=,e12.4,2x,4hrpl=,e12.4,2x,4hzpl=,e12.4,2x,
     $       4hrzd=,e12.4,2x,4hzzd=,e12.4)
 1102 format(1h ,4hamg=,e12.4,2x,6habase=,e12.4,2x,7hcddrog=,e12.4)
 1103 format(1h ,5hamug=,e12.4,2x,2hd=,e12.4,2x,3hal=,e12.4,2x,4hdds=,
     $       e12.4,2x,5hpicf=,e12.4,2x,3hcd=,e12.4)
 1203 format(1h ,3x,6halpha=,e12.4,5x,7hcappal=,e12.4)
 1301 format(1h1,5x,3hrpl,9x,5hrdrog,7x,3hzpl,9x,5hzdrog,7x,3hsep,9x,
     $       1ht,11x,4hrppl,8x,4hzppl,8x,4hthpl)
c
      end
```