Final Technical Report IITRI J6176
Contract DAHC-04-69-C-0056

FRAGMENTATION HAZARDS TO UNPROTECTED PERSONNEL

Department of Defense Explosives
Safety Board
Forrestal Building, Room GB270
Washington, D. C. 20314

D. I. Feinstein

Engineering Mechanics Division
IIT Research Institute
10 West 35th Street
Chicago, Illinois 60616

January 1972

Final Report for Period July 1971 to December 1971

Distribution of tnis document is unlimited.

BEST

AVAILABLE

COPY

FOREWORD

Under Contract DAHC-04-69-C-0056 between IIT Research Institute (IITRI) and the U. S. Army Research Office-Durham, IITRI is performing a study of fragmentation hazards to unprotected personnel. This is the final report of the study which was carried out during the period July 1971 to December 1971.

The work was conducted for the Department of Defense Explosives Safety Board and supervised by Col. W. Cameron III, Chairman, Lt. Col. J. D. Coder, Project Manager, Dr. T. A. Zaker, Explosives Scientist and Mr. R. G. Perkins, Safety Engineer. Principal contributors to the presently reported work include L. A. C. Barbarek, D. I. Feinstein, H. H. Nagaoka, J. D. Rouse and J. R. Wingfield. Special acknowledgement is made to Dr. Zaker for providing trajectory algorithms and overall technical direction to this study.

Respectfully submitted,
IIT RESEARCH INSTITUTE

D. I. Feinstein
Research Engineer

APPROVED:

E. P. Bergmann, Manager
Civil Engineering Systems
and Explosives

Unclassified
Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) IIT Research Institute 10 West 35th Street Chicago, Illinois 60616	2a. REPORT SECURITY CLASSIFICATION 2b. GROUP N/A

3. REPORT TITLE
FRAGMENTATION HAZARDS TO UNPROTECTED PERSONNEL

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)
Final Report (July 1971 to December 1971)

5. AUTHOR(S) (First name, middle initial, last name)
D. I. Feinstein

6. REPORT DATE January 1972	7a. TOTAL NO. OF PAGES 202	7b. NO. OF REFS 5
8a. CONTRACT OR GRANT NO. DAHC-04-69-C-0056 b. PROJECT NO. c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) J-6176 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)	

10. DISTRIBUTION STATEMENT
Distribution of this document is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY Dept. of Defense Explosives Safety Bd. Forrestal Building, Room GB270 Washington, D. C. 20314

13. ABSTRACT A computer program has been developed and documented that generates the information necessary to establish minimum separation distances between various munition types and personnel in order to mitigate fragment hazards. This information includes the fragment density at any point from an accidental detonation and the probability of injury to personnel within the hazardous area.

Published munition effectiveness data have been modified, utilizing statistical techniques, to emphasize a conservative approach, with respect to safety, by a better resolution of heavier fragments. The modified munition effectiveness data, which is input to the computer program have been included as a technical data appendix.

Computed results have been obtained for seven munitions and are presented in the form of contours of constant fragment density and damage probability in the horizontal plane surrounding the munition of interest. In addition, these contours have been simplified to yield an average value of the hazard as a function of range from the accidental explosion.

DD FORM 1 NOV 65 1473

UNCLASSIFIED
Security Classification

ABSTRACT

A computer program has been developed and documented that generates the information necessary to establish minimum separation distances between various munition types and personnel in order to mitigate fragment hazards. This information includes the fragment density at any point from an accidental detonation and the probability of injury to personnel within the hazardous area.

Published munition effectiveness data have been modified, utilizing statistical techniques, to emphasize a conservative approach, with respect to safety, by a better resolution of heavier fragments. The modified munition effectiveness data, which is input to the computer program, have been included as a technical data appendix.

Computed results have been obtained for seven munitions and are presented in the form of contours of constant fragment density and damage probability in the horizontal plane surrounding the munition of interest. In addition, these contours have been simplified to yield an average value of the hazard as a function of range from the accidental explosion.

CONTENTS

Section		Page
1	INTRODUCTION	1
	1.1 Problem Background	2
	1.2 Program Objectives	3
	1.3 Program Accomplishments	4
	1.4 Program Highlights	4
2	THE FRAGMENT HAZARD MODEL	5
	2.1 Trajectory Analysis	7
	2.2 Fragment Density Computation	12
	2.3 Target Damage Probability	17
	2.4 Description of Model Output	19
3	MUNITION EFFECTIVENESS DATA	20
	3.1 Documentation of Original Fragmentation Arena Data	20
	3.2 Revision of Munition Effectiveness Data	25
	3.3 Model Sensitivity to Input Data	37
4	RESULTS, CONCLUSIONS, AND RECOMMENDATIONS	44
	4.1 Results	44
	4.2 Conclusions	46
	4.3 Recommendations	46
REFERENCES		48
APPENDIX		
A	Program User's Manual	A-1
B	Contours of Fragment Number Densities and Injury Probabilities	B-1
C	Average Curves of Fragment Number Densities and Injury Probabilities	C-1
D	Computer Program Listings	D-1
E	Munitions Effectiveness Data (Classified-Published Separately)	

ILLUSTRATIONS

Figure		Page
1	Flow Chart Fragment Hazard Model	6
2	Coordinate Systems and Trajectory Geometry	8
3	Fragment Source Geometry	14
4	Fragmentation Arena Diagram (500 Lb MK 82-1 Bomb)	22
5	Cumulative Distribution of Mass for the 155 MM Munition	27
6	Distribution of Mass in the Category 200 - 250 Grains for the 155 MM Munition	29
7	Average Polar Zone Frequency Versus Mass Category	30
8	Cumulative Distribution for 500 LB Bomb	34
9	Cumulative Distribution for 750 LB Bomb	36
10	Cumulative Distribution of Mass for the 5 in. Munitions	38
11	Cumulative Distributions of Mass for the 8 in. Munitions	39
12	Cumulative Distribution of Mass for the 175 MM Munitions	40
13	175 MM Gun Shell M437A2 (Comp B Load) (Unaltered Data)	41
14	175 MM Gun Shell M437A2 (Comp. B Load) (Altered Data)	42
B-1	500 LB Low Drag Bomb Mark 82 Mod 1 (H-6 Load)	B-2
B-2	750 LB Bomb M117A2 (Tritional Load)	B-3
B-3	105 MM Howitzer Shell M1 (Comp B Load)	B-4
B-4	155 MM Howitzer Shell M107 (Comp B Load)	B-5
B-5	175 MM Gun Shell M437A2 (Comp B Load)	B-6
B-6	5"/38 Projectile Mark 49 (Comp A3 Load)	B-7
B-7	8"/55 Projectile Mark 25 (Explosive D Load)	B-8
C-1	500 Lb Low Drag Bomb Mark 82 Mod 1 (H-6 Load)	C-2
C-2	750 Lb Bomb M117A2 (Tritional Load)	C-3
C-3	105 MM Howitzer Shell M1 (Comp B Load)	C-4
C-4	155 MM Howitzer Shell M107 (Comp B Load)	C-5
C-5	175 MM Gun Shell M437A2 (Comp. B Load)	C-6
C-6	8"/55 Projectile Mark 25 (Explosive D Load)	C-7
C-7	8"/55 Projectile Mark 25 (Explosive D Load)	C-8

TABLES

Table		Page
1	Typical Format of Experimental Data for Munition of Interest	13
2	Estimated Fragment Sample Size Data for Given Arena Tests	23
3	Data Summary for Fragments Greater than 150 Grains	24
4	Summary Table for Three Principal Directions at 1 Hit per 600 sq ft	45

FRAGMENTATION HAZARDS TO UNPROTECTED PERSONNEL

1. INTRODUCTION

Existing quantity-distance standards for the manufacture, handling, and storage of munitions are based on the net weight of explosive filler contained in the devices in an unsubdivided magazine, or operating building unit. Safe distances are prescribed in tabular form essentially proportional to the cube root of the explosive weight.

Because peak blast pressures from explosions of different yield are the same at distances scaled by the cube root of the respective explosive weights, existing standards imply that the acceptable risk to a given target is based on a peak blast overpressure criterion alone. On the other hand, the field of fragments projected to the far field from accidental explosion of a munition store, consisting of inert munition component fragments and secondary fragments from any enclosure, does not satisfy the same similarity rules as does the airblast. Thus, defining an acceptable blast overpressure level at a target implies the acceptance of different levels of risk of damage by fragments, depending on the quantity and composition of the munition store and its enclosure.

To develop quantity-distance standards based on consistent blast and fragment hazard levels requires determination of the damage risk due to fragments from accidental explosions as a function of the quantity and type of munitions, and of the characteristics of the source environment and the vulnerability of the target. A previous report (Ref. 1),* was a first attempt aimed at applying engineering analysis, supplementary experimental efforts, and currently available data on fragmentation and damage criteria to the problem of estimating fragment hazards at explosives manufacturing and storage sites. While that report resulted in a fragment hazard model and a set of fragment density and damage probability maps for a variety of weapons and targets, this report is concerned with:

*References listed at the end of the text.

(1) Refining that model and making its corresponding computer algorithm usable by personnel moderately experienced at ballistic calculations,

(2) Revising near-field fragment mass distributions to more accurately reflect their use as input in estimating fragment hazards,

(3) Developing a new set of fragment density and damage probability maps; studying the effect of specific personnel protection criteria for unprotected personnel, and

(4) Developing simplified two-dimensional relationships of fragment density and damage probability as a function of range.

1.1 Problem Background

Under Contract No. DAHC-04-69-C-0056 with the U.S. Army Research Office-Durham, IITRI has been conducting a series of investigations concerning fragment hazards associated with accidental detonation of munitions. This work has been performed under the direction of the Department of Defense Explosive Safety Board.

Phase I of this study was concerned with establishing quantitative damage criteria in terms of fragment mass, velocity, and attack angle for various targets including standing personnel, vehicles, aircraft, buildings and open weapon stores. In Phase II an analytical model was developed to predict the density of fragments and the probability of damage to the targets considered in Phase I from explosion of individual munitions of various types. These included gun projectiles and general-purpose bombs. Here damage probability contours were obtained in polar coordinates for a horizontal orientation of the munition axis in each case. Phase III attempted to extend the fragment hazard model for individual munitions to the case of multiple munitions in open stores (Ref. 2). The result was a limited demonstration that an analytic model could be developed to describe the initial fragment field of a stack of munitions. However, it was also brought out that this initial fragment field was often related to munition case design, stack configuration and mode of initiation.

1.2 Program Objectives

In the current research activity the intent has been to develop an analytic tool, usable by personnel moderately experienced at ballistic calculations, and capable of generating the information necessary to establish the minimum separation distance to personnel. The objectives of the study were:

(1) To develop and document a computer algorithm which uses munition effectiveness tables as input and computes fragment densities and damage probabilities to unprotected personnel based upon the following criteria:

- A hazardous fragment has a kinetic energy of 58 ft-lbs or greater, and
- An acceptable density of hazardous fragments is not more than one per 600 sq ft.

(2) To develop a rational scheme for revising published munition effectiveness data to more accurately reflect its use as input in estimating fragment hazards.

(3) To develop a simplified means of relating fragment density and damage probability to a radial distance over a fixed sector of the ground plane.

(4) To utilize the computer algorithm and the revised data in order to compute the fragment density and damage probability from the explosion of a single round of each of the following seven munitions:

- 500 lb low drag bomb Mark 82 Mod 1 (H-6 load)
- 750 lb Bomb M117A2 (Tritional Load)
- 105mm Howitzer Shell M1 (Composition B Load)
- 155mm Howitzer Shell M107 (Composition B Load)
- 175mm Gun Shell M437A2 (Composition B Load)
- 5"/38 Projectile Mark 49 (CCMP A-3 Load)
- 8"/55 Projectile Mark 25 (Explosive D Load)

1.3 Program Accomplishments

The major result of this study has been the development and documentation of a computer algorithm which will allow safety personnel, moderately experienced at ballistic calculations, to make the computations necessary to establish separation distances for personnel due to fragment hazards. A rational statistical scheme has also been developed and demonstrated which revises published munition effectiveness data to more conservatively reflect its use in estimating fragment hazards. A set of computations were made for seven munitions and the results are presented as contour maps of total fragment number densities, damaging fragment number densities and injury probabilities. The contour maps have also been simplified to yield curves of number density and injury probability as a function of radial distance from the munition source within a constant sector of the ground plane. (i.e., the nose, base and side-spray sectors) These results serve both to demonstrate the capability of the computer algorithm and also as design aids for explosive safety personnel.

1.4 Program Highlights

The following sections of this report are organized in such a way as to first present the analysis on which the fragment hazard computer model is based, to next present an analysis of the munition effectiveness data which is input to that computer model and to finally summarize the study in the form of conclusions reached and recommendations as to model improvement.

Appendices, covering the study's deliverable items, are included following the conclusion of the report. These appendices include a user manual for operation of the fragment hazard computer program, a listing of the computer program, the contours and simplified curves describing the fragment hazard associated with unprotected personnel exposed to these

seven munitions. The revised munition effectiveness data for the seven munitions were published as a separate classified document.

2. THE FRAGMENT HAZARD MODEL

This section describes the analysis which forms the basis for the computational model for predicting fragment density and injury probability contours for various munitions The mathematical model, illustrated in Fig. 1, is limited to the consideration of the single munition without environmenta protection. The model has been programmed for digital computa tion, is modular and accepts as inputs:

(1) The spatial distribution of fragment masses and velocities for single munitions, which are defined for intervals of polar angle. (i.e., munition effectiveness tables)

(2) The k-factors for the individual munitions, which express the relationship between fragment masses and projected areas for various munition types.

(3) Vulnerability criteria for targets of interest. (e.g., kinetic energy and critical densities)

Output of the model includes:

(1) Fragment density contours showing distances to isodensity lines for all azimuths. Contours can be printed for all fragments or for various classes of fragments.

(2) Injury/damage probability contours, showing ground distances to isoprobability curves at all azimuths for various munition/target combinations.

Elements of this model include the following steps:

(1) Munition performance data is converted to an internal form. All data is normalized into a a common internal form, wherein the average fragment velocity and the average mass and number density for each of several fragment mass categories are represented as tabular functions defined everywhere on the surface of a sphere of unit radius located at the origin. Gaps in the original data are filled in, and in addition the data may be smoothed if this seems desirable.

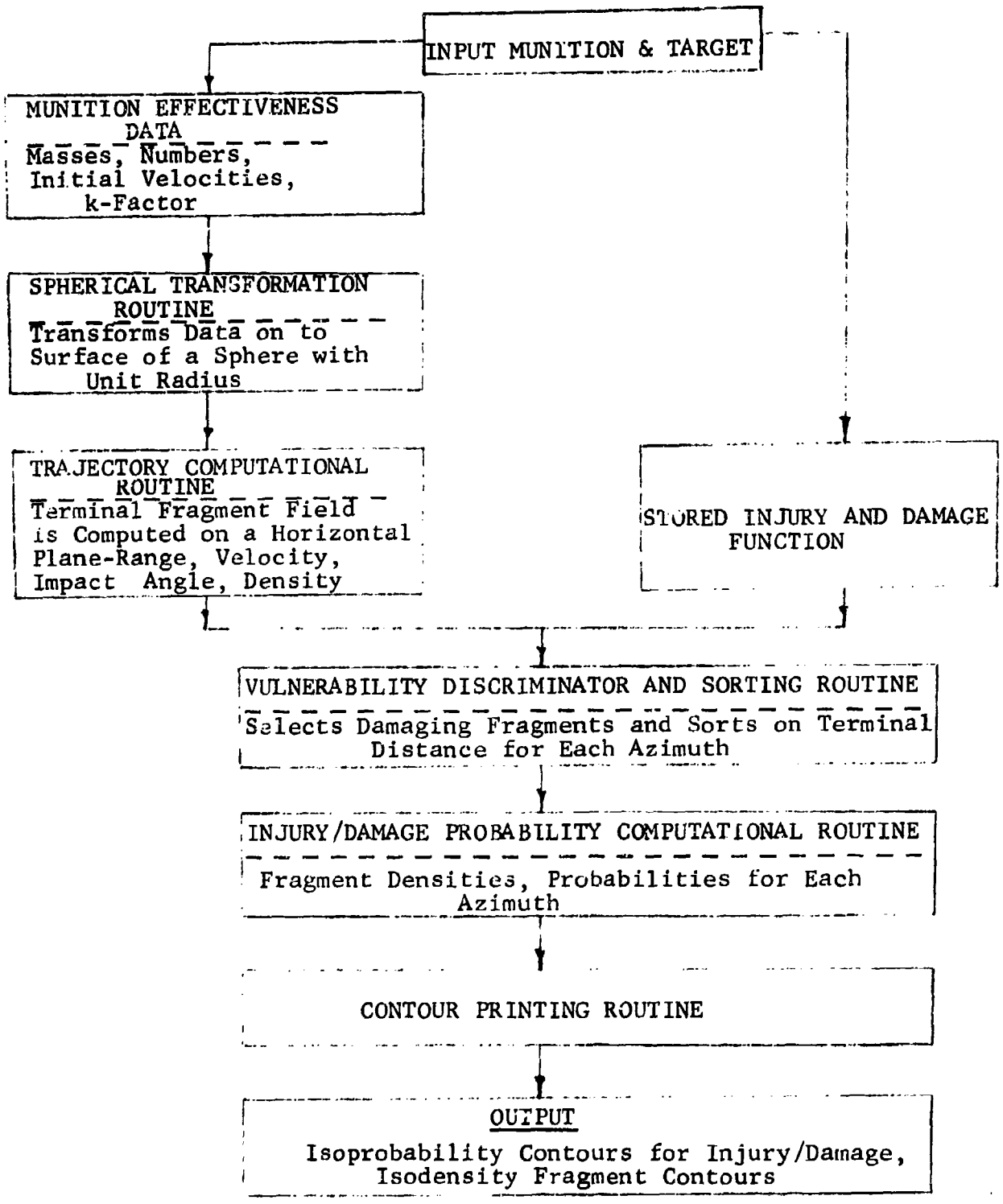


Fig. 1 FLOW CHART FRAGMENT HAZARD MODEL

(2) A terminal fragment field is computed on a selected horizontal plane (i.e., the plane is above, below or passing through the origin) utilizing routines for evaluating fragment trajectories. The quantities computed for each mass category are impact velocity, impact angle and number of fragments per sq ft in a plane normal to the impact angle. These quantities are expressed as tabular functions of range and azimuth for output processing.

(3) Selected functions of the fragment field quantities computed in step 2 are computed and plotted in this step. A wide range of functions are available. The basic functions involve the fragment field alone, and do not consider characteristics of a target. Examples of these functions are number of fragments per sq ft, and total fragment kinetic energy per sq ft.

The target functions use target characteristics to determine the number of damaging fragments and the probability of damage at every point in the field. These functions use tables or formulas to determine whether or not a fragment is damaging, by finding the minimum velocity required by that fragment to damage the target. The fragment is considered to be damaging if its velocity exceeds the threshold.

(4) Fragment field functions may be plotted either on the printer, or on an off-line plotter if the output tape is run through an appropriate post-processor.

2.1 Trajectory Analysis

Large quantities of terminal ballistic property data are used in developing the outputs of the computational model described above. These data are generated from the equations of motion for the fragments. Since these computations represent the bulk of the computational burden involved in exercising the model and their accurate evaluation is essential, it is desirable to utilize a highly efficient numerical procedure for calculating trajectories.

Figure 2 illustrates the motion of a fragment moving under the influence of aerodynamic drag and gravity forces in nonrotating local coordinates $\bar{x}$, $\bar{y}$ tangent and normal to the trajectory. The corresponding equations of motion are:

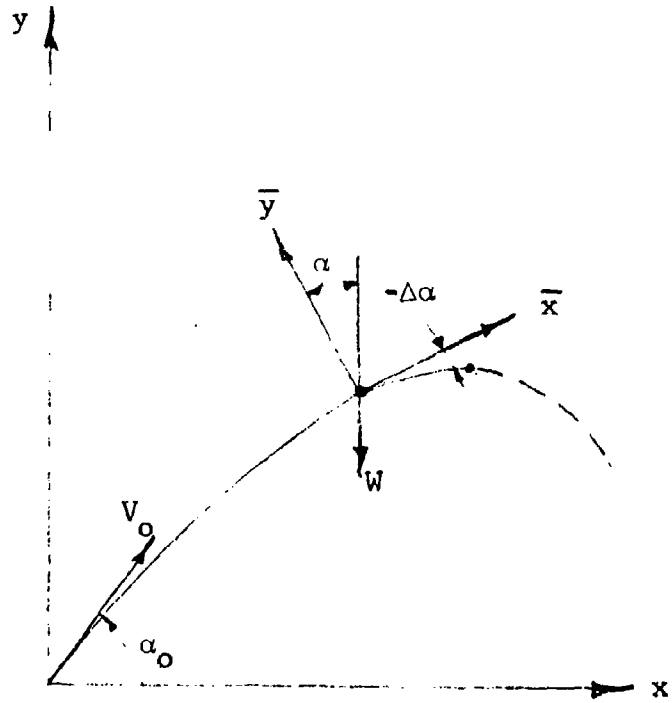


Fig. 2 COORDINATE SYSTEMS AND TRAJECTORY GEOMETRY

$$\ddot{\bar{x}} + \beta v \dot{\bar{x}} + g \sin\alpha = 0 \qquad (1)$$

$$\ddot{\bar{y}} + \beta v \dot{\bar{y}} + g \cos\alpha = 0 \qquad (2)$$

where dots denote differentiation with respect to time t. In these equations, g is the acceleration of gravity, v is the speed in the path, and α is the angle between the $\bar{x}$ - axis and the horizontal. Instantaneously we have $\dot{\bar{x}} = v$ and $\dot{\bar{y}} = 0$.

The aerodynamic coefficient β is given by

$$\beta = C_D w\ A/2W \qquad (3)$$

where C_D is the drag coefficient, w is the specific weight of air, A is the cross-sectional area of the fragment normal to the flight direction, and W is the fragment weight. The fragment area and weight are related empirically (Ref. 3) through a ballistic density k as follows

$$W = k\ A^{3/2} \qquad (4)$$

In terms of k, the aerodynamic coefficient becomes

$$\beta = C_D\ w/2(k^2W)^{1/3} \qquad (5)$$

An approximate local solution to the equations of motion is obtained by separating the displacement into two parts, one a basic solution satisfying the local initial conditions and the equations of motion with gravity absent, and the other a pair of perturbations satisfying the linearized residual equations (Ref. 4). The results, applicable for small departures of the trajectory from the local initial tangent, are equivalent to difference equations appropriate to an arbitrary time step in a numerical integration of the complete trajectory.

The displacement $\bar{x}$ is assumed to be of the form

$$\bar{x} = \bar{x}_o + \bar{x}_p \qquad (6)$$

where the basic solution $\bar{x}_o$ satisfies

$$\ddot{\bar{x}}_o + \beta\dot{\bar{x}}_o^2 = 0 \tag{7}$$

and the initial condition $\dot{\bar{x}}_o = v$, while the perturbation $\bar{x}_p$ satisfies the associated residual of Eq. (1).

The drag coefficient C_D in general depends on the Mach number, and the atmospheric weight density w is a function of altitude. If both of these factors are assumed to be constant during the time interval of interest, however, the aerodynamic coefficient β is a constant and Eq. (7) is easily integrated. The results are

$$\bar{x}_o = [\log (1+u)] /\beta \tag{8}$$

$$\dot{\bar{x}}_o = v_o/(1+u) \tag{9}$$

where

$$u \equiv \beta v_o t \tag{10}$$

In these expressions, t is measured from the time at which the fragment is at the local coordinate origin in Fig. 1, and v_o is the value of v at that time.

Substituting Eq. (6) and (7) into the equations of motion, expanding v in binomial series, and neglecting terms of second order and higher in $\dot{\bar{x}}_p$ and $\dot{\bar{y}}$, we reach the following results:

$$\ddot{\bar{x}}_p + 2\beta\dot{\bar{x}}_o\dot{\bar{x}}_p + g \sin\alpha = 0 \tag{11}$$

$$\ddot{\bar{y}} + \beta\dot{\bar{x}}_o\dot{\bar{y}} + g \cos\alpha = 0 \tag{12}$$

These equations are linear in the displacement perturbations $\bar{x}_p$ and $\bar{y}$, and can be integrated analytically by standard methods. The displacement and velocity perturbations are

$$\bar{x}_p = -(g/2)\ t^2 \sin\alpha\ (1+u/3)/(1+u) \tag{13}$$

$$\bar{y} = -(g/2)\ t^2 \cos\alpha\,[\,u(1+u/2)-\log(1+u)]/u^2 \tag{14}$$

$$\dot{\bar{x}}_p = -gt \sin\alpha\,[\,1+u(1+u/3)\,]/(1+u)^2 \tag{15}$$

$$\dot{\bar{y}} = -gt \cos\alpha\ (1+u/2)/(1+u) \tag{16}$$

where u is defined as before by Eq. (10).

The leading factors on the right in the foregoing equations express the position and velocity changes due to gravity in the elementary case of a drag-free trajectory. The multipliers containing u all approach unity as u vanishes, and can be viewed as corrections on the effect of gravity due to drag.

The drag coefficient and atmospheric density are assumed to be constant during each time step at their values at the beginning of the step. The method is self-starting in that the position and velocity changes are computed from initial values at the current step only.

Initial values $v = V_o$ and $\alpha = \alpha_o$ are assumed to be given at the fixed coordinate origin in Fig. 2. Equations (8), (9), and (13) through (16) give directly the displacement and velocity components after a typical time step t in the local coordinates. With respect to the fixed coordinates, the displacements during the time step are obtained from the relations

$$\Delta x = \bar{x} \cos\alpha - \bar{y} \sin\alpha \tag{17}$$

$$\Delta y = \bar{x} \sin\alpha + \bar{y} \cos\alpha \tag{18}$$

while the rotation of the trajectory tangent is given by

$$\Delta\alpha = \tan^{-1} (\dot{\bar{y}}/\dot{\bar{x}}) \tag{19}$$

For low register trajectories, (i.e., those launched at angles less than that corresponding to maximum range) the above analytic perturbation equations with $\alpha = \alpha_o$ furnish an approximate solution for the complete trajectory in one time step t, the total time of flight. At impact, the expressions for the position coordinates x and y are of the form:

$$x = (\bar{x}_o + \bar{x}_p) \cos\alpha_o - \bar{y} \sin\alpha_o \tag{20}$$

$$y = (\bar{x}_o + \bar{x}_p) \sin\alpha_o + \bar{y} \cos\alpha_o = 0 \tag{21}$$

where $\bar{x}_o$, $\bar{x}_p$, and $\bar{y}$ are given by Eq. (8), (13), and (14).

2.2 Fragment Density Computation

Munition effectiveness data are used to derive initial conditions for the ballistic trajectories of fragments. These data are in the form of initial velocity and number of fragments, in each of several mass categories, and are functions of polar angle measured from the nose of the munition. The resulting fragment density at any point of interest can be computed deterministically from the known terminal points of fragments in all the mass intervals in each polar zone.

An individual munition is regarded as a nonisotropic point source of fragments which is rotationally symmetric about its longitudinal (nose-to-base) axis. Thus, the properties of the fragments emitted by a single munition are functions of polar angle θ measured from the nose. The format of typical munition effectiveness data (Ref. 5) is shown in Table 1. Fragments in all mass intervals are assumed to be emitted from a given polar zone at the same velocity.

A single munition is assumed to be detonated at a level ground surface, with the munition axis horizontal. In order to determine the probability of damage to targets at various distances and directions from the source, it is first necessary to calculate the number densities of fragments of different terminal ballistic properties in the field surrounding the source.

The fragments from a munition, considered as a point source on the ground, may be regarded as originating from positions on a hemispherical envelope enclosing the source. Each point of origin on the hemisphere defines an elevation angle α_o and an azimuth ϕ in the horizontal plane measured from the nose of the munition.

In Fig. 3, let u, v, and w be rectangular coordinate axes centered at the source, with w the vertical direction and the nose of the munition in the positive v-direction. Let R_o be the (arbitrary) hemisphere radius, and θ be the polar angle measured from the nose of the munition.

TABLE 1

TYPICAL FORMAT OF EXPERIMENTAL DATA FOR MUNITION OF INTEREST

Mass Intervals (grains) / Polar Zone (degrees)	$o - m_1$		$m_1 - m_2$		. . . .	$m_8 - m_9$		Above m_9		Total Number of Fragments	Initial Velocity
	W_{AV}	$\bar{N}$	W_{AV}	$\bar{N}$		W_{AV}	$\bar{N}$	W_{AV}	$\bar{N}$		
0 - 5	-	-	-	-	. . . .	-	-	-	-	-	-
5 - 10	-	-	-	-	. . . .	-	-	-	-	-	-
10 - 15	-	-	-	-	. . . .	-	-	-	-	-	-
.	.	.	.	.	. . . .	.	.	.	.	.	.
.	.	.	.	.	. . . .	.	.	.	.	.	.
.	.	.	.	.	. . . .	.	.	.	.	.	.
.	.	.	.	.	. . . .	.	.	.	.	.	.
170 - 175	-	-	-	-	. . . .	-	-	-	-	-	-
175 - 180	-	-	-	-	. . . .	-	-	-	-	-	-

Where: W_{AV} is average weight, $\bar{N}$ is average number of fragments.

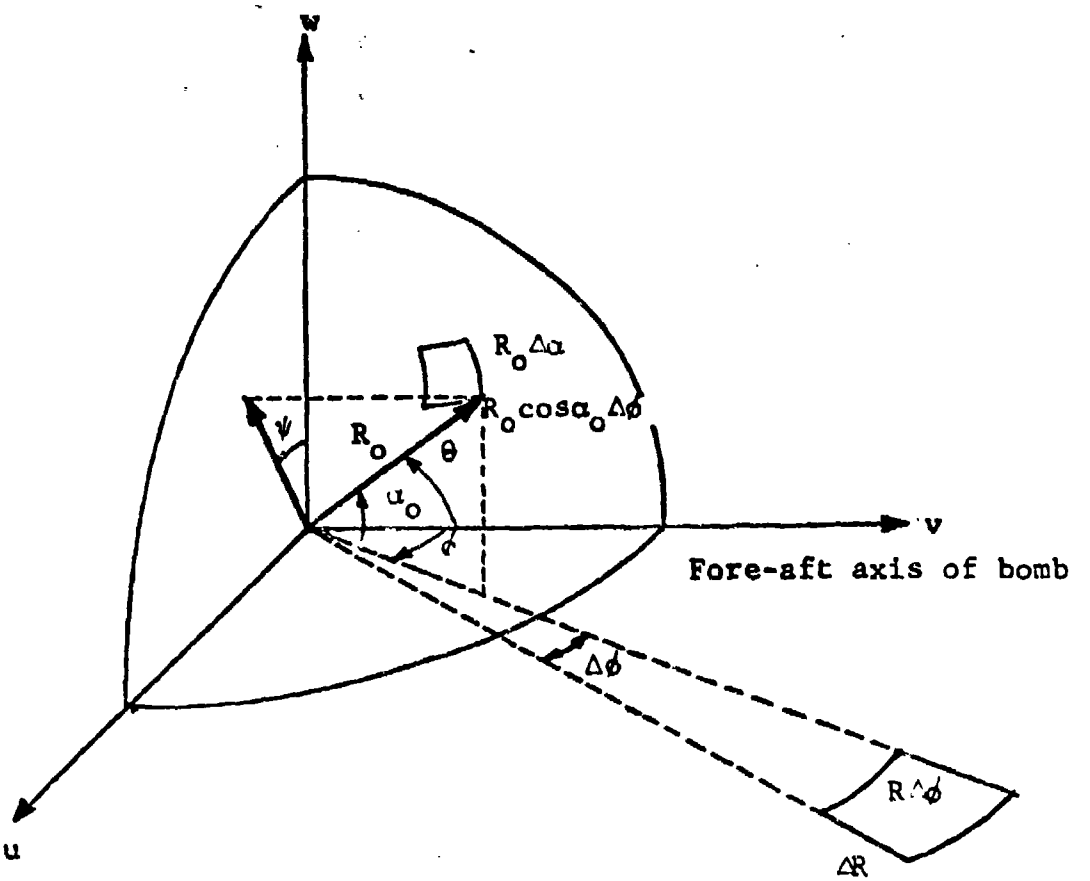


Fig. 3 FRAGMENT SOURCE GEOMETRY

Transformation formulas between the spherical coordinates (θ, ψ) with the munition nose at the pole and the spherical coordinates (α_o, ϕ) with the pole vertical give the following relations:

$$\cos \theta = \cos \alpha_o \cos \phi \tag{22}$$

$$\cos \psi = \sin \alpha_o / \sin \theta \tag{23}$$

Fragments on ballistic trajectories in the absence of wind travel in planes of constant azimuth ϕ.

For an individual munition, the initial fragment properties are functions of polar angle θ only. Therefore, for given elevation angle α_o and azimuth ϕ, the associated polar angle can be computed from Eq. (22) and all initial fragment properties at that point on the hemisphere can be obtained from munition performance data.

Now consider the fragments of a single mass interval, all having the same average ballistic properties, emitted from a particular munition at a point (α_o, ϕ) on the enclosing hemisphere. For simplicity, consider target points in the ground plane of Fig. 3. The terminal point corresponding to this initial point on the hemisphere is uniquely determined for each mass category through the associated ballistic trajectory.

The family of trajectories for fragments of a given average mass (i.e., those in one mass interval) can be thought of as a mapping of points on the hemisphere into points on the ground plane. Fragments of one mass interval originating from an element of area $R_o^2 \cos\alpha_o \Delta\phi \Delta\alpha_o$ on the hemisphere are projected into the element of area $R\Delta\phi\Delta R$ on the ground plane. Thus if n_i^o is the number density of fragments in mass interval i at the source hemisphere and n_i is the number density at the terminal point R, we have

$$n_i = n_i^o \frac{R_o^2 \cos \alpha_o}{R|dR/d\alpha_o|} \tag{24}$$

This formula permits computation of the number density of fragments in each mass interval originating from a point on the source hemisphere at any azimuth ϕ, since for given initial ballistic properties the terminal point R is known from trajectory calculations. To calculate the number density across a plane normal to the final trajectory, equation (24) is replaced by

$$n_i = n_i^o \frac{R_o^2 \cos \alpha_o}{R\left|-\sin \alpha_f \frac{dR}{d\alpha_o}\right|} \tag{25}$$

where α_f is the final trajectory elevation angle.

For an individual munition the mapping derivative $dR/d\alpha_o$ is given by

$$\frac{dR}{d\alpha_o} = \frac{\partial R}{\partial \alpha_o} + \frac{\partial R}{\partial V_o} \frac{\partial \theta}{\partial \alpha_o} \frac{dV_o}{d\theta} \tag{26}$$

where V_o is the initial velocity. From Eq. (22) we have

$$\frac{\partial \theta}{\partial \alpha_o} = \sin \alpha_o \cos \phi / \sin \theta \tag{27}$$

The partial derivatives $\partial R/\partial \alpha_o$ and $\partial R/\partial V_o$ can be obtained analytically for lower register fragments; however, for upper register fragments, they must be obtained by numerical differentiation of a precalculated ballistic file. The analytic expressions for the lower register are

$$\frac{\partial R}{\partial \alpha_o} = \frac{2R - \bar{x}_o \cos\alpha_o}{\tan 2\alpha_o} - \frac{\bar{x}_o \cos \alpha_o}{\sin 2\alpha_o} - \frac{2R - \bar{x}_o \cos\alpha_o}{\tan \alpha_f} \tag{28}$$

$$\frac{\partial R}{\partial V_o} = -2(R - \bar{x}_o \cos\alpha_o + \bar{x}_o \sin\alpha_o / \tan\alpha_f)/V_o \tag{29}$$

The derivative $dV_o/d\theta$ is determined from munition effectiveness data.

The computer model, moving along rays of azimuth in the horizontal plane, determines the terminal properties of lower register fragments at preselected radial increments of range

out to maximum range. At maximum range the corresponding elevation angle is noted and the remaining elevation angle up to 90 degrees is divided into equal increments. The model, utilizing the multi-step trajectory routine with a fixed drag coefficient of 1.28, calculates the terminal properties of the upper register fragments corresponding to each increment of elevation angle. The fragment number density can be determined exactly at prescribed terminal points for lower register fragments and, by interpolation, at these same points for the upper register fragments. These contributions are accumulated to give the total fragment density at the preselected radial distances. By symmetry it is only necessary to perform the analysis in half the horizontal plane to one side of the munition axis.

For a particular target, nondamaging fragments based on the applicable target vulnerability criterion are excluded from the accumulated number density. The number densities so calculated represent, in a statistical sense, expected values of the numbers of impacts per unit area. That is, these results, calculated deterministically as a function of position in the horizontal plane, are the numbers of damaging impacts per unit area to be expected on the average. To calculate the associated probability of damage (i.e., impact by one or more damaging fragments) to a particular target requires a statistical representation of the process, incorporating these expected values of fragment number density. In this study the target is a sphere of unit cross sectional area and fragments are considered to intersect this target across a plane normal to the final trajectory direction.

2.3 Target Damage Probability

Calculation of the probability of damage to a particular target requires a statistical representation of the process of impact by damaging fragments, involving the target area and the expected number of impacts per unit area. The number density of fragments is a function of position of the spherical target in the horizontal plane of interest.

Consider a target of projected area A_i normal to the trajectory of fragments of mass interval i at the associated terminal point. Let $\bar{n}_i$ be the expected number of impacts per unit area of fragments satisfying the applicable target vulnerability criterion. The impact process is assumed to be uniformly random in the immediate neighborhood of the point of interest. That is, impact by a damaging fragment is equally likely on all equal elements of area in the vicinity of the point. Statistically, we then have what is termed a Poisson process. The probability $p_i(j)$ of exactly j impacts on the target by damaging fragments of mass interval i is given by

$$p_i(j) = \frac{(\bar{n}_i A_i)^j \exp(-\bar{n}_i A_i)}{j!} \tag{30}$$

The probability of impact by none of the damaging fragments of mass interval i is therefore

$$p_i(0) = \exp(-\bar{n}_i A_i) \tag{31}$$

The target is assumed to be damaged if struck by one or more damaging fragments of any of the mass intervals. On this basis, the probability of damage to the target, q, is given by

$$q = 1 - \exp(-\Sigma\, \bar{n}_i A_i) \tag{32}$$

This last formula permits direct computation of the probability of damage to a given target at every point of the horizontal plane of interest from the accumulated number densities of damaging fragments and the associated projected areas. The target in this study was a standing man. The man was considered to present a projected area varying from 1.33 sq ft in plan to 9.0 sq ft frontal area. A_i was then calculated as the sum of the projections of both a vertical and a horizontal target on a plane normal to the final trajectory direction. The vertical target had an area of 9.0 sq ft and the horizontal target had an area of 1.33 sq ft.

2.4 Description of Model Output

Contour maps, describing each of the seven munitions of interest, listed in Section 1.2, were produced utilizing the computer model and are included as Appendix B. For each munition there are three contour maps: one describing the fragment flux (i.e., target hits per sq ft) due to all fragments; another describing the fragment flux of all fragments with terminal energy exceeding 58 ft-lbs; and a third which presents the probability of serious injury to standing personnel computed as described in the previous section.

Another set of figures is shown in Appendix C. Here again, there are three sets of figures per munition. These are simplified curves, derived from the corresponding contour maps, and represent fragment density and damage probability in certain sectors of the ground plane as functions of radial distance only. There are three of these fixed sectors: that is, 10-degree sectors in the base and nose of the munition and a third sector representing a peak side-spray. This third sector varied somewhat among all of the munitions, but was usually defined by a line of peak values appearing within $\pm$ 10 degrees of the 90-degree azimuth. This line of peak values did not usually include the outermost or lowest value contour. The average value of this contour in the side spray sector was determined by inspection from the maps.

Both the contours and the corresponding curves represent the fragment field from a minimum range of 250 ft out to the prescribed limit of effect (i.e., one hit per 6000 sq ft for fragment densities and a probability value of 0.0001). The minimum range is a function of a prescribed input variable and was chosen as 250 ft in this study. Since the results represent a single unit of munition and it is anticipated that future use of these results may be directed at multiple unit stacks, it seems reasonable to assume that the far fields results will be of more interest and the minimum range of 250 ft will be quite sufficient.

3. MUNITION EFFECTIVENESS DATA

Previous studies (Ref. 2) have indicated published munition effectiveness data, in their present form, are not suitable for far-field fragment hazard analysis. In general, these data place heavier fragments into one or two broad weight intervals with a corresponding average weight. Since the heavier fragments travel greater distances and their resulting terminal kinetic energy effect is high, it is necessary that these fragments be resolved into an adequate number of weight intervals.

This section of the report describes several tasks which were undertaken to provide a systematic approach to both illustrating the need for and subsequently the methods utilized in revising munition effectiveness tables.

3.1 Documentation of Original Fragmentation Arena Data

In collaboration with the Ballistics Research Laboratory at Aberdeen Proving Grounds, the fragmentation arena test procedure and data analysis techniques used to obtain munition effectiveness data were reviewed. Arrangements were made to examine the original arena test firing records at the APG Technical Library for the following munitions:

- 105 mm Howitzer Projectile M1
- 155 mm Howitzer Projectile M107
- 175 mm Gun Projectile M437A2
- 750 lb General Purpose Bomb M117A2 (APG Data)

Subsequently, contacts were established at NWL to obtain similar arena test data for the following munitions.

- 5-in/38 Projectile Mark 49 Mod 0 (VT)
- 8 in/55 Projectile Mark 25 Mod 1 (HC)
- 500 lb Low Drag Bomb Mark 82 Mod 1
- 750 lb General Purpose Bomb M117A2 (Eglin AFB Data)

For each munition the following information was documented.

- Test munition physical measurements
- Specifications for each arena test facility
- Listing of mass groups and polar zones
- Individual listing of fragment weights greater than 150 grains.

An understanding of the fragmentation arena test techniques is an important factor in the proper assessment of the larger and more hazardous fragments. The arena test is used to determine fragment mass, velocity and spatial distribution of high-explosive munitions. The munition axis of symmetry is located horizontally and is taken as the polar axis. Designating the polar angle by θ and the azimuth angle as ϕ, the fragmentation characteristics are a function of θ, and gravity effects are assumed negligible. An arena is constructed of an appropriate size as illustrated in Fig. 4. The arena test is designed to sample fragmentation characteristics in various polar angle intervals ranging between $\theta = 0$ degree at the nose, and $\theta = 180$ degrees at the base of the munition. These data samples are used to predict the fragmentation characteristics for the entire munition.

An important aspect of the arena test procedure is the relationship between the sample size obtained from the incremental polar zone on the arena recovery panel and the corresponding munition polar zone. Calculations were made from actual arena setup data to estimate the magnification factor associated with the integrated munitions data. These data, summarized in Table 2, indicate that only about 2 to 7 percent of the fragments are sampled in the 90 degree polar angle region (side spray) for each test. Thus, each fragment must be multiplied by a factor of 15 to 60 to obtain integrated data. It should be noted that the fragment sample size is about doubled at the 30 or 150 degree polar angle regions, and then increases rapidly to near 100 percent at 0 and 180 degrees (nose and base regions).

Table 3 summarizes data associated with fragments in excess of 150 grains for the munitions of interest. The 150 grain fragment was assumed to be the minimum damaging fragment mass projected at low elevations. Table 3 indicates that fragments in excess of 150 grains represent only 5 to 18 percent of the

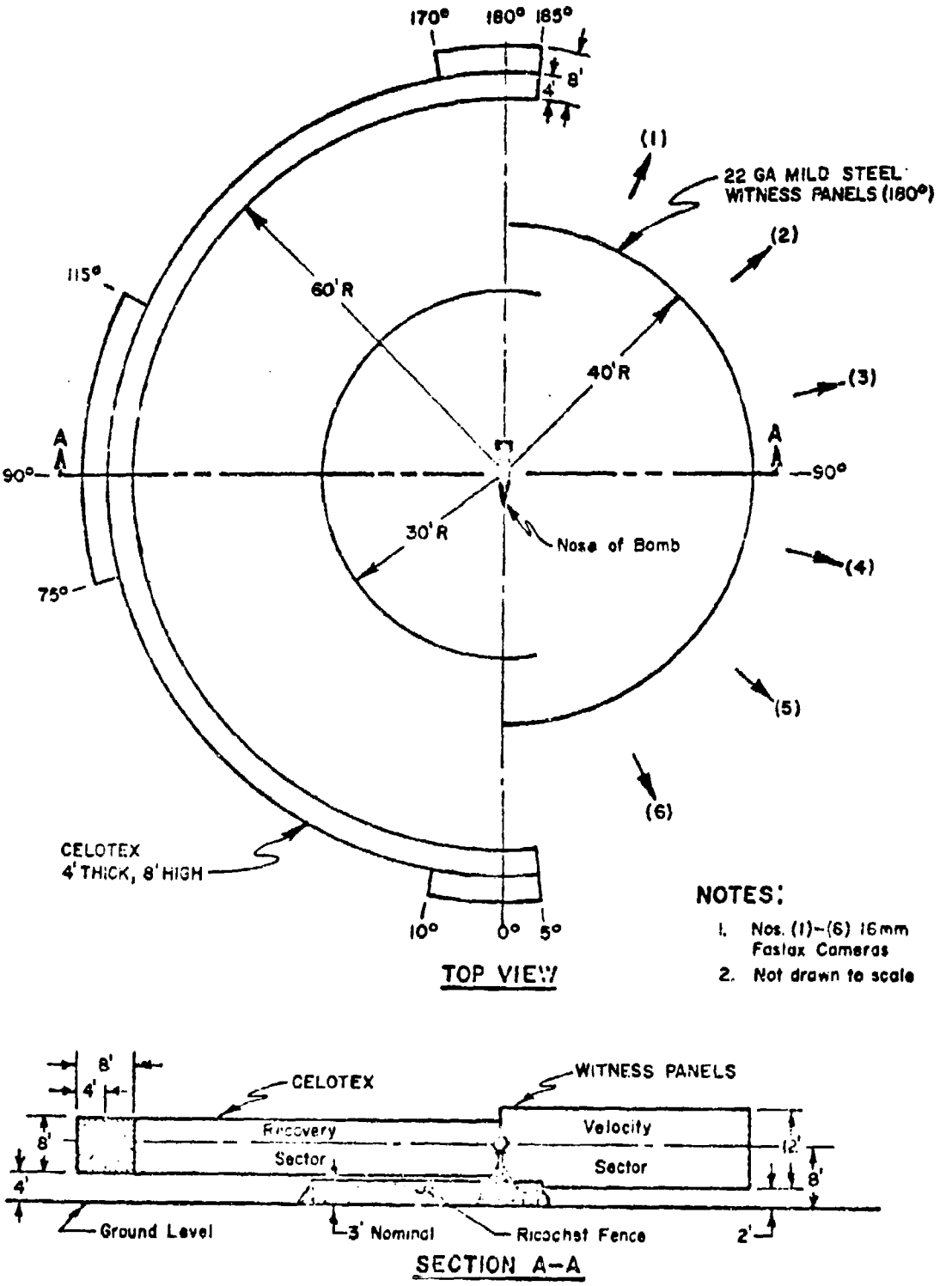
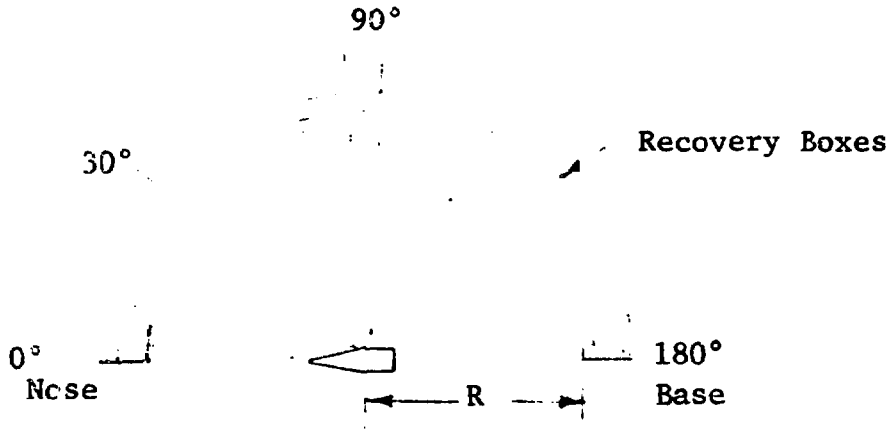


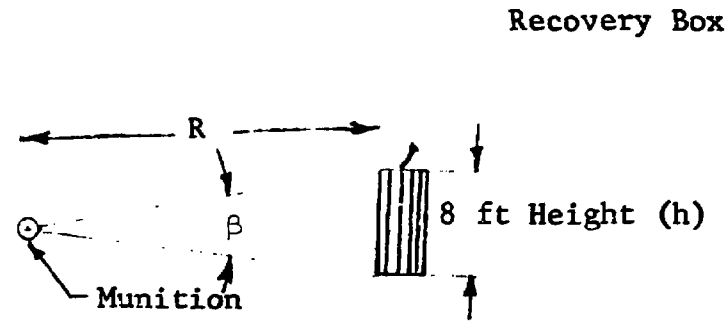
Fig. 4 FRAGMENTATION ARENA DIAGRAM
(500 LB MK 82-1 BOMB)

TABLE 2 ESTIMATED FRAGMENT SAMPLE SIZE DATA FOR GIVEN ARENA TESTS

Munition Type	Arena Radius (R) (ft)	Recover Box Height (h) (ft)	90 Degree Polar Zone Region (Side Spray)		
			Zone Angle β (deg)	Sample Size (β/360°) (%)	Mag. Factor (μ)
105 mm Projectile	20	8	22	6.1	16.4
155 mm Projectile	28	8	16	4.4	22.8
175 mm Projectile	31	8	14.5	4.0	25.0
5 in./38 Projectile	17	8	25	2.0	14.3
8 in./55 Projectile	40	8	11	3.1	32.3
500 lb Bomb	60	8	7.5	2.1	42.6
750 lb Bomb	75	8	6	1.7	58.8



Plan View of Arena



90° Polar Angle Arena Cross-Section

TABLE 3. DATA SUMMARY FOR FRAGMENTS GREATER THAN 150 GRAINS

Munition Type	Metal Weight (lbs)	Total No. of Fragments NT	Data for M > 150 Grains		
			N	%(NT)	% WT
105 mm Projectile	26.1	5,293	315	5.9	52
155 mm Projectile	77.8	6,994	943	13.5	70
175 mm Projectile	111.3	10,819	1364	12.6	78
5 in./38 Projectile	21.8	15,888	722	4.5	58
8 in./55 Projectile	231.0	8,661	1570	18.1	93
500 lb Bomb	320.9	22,114	2564	11.6	81
750 lb Bomb	339.0	24,148	3650	15.1	86

Notes: WT - Total Metal Weight Considered

NT - Total Number of Fragments (Calculated)

N - Number of Fragments Greater than 150 Grains (Approximate)

% NT - Percent No. of Fragments Greater than 150 Grains

% WT - Percentage of Total Weight for Fragments Greater than 150 Grains

total number of fragments but comprise 52 to 93 percent of the total case metal weight considered. In general, the weight groups beyond 150 grains are few in number and wide in weight interval. Since these heavier fragments are few in number and averaged over wide weight intervals, the data samples developed from arena tests are statistically inadequate in describing the characteristics for these potentially damaging fragments. As an example, the munition effectiveness data may list 60 fragments in one polar zone, each weighing 1100 grains and with a common velocity, while all other zones are free from that category of fragments. As input to the hazard model, these data could yield a ring-shaped hazardous region in the far field. It may be more reasonable to assume these fragments to be distributed statistically such as 30 units at 900 grains, 20 units at 1200 grains, and 10 units at 1500 grains, with the fragments being dispersed over three polar zones. In this regard, the distribution characteristics associated with fragments less than 150 grains could be used to predict the statistical distribution characteristics for the heavier fragments.

It should be emphasized that the munitions effectiveness data are valid statistically for characterizing munition performance for fragments of interest to the user. In general, the heavier fragments are only considered in the arena test procedure to assure that a conservation of weight is maintained between the recovered fragments in the arena sample and the integrated munition effectiveness data.

3.2 Revision of Munition Effectiveness Data

Seven sets of data, corresponding to the seven munitions of interest, were considered and of these, six were altered to overcome one of the following two fundamental deficiencies:

1) Gaps - Toward the heavier mass categories, "data voids" appeared making it difficult to approximate the cumulative distribution of mass as a continuous function.

2) Refinement - The cumulative distribution of mass loses definition where the final mass categories are an amalgam of all mass greater than the upper limit of the preceding category.

The first of these deficiencies were more predominate in the 175 mm and 155 mm munitions while the remaining sets of data, with the exception of the 105 mm, had poor refinement in the heaviest mass category. Each problem was handled differently, and in general, the problem of refinement was the most amenable since it was not necessary to add "fictitious" mass categories as it was in the case of data with void deficiencies.

It is prudent to comment that the best improvement which can be made on the experimental data is increasing the sample size. It is recognized then that with restricted data collection techniques and limited sample sizes, extrapolation and statistical inference must be conservative and cautious.

The basic indicators which were used to establish trends and to make decisions, were the average mass ($\bar{m}$) per mass category, the associated standard deviation (σ), the ratio $\sigma/\bar{m}$, and the average mass frequency (f) for each mass category.

The primary data for a particular munition was processed to print out a matrix of fragment weights and frequencies with mass categories as columns and polar zones as rows. Calculated information consisted of average mass $\bar{m}_i$ over all polar zones j within each mass category i, the corresponding average frequency f_i, the standard deviation σ_i, a ratio $\sigma_i/\bar{m}_i$ for each mass category, the total weight $W = \Sigma\, \bar{m}_{ij} f_{ij}$, and the cumulative distribution of weight

$$F_k = \sum_{i=1}^{k} \frac{\Sigma\, \bar{m}_{ij} f_{ij}}{W} \quad k = 1,2 \ldots\ldots N \tag{33}$$

for all N mass categories. The two fundamental deficiencies noted in the data were identified by inspecting the cumulative frequency curve for obvious gaps in the domain of the function and by observing anomalies in trend of standard deviation values for the mass categories.

Two cases which will serve to illustrate the method.

A. <u>The 155 mm Munition</u>

The cumulative distribution curve (Fig. 5) was plotted and it was noted that although the average mass categories range up

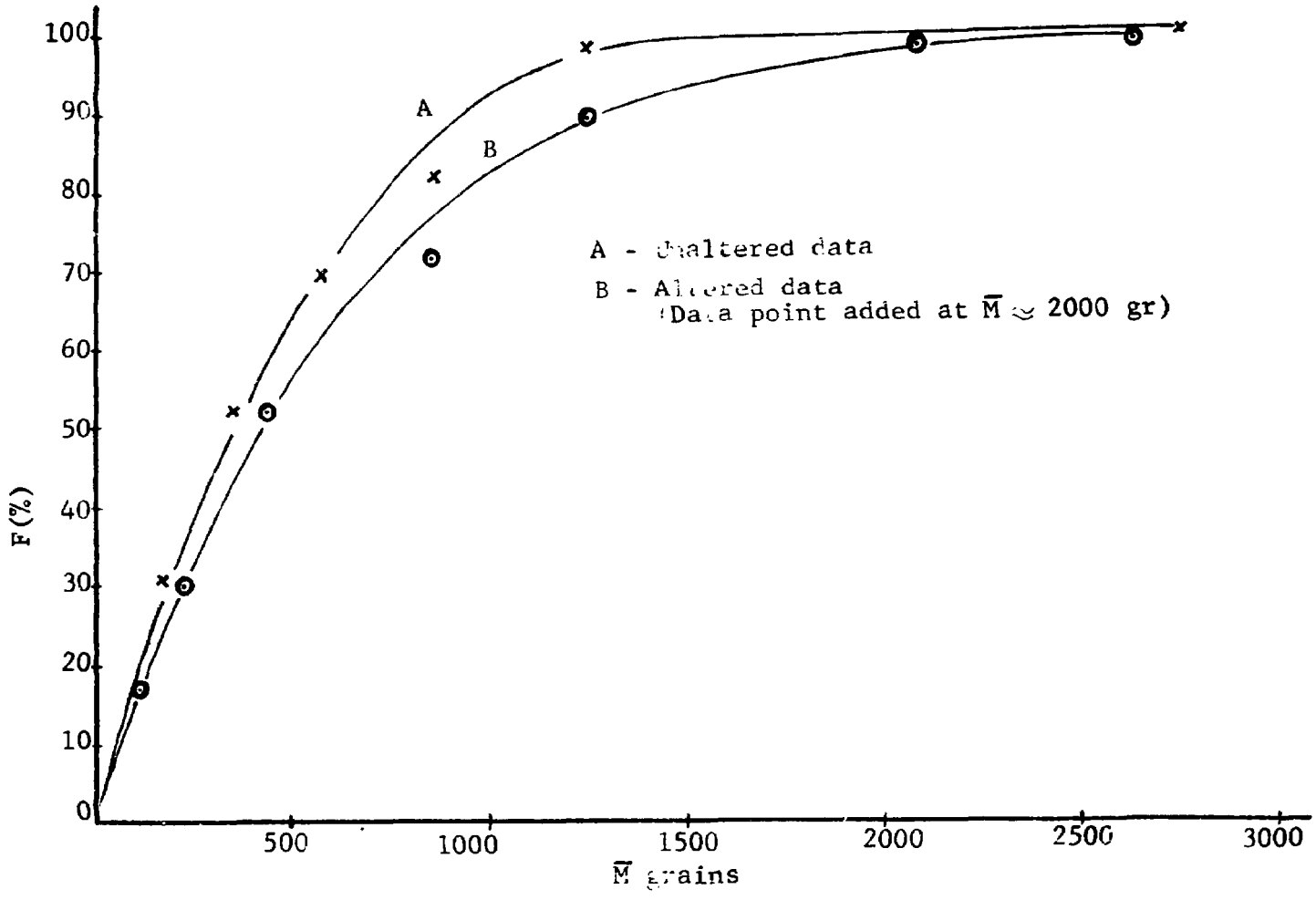


Fig. 5 CUMULATIVE DISTRIBUTION OF MASS FOR THE 155 MM MUNITION

to about 2800 grains, almost 100 percent of the weight has been accumulated at about 1200 grains. By inspection of the curve in Fig. 5 it was decided to start by adding a data point at about 2000 grains so that that portion of the curve would be less sparcely populated. The next decisions must phrase an answer to the questions: how much, and in what manner?

First and second moments were used as the indices of central measure and dispersion of the distribution of mass and frequency across the polar zones within a mass category. In general, the data available in the higher mass categories is not sufficient to construct a reasonable frequency histogram. In the lower mass categories it becomes possible to do this and in order to obtain a graphical sense for the distribution of this mass, histograms were constructed for several categories. Figure 6 depicts the weight distribution in the 200-250 category for the 155 munition. There is another distribution which will be used as a decision aid. This is the frequency of mass over all mass categories with polar zones constant. This curve is shown in Fig. 7.

Essentially then, we have a two-dimensional polar zone - mass frequency matrix in which we would like to make additional entries. Toward this end trends are identified in two directions: across columns (mass categories), and across rows (polar zones). The intersection of these trends at the point in question allows an estimate of the permissible entries which can be made at this point.

It was noted that the ratio $\sigma_i/\overline{m}_i$ tended to be fairly constant and, in case of the 155 mm munition, the average ratio was on the order of .061. Using this in conjunction with the intention of adding mass at $\overline{m} = 2000$, the dispersion measure is

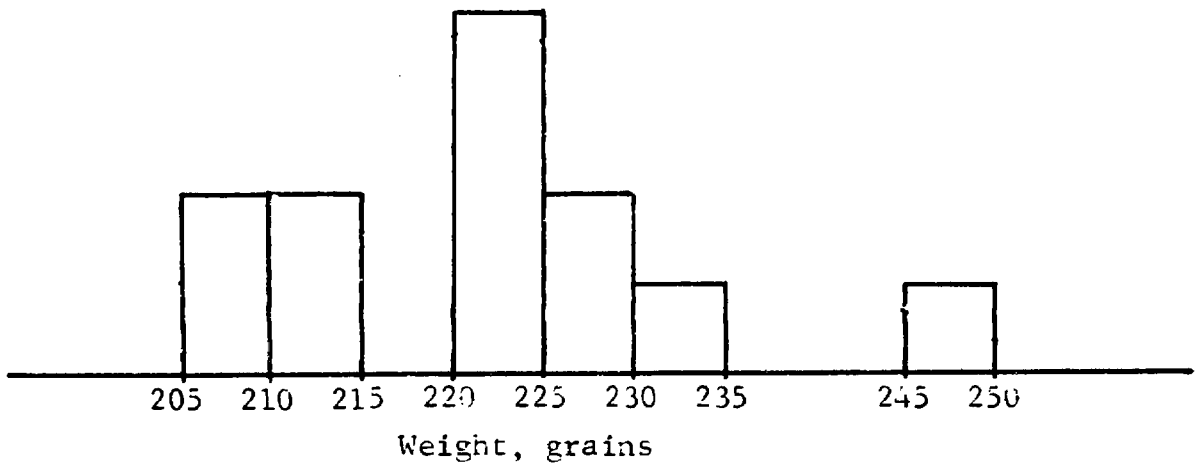


Fig. 6 DISTRIBUTION OF MASS IN THE CATEGORY 200 - 250 GRAINS FOR THE 155 MM MUNITION

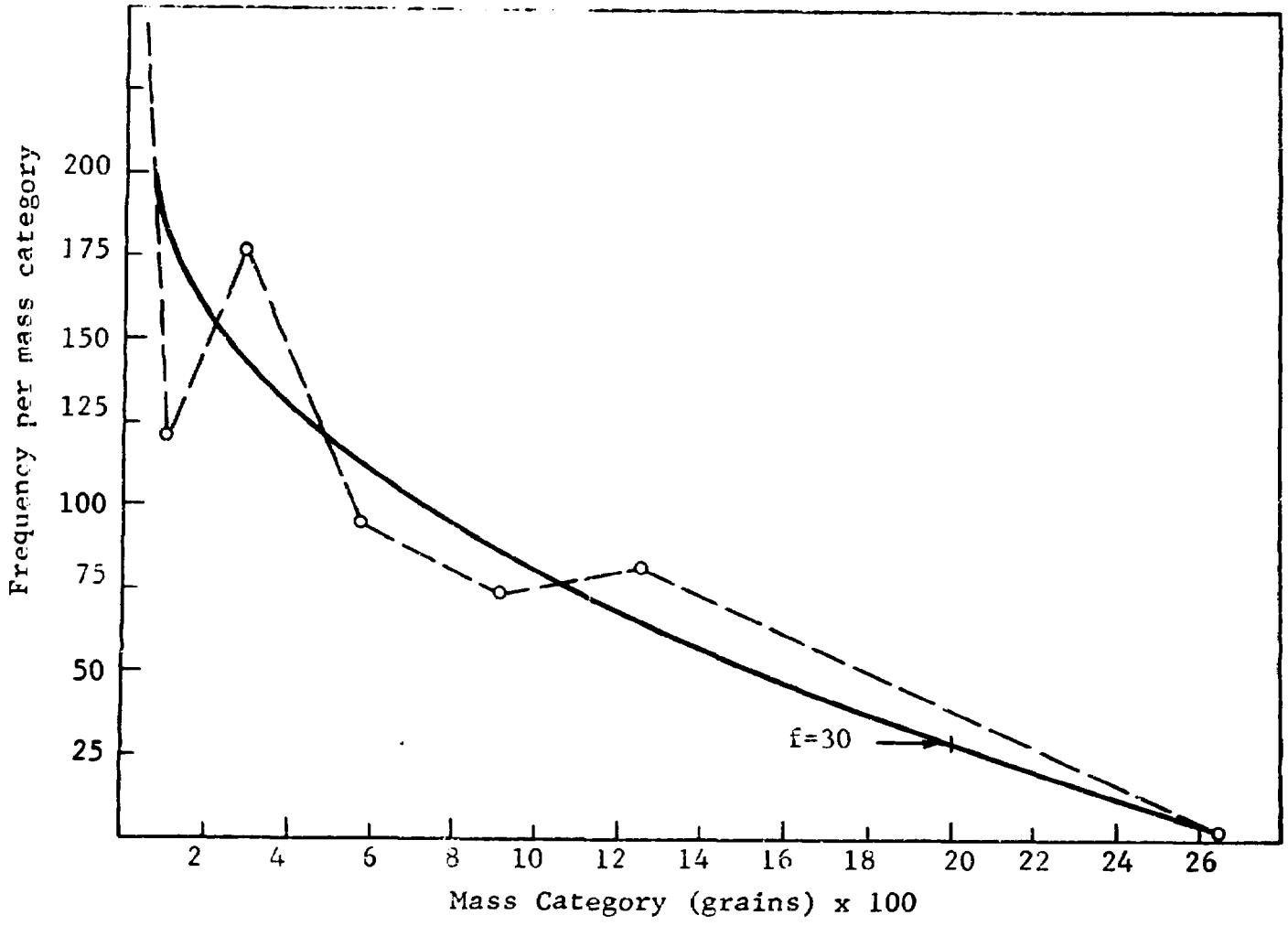


Fig. 7 POLAR ZONE FREQUENCY VERSUS MASS CATEGORY

estimated as .061 x 2000 = 122. The plot of polar zone frequency versus mass category (Fig. 7) was consulted, and a frequency of 30 was associated with the average mass of 2000 grains. By observing the trend across the mass categories, attention focused on polar zone 7 (60-70 degrees) and polar zone 11 (100-110 degrees). Further, by noting the distribution of frequencies across the polar zones in the previous categories, the same proportions are used to obtain:

Mass Category $\bar{m}$	=	2000 grains
Frequency f	=	30
Polar zones receiving entries	=	7 and 11
The frequency in previous mass category $\bar{m}$ = 1230	=	79
Distribution of mass in zones 7 and 11 for $\bar{m}$ = 1230	=	11/79 for zone 7 68/79 for zone 11
Distribution of mass in zones 7 and 11 for $\bar{m} \approx 2000$	=	$11/79 \times 30 \approx 4$, zone 7 $68/79 \times 30 \approx 26$, zone 11

Now using σ estimated at 122 for $\bar{m}$ = 2000, we obtain

$$\bar{m} + \sigma = 2122$$
$$\bar{m} - \sigma = 1878$$

It should be noted at this point that the selection of $\bar{m}$, σ and consequently $\bar{m} \pm \sigma$ impose an implied condition of symmetry for the distribution of mass in that particular $\bar{m}$ category. This mass is further apportioned so as to agree with the trend generally defined by the preceding adjacent category thus bending the implied symmetry to conform with a distribution which is most likely not symmetrical. When data is added in this way the recalculated average mass will not generally agree with the original average mass $\bar{m}$, which was a somewhat arbitrary selection to begin with. The new $\bar{m}$ will be fairly close and in case of the 155 munition the data point was added at 2090 instead of $\bar{m}$ = 2000. The completed profile for the new "fictitious" mass category consists of the following:

1. All polar zones empty except 7 and 11

2. A mass of 1878 grains with a frequency of 4 is entered in polar zone 7
3. A mass of 2122 grains with a frequency of 26 is entered in polar zone 11.

To complete the addition of this new mass category we now consider the conservation of total mass effects. Obviously after the addition of a new mass the total mass T_1 has been increased by some increment T_2. Thus, we have from the original or primary data

$$\Sigma\Sigma\ f_{ij}\ m_{ij} = T_1 \tag{34}$$

where m_{ij} is the fragment mass in the jth polar zone in the ith mass category, and f_{ij} is the associated frequency. Now with the added data points we have

$$\Sigma\Sigma\ f_{ij}m_{ij} + \Sigma\Sigma\ f^*_{ij}\ m^*_{ij} = T_1 + T_2 \tag{35}$$

where m^*_{ij} and f^*_{ij} are the added data for the additional mass categories $\bar{m}_i$. The objective is to scale down all f_{ij}, and f^*_{ij} proportionately such that for new f_{ij},

$$\Sigma\Sigma\ f_{ij}m_{ij} + \Sigma\Sigma\ f^*_{ij}\ m^*_{ij} = T. \tag{36}$$

Toward this end, let p_{ij} be that proportion of f_{ij} which generate the new f_{ij} such that Eq. (36) is satisfied for new f_{ij}. Then for original f_{ij},

$$\sum_{i=1}^{k}\Sigma\ p_{ij}f_{ij}m_{ij} + \sum_{i=k}^{N}\Sigma\ p_{ij}f^*_{ij}m^*_{ij} = T_1 \tag{37}$$

Dividing both sides of (37) by $T_1 = \Sigma\Sigma\ f_{ij}m_{ij}$, we have

$$\frac{\Sigma\Sigma\ p_{ij}f_{ij}m_{ij}}{\Sigma\Sigma\ f_{ij}m_{ij}} + \frac{\Sigma\Sigma\ p_{ij}f^*_{ij}m^*_{ij}}{\Sigma\Sigma\ f_{ij}m_{ij}} = 1 \tag{38}$$

recalling that $\bar{x} = \dfrac{\int\int x\ f(x)\ g(x)}{\int\int f(x)\ g(x)}$

the first term in (38) is $\bar{p}$. Hence,

$$\bar{p} + \frac{\sum_{i=k}^{N}\Sigma\ p_{ij}f^*_{ij}m^*_{ij}}{\sum_{i=1}^{k}\Sigma\ f_{ij}m_{ij}} = 1 \tag{39}$$

Since the sum $\Sigma\Sigma\, p_{ij} f^*_{ij} m^*_{ij}$ is over small ranges of i and j the error will be small if we substitute $\bar{p}$ for p_{ij}

$$\bar{p} + \bar{p}\ \frac{\Sigma\Sigma\, f^*_{ij} m^*_{ij}}{\Sigma\Sigma\, f_{ij} m_{ij}} = 1 \tag{40}$$

or $\quad \bar{p} + \bar{p}\ (T_2/T_1) = 1$

and

$$\bar{p} = \frac{T_1}{T_1 + T_2} \tag{41}$$

The result shows that we can readjust all the frequencies by one constant of proportionality. Hence, a complete set of new data is generated with an added data point on the cumulative curve with the conservation of mass. The new cumulative curve, plotted from the revised data set, is shown in Fig. 5. As in every other case, the curve is shifted downward, which introduces a conservative element into the interpretation of the data.

B. <u>The 500 lb Bomb</u>

The cumulative distribution for this munition was plotted from the original data and is shown in Fig. 8. Without modification, the original data was divided into ten mass categories:

0-10, 10-20, 20-40, 40-80, 80-120, 120-150,
150-190, 190-230, 230-310, 310 + grains

The problem in this case is poor resolution. The standard deviation for example is 3.26 for category number 4, 8.33 for category number 6, 7.93 for category number 8 and 698.5 for category number 10. The standard deviation of 698.5 is not in itself unexpected. Because of its magnitude, it is an anomaly since its ratio $\sigma/\bar{m}$ = .754 is more than ten times the expected value of approximately .06. Moreover, it is a sharp break in the standard deviation trend. If we focus on mass category No. 10, then we find it labeled as 310 +. This is misleading since, in fact, mass category number 10 contains fragments which weigh as much as 5000 grains. To overcome this resolution deficiency, the 10th category was expanded into five additional categories such that the total number of mass categories became 15.

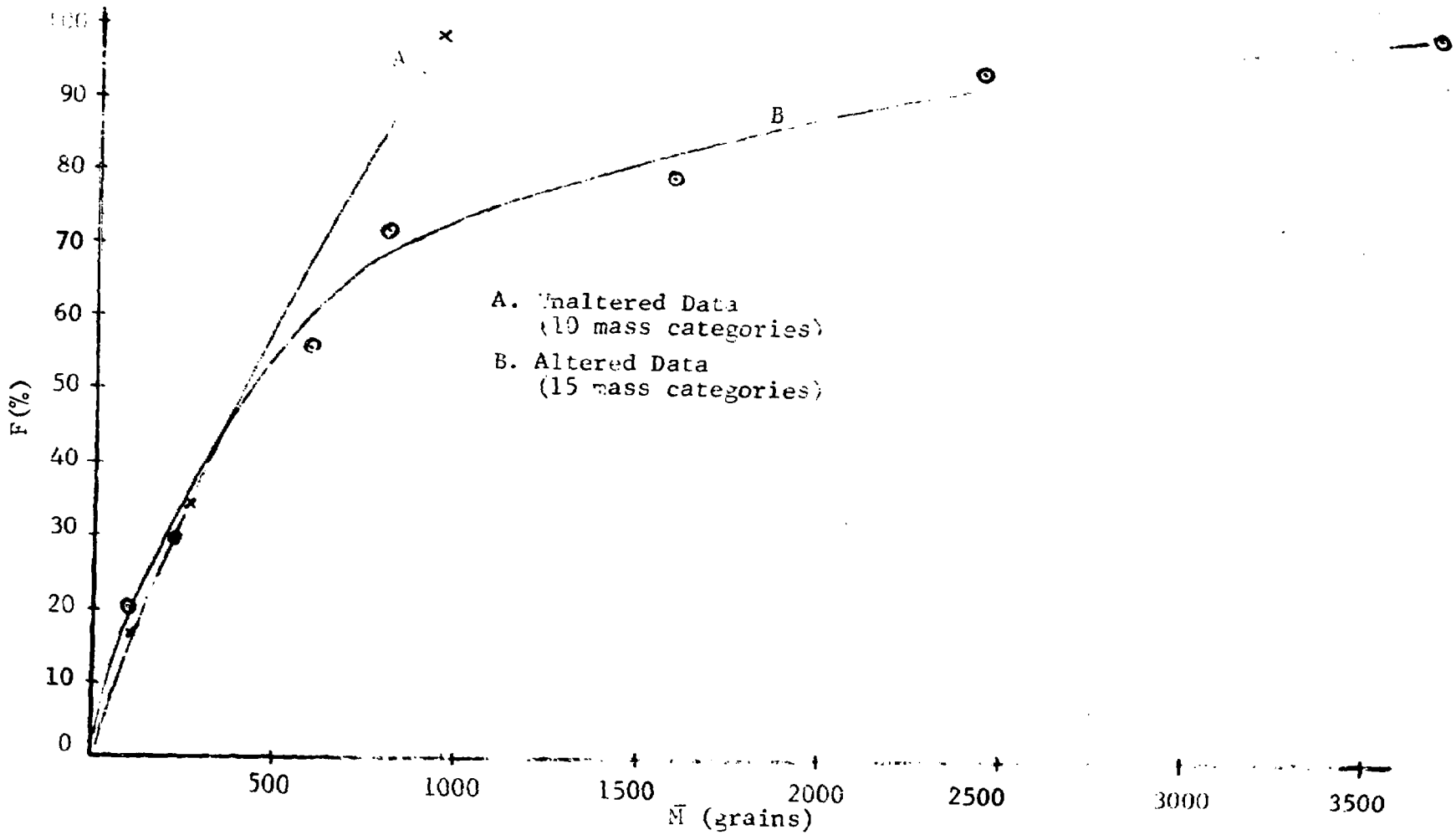


Fig. 8 CUMULATIVE DISTRIBUTION FOR 500 LB BOMB

Category	Range
10	310-450
11	450-750
12	750-1000
13	1000-2000
14	2000-3000
15	3000 +

These mass categories were then added on to the data, replacing the former category number 10. The result was a better than 50 percent improvement in the average $\sigma/\overline{m}$ for categories 10 thru 15 and a smoothing of the expected standard deviation trend. The altered cumulative distribution is shown in Fig. 8.

The data received from the Navy was not directly usable with out extensive recoding and key punching. This was done for the 750 lb bomb and the cumulative distribution appears in Fig. 9. This curve falls below the curves for the original data and its altered form. Since the lower curve implies more mass in the higher mass categories it also implies a greater damage influence range. In all cases the alteration of the data produced the same effect by lowering the cumulative distribution curve.

The following is a summary of how the remaining data were altered. No alterations were necessary for the 105 mm shell.

750 lb bomb: originally had 12 categories. The label on category 12 was 450+. Six additional categories were added as follows:

12 450-700
13 700-900
14 900-1200
15 1200-1500
16 1500-2000
17 2000-3000
18 3000 +

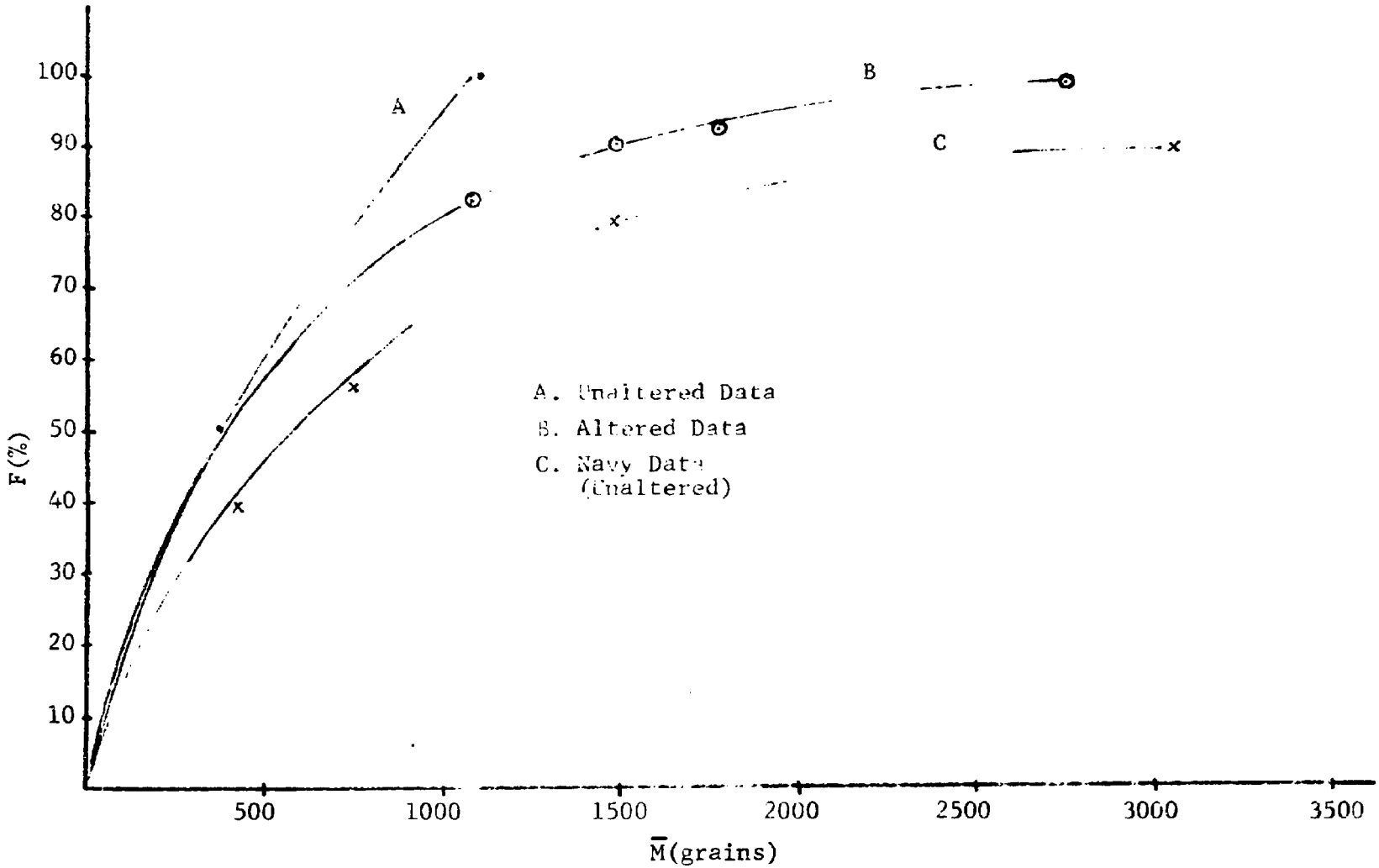


Fig. 9 CUMULATIVE DISTRIBUTION FOR 750 LB BOMB

The cumulative distribution is shown in Fig. 9

5 inch Munition: 11 mass

categories were expanded to 14 as follows:

11	250-450
12	450-700
13	700-1500
14	1500-2000

The cumulative distribution is shown in Fig. 10.

The 8 inch Munition: 10 mass

categories were expanded to 16 as follows:

10	160-300
11	300-500
12	500-700
13	700-1000
14	1000-2000
15	2000-3000
16	3000+

The cumulative distribution is shown in Fig. 11.

The 175 mm munition: two data points were added to the cumulative distribution as shown in Fig. 12. The technique was the same as described for the 155 mm munition.

A complete set of the revised munition effectiveness tables are presented in Appendix E to this report. Their format is similar to that shown in Table 1.

3.3 Model Sensitivity to Input Data

In order to gain an appreciation for the sensitivity of the fragment hazard model output to alteration of munition effectiveness data, which is input to the model, a computational experiment was conducted. The unaltered and altered data sets for the 175 mm gun shell were used as input to the model. The results of these two cases are shown respectively in Figs. 13 and 14. The primary difference between the two sets of contours is the location of the lowest value contour line.

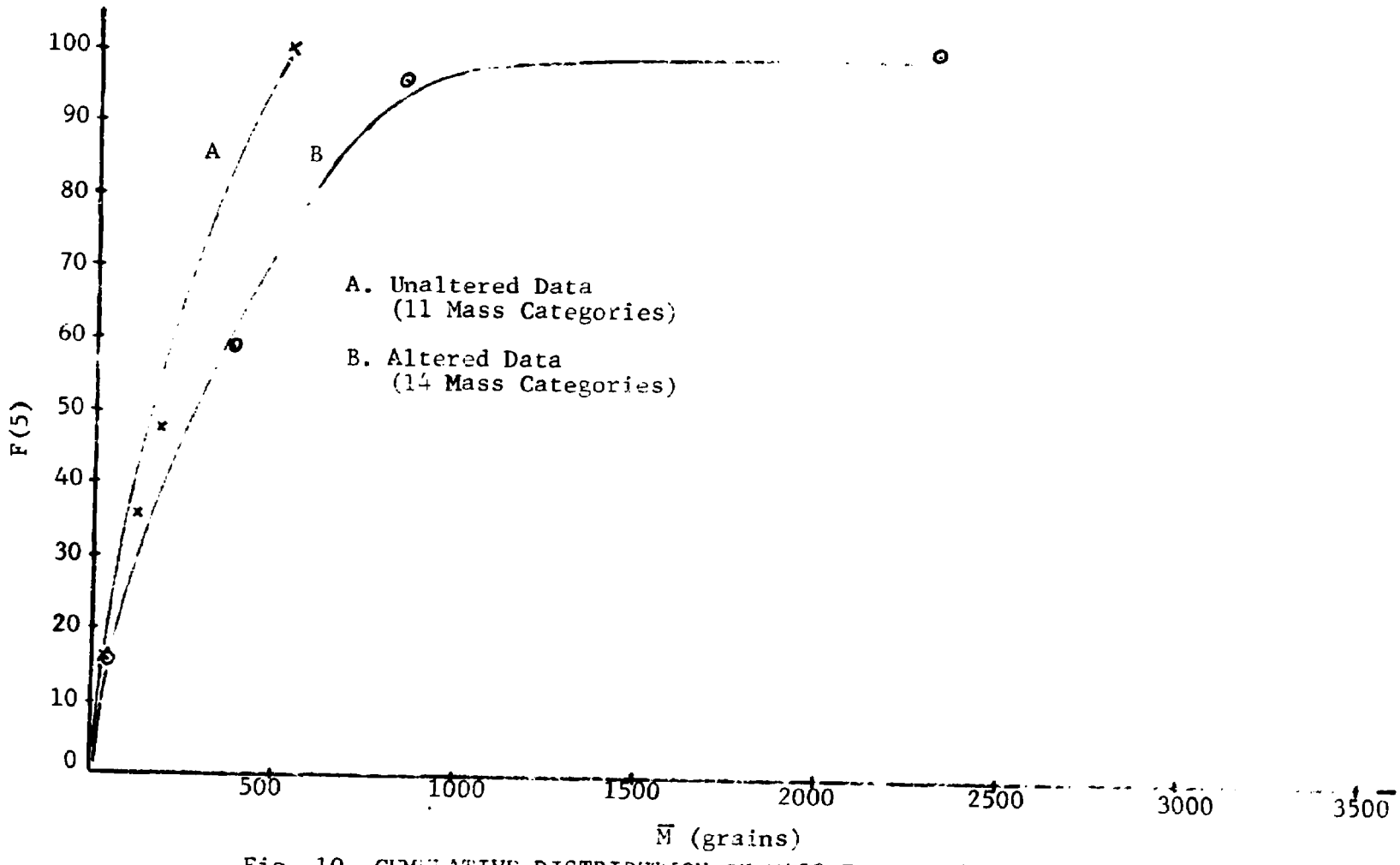


Fig. 10 CUMULATIVE DISTRIBUTION OF MASS FOR THE 5 IN. MUNITIONS

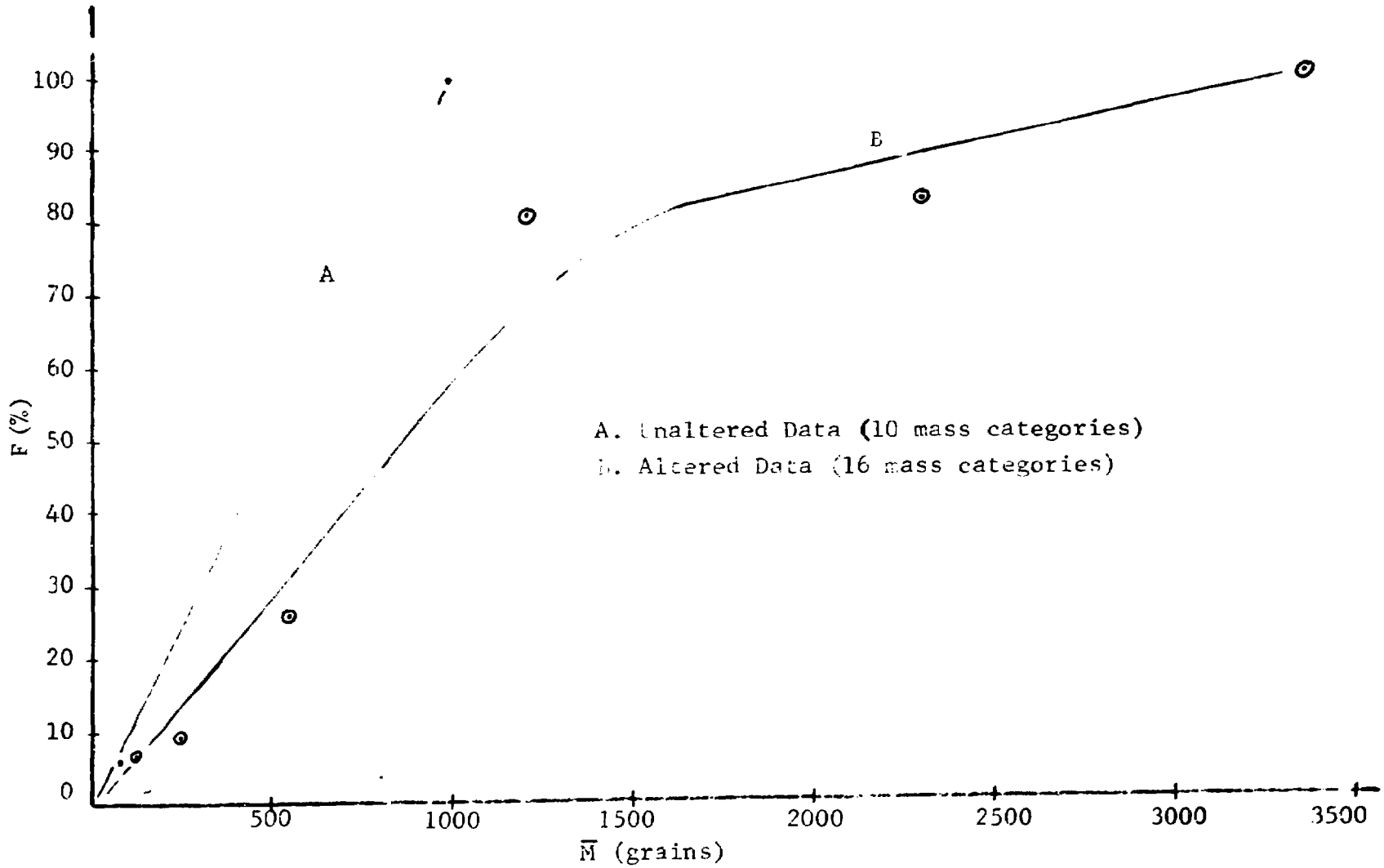


Fig. 11 CUMULATIVE DISTRIBUTIONS OF MASS FOR THE 8 IN. MUNITIONS

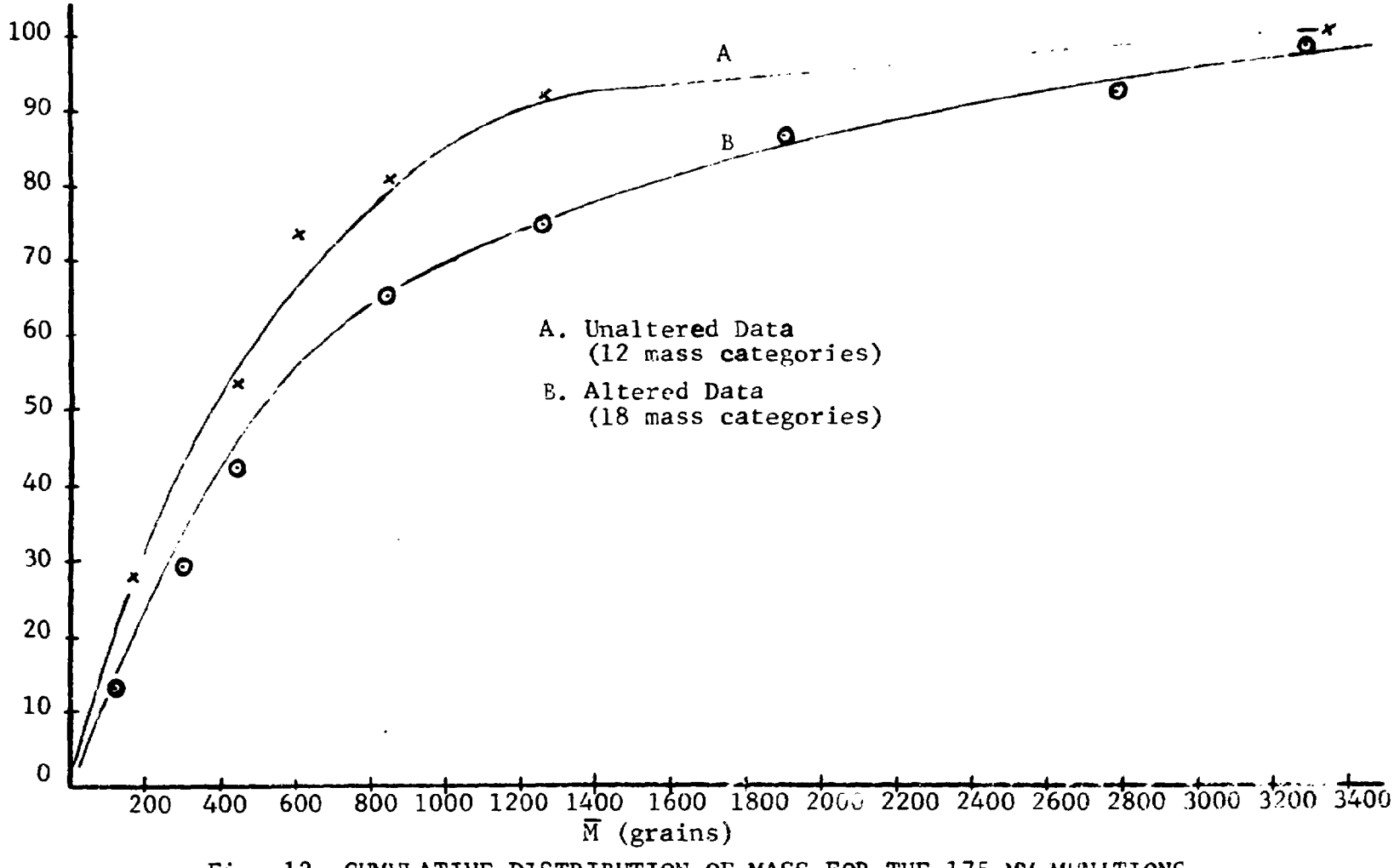
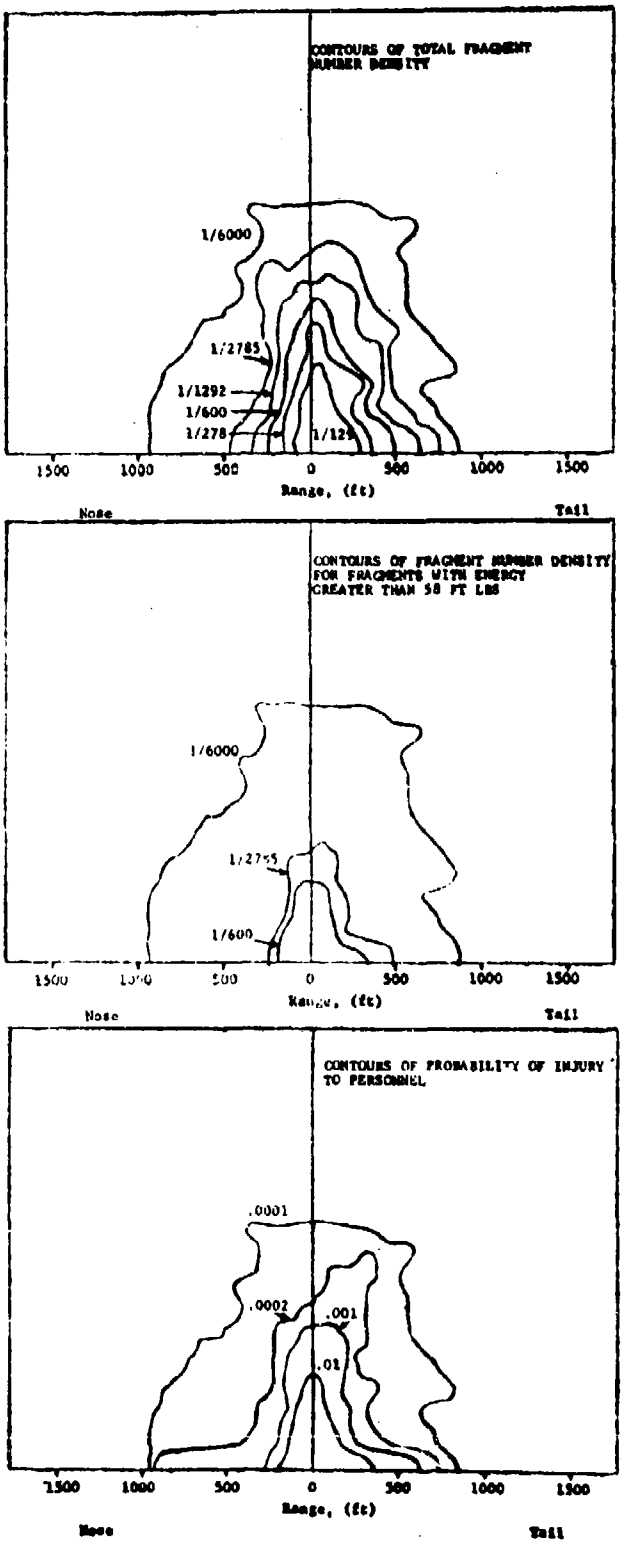


Fig. 12 CUMULATIVE DISTRIBUTION OF MASS FOR THE 175 MM MUNITIONS



Reproduced from best available copy.

Fig. 13 175 MM GUN SHELL M437A2 (COMP. B LOAD) (UNALTERED DATA)

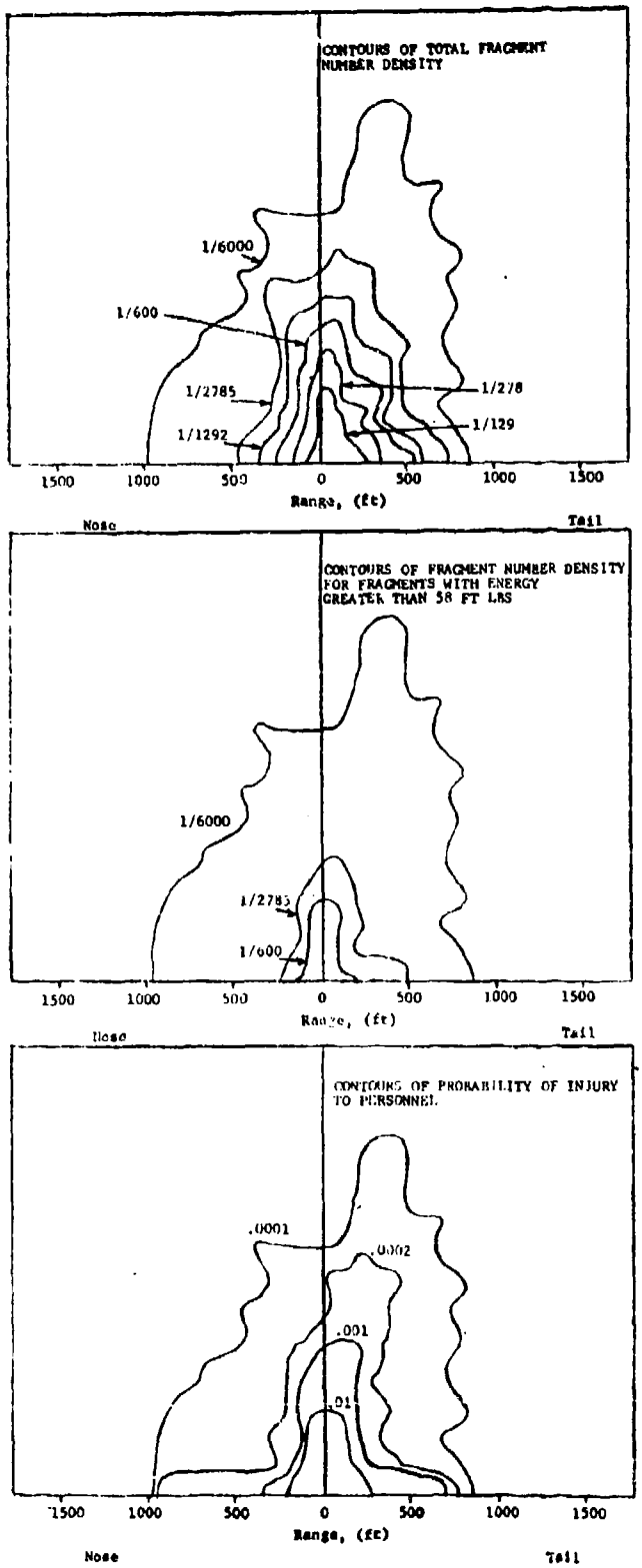


Fig. 14 175 MM GUN SHELL M437A2 (COMP. B LOAD) (ALTERED DATA)

The altered data set represents an emphasis on the resolution of the heavier fragments, (e.g., note Fig. 12) at the expense of lower weight fragments. This has the pronounced effect of shifting the location of the lowest value contour line outward for the altered data set. In terms of safety, then, the altered data set results in a more conservative set of contours.

4. RESULTS, CONCLUSIONS, AND RECOMMENDATIONS

This study has resulted in the development of a computer model which generates the information necessary in establishing minimum separation distances between various munition types and personnel in order to mitigate fragment hazards. The model specifically treats the fragment hazard associated with a single munition and has been utilized to generate single unit fragment hazard data for seven common military munitions. While single unit detonation does not represent a realistically severe accident situation, previous work indicates that multiple unit (i.e., stacks) fragment hazards may be proportional to single unit results. If this is true, and there is evidence to support such a hypothesis in the case of thin wall munitions, then the results obtained herein are directly applicable to estimating the fragment hazards associated with openly stored munitions.

4.1 Results

- Contour maps, expressing number density and injury probabilities (i.e., Appendix B), are seen to be particularly pronounced in three fixed sectors of the ground plane; that is, the nose, tail and side-spray sectors. Average values within each of these sectors have been expressed as functions of the radial distance from the explosive source. (i.e., Appendix C)

- Examination of the relationships in Appendix C indicate that the side spray sector is the critical area in the case of thick wall "shell type" munitions and to a limited degree in the thin wall "bomb" munitions.

- The curves in Appendix C also serve to illustrate how the kinetic energy criterion for personnel injury (i.e., 58 ft-lbs) reduces the applicable number density. At a given range the number density is reduced as much as an order of magnitude. It should be recognized that consideration of a less severe kinetic energy criterion will lead to lesser reduction in number density and thus a more conservative injury criterion in terms of safety.

- Table 4 summarizes the results available in Appendix C for each of the seven munitions considered. The table gives the computed distances corresponding to a fragment density of one hit in 600 ft^2 for all fragments and fragments with a terminal energy in excess of 58 ft lbs.

TABLE 4

SUMMARY TABLE FOR THREE PRINCIPAL DIRECTIONS AT 1 HIT PER 600 SQ FT (ENTRIES ARE IN RADIAL FEET)

Munition	All Fragments			Hazardous Fragments**		
	Nose	Side	Tail	Nose	Side	Tail
750 (1)*	440	1060	740	220	690	500
500 (1)	220	825	595	210	670	450
175 (1)	250	840	575	250	450	200
155 (1)	290	810	510	120	400	230
105 (0)	240	650	360	100	270	150
8 (1)	325	660	240	140	520	120
5 (1)	310	720	340	140	275	150

*(0) = Unaltered Data

(1) = Altered Data

**Hazardous fragments are defined as having in excess of 58 ft lbs kinetic energy

4.2 Conclusions

- By utilizing trajectory analysis in conjunction with stochastic treatment of experimental input data and damage functions, a mathematical model has been developed for estimating injury/damage contours for various single unit munitions.

- The mathematical model currently utilizes published munition effectiveness data which has been altered to give more emphasis to heavier fragments. It has been observed that this results in extending contour levels outward giving a more conservative result insofar as fragment safety is concerned.

4.3 Recommendations

Before making specific recommendations concerning the results generated in this study and future research efforts it is prudent to discuss the adequacy of munition effectiveness data as input to the model.

The results generated by the computer model are dependent upon and quite sensitive to the munition effectiveness data, which is input. This data was originally generated to support munition effectiveness studies and is the result of explosive tests of single unit munitions. It is the only known source of information concerning near-field estimates of munition fragment size, number and initial velocity. However, since it has been collected to be utilized in weapon effectiveness studies, it is primarily concerned with the fragments which are effective within the applicable range of the munition. This has normally led to a set of data which has a high degree of resolution, in terms of weight intervals, where the greatest number of fragments are concentrated. This unfortunately is at a rather low fragment weight (e.g., below 300 grains). The remaining fragment weight of the munition is quite substantial, but because it does not break down into very many fragments and is not always projected into the designed zones of munition effectiveness, its recorded resolution is usually quite poor.

Another inadequacy of recorded munition effectiveness data is concerned with its use in representing the basis for multiple unit munitions fragment hazard analysis. Here, the primary

concern is whether the munition fragment size, number and initial velocities will be similar for munitions in single and multiple units.

In the present study an attempt has been made to overcome the first of these deficiencies in a conservative manner. That is, existing munition effectiveness data have been altered to adequately resolve higher weight fragments in a statistical sense. Recently reduced data, developed at the Naval Weapons Laboratory, tend to support these alterations; however, the best way to resolve this problem would be the design and use of an experimental procedure aimed at obtaining arena data for hazard analysis.

The problem in assuming similar fragment characteristics for multiple and single unit stacks is a much more difficult problem to resolve. A number of detonation tests of stacked munitions have been conducted in the past where resulting fragments have been collected. In some cases the fragments have also been sized and number and weight distributions computed. Results of these studies indicate that thinwall "bomb type" munitions tend to fragment into similar size fragments for both multiple and single units. However, it is becoming quite apparent that this is not the case for thickwall "shell type" munitions. Here, fragment weight and number distributions from multiple units are quite different from single unit munitions.

In light of the input deficiencies enumerated above, the following recommendations are made:

- To take care in utilizing results obtained in this study for single unit munitions, in projecting fragment hazards associated with multiple units of similar munitions. This is especially true of "shell type" munitions.

- To compare the analytic results for the single unit 750 lb bomb obtained in the present study with experimental multiple unit results obtained in the NWC-China Lake Tests of March, 1970. Such a comparison will serve to validate our analytic procedures and the extension of our results for estimating multiple unit "bomb" fragment hazards.

- To characterize the resulting fragment number and weight distributions obtained from the June, 1970 155 mm shell tests at Yuma and the corresponding 155 mm shell tests at China Lake in December of 1971. These tests are expected to be dissimilar amongst both themselves and as compared to single unit results. However, such comparison will document these differences and the resulting distributions can be utilized by the analytic model to estimate the fragment hazard associated with both these cases.

REFERENCES

1. Zaker, T. A., et al., "Fragmentation Hazard Study, Phases I and II, IITRI Final Report J6176 for the Armed Services Explosives Safety Board under Contract DAHC-04-69-C-0056, April, 1970.

2. Feinstein, D. I., et al., "Fragmentation Hazard Study Fragment Hazards From Detonation of Multiple Munitions in Open Stores," Phase II, IITRI Final Report J6176 for the Armed Services Explosive Safety Board under Contract DAHC-04-69-C-0056, August, 1971.

3. Shaw, J. E., "A Measurement of the Drag Coefficient of High Velocity Fragments," BRL Report 744, (October 1950) (U).

4. Zaker, T. A., "Trajectory Calculations in Fragment Hazard Analysis," Minutes, Thirteenth ASESB Seminar, 101-115, September, 1971.

5. Weapons Characteristics Report, "Joint Munitions Effectiveness Manual/Air-to-Surface (U)," March 1967, (S).

APPENDIX A

PROGRAM USER'S MANUAL

APPENDIX A
PROGRAM USER'S MANUAL

The computer program for the fragment hazard model has been written in such a manner as to accept a problem-oriented input language. The following discussion describes, in detail, the various control and data statements required to execute the program. An example of the use of these statements in a sample problem is included.

Control Cards

All control cards start with column 1 = '$'. The remainder of the card is free format. Only columns 1-72 may be used for control text; columns 73-80 are not examined and may be used for anything.

Control Statement Form

Control statements are of the form:

name field1 field2 ... fieldn

where 'name' identifies the control statement, and the 'field's are parameters whose form depend on the particular control statement. 'name' and the 'field's are separated from one another by

1) one or more blanks, or
2) a comma, padded on either side by one or more blanks.

Examples:
```
$PRINT CON,DATA
$TAPE 4,SRCH=01100,BKSP
$STOP
```

If desired, comments preceded by a '$' may appear after the last field.

Example: `$STOP  $  END OF RUN`

If a control statement will not all fit on one card, it may be continued to additional cards in the following way:

1) place a comma after the last field on the first card.

2) start the second card with a '$' and a ',' (padding with blanks if desired).

3) place remaining fields after the '$' and ',' on the second card.

4) as many continuation cards as necessary may be used.

```
Example:  $ORDNANCE FORMAT=2,REDUCE,REZONE,NORM,SMOOTH,
          $,READ=3,LIST     $  READ FROM TAPE 3
```

Parameter Form

Control statement parameters may be of three forms:

1) Positional: A value which must appear in its proper order on a control statement, as the first parameter, third parameter, etc.

   Examples: 3, 14.197, T, 'TITLE A'

2) Flag: A name, or the two characters 'NO' prefixed to a name. A flag may appear anywhere in a control statement after the positional parameters (if any). A flag is associated with a logical value: this value is TRUE if the flag's name is specified, or FALSE if 'NO' plus the flag's name is specified.

   Examples: LIST, NOLIST, BCD, BIN

3) Keyword: A name, followed by either a BCD (3-8 punch) or an EBCDIC (6-8 punch) equal sign (=), which may be padded with blanks, followed by a value.

   A keyword parameter may appear anywhere in a control statement after the positional parameters (if any).

   The variable represented by the name is set to the provided value.

   Examples: ORDER=2, Z=5.5, FLAGS=XY, TITLE='TITLE A'

Value Form

There are four types of values:

1) Logical: a string of one or more characters, not containing any of the following: dollar sign ($), comma (,), equal sign (=), blank, or apostrophe ('). If a T appears in the string before an F appears, the value is TRUE. If an F appears before a T appears, or if neither an F nor a T appear, then the value is false.

   Examples: T, .TRUE., TRUFFLE and TRUE are all TRUE.
   F, .FALSE., AFTER, FALSE and NO are all FALSE.

2) Integer: one or more digits, optionally preceded by a + or - sign.

   Examples: 0, -1234, +77111

3) Real: a real number. Decimal point and exponent may be used. The exponent, if present, must be of one of the following forms:

   En, E+n, E-n, +n, -n

   where n is one or two digits.

   Examples: 0, -1234, +77111, 3.1416, 6.0238E+23, 1.-5, -.1+10, 51E6

4) Alphabetic: two forms are allowed.

   - a string of one or more characters, not containing any of the following: dollar sign ($), comma (,), equal sign (=), blank or apostrophe (').

     Examples: ABC, 123, MACH-1

   - a string of zero or more arbitrary characters, enclosed in apostrophes (''). Within the string, one apostrophe is represented by a pair of apostrophes. Either the BCD (4-8 punch) or EBCDIC (5-8 punch) apostrophe may be used.

     Examples: 'ARTHUR', 'WHICH WAY?', 'DON''T GO', 'TITLE A', '' (a null string)

Notation

When describing control cards, some possibly unfamiliar notation is used. This can be explained with an example:

$$\$\text{TAPE } n \left\{ \begin{matrix} ,\text{BCD} \\ ,\underline{\text{BIN}} \end{matrix} \right\} [\,,\text{REW}]$$

The use of { } means that only one of the arguments listed may be used. If one of the arguments is underlined, that one will be assumed if none of the group are specified. (In the example, BIN is assumed unless BCD is used. BIN and BCD may not both be used.) The use of [ ] means that the enclosed argument is not required; it may be used or not used.

$PRINT [,CON][,DATA]

Print controls

CON	Cause control cards to be printed.
DATA	Cause data cards to be printed

$STOP

End of run; execution is terminated.

$TIME

Print elapsed time since beginning of execution, and since last $TIME statement.

$TAPE n {,BCD / ,BIN} [,REW][,SKIP=k] {, SRCH=α / ,XSRCH=α} [,BKSP][,WEOF]

Tape file manipulations. If more than one of the operations (REW through WEOF) are specified, the operations take place in the order REW,SKIP, SRCH or SXRCH, BKSP,WEOF, regardless of the order in which the operations are specified.

n	Tape unit number
BCD	Tape is formatted (BCD)
BIN	Tape is unformatted (binary).
	If neither BCD nor BIN is specified, BIN is assumed.
REW	Rewind unit n.
SKIP=k	Skip k records on unit n.
SRCH=α	Search unit n, starting at its present position, for a sentinel record whose key is α (1 to 8 characters). Terminate if no such sentinel is found before the end-of-file.
XSRCH=α	Proceed as with SRCH, but if an end-of-file is encountered, rewind unit n and search the entire file. Terminate if end-of-file is again reached without finding the proper sentinel.
BKSP	Backspace unit n one record.
WEOF	Write a terminal sentinal record and an end-of-file mark on unit n.

$ORDNANCE FORMAT=n[,NSETS=n][,REDUCE][,REZONE][,NORM][,SMOOTH]
[,ORDER=n][,READ=n][,LIST][,KWIKPLOT][,SUMMARY]

Cause a title card and a set of ordnance data to be read.

FORMAT=n	Ordnance format code. =1 or 2.
NSETS=n	Number of ordnance components (fuze, casing, etc.) If ≤ 0 or omitted, NSETS=1 is assumed.

REDUCE	Halve the number of mass categories by combining each pair of categories.
REZONE	Transform data from N+1 sets of values at the boundaries of N polar zones to N sets of values at the N polar zone midpoints.
NORM	Normalize data, given as total number of fragments per mass category per polar zone, to number per unit area on a unit sphere surrounding the ordnance.
SMOOTH	Fill in gaps in data, and (if ORDER>0) apply a smoothing function to the ordnance data.
ORDER=n	Use an nth order Fourier series to smooth data (if SMOOTH is specified).
READ=n	Read ordnance data from unit n, a BCD card image unit. If n=5 or if READ=n is omitted, ordnance data is assumed to follow the $ORDNANCE statement.
LIST	List ordnance data in tabular form.
SUMMARY	List data summary information: total mass per mass category, per polar zone, etc.
KWIKPLOT	Produce printer plots of velocity curve, and of mass and number curves for each mass category, all as a function of polar angle.

An ordnance title card is read immediately after a $ORDNANCE statement is encountered. Columns 1-72 contain a description of the ordnance.

Two basic formats are defined for ordnance data (FORMAT=1 and 2). Within the context of each format, several data conditioning options are available (REDUCE, REZONE, NORM, SMOOTH, ORDER=n).

Ordnance data is read from cards if the READ field is omitted, or if READ=5. If a READ field specifies some other I/O unit, input is read from that unit in card image form. This unit is assumed to be properly positioned.

The first card (or card image) of a set of ordnance data is as follows, regardless of the type of ordnance data.

COLS	FORMAT	NAME	DESCRIPTION
1-5	I5	NMORD	Number of mass categories. If NMORD> 30 (maximum number of mass categories), REDUCE must be specified to cut the number of mass categories in half.
6-10	I5	NVORD	Number of measurement positions along a meridian from nose to tail. Measurements may be made either at the center or at the edges of equi-spaced polar zones. In the latter case, REZONE must be specified NVORD<37.
11-15	I5	IDORD	Ordnance I.D. number 0≤IDORD≤99.
16			(ignored)
17	A1	IVO	Ordnance modification code. If left blank, IVO='0' is assumed.
18-20			(ignored)
21-30	E10.0	XKORD	Average fragment drag factor (grains/in.^{3})

Format 1 Ordnance Data

Data of format 1 consists of NVORD groups of cards, each containing:

- ⌈NMORD/4⌉* cards, in (4(2F5.1,5X))format, containing NMORD pairs of values: the average fragment mass (grains) and the average number of fragments** for each mass category, for one polar zone.
- one card, in (50X,E10.1) format, containing the average fragment initial velocity (fps) for one polar zone.

Format 2 Ordnance Data

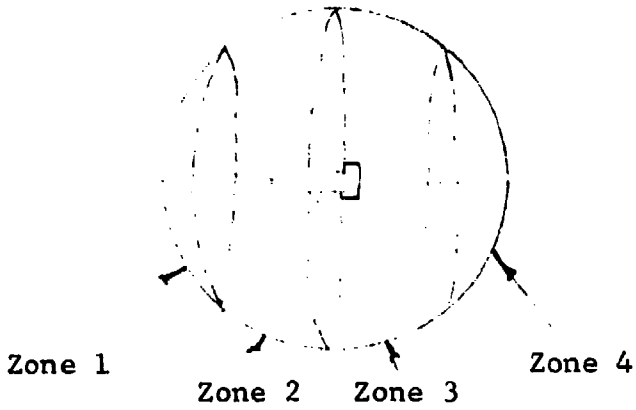
Data of format 2 is split into two parts.

Part 1 contains NSETS of cards, each consisting of ⌈NMORD/4⌉ groups of NVORD cards. Each card is in (8E10.0) format and contains four pairs of values: the average fragment mass (grains) and the average number of fragments for four mass categories, for one polar zone.

---

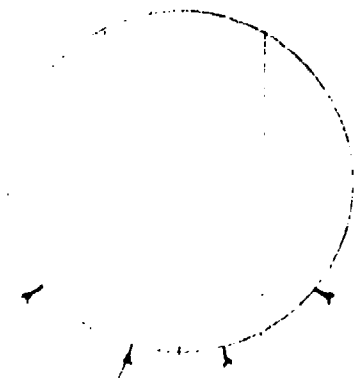
*⌈X⌉ means the smallest integer not less than X.
Examples: ⌈3.14⌉=4, ⌈14⌉=14, ⌈7.00001⌉=⌈7.9999⌉=8.

**Fragment number information in the form of a count of the total number of fragments in a polar zone may be used, provided that the NORM parameter is specified on the $ORDNANCE statement.

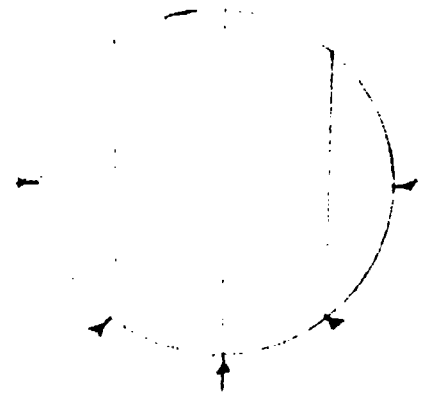


Hypothetical Sphere of Unit Radius Surrounding Ordnance.

Sphere divided into four polar zones, each of equal latitude range.



Measurements Taken at Center of Zones



Measurements Taken at Zone Boundaries (REZONE Parameter Required on $ORDNANCE Statement)

Part 2 contains NSETS sets of cards, each consisting of NVORD cards in (50X,E10.0) format, each of which contains the average fragment initial velocity (fps) for one polar zone.

$BOOM [,FLAGS=α][,NBAR=n][,ORDER=n][,Z=h][,DRNG=r][,NRNG=n]
[,NEL=n][,NAX=n]

Generate a fragment distribution using ordnance data read according to the prior $ORDNANCE statement. Write the fragment distribution field, with a sentinel record on binary unit 4.

FLAGS=α — α consists of one or two characters. If α is one character, the second flag character is assumed to be zero. If the FLAGS field is omitted, both flag characters are assumed to be zero. The flag characters are used in the sentinel I.D. for the fragment field output.

NBAR=n — Number of barriers placed in the vicinity of the ordnance. If omitted, NBAR=0 is assumed. Barrier description cards, if any, are read after the fragment field title card, NBAR $\leq$ 1.

Z=h — height of target plane (in feet) above ordnance. If Z<0, the ordnance is above the target plane.

DRNG=r — spacing of range points, in feet.

NRNG=n — number of range points on an azimuth ray, NRNG $\leq$ 20.

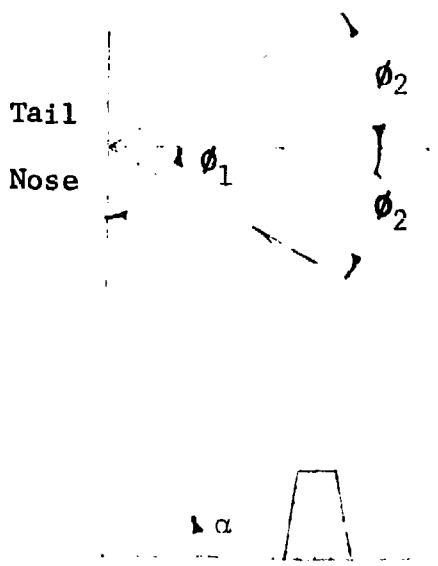
NEL=n — Number of elevation angles to be considered (high register fragments), NEL $\leq$ 18.

NAZ=n — number of azimuth rays in fragment field (a semicircle), NAZ $\leq$ 36.

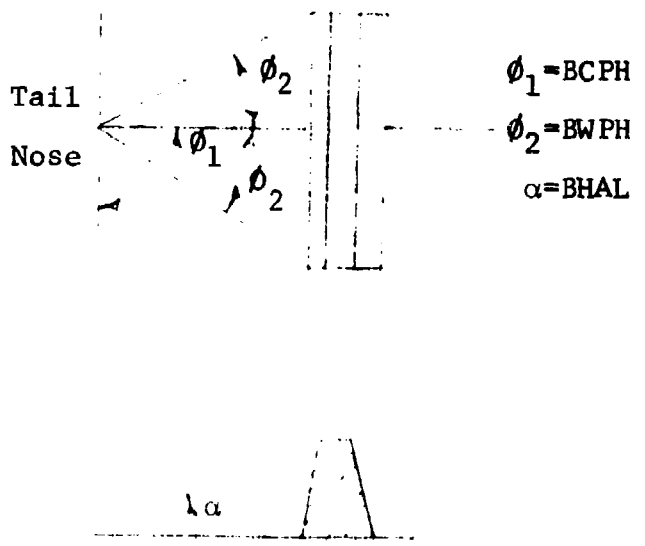
The following cards are read immediately after a $BOOM statement is encountered:

- A title card. Columns 1-72 contain a description of the fragment field.
- If NBAR>0, NBAR barrier description cards are read. The format of each card is:

COLS	FORMAT	NAME	DESCRIPTION
1-5	I5	IBTYP	Barrier type code. 1=barrier is a circular arc, with ordnance at center. 2=barrier is straight and is closest to the ordnance at its midpoint.
5-10			(ignored)
11-20	E10.0	BCPH	Azimuth angle (deg) of barrier midpoint
21-30	E10.0	BWPH	Extent of barrier in azimuth (deg) on either side of midpoint.
31-40	E10.0	BHAL	Angular height of barrier (deg), as seen from the ordnance. If IBTYP=2, this angular height is measured at the midpoint of the barrier.



Circular Barrier



Straight Barrier

```
$OUTPUT { ,TABLES                          }
        { ,PARAMS                          }
        { ,LIMITS                          }
        { ,EXEC[ ,PLOT][ ,KWIKPLOT]        } [ ,LIST]
        { ,RANGE                           }
        { ,SOJAC                           }
```

Read function definitions or compute and plot functions of the fragment field variables (velocity, striking angle, and number per square foot, as functions of range and azimuth).

The fragment field is read from binary unit 4, which is presumed to be properly positioned.

TABLES — Read Damage Threshold Velocity Tables or Formula Coefficients*.

Between two (2) and five (5) cards are read for each Table or Formula. The formats of these cards are given below. The end of input is marked by a card containing blanks or zeroes in columns 1 and 2.

PARAMS — Read a title card, and fragment threshold and target description cards for each fragment field function.

Between two (2) and four(4) cards are read for each function. The formats of these cards are given below. The end of input is marked by a card containing blanks or zeroes in columns 1 and 2.

LIMITS — Read an output parameter card for each fragment field function.

The format of this card is given below. The end of input is marked by a card containing blanks or zeroes in columns 1 and 2.

*The formula used is the empirical relationship derived under project THOR.

EXEC

Compute and plot specified functions of a previously computed fragment field. Functions are specified on the data card following this control card. The format of this card is given below.

The fragment field is read from binary unit 4, which is presumed to be properly positioned.

The functions are specified using two digit codes, whose meanings are:

01 total number of fragments (per sq ft)

02 total fragment mass (lb)

03 total fragment momentum (lb-sec/ft^2)

04 total fragment kinetic energy (ft-lb/ft^2)

05-07 (unused)

08* average fragment impact velocity (ft/sec)

09* average fragment impact angle (radians)

1j number of damaging fragments, using table j.

2j probability of damage, using table j.

3j compute and plot 1j and 2j together (this is cheaper than specifying 1j and 2j separately)

4j number of damaging fragments, using Thor coefficient set j.

5j probability of damage, using Thor coefficient set j.

6j compute plot 4j and 5j together (this is cheaper than specifying 4j and 5j separately)

7j-9j (unused)

Up to eight (8) functions may be computed and plotted at one time. To plot more than eight functions, the fragment field file must be repositioned, and another set of plots requested.

*Functions 08 and 09 must both be specified if either is, and must be the last function codes specified.

Note that a function code of 3j or 6j causes two plots to be generated, and that specification of functions 08 and 09 counts for 3 functions, even though only two plots are generated.

PLOT	cause function values to be written to the plot file (file 2) for later processing by a contour plotting program.
KWIKPLOT	generate contour plots of functions values on the printer.
RANGE	read plot range parameters from data card following this control card. The format of this card is given below.
SOJAC	read characters to be used for printer plotter (see KWIKPLOT) from data card following this control card. The format of this card is given below.
LIST	list data read with this control card.

Data cards read after a $OUTPUT,TABLES statement. Between two and five cards are read for each table or formula coefficient group.

COLS	FORMAT	NAME	DESCRIPTION
Heading Card			
1	I1	IT	type code. If IT=1,2 or 3, table IS is to be read. If IT=4,5 or 6, coefficient group IS is to be read. IF IT=0 and IS=0, the end of this group of data cards has been reached.
2	I1	IS	table or coefficient group number. $0 \leq IS \leq 9$.
3-5			(ignored)
6-10	I5	NTB	(used only for tables) number of M-V pairs to be read. $NTB \leq 14$.
11-15	I5	JDUM	(used only for tables) interpolation mode. All table values are read in true form, but linear interpolation of the table during processing may require either a log or linear scale, according to JDUM: =00 M linear, V linear =01 M linear, V log =10 M log, V linear =11 M log, V log
Coefficients (formula coefficients only)			
1-70	7E10.0	(COEF(I),I=1,7)	- Thor formula coefficients, group IS.
Fragment Mass Values (tables only)			
NTB masses are read, 7 per card.			
1-70	7E10.0	(XTB(I),I=1,7)	- Fragment mass values, (grains) for table IS.
Damage Threshold Velocity Values (tables only)			
NTB velocities are read, 7 per card.			
1-70	7E10.0	(YTB(I),I=1,7)	- Damage threshold velocities (ft/sec) for table IS.

Data cards read after a $OUTPUT,PARAMS statement. Between two and four cards are read for each fragment field function.

COLS.	FORMAT	NAME	DESCRIPTION
Heading Card			
1	I1	IT	Function type. If IT=0 and IS=0, the end of this group of data cards has been reached. If IT=0 and IS>0, then the function requires neither a table nor a Thor formula, and the function type is given by IS. IF IT>0, then IT determines the function type. =1 number of damaging fragments per sq ft, using table IS. =2 probability of damage, using table IS. =4 number of damaging fragments per sq ft using coefficient set IS in the Thor formula. =5 probability of damage using coefficient set IS in the Thor formula.
2	I1	IS	table number (if IT=1 or 2), coefficient group number (if IT=4 or 5), or function type (if IT=0): =1 total number of fragments per sq ft =2 total fragment mass (lb) per sq ft =3 total fragment momentum (lb-sec) per sq ft =4 total fragment kinetic energy (ft-lb) per sq ft =8 average fragment impact velocity (ft/sec) =9 average fragment impact angle (radians)
3			(ignored)
4	A1	IVP	Function variant code. A blank is treated as a 'zero' character. IVP is used to distinguish plots of the same function, but with different parameters.
5			(ignored)
6-10	I5	NF	Number of parameters ($0 \leq NF \leq 14$) If $NF < 14$, parameters not read are set to zero.

COLS.	FORMAT	NAME	DESCRIPTION

Title Card

1-72	12A6	(ITTLF(I),I=1,12)	- function title, which will appear on all plots of this function.

Parameter Card(s)

NF parameters are read, 7 per card. If NF=0, no cards are read.

1-70	7E10.0	(F(I),I=1,NF)	- parameters affecting function performance.

Parameters 1 through 7 have the same meaning for all functions:

F(1) miminum fragment weight (grams). All lighter fragments are ignored.

F(2) minimum fragment velocity (ft/sec). All slower fragments are ignored.

F(3) minimum fragment momentum (lb-sec). All fragments whose momentum is too small are ignored.

F(4) minimum fragment kinetic energy (ft-lb). All fragments whose kinetic energy is too small are ignored.

F(5) - F(7) (reserved)

Parameters 8 through 14 are used only if IT>0.

F(8) target plan area (ft^2)

F(9) average target elevation area (ft^2)

F(10) - F(11) are used only if IT=4 or 5.

F(10) target thickness (in.)

F(11) target hardness, where applicable.

F(12) - F(14) (reserved)

Data cards read after a $OUTPUT,LIMITS statement. One card is read for each function.

COLS.	FORMAT	NAME	DESCRIPTION
1	I1	IT	Function code (see PARAMS discussion above)
2	I2	IS	subcode or table code (see PARAMS discussion above) Note that if IT=0 and IS=0 the end of this group of data cards has been reached.
3-10			(unused)
11-14	I4	ISM	Plot smoothing code. IF ISM=0, do no smoothing. IF ISM=1 to 5, smooth by replacing each value by the average of the values in a 2*ISM+1 raster square centered on that value.
15	I1	ISC	Plot scale code. ISC=0 to request a linear plot. ISC=1 to request a plot of the log of the function.
16-20			(unused)
21-30	E10.0	OZMIN	minimum value to be plotted.
31-40	E10.0	OZMAX	maximum value to be plotted. If OZMIN≥OZMAX, the plot range is set internally.

COLS.	FORMAT	NAME	DESCRIPTION

Data card read after $OUTPUT,EXEC statement. See discussion of the EXEC parameter for the list of valid function codes.

1-2	I2		first function code
3-5			(ignored)

Subsequent function codes go in columns 6-7, 11-12, etc.

Data card read after $OUTPUT,SOJAC statement.

1-5			(ignored)
6-10	I5	NZ	number of function value ranges to be plotted on printer. NZ≤50
11-19			(ignored, leave blank)
20-69	50A1	(CH(I),I=1,NZ)	- plot characters for each function value range.

Data card read after $OUTPUT,RANGE statement.

1-5			(ignored)
6-10	I5	NX	number of resolution increments in X direction (tail to nose). NX≤81.
11-15	I5	NY	number of resolution increments in Y direction (laterally from waist of munition). Normally NY=(NX-1)/2+1. NX≤41.
16-20			(unused)
21-30	E10.0	XMIN	minimum X value (ft)
31-40	E10.0	YMIN	minimum Y value (ft). Normally YMIN=0.
41-50	E10.0	DX	X increment (ft). Normally NX,XMIN and DX are set so that XMAX=-XMIN where XMAX is XMIN+DX*(NX-1).
51-60	E10.0	DY	Y increment (ft). Normally NY and DY are set so that YMAX=XMAX where YMAX is DY*(NY-1) if YMIN=0.

Sample Data Deck

The deck listed on the next page is set up to do the following job:

1) Read munition data for munition No. 4 (mod 1). Gaps in the data are to be filled in, and the data is to be listed neatly in tabular form.

2) A fragment field is to be computed, on a polar grid consisting of 8 ranges spaced 250 feet apart, and 18 azimuth rays, spread evenly over a half-circle (at azimuths 5, 15, ..., 175 deg). After the fragment field has been written onto tape 4, an end of file is written and the tape is rewound.

3) Output specifications are read. These include
   a. the printer-plotter character set,
   b. the size and resolution of the plot,
   c. the trivial (0-velocity) fragment threshold velocity table,
   d. the function limits and scale (all are to be plotted in log units), and
   e. the function titles and parameters.

4) Plots are generated. The function specifications 01 and 30 cause functions 01, 10 and 20 to be generated and plotted.

```
$PRINT CONTROL,DATA
$ORDNANCE FORMAT=2,NORM,SMOOTH,LIST
MUNITION 04 MOD 1
   16    36     4-1    660.
```

(The munition data goes here.)

```
$BOOM NAZ=18,DRNG=250.,NRNG=8     $    2000 FOOT RADIUS, 10 DEG AZ SECTORS
GROUND LEVEL FRAGMENT FIELD
$TIME
$TAPE 4,WEOF
$TAPE 4,REWIND
$OUTPUT SOUAC LIST
          9          -123456789+
$OUTPUT RANGE LIST
         81    41      -2000.     0.          50.         50.
$OUTPUT TABLES LIST
10        2    0
0.         100000.
0.         0.
  END
$OUTPUT LIMITS LIST
01              1      .00016667 .16667
10              1      .00016667 .16667
20              1      .0001     .1
  END
$OUTPUT PARAMETERS LIST
01        4
AREAL DENSITY OF ALL FRAGMENTS (TARGET IS SPHERE OF UNIT CROSS-SECTION)
0.         0.          0.          0.
10        9
AREAL DENSITY OF DAMAGING FRAGMENTS (K.E. GT 58 FT-LB)
0.         0.          0.          58.
1.33       9.0
20        9
PROBABILITY OF HIT BY DAMAGING FRAGMENT (K.E. GT 58 FT-LB)
0.         0.          0.          58.
1.33       9.0
  END
$OUTPUT EXEC LIST KWIKPLOT
01    30
$TIME
$STOP
```

Reproduced from best available copy.

Sentinel Records

The munition data file (if one is used), the fragment field file, and the output function file (if one is used) each consist of several groups of data. Each group is preceded by a sentinel record which identifies the group, as described in the following section. In addition a special sentinel record is written after the last data group.

A sentinel record exists in two forms, one for formatted files (the BCD munition data file), and one for unformatted files (the binary fragment field file and the binary output function file).

Formatted sentinel record:

char. 1-6	'*HEAD*'
char. 7-14	an 8-character search key. (If fewer than 8 characters, blanks are added.)

Search for a formatted data group using the statement:

```
$TAPE n,BCD,SRCH=key,BKSP
```

Unformatted sentinel record:

word 1*	'*HEAD*'
word 2**	an 8-character search key. (If fewer than 8 characters, blanks are added.)

Search for an unformatted data group using the statement:

```
$TAPE n,SRCH=key,BKSP
```

*Two words are needed when using a computer with four characters per word.

**Two words are needed when using a computer with four or six characters per word.

Sentinel Record Keys

1. A 3-character key of the form

   XXI

   is used for munition data. (BCD tape)

   XX = munition ID number. $00 \leq XX \leq 99$

   I = a modification code. Normally I=0.

2. A 5-character key of the form

   XXIJK

   is used for fragment field tables. (Binary tape 4)

   XX,I as above

   J,K = fragment field modification code to reflect changes in resolution, target plane altitude, barrier placement, etc.). Normally J=K=0.

3. An 8-character key of the form

   XXIJKYYL

   is used for output function tables. (Binary tape 2)

   XX,I,J,K as above

   YY = output function code. (See $OUTPUT discussion.)

   L = output function modification code(to reflect changes in threshold or target parameters, etc.). Normally L=0.

4. The sentinel record which should end a file has an 8-character key which is

   99999999

   A sentinel record may be written with the statement

   $TAPE n[,BCD],WEOF

Timing

The generation of the fragment field is the most time consuming part of the program. Several parameters affect timing in this phase.

NM	number of mass categories into which munition data is divided
NAZ	number of azimuth angles
NEL	number of elevation angles
DRNG	range increment (ft)

All operations are performed for each mass category at each azimuth. Low-register fragment trajectories are computed for ranges R_i=DRNG*i, out to $R_i=R_{max}$, which corresponds to an initial elevation angle of α_{max}. High-register fragment trajectories are computed for elevations $\alpha_i=(i-1/2)\Delta\alpha$, where $\Delta\alpha=90°/NEL$, and where $\alpha_{max}<\alpha_i<90°$ holds.

Thus the computation time for low register fragment trajectories increases as DRNG decreases, and the computation time for high register fragment trajectories increases as NEL increases. So the expected computation time is given by

T=A*NM*NAZ*(1+B/DRNG+C*NEL)

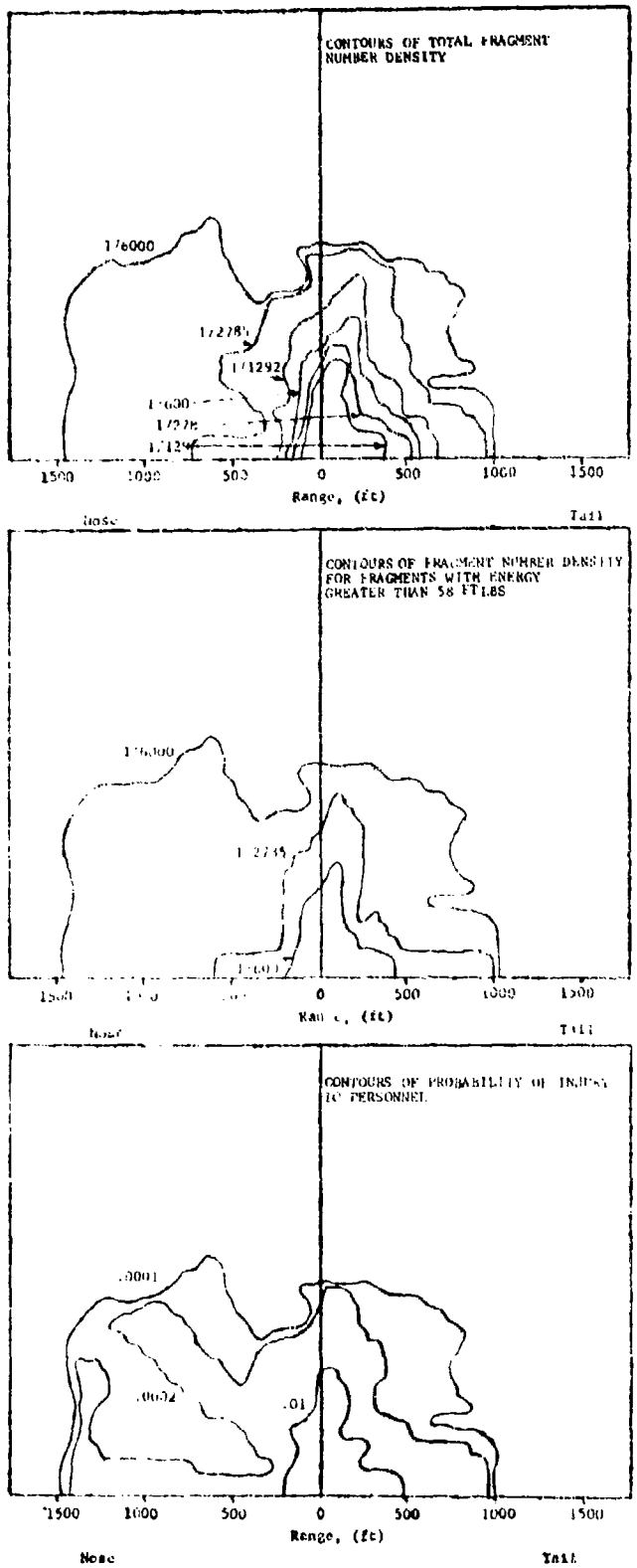
Because all runs done for this project used the same values of NAZ, DRNG and NEL, the constants A, B and C cannot be determined at this time. However, the approximate relation (in sec)

T=7 x NM

and has been observed on the Univac 1108, with NAX=18, DRNG=250 and NEL=18.

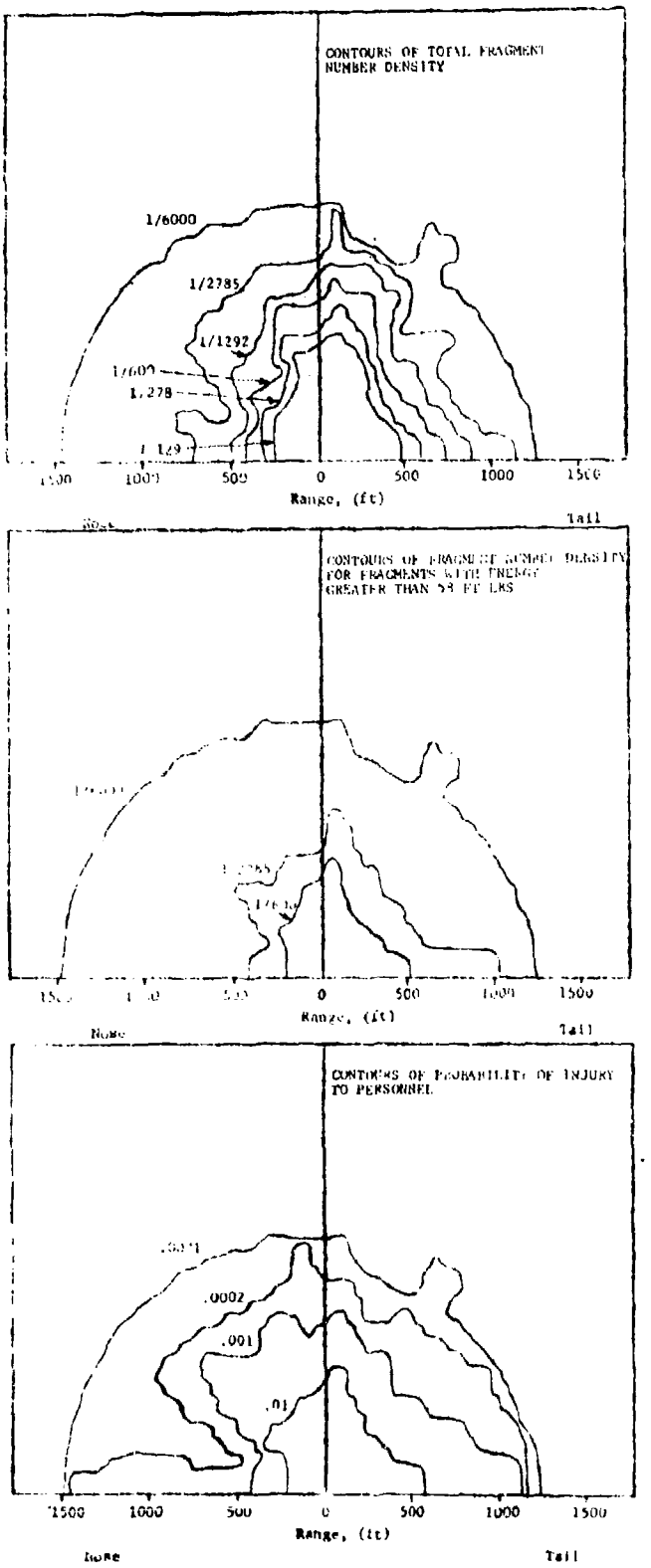
APPENDIX B

CONTOURS OF FRAGMENT NUMBER DENSITIES AND INJURY PROBABILITIES



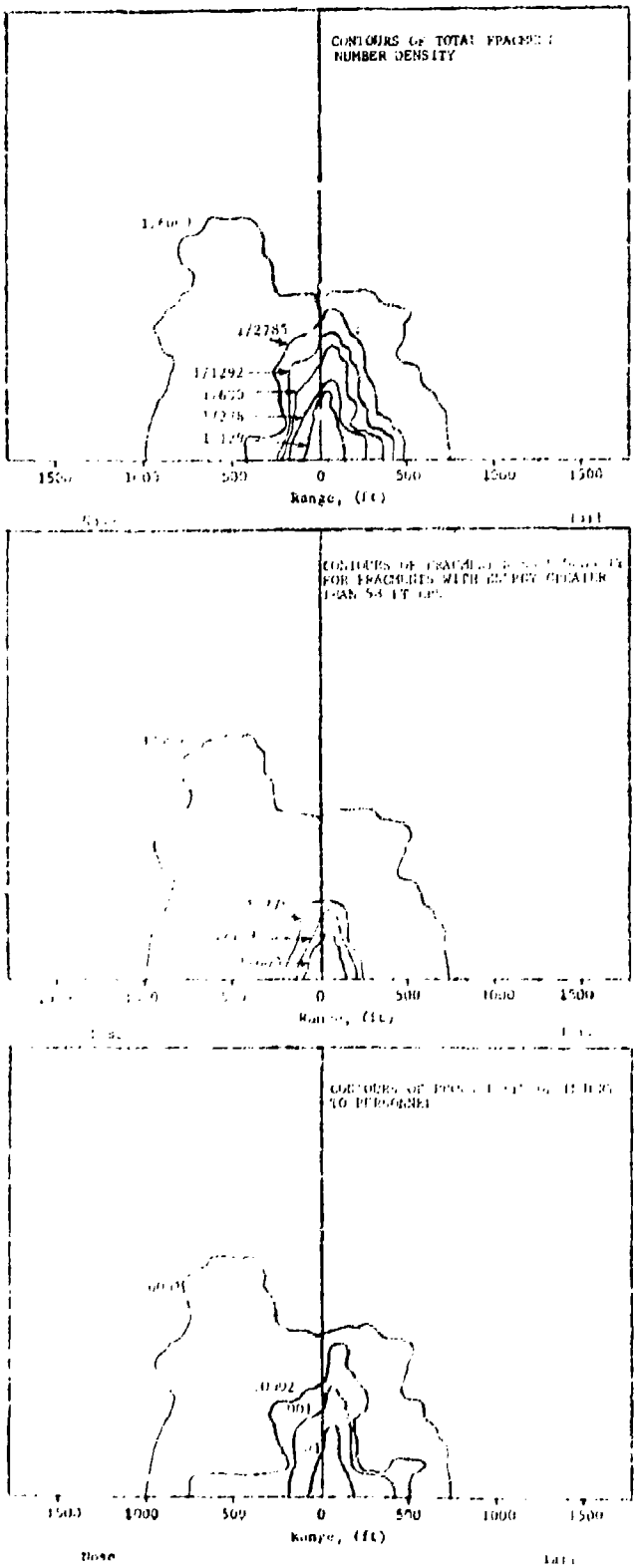
Reproduced from best available copy.

Fig. B1 500 LB LOW DRAG BOMB MARK 82 MOD 1 (H-6 LOAD)



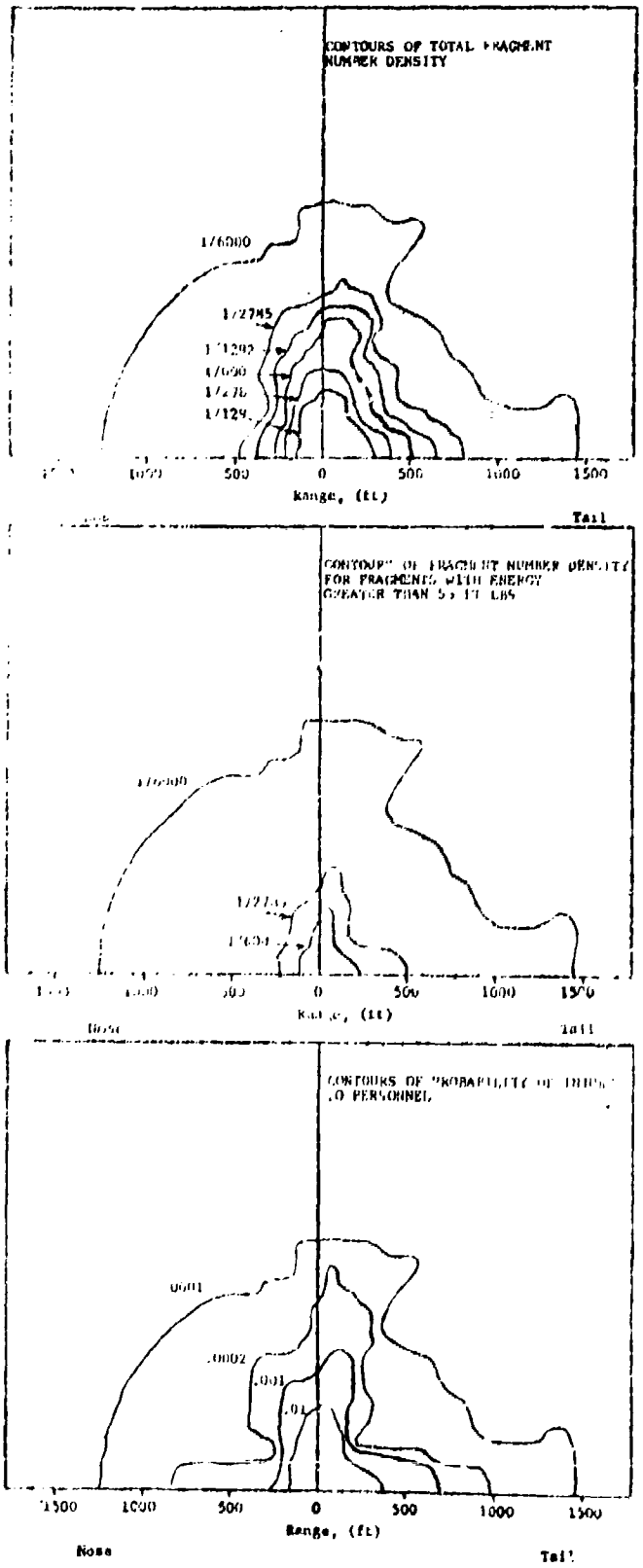
Reproduced from best available copy.

Fig. B2 750 LB BOMB M117A2
(TRITIONAL LOAD)



Reproduced from best available copy.

Fig. B3 105 MM HOWITZER SHELL M1
(COMP. B LOAD)



Reproduced from best available copy.

Fig. B4 155 MM HOWITZER SHELL M107 (COMP. B LOAD)

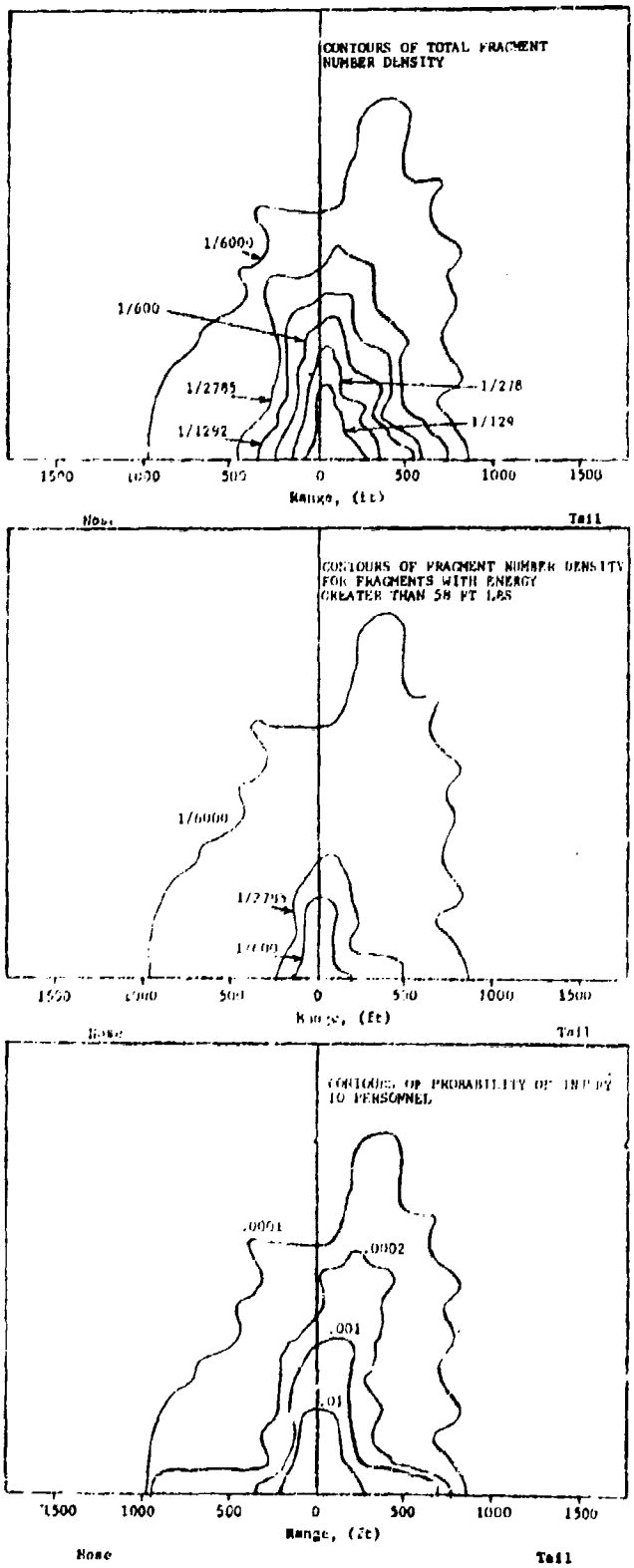
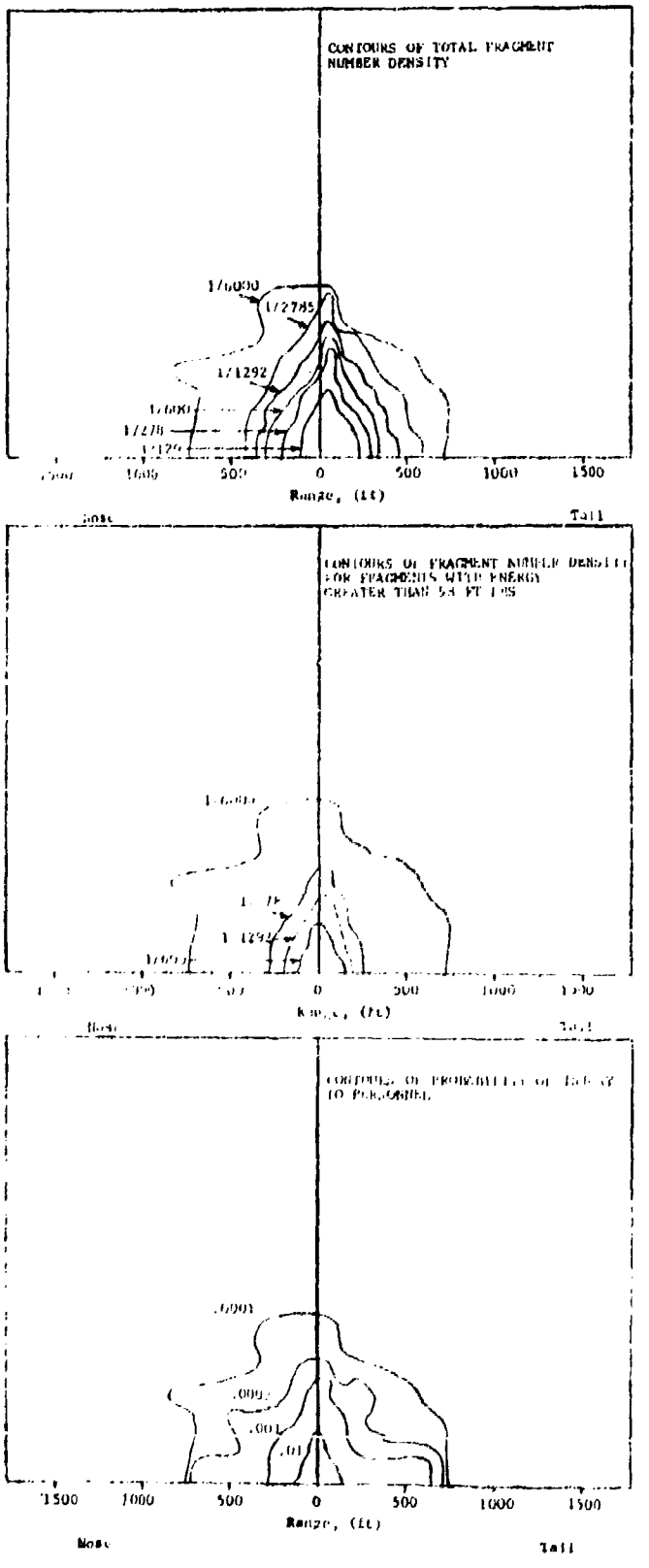
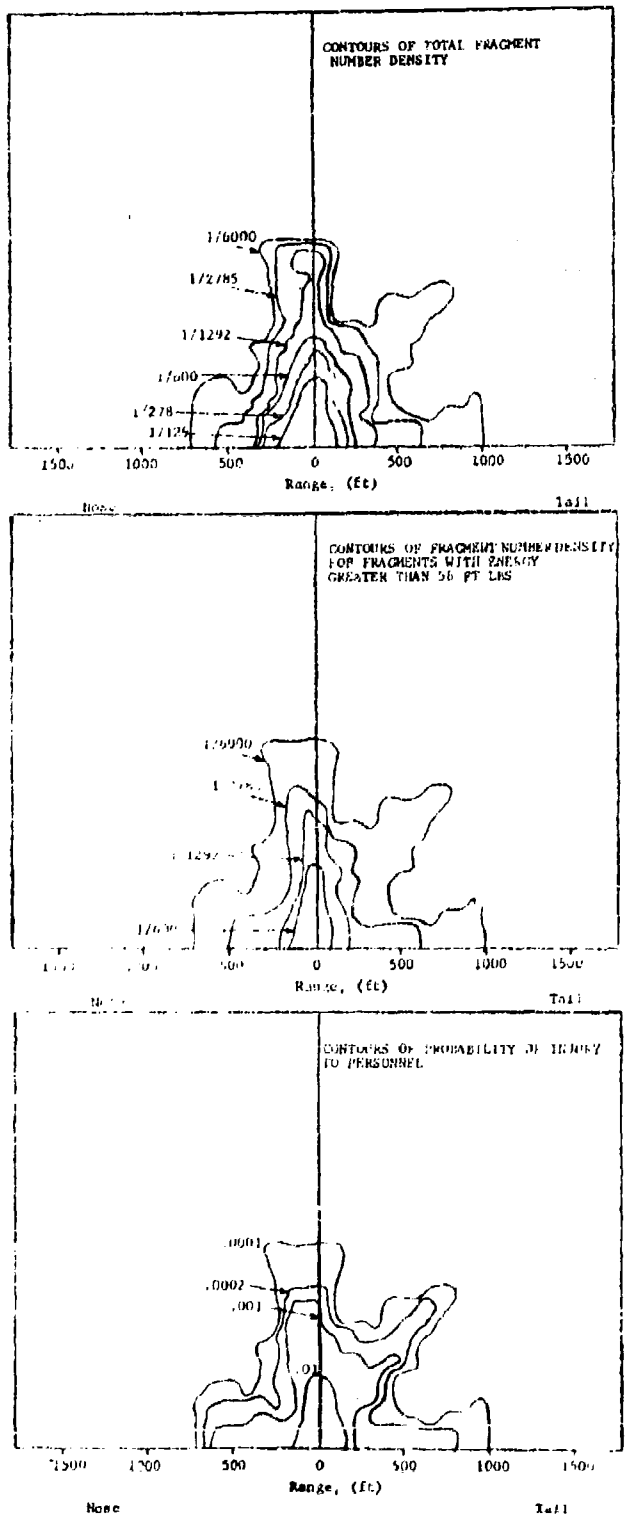


Fig. B5 175 MM GUN SHELL M437A2 (COMP. B LOAD)



Reproduced from best available copy.

Fig. B6 5"/38 PROJECTILE MARK 49 (COMP. A3 LOAD)

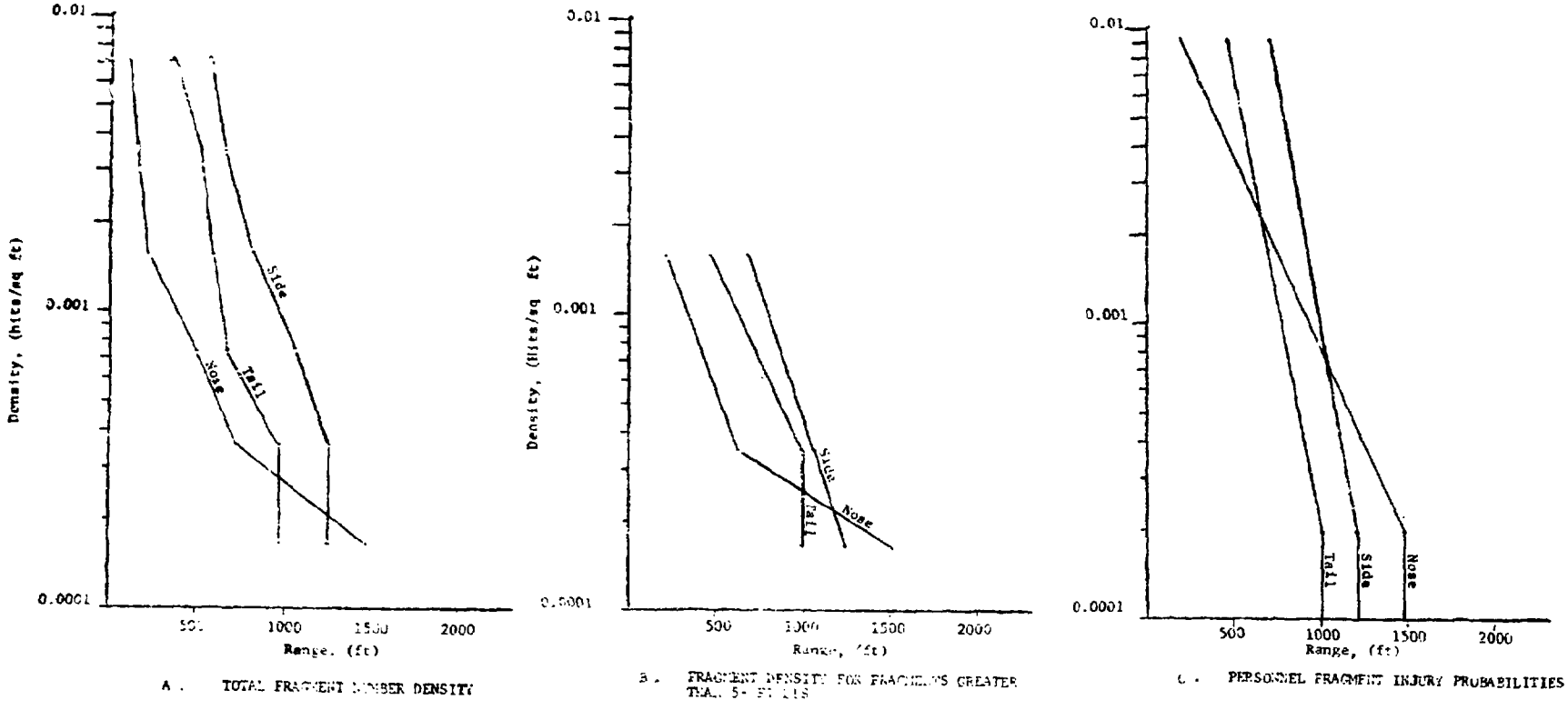


Reproduced from best available copy.

Fig. B7 8"/55 PROJECTILE MARK 25 (EXPLOSIVE D LOAD)

APPENDIX C

AVERAGE CURVES OF FRAGMENT NUMBER DENSITIES AND INJURY PROBABILITIES



Reproduced from best available copy.

Fig. C1 500 LB LOW DRAG BOMB MARK 82 MOD 1 (H-6 LOAD)

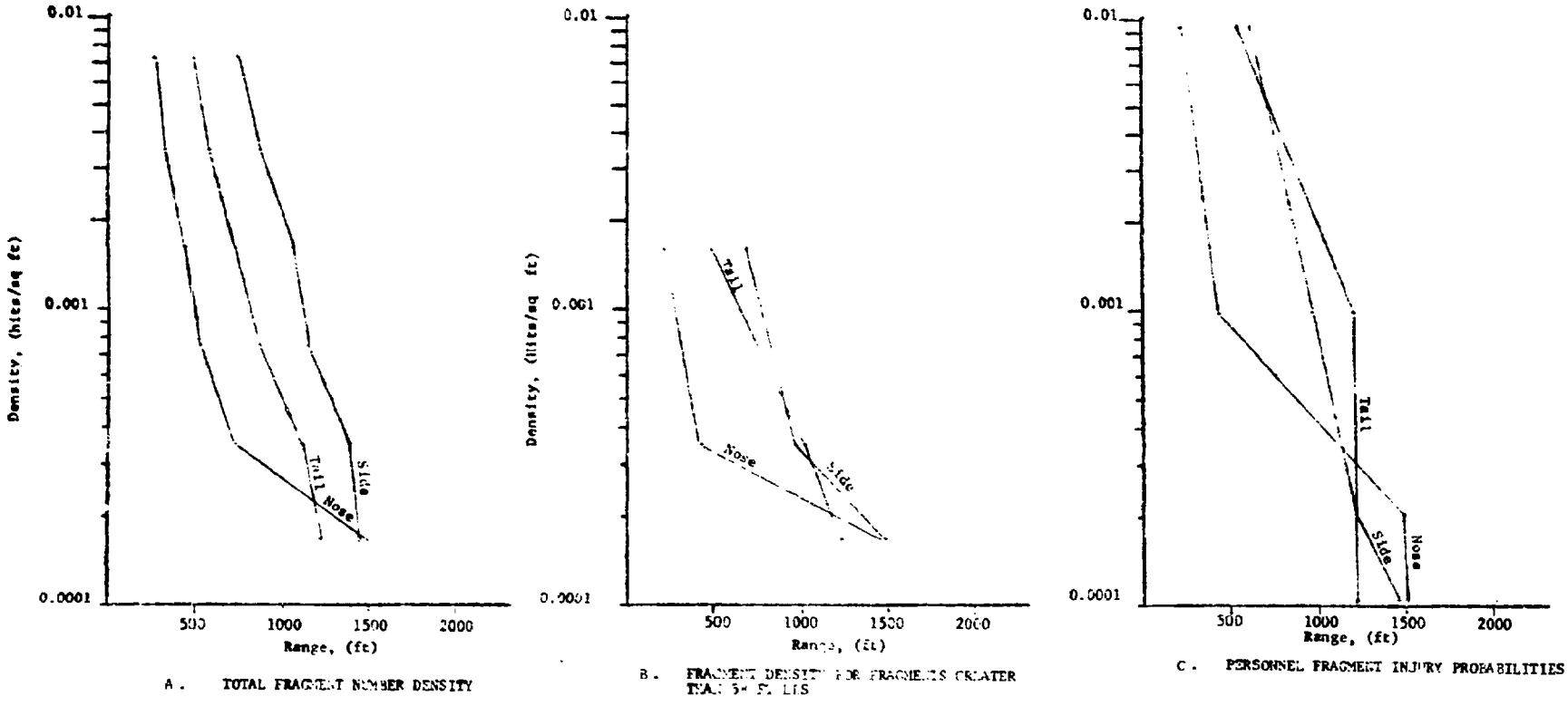
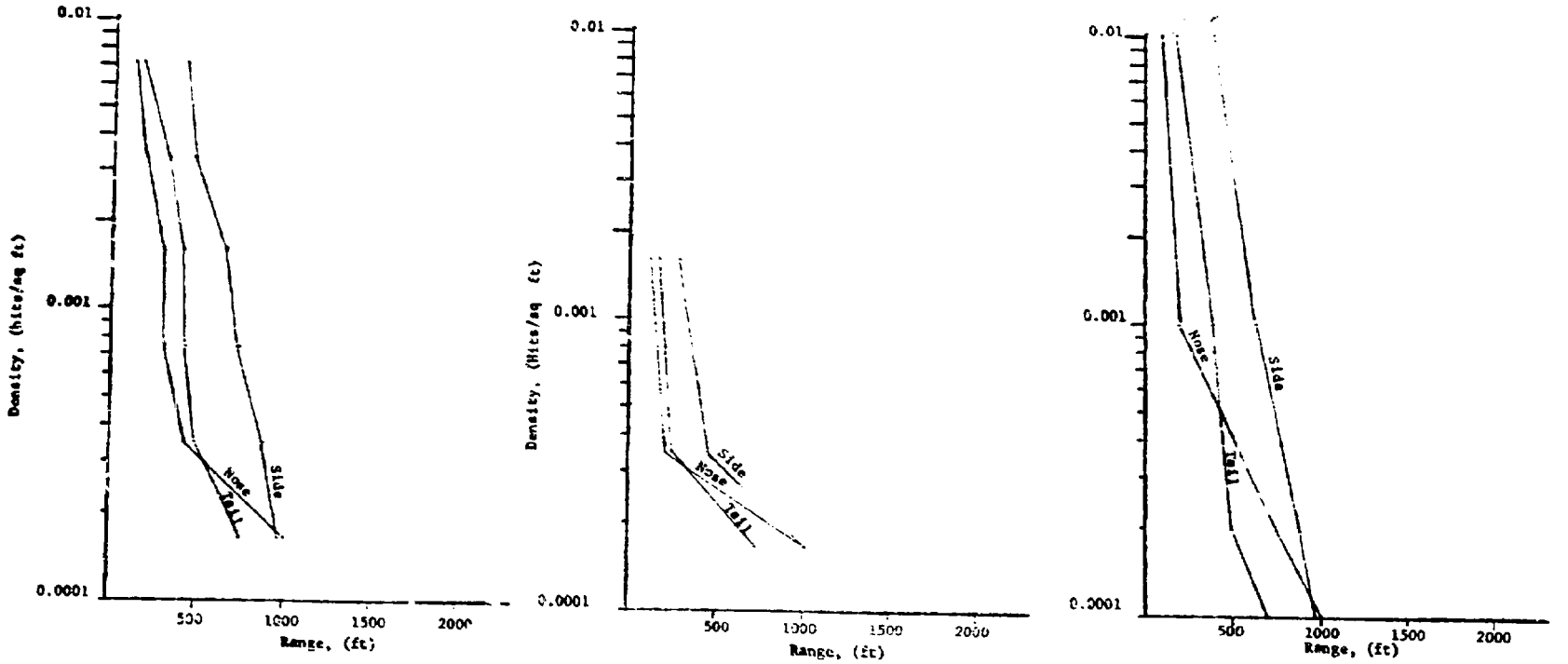


Fig. C2 750 LB BOMB M117A2 (TRITIONAL LOAD)

Reproduced from best available copy.



Reproduced from best available copy.

Fig. C3 105 MM HOWITZER SHELL M1 (COMP. B LOAD)

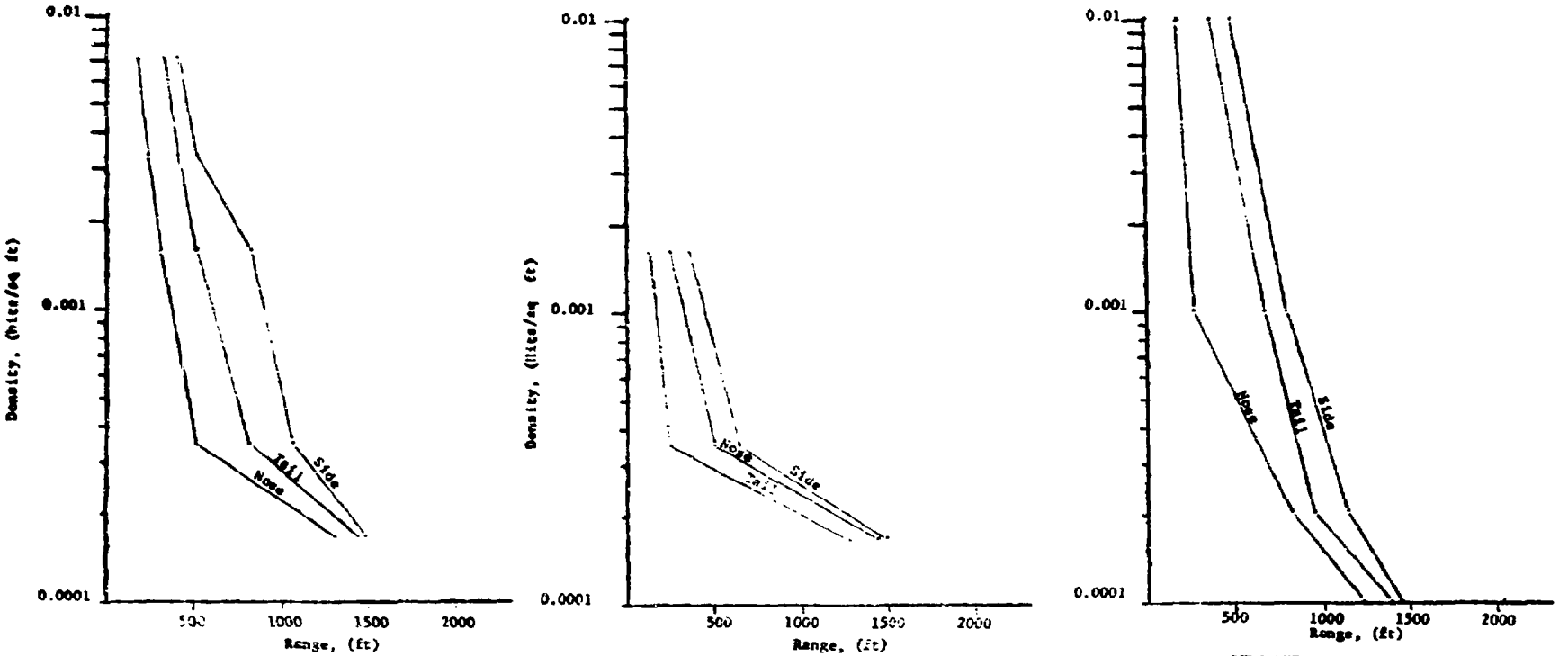


Fig. C4 155 MM HOWITZER SHELL M107 (COMP. B LOAD)

Reproduced from best available copy.

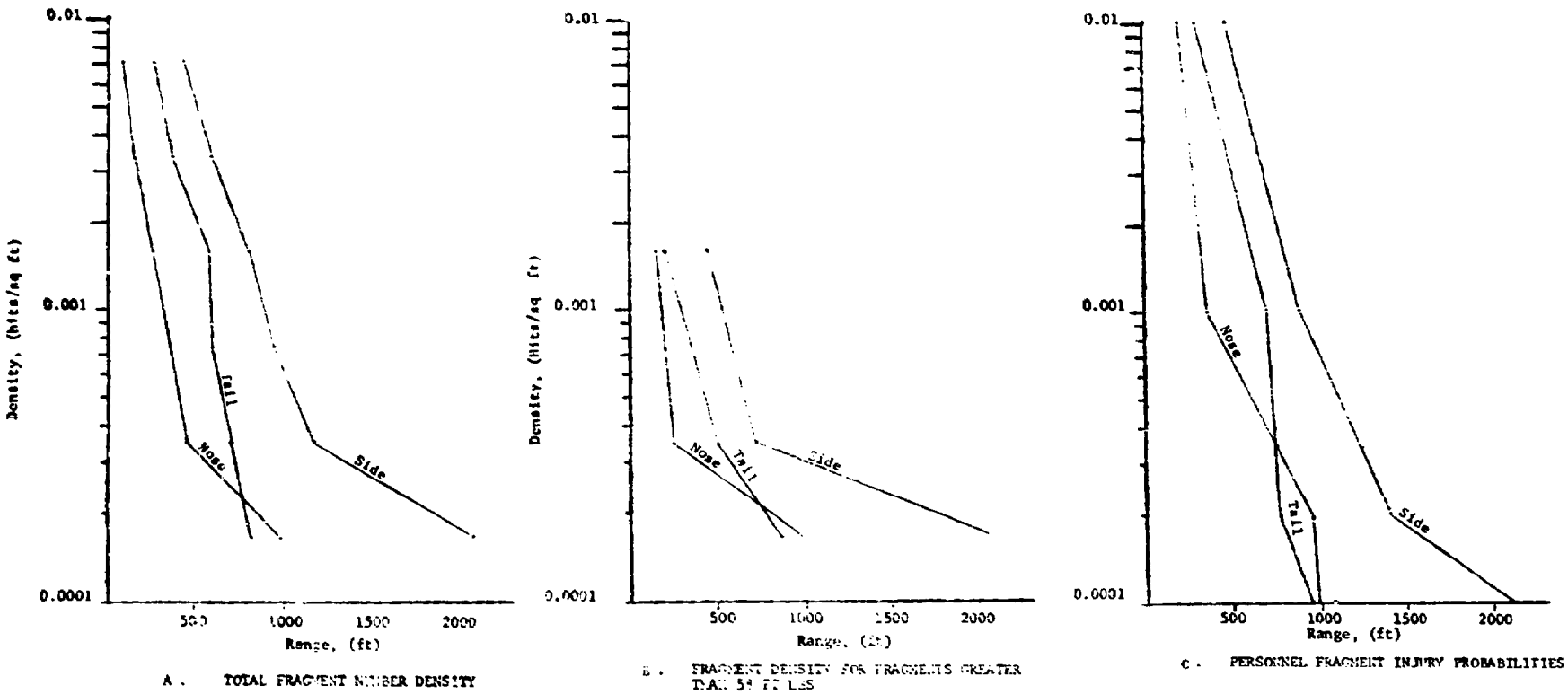


Fig. C5 175 MM GUN SHELL M437A2 (COMP. B LOAD)

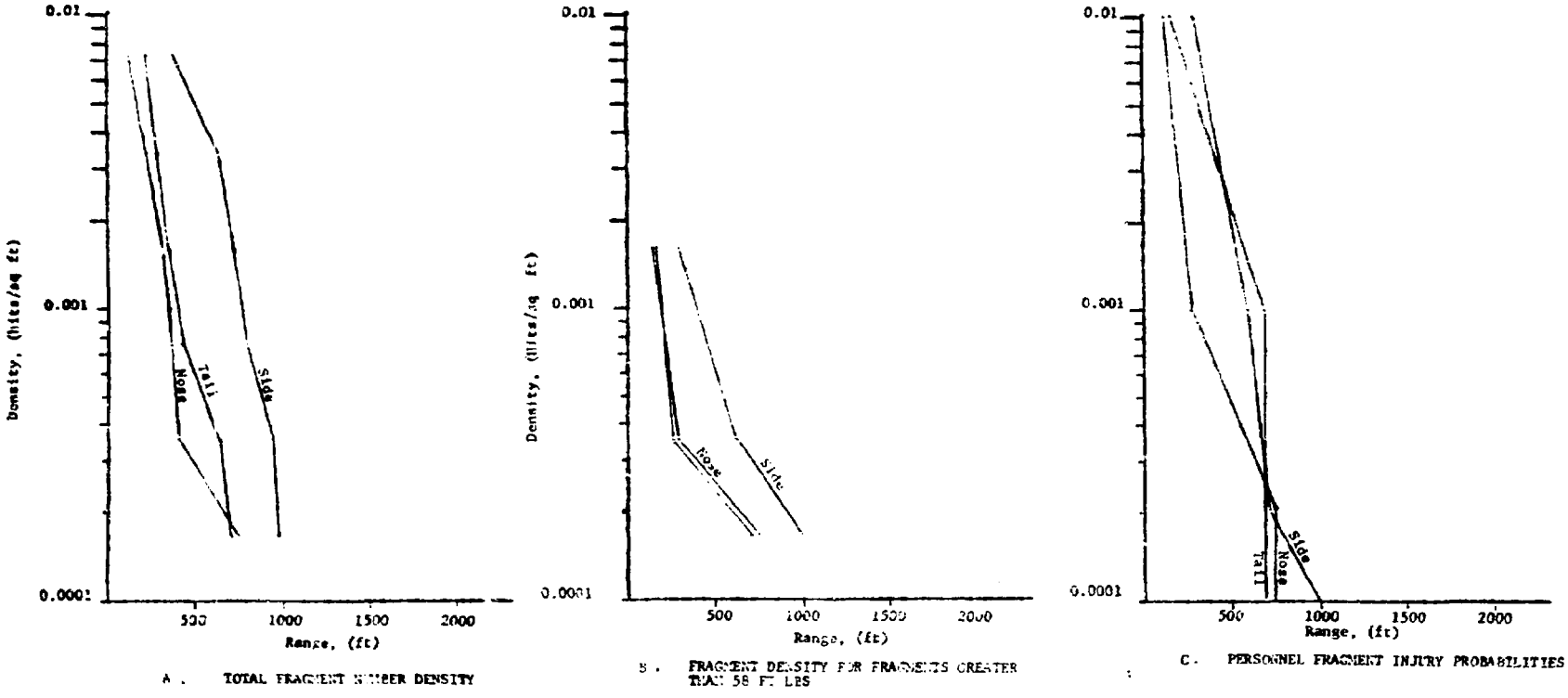


Fig. C6 8"/55 PROJECTILE MARK 25 (EXPLOSIVE D LOAD)

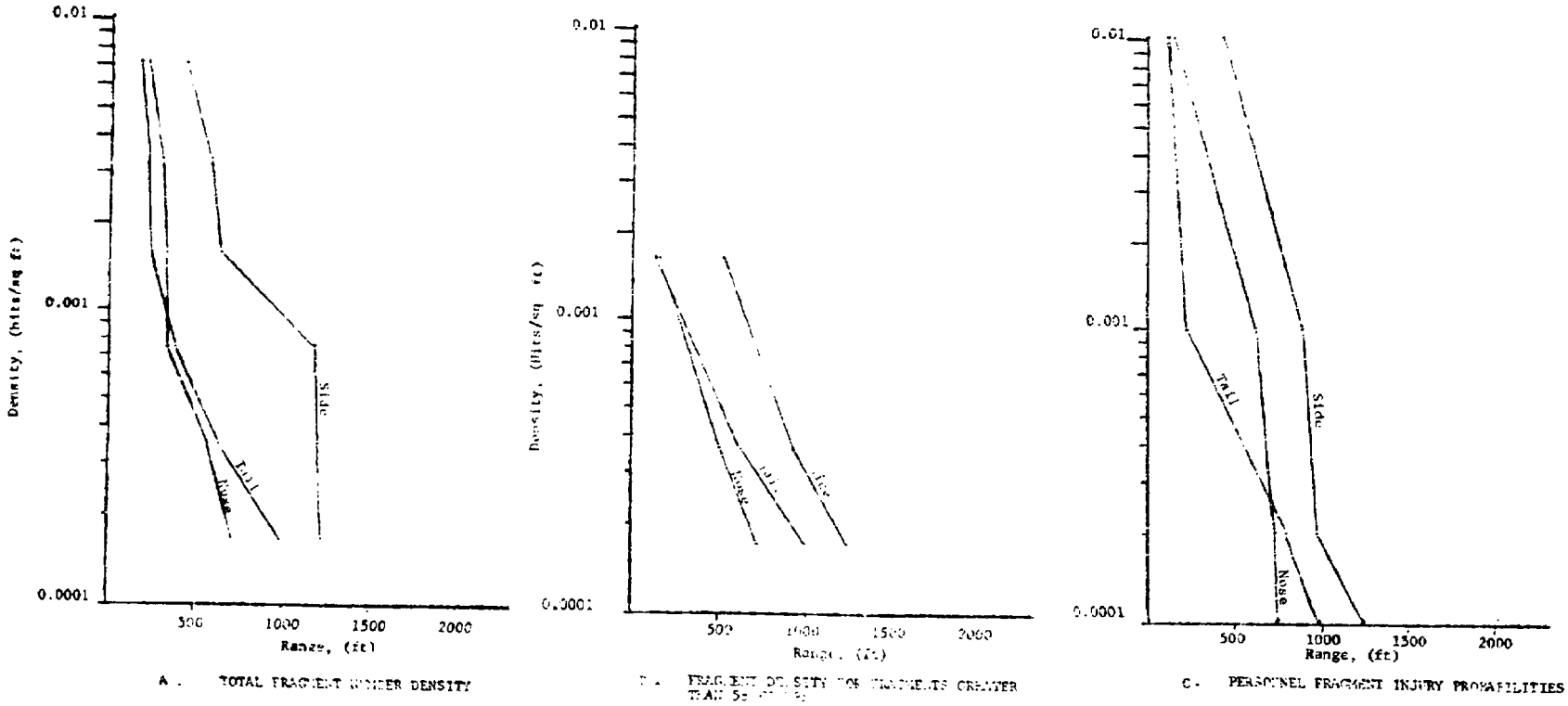


Fig. C7 8"/55 PROJECTILE MARK 25 (EXPLOSIVE D LOAD)

APPENDIX D

COMPUTER PROGRAM LISTINGS

```
C     GIVEN STD TRAJECTORY TABLES (TAPE1) AND ORDNANCE DATA (INPUT OR   TRAG0010
C     RCD TAPE), GENERATE TABLE OF FRAGMENT INFO. ON SEMI-CIRCULAR GRID.TRAG0020
C     STORE FRAGMENT FIELD TABLES ON TAPE 4, IF REQUESTED.              TRAG0030
C     THEN COMPUTE OPTIONAL FUNCTIONS OF FRAGMENT INFO.  OUTPUT FUNCTIONTRAG0040
C     I.D. CODES AND VALUES (TAPE2)                                     TRAG0050
C                                                                       TRAG0060
C  CONTROL CARD TABLES AND GENERAL PROGRAM CONTROL VARIABLES            TRAG0070
      COMMON /CNTRL/ GCCRD,GCCPRT,KTCD,FCC(8),KARD(14),PC,PS,PD,PP,     TRAG0080
     * TXTIME(2),TXDATE(2),WHY(20),KKARD(800)                            TRAG0090
      LOGICAL GCCRD,GCCPRT,WHY                                          TRAG0100
      INTEGER TXTIME,TXDATE,FCC,PC,PS,PD,PP                             TRAG0110
C                                                                       TRAG0120
      EQUIVALENCE (KKARD(12),KTYPE)                                     TRAG0130
C                                                                       TRAG0140
      LOGICAL PRTCCD                                                    TRAG0150
      EQUIVALENCE (KKARD(497),PRTCCD)                                   TRAG0160
C                                                                       TRAG0170
C  SYNTAX TABLES                                                        TRAG0180
      INTEGER XSYN(64),TSYN(64),SSYN(4),SYND,SYNB                       TRAG0190
      LOGICAL SYNTAX                                                    TRAG0200
      EXTERNAL CCARD                                                    TRAG0210
C***  SYND IS A MARKING CHARACTER TO POINT AT SYNTAX ERRORS (11-7-8 PCH)TRAG0220
      DATA NSSYN,SYND,SYNB/4,1Ha,1H /,                                  TRAG0230
     * TSYN/5*0,4,30*0,5,5,0,2,3*0,6,0,0,3,11*0,6,5*0/                  TRAG0240
C***  STD CONTROL-CARD SYNTAX RULES, MODIFIED FOR $=END OF STATEMENT    TRAG0250
C***  KKARD(4)=2 (NEW STMNT ON SAME CARD) SHOULD THEN NEVER APPEAR      TRAG0260
      DATA XSYN/                                                        TRAG0270
     * 00202000200,000000020000, 00202000400,000000020011,             TRAG0280
     * 00202000200,000000050000, 00202000200,000000070000,             TRAG0290
     * 00202000400,000000050006, 00202000300,000000160000,             TRAG0300
     * 00202000400,000000070010, 00202000300,000000260000,             TRAG0310
     * 00202000000,000010014001 2, 00202000200,000000000013,           TRAG0320
     * 00202000100,000000000000, 00202000400,000000140015,             TRAG0330
     * 00202000300,000000160022, 00202000400,000000160017,             TRAG0340
     * 00204000102,000000200024, 00701000403,000000000000,             TRAG0350
     * 00701000402,000000000000, 00202000400,000060000023,             TRAG0360
     * 00202000100,000060000024, 00202000000,000200330025,             TRAG0370
     * 00202000600,000100026000, 00202000100,000000270030,             TRAG0380
     * 00701000404,000000000000, 00202000600,000000310032,             TRAG0390
     * 00202000600,000070026001 4, 00701000000,000070026000,           TRAG0400
```

Reproduced from best available copy.

```
     * 00202000400,0000000330034, 00202000500,0000300360035,          TRAG0410
     * 00300000000,0000500150000, 00202000400,0000000360037,          TRAG0420
     * 00202000000,0000400140040, 00202000600,0000000260000/          TRAG0430
C  MISCELLANEOUS DECLARATIONS                                          TRAG0440
      LOGICAL COMMNT                                                   TRAG0450
      DATA KNCOMM/1H*/                                                 TRAG0460
C                                                                      TRAG0470
C  ONCE-ONLY CODE                                                      TRAG0480
C                                                                      TRAG0490
      CALL DATE(9,TXDATE)                                              TRAG0500
      CALL TOD(8,TXTIME)                                               TRAG0510
      CALL KLOCK                                                       TRAG0520
      CPTIME = 0.                                                      TRAG0530
      WRITE(6,8) TXTIME,TXDATE                                         TRAG0540
    8 FORMAT(1H1,30X,6HTIME   ,A6,A2,5X,6HDATE   ,A6,A3//)             TRAG0550
      KKARD(4) = 1                                                     TRAG0560
C                                                                      TRAG0570
C  READ CONTROL CARD AND INTERPRET                                     TRAG0580
C                                                                      TRAG0590
  100 IF(QCCRD) READ(5,101,END=200) KARD                               TRAG0600
  101 FORMAT(14A6)                                                     TRAG0610
      QCCRD = .TRUE.                                                   TRAG0620
      IF (KKARD(4).EQ.2) GO TO 112                                     TRAG0630
      KTCD = KTCD+1                                                    TRAG0640
      QCCPRT = PRTCCD                                                  TRAG0650
C   * IN CC1 MEANS COMMENT                                             TRAG0660
      COMMNT = KOM(KARD,1,1,KNCOMM,1).EQ.0                             TRAG0670
      IF (COMMNT) GO TO 106                                            TRAG0680
      IF(.NOT.PRTCCD) GO TO 110                                        TRAG0690
C  LIST CONTROL CARD IF SWITCH IS ON                                   TRAG0700
  106 WRITE(6,FCC) PC,KTCD,KARD                                        TRAG0710
      PC = PS                                                          TRAG0720
      IF (COMMNT) GO TO 100                                            TRAG0730
C  SEARCH FOR C.C. NAME, AND INTERPRET CARD                            TRAG0740
  110 KSYN = 1                                                         TRAG0750
      LSYN = 72                                                        TRAG0760
  112 I = KKARD(4)                                                     TRAG0770
      KKARD(4) = 1                                                     TRAG0780
      IF(SYNTAX(KARD,KSYN,LSYN,XSYN,I,TSYN,CCARD,SSYN,NSSYN,KKARD))    TRAG0790
     * GO TO 119                                                       TRAG0800
```

```
      IF(KTYPE.LE.0) WHY(1)=.TRUE.                                      TRAG0810
      IF(KTYPE.GT.0) WHY(KTYPE)=.TRUE.                                  TRAG0820
      IF(.NOT.GCCPRT) WRITE(6,FCC) PC,KTCD,KARD                         TRAG0830
      GCCPRT = .TRUE.                                                   TRAG0840
      IF(KKARD(13).GT.0) KSYN=KKARD(13)                                 TRAG0850
      WRITE(6,111) (SYNB,I=1,KSYN),SYND                                 TRAG0860
  111 FORMAT(17X,10A4)                                                  TRAG0870
      WRITE (6,113)                                                     TRAG0880
  113 FORMAT(44H ******  ERROR IN ABOVE CONTROL CARD  ******)           TRAG0890
      CALL FLUSH                                                        TRAG0900
      GO TO 100                                                         TRAG0910
  119 IF(KKARD(4).GE.3) GO TO 100                                       TRAG0920
      IF (KKARD(4).EQ.2) GCCRD=.FALSE.                                  TRAG0930
      GO TO (100,200,200,400,500,600,700,800,900                        TRAG0940
     * ),KTYPE                                                          TRAG0950
C                                                                       TRAG0960
C  END OF PROCESSING                                                    TRAG0970
C                                                                       TRAG0980
  200 STOP                                                              TRAG0990
C                                                                       TRAG1000
C  COMPUTE BASIC FRAGMENTATION INFO. TABLE                              TRAG1010
C                                                                       TRAG1020
  400 CALL LINK1(3)                                                     TRAG1030
      GO TO 9000                                                        TRAG1040
C                                                                       TRAG1050
C  DEFINE TYPES OF OUTPUT FUNCTIONS.  ALL MUST YIELD VALUES AT (R,PHI)  TRAG1060
C  DEPENDENT ONLY ON R. PHI. AND FRAG. INFO AT (R,PHI).                 TRAG1070
C                                                                       TRAG1080
  500 CALL LINK2                                                        TRAG1090
      GO TO 9000                                                        TRAG1100
C                                                                       TRAG1110
C  POSITION A FILE                                                      TRAG1120
C                                                                       TRAG1130
  600 CALL TAPE                                                         TRAG1140
      GO TO 9000                                                        TRAG1150
C                                                                       TRAG1160
C  READ, NORMALIZE, SMOOTH (AND LIST) ORDNANCE DATA                     TRAG1170
C                                                                       TRAG1180
  700 CALL LINK1(1)                                                     TRAG1190
      GO TO 9000                                                        TRAG1200
```

```
C                                                                       TRAG1210
C  READ AND PRINT CLOCK                                                 TRAG1220
C                                                                       TRAG1230
  800 CALL KLOCK(DUM)                                                   TRAG1240
      DUM1 = DUM-CPTIME                                                 TRAG1250
      IF (PC.EQ.PS) PC=PD                                               TRAG1260
      WRITE (6,801) PC,DUM,DUM1                                         TRAG1270
  801 FORMAT(A1,33X,27H******  C.P. TIME   TOTAL =,F8.3,9H SEC  (UP,    TRAG1280
     * F7.3,13H SEC)  ******)                                           TRAG1290
      CPTIME = DUM                                                      TRAG1300
      PC = PD                                                           TRAG1310
      GO TO 100                                                         TRAG1320
C                                                                       TRAG1330
C  UNUSED STATEMENTS                                                    TRAG1340
C                                                                       TRAG1350
  900 CONTINUE                                                          TRAG1360
      GO TO 200                                                         TRAG1370
C                                                                       TRAG1380
C  AFTER EVERY COMMAND, TEST ABORT FLAG BEFORE READING NEXT COMMAND     TRAG1390
C                                                                       TRAG1400
 9000 IF(WHY(2)) GO TO 200                                              TRAG1410
      GO TO 100                                                         TRAG1420
C                                                                       TRAG1430
      END                                                               TRAG1440
```

```
      BLOCK DATA                                                        BLK 0010
      COMMON /CNTRL/ GCCRD,GCCPRT,KTCD,FCC(8),KARD(14),PC,PS,PD,PP,     BLK 0020
     *  XCOMF(2),TXDATE(2),WHY(20),KKARD(800)                           BLK 0030
      DATA GCCRD,KTCD,FCC,PC,PS,PD,PP,WHY/.TRUE.,0,                     BLK 0040
     * 26H(A1,8HCARD NO.,I5,5X,14A6),3*0,                               BLK 0050
     * 1H ,1H ,1H0,1H1, 20*.FALSE./                                     BLK 0060
      DATA (KKARD (I),I=1, 124)/10*0,  76,9*0,                          BLK 0070
     * 1, 797, 501, 501, 504,   1,   0, 2, 777, 505, 505, 504,   1,   0,BLK 0080
     * 2, 777,   0,   0,   0,   0,   0, 4, 761, 505, 505, 512,   1,   0,BLK 0090
     * 5, 737, 513, 513, 525,   1,   0, 6, 685, 526, 527, 537,   1,   0,BLK 0100
     * 7, 637, 538, 538, 551,   1,   0, 8, 589, 552, 552, 551,   1,   0,BLK 0110
     *      48*0/                                                        BLK 0120
      DATA (KKARD (I),I= 125, 276)/                                     BLK 0130
     *    81000, 593,0, 0,0,0,                  0,0,                     BLK 0140
     *    81000, 597,0, 0,0,0,                  0,0,                     BLK 0150
     *    81000, 601,0, 0,0,0,                  0,0,                     BLK 0160
     *        1, 605,0, 0,0,0,     .FALSE.      ,0,                     BLK 0170
     *        2, 609,0, 0,0,0,     5            ,0,                     BLK 0180
     *        1, 613,0, 0,0,0,     .FALSE.      ,0,                     BLK 0190
     *        1, 617,0, 0,0,0,     .FALSE.      ,0,                     BLK 0200
     *        1, 621,0, 0,0,0,     .FALSE.      ,0,                     BLK 0210
     *        1, 625,0, 0,0,0,     .FALSE.      ,0,                     BLK 0220
     *    82002, 629,0, 0,0,0,                  0,0,                     BLK 0230
     *        2, 633,0, 0,0,0,     1            ,0,                     BLK 0240
     *    81000, 641,0, 0,0,0,                  0,0,                     BLK 0250
     *    81000, 645,0, 0,0,0,                  0,0,                     BLK 0260
     *    81000, 649,0, 0,0,0,                  0,0,                     BLK 0270
     *        1, 653,0, 0,0,0,     .FALSE.      ,0,                     BLK 0280
     *        1, 657,0, 0,0,0,     .FALSE.      ,0,                     BLK 0290
     *        3, 661,0, 0,0,0, 6H       ,6H       ,                     BLK 0300
     *        3, 665,0, 0,0,0, 6H       ,6H       ,                     BLK 0310
     *        1, 669,0, 0,0,0,     .FALSE.      ,0/                     BLK 0320
      DATA (KKARD (I),I= 277, 428)/                                     BLK 0330
     *        1, 673,0, 0,0,0,     .FALSE.      ,0,                     BLK 0340
     *        1, 677,0, 0,0,0,     .FALSE.      ,0,                     BLK 0350
     *        2, 681,0, 0,0,0,     0            ,0,                     BLK 0360
     *    82002,   0,0, 0,0,0,                  0,0,                     BLK 0370
     *    81000, 689,0, 0,0,0,                  0,0,                     BLK 0380
     *    81000, 693,0, 0,0,0,                  0,0,                     BLK 0390
     *    81000, 697,0, 0,0,0,                  0,0,                     BLK 0400
```

Reproduced from best available copy.

```
*        1, 701,0, 0,0,0,     .FALSE.      ,0,                          BLK 0410
*        1, 705,0, 0,0,0,     .FALSE.      ,0,                          BLK 0420
*        1, 709,0, 0,0,0,     .FALSE.      ,0,                          BLK 0430
*        1, 713,0, 0,0,0,     .FALSE.      ,0,                          BLK 0440
*        1, 717,0, 0,0,0,     .FALSE.      ,0,                          BLK 0450
*        1, 721,0, 0,0,0,     .FALSE.      ,0,                          BLK 0460
*        1, 725,0, 0,0,0,     .FALSE.      ,0,                          BLK 0470
*        1, 729,0, 0,0,0,     .FALSE.      ,0,                          BLK 0480
*        1, 733,0, 0,0,0,     .FALSE.      ,0,                          BLK 0490
*        2, 741,0, 0,0,0,     36           ,0,                          BLK 0500
*        2, 745,0, 0,0,0,     18           ,0,                          BLK 0510
*        2, 749,0, 0,0,0,     0            ,0/                          BLK 0520
 DATA (KKARD (I),I= 429, 500)/                                          BLK 0530
*        5, 753,0, 0,0,0,     0            ,0,                          BLK 0540
*        2, 757,0, 0,0,0,     0            ,0,                          BLK 0550
*     2003, 773,0, 0,0,0, 6H00     ,6H       ,                          BLK 0560
*        5, 765,0, 0,0,0,     250.         ,0,                          BLK 0570
*        2, 769,0, 0,0,0,     16           ,0,                          BLK 0580
*    81000, 781,0, 0,0,0,                 0,0,                          BLK 0590
*    81000, 785,0, 0,0,0,                 0,0,                          BLK 0600
*        1, 789,0, 0,0,0,     .FALSE.      ,0,                          BLK 0610
*        1, 793,0, 0,0,0,     .FALSE.      ,0/                          BLK 0620
 DATA (KKARD (I),I= 501, 588)/                                          BLK 0630
*     493, 485, 477, 469, 461, 453, 445, 437, 429, 421, 413, 405,       BLK 0640
*     397, 389, 381, 373, 365, 357, 437, 349, 341, 333, 325, 317,       BLK 0650
*     309, 301, 293, 285, 277, 269, 261, 253, 245, 237, 229, 221,       BLK 0660
*     213, 205, 197, 189, 181, 173, 165, 437, 157, 373, 349, 149,       BLK 0670
*     141, 133, 125,                                                    BLK 0680
*      37*0/                                                            BLK 0690
 DATA (KKARD (I),I= 589, 740)/                                          BLK 0700
*    6HTIM    ,6H       , -3,   0,    6HZZ7    ,6H       ,  3,   0,     BLK 0710
*    6HYY7    ,6H       ,  3,   0,    6HXX7    ,6H       ,  3,   0,     BLK 0720
*    6HSUM    ,6H       , -3,   0,    6HREAD   ,6H       , -4,   0,     BLK 0730
*    6HSMOOTH,6H        , -6,   0,    6HNORM   ,6H       , -4,   0,     BLK 0740
*    6HREZONE,6H        , -6,   0,    6HREDUCE,6H        , -6,   0,     BLK 0750
*    6HFOR    ,6H       , -3,   0,    6HNSET   ,6H       , -4,   0,     BLK 0760
*    6HORD    ,6H       , -3,   0,    6HZZ6    ,6H       ,  3,   0,     BLK 0770
*    6HYY6    ,6H       ,  3,   0,    6HXX6    ,6H       ,  3,   0,     BLK 0780
*    6HWE     ,6H       , -2,   0,    6HBK     ,6H       , -2,   0,     BLK 0790
*    6HSR     ,6H       , -2,   0,    6HXS     ,6H       , -2,   0,     BLK 0800
```

```
*     6HPE     ,6H       , -2,    0,     6HBI     ,6H       , -2,    0,     BLK 0810
*     6HPC     ,6H       , -2,    0,     6HSK     ,6H       , -2,    0,     BLK 0820
*     6HTAPE   ,6H       ,  4,    0,     6HZ75    ,6H       ,  3,    0,     BLK 0830
*     6HYY5    ,6H       ,  3,    0,     6HXX5    ,6H       ,  3,    0,     BLK 0840
*     6HLIM    ,6H       , =3,    0,     6HPA     ,6H       , -2,    0,     BLK 0850
*     6HKW     ,6H       , -2,    0,     6HTA     ,6H       , -2,    0,     BLK 0860
*     6HPL     ,6H       , -2,    0,     6HLIST   ,6H       , -4,    0,     BLK 0870
*     6HEX     ,6H       , =2,    0,     6HRA     ,6H       , -2,    0,     BLK 0880
*     6HSO     ,6H       , -2,    0,     6HOUT    ,6H       , -3,    0/     BLK 0890
 DATA (KKARD (I),I= 741, 800)/                                              BLK 0900
*     6HNA     ,6H       , -2,    0,     6HNF     ,6H       , -2,    0,     BLK 0910
*     6HNB     ,6H       , -2,    0,     6HZ      ,6H       , -1,    0,     BLK 0920
*     6HOR     ,6H       , -2,    0,     6HBOOM   ,6H       ,  4,    0,     BLK 0930
*     6HOR     ,6H       , -2,    0,     6HNR     ,6H       , -2,    0,     BLK 0940
*     6HFL     ,6H       , -2,    0,     6HSTOP   ,6H       ,  4,    0,     BLK 0950
*     6HYY1    ,6H       ,  3,    0,     6HXX1    ,6H       ,  3,    0,     BLK 0960
*     6HDA     ,6H       , -2,    0,     6HCO     ,6H       , -2,    0,     BLK 0970
*     6HPRINT  ,6H       ,  5,    0/                                        BLK 0980
 END                                                                        BLK 0990
```

Reproduced from best available copy.

```
      SUBROUTINE FLUSH                                                  FLUS0010
C                                                                       FLUS0020
C     IN THE EVENT OF AN ERROR, DATA CARDS ARE SKIPPED UNTIL THE NEXT   FLUS0030
C     CONTROL CARD IS FOUND. (ALL CONTROL CARDS HAVE A '$' IN CC.1)     FLUS0040
C     COMMENT CARDS ('*' IN CC.1) ARE LISTED                            FLUS0050
C                                                                       FLUS0060
      COMMON /CNTRL/ QCCRD,QCCPRT,KTCD,FCC(8),KARD(14),PC,PS,PD,PP,     FLUS0070
     * TXTIME(2),TXDATE(2),WHY(20),KKARD(600)                           FLUS0080
      LOGICAL QCCRD,QCCPRT,WHY                                          FLUS0090
      INTEGER TXTIME,TXDATE,FCC,PC,PS,PD,PP                             FLUS0100
C                                                                       FLUS0110
      LOGICAL Q                                                         FLUS0120
      DATA IPE,IRS/1H$,1H*/                                             FLUS0130
C                                                                       FLUS0140
      PC = PD                                                           FLUS0150
      Q = .FALSE.                                                       FLUS0160
    1 READ (5,2,END=200) KARD                                           FLUS0170
    2 FORMAT(14A6)                                                      FLUS0180
      IF (KOM(KARD,1,1,IPE,1).EQ.0) GO TO 20                            FLUS0190
      IF (KOM(KARD,1,1,IRS,1).EQ.0) GO TO 10                            FLUS0200
      IF(Q) GO TO 1                                                     FLUS0210
      Q = .TRUE.                                                        FLUS0220
      WRITE(6,4) KARD                                                   FLUS0230
    4 FORMAT(1H0,18X,14A6)                                              FLUS0240
      WRITE(6,5)                                                        FLUS0250
    5 FORMAT(35H0******  INPUT DATA SKIPPED  ******)                    FLUS0260
      GO TO 1                                                           FLUS0270
C  PRINT COMMENTS AS THEY APPEAR                                        FLUS0280
   10 WRITE(6,4) KARD                                                   FLUS0290
      PC = PS                                                           FLUS0300
      GO TO 1                                                           FLUS0310
C  SET 'CONTROL CARD READY' FLAG WHEN CONTROL CARD IS FOUND             FLUS0320
   20 PC = PD                                                           FLUS0330
      QCCRD = .FALSE.                                                   FLUS0340
      RETURN                                                            FLUS0350
C  CAUSE PROGRAM TERMINATION IF END OF DATA ENCOUNTERED.                FLUS0360
  200 WHY(2) = .TRUE.                                                   FLUS0370
      RETURN                                                            FLUS0380
      END                                                               FLUS0390
```

```
      SUBROUTINE TAPE                                                   TAPE0010
C                                                                       TAPE0020
C     GENERAL FILE POSITIONING ROUTINE                                  TAPE0030
C                                                                       TAPE0040
C     FOR FILE SEARCH COMMANDS, IT IS ASSUMED THAT THERE ARE SENTINAL   TAPE0050
C     RECORDS, WHICH CONSIST OF THREE WORDS..                           TAPE0060
C          1)  6H*HEAD*                                                 TAPE0070
C          2)  1ST 6 CHARACTERS OF IDENTIFICATION (LEFT-JUST.)          TAPE0080
C          3)  LAST 2 CHARACTERS OF IDENTIFICATION (BLANK IF ID IS      TAPE0090
C              NOT OVER 6 CHARACTERS LONG)                              TAPE0100
C     THE IDENTIFICATION OF THE END-OF-FILE SENTINEL IS 6H999999,2H99,  TAPE0110
C     WRITTEN JUST PRIOR TO AN END-OF-FILE.                             TAPE0120
C     THE 'SRCH' COMMAND SEARCHES FROM THE CURRENT FILE POSITION TO END TAPE0130
C     OF FILE.  THE 'XSRCH' COMMAND PROCEEDS AS 'SRCH', BUT REWINDS AND TAPE0140
C     SEARCHES THE WHOLE FILE IF AN END-OF-FILE IS READ.                TAPE0150
C     IN BOTH CASES, AN ERROR IS PRINTED, AND WHY(6) IS SET IF THE      TAPE0160
C     DESIRED SENTINAL RECORD IS NOT FOUND.                             TAPE0170
C     OPERATIONS ARE PERFORMED IN THE FOLLOWING ORDER IF MORE THAN ONE  TAPE0180
C     OPERATION IS SPECIFIED..                                          TAPE0190
C          1)  REWIND                                                   TAPE0200
C          2)  SKIP=N                                                   TAPE0210
C          3)  SR=ID OR XS=ID                                           TAPE0220
C          4)  BKSP                                                     TAPE0230
C          5)  WEOF                                                     TAPE0240
C                                                                       TAPE0250
      COMMON /CNTRL/ QCCRD,QCCPRT,KTCD,FCC(8),KARD(14),PC,PS,PD,PP,     TAPE0260
     * TXTIME(2),TXDATE(2),WHY(20),KKARD(800)                            TAPE0270
      LOGICAL QCCRD,QCCPRT,WHY                                          TAPE0280
      INTEGER TXTIME,TXDATE,FCC,PC,PS,PD,PP                             TAPE0290
C                                                                       TAPE0300
      INTEGER ISRC(2),IXSRC(2)                                          TAPE0310
      LOGICAL IFMODE,IFBIN,IFRWD,IFBKSP,IFWEOF,IFSRWD                   TAPE0320
      EQUIVALENCE (KKARD(297),IFSKIP),(KKARD(289),IFMODE),              TAPE0330
     * (KKARD(281),IFBIN),(KKARD(273),IFRWD),(KKARD(263),KXSRC),        TAPE0340
     * (KKARD(265),IXSRC),(KKARD(255),KSRC),(KKARD(257),ISRC),          TAPE0350
     * (KKARD(249),IFBKSP),(KKARD(241),IFWEOF),(KKARD(303),KFILE),      TAPE0360
     * (KKARD(305),IFILE)                                               TAPE0370
C                                                                       TAPE0380
      INTEGER IFSARG(2),IFLAB(2),IHEND(2)                               TAPE0390
      DATA IFSARG/2*1H /,IFHEAD,IHEND/6H*HEAD*,6H999999,2H99/           TAPE0400
```

Reproduced from best available copy.

```
      DATA NWW,NCN/2,8/                                                 TAPE0410
C                                                                       TAPE0420
      IF(KFILE.LE.0) GO TO 640                                          TAPE0430
      IF(IFILE.LE.0) GO TO 640                                          TAPE0440
      IF(IFILE.EQ.5 .OR. IFILE.EQ.4) GO TO 645                          TAPE0450
      IF(IFMODE .AND.IFBIN) GO TO 640                                   TAPE0460
      IF(KSRC*KXSRC.GT.0) GO TO 640                                     TAPE0470
      WHY(6) = .FALSE.                                                  TAPE0480
C  REWIND IF REQUESTED.                                                 TAPE0490
      IF (IFRWD) REWIND IFILE                                           TAPE0500
      IF (IFSKIP.LE.0) GO TO 620                                        TAPE0510
C  SKIP RECORDS IF REQUESTED.                                           TAPE0520
      IF (IFMODE) GO TO 610                                             TAPE0530
      DO 601 I=1,IFSKIP                                                 TAPE0540
  601 READ (IFILE) IDUM                                                 TAPE0550
      GO TO 620                                                         TAPE0560
  610 DO 611 I=1,IFSKIP                                                 TAPE0570
  611 READ (IFILE,612) IDUM                                             TAPE0580
  612 FORMAT(A1)                                                        TAPE0590
C  SEARCH FOR SENTINEL RECORD                                           TAPE0600
  620 IF(KSRC.LE.0 .AND. KXSRC.LE.0) GO TO 650                          TAPE0610
      IF(KSRC.GT.0) CALL MOVE(IFSARG,1,NCN,ISRC,1)                      TAPE0620
      IF(KXSRC.GT.0) CALL MOVE(IFSARG,1,NCN,IXSRC,1)                    TAPE0630
      IFSRWD = .FALSE.                                                  TAPE0640
  621 IF(IFMODE) GO TO 622                                              TAPE0650
      READ (IFILE,END=630) IDUM,IFLAB                                   TAPE0660
      GO TO 625                                                         TAPE0670
  622 READ (IFILE,624,END=630) IDUM,IFLAB                               TAPE0680
  624 FORMAT(3A6)                                                       TAPE0690
  625 IF(IDUM.NE.IFHEAD) GO TO 621                                      TAPE0700
      IF (KOMW(IFSARG,NWW,IFLAB).NE.0) GO TO 621                        TAPE0710
  650 IF(IFBKSP) BACK SPACE IFILE                                       TAPE0720
      IF(.NOT.IFWEOF) GO TO 100                                         TAPE0730
      IF(IFMODE) GO TO 651                                              TAPE0740
      WRITE(IFILE) IFHEAD,IHEND                                         TAPE0750
      GO TO 652                                                         TAPE0760
  651 WRITE(IFILE,624) IFHEAD,IHEND                                     TAPE0770
  652 END FILE IFILE                                                    TAPE0780
      GO TO 100                                                         TAPE0790
  630 IF(IFSRWD .OR. KXSRC.LE.0) GO TO 635                              TAPE0800
```

```
      IFSRWD = .TRUE.                                                   TAPE0810
      REWIND IFILE                                                      TAPE0820
      GO TO 621                                                         TAPE0830
  635 IF(.NOT.QCCPRT) WRITE(6,FCC) PC,KTCD,KARD                         TAPE0840
      QCCPRT = .TRUE.                                                   TAPE0850
      WRITE (6,636) IFSARG                                              TAPE0860
  636 FORMAT(19H0******  SENTINAL (,A6,A2,12H) NOT FOUND./)             TAPE0870
      GO TO 645                                                         TAPE0880
  640 IF(.NOT.QCCPRT) WRITE(6,FCC) PC,KTCD,KARD                         TAPE0890
      QCCPRT = .TRUE.                                                   TAPE0900
      WRITE(6,641)                                                      TAPE0910
  641 FORMAT(48H0******  INCONSISTENT OR INCORRECT PARAMETER(S)./)      TAPE0920
  645 WHY(6) = .TRUE.                                                   TAPE0930
      PC = PC                                                           TAPE0940
  100 RETURN                                                            TAPE0950
      END                                                               TAPE0960
```

Reproduced from best available copy.

```
      SUBROUTINE LINK1(L)                                               LINK0010
C                                                                       LINK0020
C     CALL LINK 1 ROUTINES                                              LINK0030
C                                                                       LINK0040
      IF (L-2) 10,20,30                                                 LINK0050
   10 CALL ORDNAN                                                       LINK0060
      RETURN                                                            LINK0070
   20 CALL EXIT                                                         LINK0080
      RETURN                                                            LINK0090
   30 CALL BOOM                                                         LINK0100
      RETURN                                                            LINK0110
      END                                                               LINK0120
```

```
      BLOCK DATA                                                        BLKB0010
C  ORDNANCE TABLES                                                      BLKB0020
      COMMON /ORDCM/ MVORD,MMORD,ICORD,IVO,XKORD,                       BLKB0030
     * TTLORD,NVORD,NMORD,DTH,DTH2,                                     BLKB0040
     * VORD(37),AMORD(30),DORD(37,30),WORD(37,30)                       BLKB0050
      INTEGER TTLORD(10)                                                BLKB0060
      DATA NVORD,NMORD/37,30/                                           BLKB0070
      END                                                               BLKB0080
```

Reproduced from best available copy.

```
      SUBROUTINE ORDNAN                                                 ORDN0010
C                                                                       ORDN0020
C  CONTROL CARD TABLES AND GENERAL PROGRAM CONTROL VARIABLES            ORDN0030
      COMMON /CNTRL/ QCCRD,QCCPRT,KTCD,FCC(8),KARD(14),PC,PS,PD,PP,     ORDN0040
     * TXTIME(2),TXDATE(2),WHY(20),KKARD(800)                            ORDN0050
      LOGICAL QCCRD,QCCPRT,WHY                                          ORDN0060
      INTEGER TXTIME,TXDATE,FCC,PC,PS,PD,PP                             ORDN0070
C                                                                       ORDN0080
      INTEGER AFFORD                                                    ORDN0090
      LOGICAL PRTDAT,QCBORD,REZONE,NRMORD,SMORD,LIST                    ORDN0100
      LOGICAL QWKPLT,SUMRY                                              ORDN0110
      EQUIVALENCE (KKARD(489),PRTDAT),(KKARD(209),NSETS),               ORDN0120
     * (KKARD(201),IERORD),(KKARD(193),QCBORD),(KKARD(185),REZONE),     ORDN0130
     * (KKARD(177),NRMORD),(KKARD(169),SMORD),(KKARD(161),INU),         ORDN0140
     * (KKARD(377),LIST),(KKARD(441),AFFORD),(KKARD(353),QWKPLT),       ORDN0150
     * (KKARD(153),SUMRY)                                               ORDN0160
C                                                                       ORDN0170
C  ORDNANCE TABLES                                                      ORDN0180
      COMMON /ORDCM/ MVORD,MMORD,IDORD,IVO,XKORD,                       ORDN0190
     * TTLORD,NVORD,NMORD,DTH,DTH2,                                     ORDN0200
     * VORD(37),AMORD(30),DORD(37,30),WORD(37,30)                       ORDN0210
      INTEGER TTLORD(12)                                                ORDN0220
C                                                                       ORDN0230
      REAL COFORD(21)                                                   ORDN0240
      EXTERNAL ORDPLT                                                   ORDN0250
      REAL ZAREA(36),DTZ(36),WTZ(36),WAZ(36),WTM(30),DTM(30),DAM(30),   ORDN0260
     * SDW(30),SDD(30),ORG(4,30),ORGXX(122)                             ORDN0270
      INTEGER CHOP(9)                                                   ORDN0280
      DATA CHOP/1HV,1HD,1HW,1HD,1HP,1HD,1HW,1HD,1HW/                    ORDN0290
      DATA PI/3.14159265/,MXFORD/10/                                    ORDN0300
      DATA TXWGT,TXNO/6HWGT. (,5H) NO./                                 ORDN0310
      EQUIVALENCE (ORGXX(1),ORG(1,1)),(ORGXX(121),VMN),                 ORDN0320
     * (ORGXX(122),VMX)                                                 ORDN0330
C                                                                       ORDN0340
      WHY(7) = .FALSE.                                                  ORDN0350
      IF(INU.NE.5 .AND. (WHY(1).OR.WHY(6))) RETURN                      ORDN0360
      SUMRY=SUMRY.OR.LIST.OR.QWKPLT                                     ORDN0370
      READ (5,160) TTLORD                                               ORDN0380
  160 FORMAT(12A6)                                                      ORDN0390
      READ (INU,161) NMORD,NVORD,IDORD,IVO,XKORD                        ORDN0400
```

```
  161 FORMAT(3I5,1X,A1,3X,F10.0)                                        ORDN0410
      IF(IVO.EQ.PS) IVO=PD                                              ORDN0420
      WRITE(6,201) TTLORD,NMORD,NVORD,IDORD,IVO,XKORD                   ORDN0430
  201 FORMAT(1H0,18X,12A6//19X,7HNMORD =,I3,5X,7HNVORD =,I3,5X,         ORDN0440
     * 6HI.D. =,I6,1H=,A1,5X,12HAVG OR/IN2 =,F10.2)                    ORDN0450
      IF (NMORD.LE.0 .OR. NVORD.LE.0 .OR. NVORD.GT.MVORD .OR.           ORDN0460
     * XKORD.LE.0.) STOP                                               ORDN0470
      IF(NSETS.LE.0) NSETS=1                                            ORDN0480
      IF(IFRORD.EQ.1) CALL ORDIN1(NMORD,NVORD,NSETS,WORD,DORD,VORD,     ORDN0490
     * QCBORD,PRTDAT,KTCD,INU,MVORD)                                    ORDN0500
      IF(IFRORD.EQ.2) CALL ORDIN2(NMORD,NVORD,NSETS,WORD,DORD,VORD,     ORDN0510
     * QCBORD,PRTDAT,KTCD,INU,MVORD)                                    ORDN0520
      PC = PD                                                           ORDN0530
      IF (PRTDAT) PC=PF                                                 ORDN0540
      IF(.NOT.REZONE) GO TO 210                                         ORDN0550
C  IF REDUCTION OF ZONES IS REQUESTED                                   ORDN0560
      IDUM = NVORD-1                                                    ORDN0570
      DO 230 I=1,NMORD                                                  ORDN0580
      DORD(1,I) = DORD(1,I)*2.                                          ORDN0590
      DORD(NVORD,I) = DORD(NVORD,I)*2.                                  ORDN0600
      DO 231 J=1,IDUM                                                   ORDN0610
      DUM = DORD(J,I)+DORD(J+1,I)                                       ORDN0620
      IF (DUM.LE.0.) GO TO 232                                          ORDN0630
      WORD(J,I) = (WORD(J,I)*DORD(J,I)+WORD(J+1,I)*DORD(J+1,I))/DUM     ORDN0640
      GO TO 231                                                         ORDN0650
  232 WORD(J,I) = 0.                                                    ORDN0660
  231 DORD(J,I) = DUM/2.                                                ORDN0670
  230 CONTINUE                                                          ORDN0680
      NVORD = IDUM                                                      ORDN0690
  210 DTH = PI/FLOAT(NVORD)                                             ORDN0700
      DTH2=DTH/2.                                                       ORDN0710
      DPHP = 180./FLOAT(NVORD)                                          ORDN0720
C  GENERATE POLAR ZONE AREA (ON UNIT SPHERE)                            ORDN0730
      DUM = DTH2                                                        ORDN0740
      PDUM = 4.*PI*SIN(DUM)                                             ORDN0750
      DO 211 I=1,NVORD                                                  ORDN0760
      ZAREA(I) = SIN(DUM)*PDUM                                          ORDN0770
  211 DUM = DUM+DTH                                                     ORDN0780
      IF(.NOT.NRMORD) GO TO 220                                         ORDN0790
C  NORMALIZE                                                            ORDN0800
```

Reproduced from best available copy.

```
      DO 215 I=1,NVORD                                                  ORDN0810
      DO 215 J=1,NMORD                                                  ORDN0820
  215 DORD(I,J) = DORD(I,J)/ZAREA(I)                                    ORDN0830
C  CLEAR STD. DEVIATION CELLS                                           ORDN0840
  220 SDV = 0.                                                          ORDN0850
      DO 249 I=1,NMORD                                                  ORDN0860
      SDW(I) = 0.                                                       ORDN0870
  249 SDD(I) = 0.                                                       ORDN0880
      IF(.NOT.SMORD) GO TO 270                                          ORDN0890
C  SMOOTH VORD, DORD, AND WORD                                          ORDN0900
C  1) TRY TO *FILL IN* WORD(*,*)=0 ENTRIES                              ORDN0910
      DO 221 I=1,NMORD                                                  ORDN0920
      DUM = 0.                                                          ORDN0930
      J = 1                                                             ORDN0940
  228 IF(WORD(J,I).GT.0.) GO TO 222                                     ORDN0950
      IF(J.GT.1) DUM=WORD(J-1,I)                                        ORDN0960
      IF(J.GE.NVORD) GO TO 241                                          ORDN0970
      DO 223 K=J,NVORD                                                  ORDN0980
      IF(WORD(K,I).GT.0.) GO TO 224                                     ORDN0990
  223 CONTINUE                                                          ORDN1000
C  A) TRAILING ZEROES -- SET TO LAST NON-ZERO VALUE (IF ANY)            ORDN1010
  241 DO 225 K=J,NVORD                                                  ORDN1020
  225 WORD(K,I)=DUM                                                     ORDN1030
      GO TO 221                                                         ORDN1040
  224 K1 = K-1                                                          ORDN1050
      IF(J.GT.1) GO TO 226                                              ORDN1060
C  B) LEADING ZEROES -- SET TO 1ST NON-ZERO VALUE                       ORDN1070
      DO 227 L=1,K1                                                     ORDN1080
  227 WORD(L,I) = WORD(K,I)                                             ORDN1090
      J = K                                                             ORDN1100
      GO TO 222                                                         ORDN1110
C  C) INTERMEDIATE ZEROES -- FILL IN VIA LINEAR INTERPOLATION           ORDN1120
  226 DUM = WORD(J-1,I)                                                 ORDN1130
      DK = (WORD(K,I)-DUM)/FLOAT(K-J+1)                                 ORDN1140
      DO 242 L=J,K1                                                     ORDN1150
      DUM = DUM+DK                                                      ORDN1160
  242 WORD(L,I) = DUM                                                   ORDN1170
      J = K                                                             ORDN1180
  222 J = J+1                                                           ORDN1190
  229 IF(J.LE.NVORD) GO TO 228                                          ORDN1200
```

```
  221 CONTINUE                                                          ORDN1210
C  2) SMOOTH ALL CURVES USING FOURIER SERIES                            ORDN1220
C     NOTE,, MASS AND NUMBER NOT CONSERVED EXACTLY BY THESE FITS        ORDN1230
      IF (AFFORD.LE.0) GO TO 270                                        ORDN1240
      AFFORD = MIN0(AFFORD,MXFORD)                                      ORDN1250
      CALL FOUR(VORD,COFORD,NVORD,AFFORD)                               ORDN1260
      CALL FORGET(WTZ,COFORD,NVORD,AFFORD)                              ORDN1270
      DO 246 J=1,NVORD                                                  ORDN1280
      SDV = SDV+(WTZ(J)-VORD(J))**2                                     ORDN1290
  246 VORD(J) = WTZ(J)                                                  ORDN1300
      SDV = SQRT(SDV/FLOAT(NVORD))                                      ORDN1310
      DO 243 I=1,NMORD                                                  ORDN1320
      CALL   FOUR(WORD(1,I),COFORD,NVORD,AFFORD)                        ORDN1330
      CALL FORGET(WTZ,COFORD,NVORD,AFFORD)                              ORDN1340
      DO 244 J=1,NVORD                                                  ORDN1350
      SDW(I) = SDW(I)+(WTZ(J)-WORD(J,I))**2                             ORDN1360
  244 WORD(J,I) = WTZ(J)                                                ORDN1370
      SDW(I) = SQRT(SDW(I)/FLOAT(NVORD))                                ORDN1380
      CALL   FOUR(DORD(1,I),COFORD,NVORD,AFFORD)                        ORDN1390
      CALL FORGET(WTZ,COFORD,NVORD,AFFORD)                              ORDN1400
      DO 245 J=1,NVORD                                                  ORDN1410
      SDD(I) = SDD(I)+(WTZ(J)-DORD(J,I))**2                             ORDN1420
  245 DORD(J,I) = WTZ(J)                                                ORDN1430
      SDD(I) = SQRT(SDD(I)/FLOAT(NVORD))                                ORDN1440
  243 CONTINUE                                                          ORDN1450
C  DETERMINE VARIOUS AVERAGES AND TOTALS                                ORDN1460
  270 VA = 0.                                                           ORDN1470
      WT = 0.                                                           ORDN1480
      DT = 0.                                                           ORDN1490
      VMN = +1.E+38                                                     ORDN1500
      VMX = -VMN                                                        ORDN1510
      DO 273 I=1,NMORD                                                  ORDN1520
      ORG(1,I) = VMN                                                    ORDN1530
      ORG(2,I) = VMX                                                    ORDN1540
      ORG(3,I) = VMN                                                    ORDN1550
  273 ORG(4,I) = VMX                                                    ORDN1560
      DO 271 J=1,NVORD                                                  ORDN1570
      DUM = AMAX1(VORD(J),0.)                                           ORDN1580
      VMN = AMIN1(VMN,DUM)                                              ORDN1590
      VMX = AMAX1(VMX,DUM)                                              ORDN1600
```

Reproduced from best available copy.

```
      WTZ(J) = 0.                                                       ORDN1610
      DTZ(J) = 0.                                                       ORDN1620
      DO 272 I=1,NMORD                                                  ORDN1630
      UNIT = AMAX1(DORD(J,I),0.)                                        ORDN1640
      COUNT = UNIT*ZAREA(J)                                             ORDN1650
      WEIGHT = AMAX1(WORD(J,I),0.)                                      ORDN1660
      WTZ(J) = WTZ(J)+COUNT*WEIGHT                                      ORDN1670
      DTZ(J) = DTZ(J)+COUNT                                             ORDN1680
      ORG(1,I) = AMIN1(ORG(1,I),WEIGHT)                                 ORDN1690
      ORG(2,I) = AMAX1(ORG(2,I),WEIGHT)                                 ORDN1700
      ORG(3,I) = AMIN1(ORG(3,I),UNIT)                                   ORDN1710
  272 ORG(4,I) = AMAX1(ORG(4,I),UNIT)                                   ORDN1720
      WAZ(J) = WTZ(J)/DTZ(J)                                            ORDN1730
      VA = VA+VORD(J)*DTZ(J)                                            ORDN1740
      WT = WT+WTZ(J)                                                    ORDN1750
  271 DT = DT+DTZ(J)                                                    ORDN1760
      VA = VA/WT                                                        ORDN1770
      WA = WT/DT                                                        ORDN1780
      DO 274 I=1,NMORD                                                  ORDN1790
      DAM(I) = 0.                                                       ORDN1800
      WTM(I) = 0.                                                       ORDN1810
      DTM(I) = 0.                                                       ORDN1820
      DO 275 J=1,NVORD                                                  ORDN1830
      WEIGHT = AMAX1(WORD(J,I),0.)                                      ORDN1840
      UNIT = AMAX1(DORD(J,I),0.)                                        ORDN1850
      COUNT = UNIT*ZAREA(J)                                             ORDN1860
      WTM(I) = WTM(I)+WEIGHT*COUNT                                      ORDN1870
  275 DTM(I) = DTM(I)+COUNT                                             ORDN1880
      DAM(I) = DTM(I)/(4.*PI)                                           ORDN1890
  274 AMORD(I) = WTM(I)/DTM(I)                                          ORDN1900
C  LIST FINAL DATA IN COLUMNS IF REQUESTED                              ORDN1910
  280 IF(.NOT.LIST) GO TO 250                                           ORDN1920
      M2 = MIN0(5,NVORD)                                                ORDN1930
      WRITE (6,281) PC,(TXWGT,I,TXNO,I=1,M2)                            ORDN1940
  281 FORMAT(A1,63X,15HMASS CATEGORIES/                                 ORDN1950
     * 12H0 POLAR ZONE,12X,8HVELOCITY,5(7X,A6,I2,A5))                   ORDN1960
      PHP1 = 0.                                                         ORDN1970
      PC = PD                                                           ORDN1980
      DO 282 J=1,NVORD                                                  ORDN1990
      PHP2 = PHP1+DPHP                                                  ORDN2000
```

```
      WRITE(6,283) PC,PHP1,PHP2,NORD(J),(WORD(J,I),DORD(J,I),I=1,M2)     ORDN2010
  283 FORMAT(A1,F5.1,1H-,F5.1,12X,A12,1)                                 ORDN2020
      PHP1 = PHP2                                                        ORDN2030
  282 PC = PS                                                            ORDN2040
  284 M1 = M2+1                                                          ORDN2050
      IF(M2.GE.NMORD) GO TO 289                                          ORDN2060
      M2 = MIN0(NMORD,M1+5)                                              ORDN2070
      PC = PP                                                            ORDN2080
      WRITE(6,285) PC,(TXNT,I,TXNT,I=M1,M2)                              ORDN2090
  285 FORMAT(A1,63X,15H MSS CATEGORIES/                                  ORDN2100
     * 12H  POLAR ZONE,6(7X,A6,I2,A5))                                   ORDN2110
      PHP1 =0.                                                           ORDN2120
      PC = PP                                                            ORDN2130
      DO 286 J=1,NVORD                                                   ORDN2140
      PHP2 = PHP1+DPHP                                                   ORDN2150
      WRITE(6,287) PC,PHP1,PHP2,(NORD(J,I),DORD(J,I),I=M1,M2)            ORDN2160
  287 FORMAT(A1,F5.1,1H-,F5.1,12F10.1)                                   ORDN2170
      PHP1 = PHP2                                                        ORDN2180
  286 PC = PS                                                            ORDN2190
      GO TO 284                                                          ORDN2200
  289 PC = PP                                                            ORDN2210
C  LIST SUMMARY OF DATA IF REQUESTED                                     ORDN2220
  250 IF (.NOT.SUMRY) GO TO 260                                          ORDN2230
      WRITE(6,257)                                                       ORDN2240
  257 FORMAT(1H1,57X,15HSUMMARY OF DATA//                                ORDN2250
     * 12H  POLAR ZONE,5X,10HTOTAL MASS,7X,8HNO. FRAG,7X,8HAVG MASS/)    ORDN2260
      PHP1 = 0.                                                          ORDN2270
      PC = PP                                                            ORDN2280
      DO 256 J=1,NVORD                                                   ORDN2290
      PHP2 = PHP1+DPHP                                                   ORDN2300
      WRITE(6,251) PHP1,PHP2,WTZ(J),DTZ(J),WAZ(J)                        ORDN2310
  251 FORMAT(1X,F5.1,1H-,F5.1,3F15.1)                                    ORDN2320
  256 PHP1 = PHP2                                                        ORDN2330
      WRITE(6,252) WT,DT,WA                                              ORDN2340
  252 FORMAT(12H0  ALL ZONES,3F15.1//)                                   ORDN2350
      WRITE(6,253) VMN,VA,VMX,SDV                                        ORDN2360
  253 FORMAT(1H1,53X,23HSUMMARY OF DATA (CONT.)//                        ORDN2370
     * 34X,3HLOW,8X,7HAVERAGE,11X,4HHIGH,8X,7HSTD DEV,10X,5HTOTAL//      ORDN2380
     * 14X,8HVELOCITY,4F15.1)                                            ORDN2390
      DO 254 I=1,NMORD                                                   ORDN2400
```

Reproduced from best available copy.

```
254 WRITE(6,255) I,ORG(3,I),DAM(I),ORG(4,I),SDD(I),DTM(I),                ORDN2410
   *              ORG(1,I),AMORD(I),ORG(2,I),SDM(I),WTM(I)                ORDN2420
255 FORMAT(11H0MASS CAT, ,I2,9H NO. FRAG,5F15.1/16X,4HMASS,5F15.1)        ORDN2430
260 IF(.NOT.QWKPLT) GO TO 100                                             ORDN2440
    NPLOT = NMORD/4+1                                                     ORDN2450
    AZ1 = DPWP/2.                                                         ORDN2460
    AZ2 = 180.-AZ1                                                        ORDN2470
    CALL ORDPLT(1,E+38,PC,ORG)                                            ORDN2480
    PC = PD                                                               ORDN2490
    M1 = 1                                                                ORDN2500
    KC = 1                                                                ORDN2510
    DO 261 I=1,NPLOT                                                      ORDN2520
    M2 = MIND(NMORD,M1+3)                                                 ORDN2530
    IF (I.EQ.1) M2=MIND(M2,3)                                             ORDN2540
262 WRITE(6,263) PC,M1,M2                                                 ORDN2550
263 FORMAT(A1,10X,47HPLOT OF FINAL DENSITY AND WEIGHT TABLES, MASSES,     ORDN2560
   * I3,5H THRU,I3/)                                                      ORDN2570
    NF = 2*(M2-M1+1)                                                      ORDN2580
    IF(I.EQ.1) NF=NF+1                                                    ORDN2590
    CALL NOTARY(ORDPLT,NF,NMORD,AZ1,AZ2,121.,0.,8.,CHOP(KC))              ORDN2600
    KC = 2                                                                ORDN2610
    M1 = M2+1                                                             ORDN2620
    IF(I.NE.NPLOT) CALL ORDPLT(-1.,0.,DUM)                                ORDN2630
261 CONTINUE                                                              ORDN2640
100 PC = PD                                                               ORDN2650
    IF(PRTCAT .OR. SUMRY) PC=PD                                           ORDN2660
    RETURN                                                                ORDN2670
    END                                                                   ORDN2680
```

```
      SUBROUTINE ORDIN1(INORD,NVORD,NSETS,WORD,DORD,VORD,QCBORD,PRTDAT,  ORDI0010
     * KTCD,INU,MVORD)                                                  ORDI0020
C                                                                       ORDI0030
C     INPUT ROUTINE FOR FORMAT=1 ORDNANCE DATA                          ORDI0040
C***  NSETS.GT.1 AND QCBORD .T. NOT SUPPORTED                           ORDI0050
C                                                                       ORDI0060
      REAL WORD(MVORD,1),DORD(MVORD,1),VORD(MVORD)                      ORDI0070
      LOGICAL QCBORD,PRTDAT                                             ORDI0080
C                                                                       ORDI0090
      DO 10 J=1,NVORD                                                   ORDI0100
      READ(INU,2)  (WORD(J,I),DORD(J,I),I=1,NMORD)                      ORDI0110
    2 FORMAT(4(2F5.1,5X))                                               ORDI0120
      IF (PRTDAT) WRITE(6,4) (WORD(J,I),DORD(J,I),I=1,NMORD)            ORDI0130
    4 FORMAT(10X,10HWORD,DORD ,4(2F10.3,4X))                            ORDI0140
      READ(INU,3) VORD(J)                                               ORDI0150
    3 FORMAT(50X,F10.1)                                                 ORDI0160
      IF(PRTDAT) WRITE(6,5) VORD(J)                                     ORDI0170
    5 FORMAT(10X,4HVORD,5X,F12.1)                                       ORDI0180
   10 CONTINUE                                                          ORDI0190
      RETURN                                                            ORDI0200
      END                                                               ORDI0210
```

Reproduced from best available copy.

```
      SUBROUTINE ORDIN2(NMORD,NVORD,NSETS,WORD,DORD,VORD,QCBORD,PRTDAT,  ORDI0010
     * KTCD,INU,MVORD)                                                  ORDI0020
C                                                                       ORDI0030
C     INPUT ROUTINE FOR FORMAT=2 ORDNANCE DATA                          ORDI0040
C                                                                       ORDI0050
C***  NOTE THAT WZ AND WZD SHOULD BE DIMENSIONED FOR COL.LEN=MVORD.     ORDI0060
      REAL WORD(MVORD,4),DORD(MVORD,4),VORD(MVORD),WZ(37,2),WZD(37),    ORDI0070
     * WT(4),DT(4)                                                      ORDI0080
      LOGICAL QCBORD,QNMODD,PRTDAT                                      ORDI0090
C                                                                       ORDI0100
      NUM = NMORD                                                       ORDI0110
      IF (QCBORD) NUM=(NUM+1)/2                                         ORDI0120
      QNMODD = QCBORD .AND. NMORD.NE.NVORD/2*2                          ORDI0130
C  CLEAR ORD, ARRAYS                                                    ORDI0140
      DO 2 J=1,NVORD                                                    ORDI0150
      WZD(J)=0.                                                         ORDI0160
      VORD(J) = 0.                                                      ORDI0170
      DO 3 I=1,2                                                        ORDI0180
    3 WZ(J,I) = 0.                                                      ORDI0190
      DO 4 I=1,NUM                                                      ORDI0200
      WORD(J,I) = 0.                                                    ORDI0210
    4 DORD(J,I) = 0.                                                    ORDI0220
    2 CONTINUE                                                          ORDI0230
C  READ ORD, MASS AND NUMBER TABLES                                     ORDI0240
      DO 11 N=1,NSETS                                                   ORDI0250
      IF(PRTDAT) WRITE(6,82) N                                          ORDI0260
   82 FORMAT(10X,23HWORD,DORD FOR COMPONENT,I2)                         ORDI0270
      I1 = 1                                                            ORDI0280
   16 I2 = I1+1                                                         ORDI0290
      IF(.NOT.QCBORD) I2=I2+2                                           ORDI0300
      I2 = MIND(NUM,I2)                                                 ORDI0310
      DO 12 J=1,NVORD                                                   ORDI0320
      READ (INU,13) (WT(K),DT(K),K=1,4)                                 ORDI0330
   13 FORMAT(4(2F5.0,5X))                                               ORDI0340
      IF (PRTDAT) WRITE(6,81) (WT(K),DT(K),K=1,4)                       ORDI0350
   81 FORMAT(20X,4(2F10.3,4X))                                          ORDI0360
      K = 0                                                             ORDI0370
      DO 14 I=I1,I2                                                     ORDI0380
      K = K+1                                                           ORDI0390
      IF(.NOT.QCBORD) GO TO 15                                          ORDI0400
```

```
      K = K+1                                                           ORDI0410
      WORD(J,I) = WORD(J,I)+WT(K-1)*PI(K-1)                             ORDI0420
      DORD(J,I) = DORD(J,I)+PT(K-1)                                     ORDI0430
      WZ(J,N) = WZ(J,N)+WT(K-1)*PT(K-1)                                 ORDI0440
      IF(I.EQ.NUM .AND. NMORD) GO TO 14                                 ORDI0450
   15 WORD(J,I) = WORD(J,I)+WT(K)*PT(K)                                 ORDI0460
      DORD(J,I) = DORD(J,I)+PT(K)                                       ORDI0470
      WZ(J,N) = WZ(J,N)+WT(K)*PT(K)                                     ORDI0480
   14 CONTINUE                                                          ORDI0490
   12 CONTINUE                                                          ORDI0500
      I1 = I2+1                                                         ORDI0510
      IF(I2.LT.NUM) GO TO 16                                            ORDI0520
   11 CONTINUE                                                          ORDI0530
      NMORD = NUM                                                       ORDI0540
C  SET AVG MASS ARRAYS (DIVIDE TOTAL MASS(J,I) BY TOT. NO. (J,I)  )     ORDI0550
      DO 20 J=1,NVORD                                                   ORDI0560
      DO 20 I=1,NMORD                                                   ORDI0570
      IF(DORD(J,I).LE.0.) GO TO 22                                      ORDI0580
      WORD(J,I) = WORD(J,I)/DORD(J,I)                                   ORDI0590
      GO TO 20                                                          ORDI0600
   22 WORD(J,I) = 0.                                                    ORDI0610
   20 CONTINUE                                                          ORDI0620
C  READ AVG VEL,S(J,N), WEIGHT BY TOTAL MASSES                          ORDI0630
      DO 23 N=1,NSETS                                                   ORDI0640
      IF (PRTDAT) WRITE(6,83) N                                         ORDI0650
   83 FORMAT(10X,18HVORD FOR COMPONENT,I2)                              ORDI0660
      DO 21 J=1,NVORD                                                   ORDI0670
      READ(INU,24) VT                                                   ORDI0680
   24 FORMAT(50X,F10.0)                                                 ORDI0690
      IF (PRTDAT) WRITE(6,84) VT                                        ORDI0700
   84 FORMAT(20X,F10.1)                                                 ORDI0710
      WZD(J) = WZD(J)+WZ(J,N)                                           ORDI0720
   21 VORD(J) = VT*WZ(J,N)+VORD(J)                                      ORDI0730
   23 CONTINUE                                                          ORDI0740
C  STORE OVERALL AVG V'S -- CONSERVE MOMENTUM                           ORDI0750
      DO 30 J=1,NVORD                                                   ORDI0760
      IF(WZD(J).LE.0.) GO TO 31                                         ORDI0770
      VORD(J) = VORD(J)/WZD(J)                                          ORDI0780
      GO TO 30                                                          ORDI0790
   31 VORD(J) = 0.                                                      ORDI0800
   30 CONTINUE                                                          ORDI0810
      RETURN                                                            ORDI0820
      END                                                               ORDI0830
```

Reproduced from best available copy.

```
      SUBROUTINE ORDPLT(X,N,V)                                          ORDP0010
      REAL V(1),DM(30),DA(30),WM(30),WA(30)                             ORDP0020
C  ORDNANCE TABLES                                                      ORDP0030
      COMMON /ORDCM/ MVORD,MMORD,IDORD,IVO,XKORD,                       ORDP0040
     * TTLORD,NVORD,NMORD,DTH,DTH2,                                     ORDP0050
     * VORD(37),AMORD(30),DORD(37,30),WORD(37,30)                       ORDP0060
      INTEGER TTLORD(12)                                                ORDP0070
C                                                                       ORDP0080
C     SUPPLY FUNCTION VALUES (PLOTTING ORDNANCE DATA)                   ORDP0090
C     X = 1 TO NVORD NORMALLY.  IF X.GE.1.E+38, INITIALIZE EVERYTHING.  ORDP0100
C     IF X.LE.0., INITIALIZE NEW PLOT.                                  ORDP0110
C                                                                       ORDP0120
      IF(X.GE.1.E+38) GO TO 10                                          ORDP0130
      IF(X.LT.0.) GO TO 20                                              ORDP0140
      I = I+1                                                           ORDP0150
      K = 1                                                             ORDP0160
      IF(NM1.GT.1) GO TO 2                                              ORDP0170
      V(1) = AMAX1(0.,VORD(I))*VM+VA                                    ORDP0180
      K = 2                                                             ORDP0190
    2 DO 1 J=NM1,NM2                                                    ORDP0200
      V(K) = AMAX1(0.,VORD(I,J))*DM(J)+DA(J)                            ORDP0210
      V(K+1) = AMAX1(0.,WORD(I,J))*WM(J)+WA(J)                          ORDP0220
    1 K = K+2                                                           ORDP0230
      RETURN                                                            ORDP0240
   20 NM1 = NM2+1                                                       ORDP0250
      NM2 = MIN0(NM1+3,NMORD)                                           ORDP0260
      RETURN                                                            ORDP0270
   10 NM1 = 1                                                           ORDP0280
      NM2 = MIN0(3,NMORD)                                               ORDP0290
      YMN = 0.                                                          ORDP0300
      YMX = 8.0                                                         ORDP0310
      DY = 2.06                                                         ORDP0320
      DP = 1.82                                                         ORDP0330
      K = 1                                                             ORDP0340
      DO 11 I=1,NMORD                                                   ORDP0350
      WA(I) = V(K)                                                      ORDP0360
      WM(I) = V(K+1)                                                    ORDP0370
      DA(I) = V(K+2)                                                    ORDP0380
      DM(I) = V(K+3)                                                    ORDP0390
   11 K = K+4                                                           ORDP0400
```

```
      K = 4*NMORD                                                       ORDP0410
      VA = V(K+1)                                                       ORDP0420
      VM = V(K+2)                                                       ORDP0430
      VM = DP/(VM-VA)                                                   ORDP0440
      VA = YMN-VA*VM                                                    ORDP0450
      Y = YMN+DY                                                        ORDP0460
      DO 14 J=1,NMORD                                                   ORDP0470
      DM(J) = DP/(DM(J)-DA(J))                                          ORDP0480
      DA(J) = Y-DA(J)*DM(J)                                             ORDP0490
      WM(J) = DP/(WM(J)-WA(J))                                          ORDP0500
      WA(J) = Y-WA(J)*WM(J)                                             ORDP0510
      Y = Y+DY                                                          ORDP0520
      IF(Y.GE.YMX) Y=YMX                                                ORDP0530
   14 CONTINUE                                                          ORDP0540
      I = 0                                                             ORDP0550
      RETURN                                                            ORDP0560
      END                                                               ORDP0570
```

Reproduced from best available copy.

```
      SUBROUTINE FOUR(F,C,N,M)                                          FOUR0010
C                                                                       FOUR0020
C     FIND FOURIER COEFFICIENTS THRU ORDER M, STORE IN ARRAY C          FOUR0030
C     C(1) IS BIAS,  C(2*K) IS SIN-K COEF,  C(2*K+1) IS COS-K COEF,     FOUR0040
C     FUNCTION F IS DEFINED AT MIDPOINTS OF N EQUI-SPACED INTERVALS.    FOUR0050
C                                                                       FOUR0060
      REAL F(1),C(1)                                                    FOUR0070
      DATA PI/3.14159265/                                               FOUR0080
C                                                                       FOUR0090
      N21 = N+M+1                                                       FOUR0100
      DO 1 I=1,N21                                                      FOUR0110
    1 C(I) = 0.                                                         FOUR0120
      DX = PI/FLOAT(N)                                                  FOUR0130
      SX = SIN(DX)                                                      FOUR0140
      CX = COS(DX)                                                      FOUR0150
      SDX = 2.*SX*CX                                                    FOUR0160
      CDX = CX*CX-SX*SX                                                 FOUR0170
      DO 2 I=1,N                                                        FOUR0180
      C(1) = C(1)+F(I)                                                  FOUR0190
      IF(M.LE.0) GO TO 5                                                FOUR0200
      SNX = SX                                                          FOUR0210
      CNX = CX                                                          FOUR0220
      DO 3 K=1,M                                                        FOUR0230
      C(2*K  ) = C(2*K  )+F(I)*SNX                                      FOUR0240
      C(2*K+1) = C(2*K+1)+F(I)*CNX                                      FOUR0250
      DUM = SNX*CX+CNX*SX                                               FOUR0260
      CNX = CNX*CX-SNX*SX                                               FOUR0270
      SNX = DUM                                                         FOUR0280
    3 CONTINUE                                                          FOUR0290
    5 DUM = SX*CDX+CX*SDX                                               FOUR0300
      CX = CX*CDX-SX*SDX                                                FOUR0310
      SX = DUM                                                          FOUR0320
    2 CONTINUE                                                          FOUR0330
      C(1) = C(1)/PI*DX                                                 FOUR0340
      IF(M.LE.0) RETURN                                                 FOUR0350
      DO 4 I=2,N21                                                      FOUR0360
    4 C(I) = 2.*C(I)/PI*DX                                              FOUR0370
      RETURN                                                            FOUR0380
      END                                                               FOUR0390
```

```
      SUBROUTINE FORGE (F,C,N,M)                                        FORG0010
C                                                                       FORG0020
C     GIVEN FOURIER COEFFICIENTS THRU ORDER M, COMPUTE FUNCTION VALUES  FORG0030
C     AT MIDPOINTS OF N EQUI-SPACED INTERVALS.  COEFFICIENTS ARE        FORG0040
C     IN C. STORE FUNCTION VALUES IN F.                                 FORG0050
C                                                                       FORG0060
      REAL F(1),C(1)                                                    FORG0070
      DATA PI/3.14159265/                                               FORG0080
C                                                                       FORG0090
      DUM = PI/FLOAT(N)                                                 FORG0100
      SX = SIN(DUM)                                                     FORG0110
      CX = COS(DUM)                                                     FORG0120
      SDX = 2.*SX*CX                                                    FORG0130
      CDX = CX*CX-SX*SX                                                 FORG0140
      DO 2 I=1,N                                                        FORG0150
      F(I) = C(1)                                                       FORG0160
      IF(M.LE.0) GO TO 5                                                FORG0170
      SNX = SX                                                          FORG0180
      CNX = CX                                                          FORG0190
      DO 3 K=1,M                                                        FORG0200
      F(I) = F(I)+C(2*K)*SNX+C(2*K+1)*CNX                               FORG0210
      DUM = SNX*CX+CNX*SX                                               FORG0220
      CNX = CNX*CX-SNX*SX                                               FORG0230
      SNX = DUM                                                         FORG0240
    3 CONTINUE                                                          FORG0250
    5 DUM = SX*CDX+CX*SDX                                               FORG0260
      CX = CX*CDX-SX*SDX                                                FORG0270
      SX = DUM                                                          FORG0280
    2 CONTINUE                                                          FORG0290
      RETURN                                                            FORG0300
      END                                                               FORG0310
```

Reproduced from best available copy.

```
      SUBROUTINE BOOM                                                   BOOM0010
C                                                                       BOOM0020
C     GIVEN TRAJECTORY TABLE AND ORDNANCE DATA, COMPUTES FRAGMENT FIELD,BOOM0030
C     ORDNANCE DATA IS IN /ORDCM/,                                      BOOM0040
C     TRAJECTORY TABLE IS IN /TRAJCM/                                   BOOM0050
C     FRAGMENT FIELD IS WRITTEN OUT ON FILE 'IFRTAP'                    BOOM0060
C     FIELD IS COMPUTED ON A HORIZONTAL HALF-CIRCLE OF RADIUS           BOOM0070
C     'NRNG*DRNG' AND AT AN ALTITUDE OF 'Z'                             BOOM0080
C     LINEAR AND CIRCULAR BARRIERS MAY BE PLACED IN THE FIELD.          BOOM0090
C                                                                       BOOM0100
C  CONTROL CARD TABLES AND GENERAL PROGRAM CONTROL VARIABLES            BOOM0110
      COMMON /CNTRL/ QCCRD,QCCPRT,KTCD,FCC(8),KARD(14),PC,PS,PD,PP,     BOOM0120
     * TXTIME(2),TXDATE(2),WWW(20),KKARD(800)                           BOOM0130
      LOGICAL QCCRD,QCCPRT,WWW                                          BOOM0140
      INTEGER TXTIME,TXDATE,FCC,PC,PS,PD,PP                             BOOM0150
C                                                                       BOOM0160
      LOGICAL PRTDAT                                                    BOOM0170
      EQUIVALENCE (KKARD(459),PRTDAT),(KKARD(457),DRNG),                BOOM0180
     * (KKARD(449),IPOSOP),(KKARD(433),Z),                              BOOM0190
     * (KKARD(425),NRAD),(KKARD(417),NALF),(KKARD(409),NPH),            BOOM0200
     * (KKARD(465),NRNG)                                                BOOM0210
C                                                                       BOOM0220
C  ORDNANCE TABLES                                                      BOOM0230
      COMMON /ORDCM/ MVORD,MMORD,IDORD,IVO,XKORD,                       BOOM0240
     * TTLORD,NVORD,NMORD,DTH,DTH2,                                     BOOM0250
     * VORD(37),AMORD(30),DORD(37,30),WORD(37,30)                       BOOM0260
      INTEGER TTLORD(12)                                                BOOM0270
      EQUIVALENCE (NM,MMORD)                                            BOOM0280
C                                                                       BOOM0290
      DIMENSION PACK(20,180),KPK1(60),KPK2(60),ALFHI(30),               BOOM0300
     * TEMP(4,30,40),KDTA(30),KDTD(30),IDIP(2),IPOSOP(2)                BOOM0310
      DIMENSION IBTYP(1),BCPH(1),BVPH(1),BHAL(1),TRHAL(1)               BOOM0320
      DIMENSION CL(3),CH(3)                                             BOOM0330
      LOGICAL QBAR,QBARPH                                               BOOM0340
      INTEGER TTLFRG(12)                                                BOOM0350
      DATA PI,RADEG/3.14159265,0.0174532925/                            BOOM0360
      DATA IHDR,IDIP/6H*HEAD*,6H000000 ,2H  /                           BOOM0370
C     IFRTAP   FRAGMENT FIELD OUTPUT TAPE                               BOOM0380
C     NWN      NUMBER OF WORDS IN A TAPE I.D.                           BOOM0390
      DATA IFRTAP/4/,NWN/2/                                             BOOM0400
```

```
C       MXPH      MAX NO AZIMUTH RAYS ALLOWED                                  BOOM0410
C       MXRNG     MAX NO RANGES ALLOWED                                        BOOM0420
C       MXRAY     MAX ALLOWED NO TRAJ. CALCS PER AZ RAY                        BOOM0430
        DATA MXPH,MXRNG,MXRAY/36,20,40/                                        BOOM0440
C       FPM1, FPM2   FITTING TOLERANCES FOR 'CFIT'                             BOOM0450
        DATA FPM1,FPM2/0.,0.,5/                                                BOOM0460
C       EPS       HEIGHT TOLERANCE FOR CONVERGENCE, 1-STEP INTEGRATION         BOOM0470
C       MKOUNT    MAX NO ITERATIONS ALLOWED, 1-STEP INTEGRATION                BOOM0480
        DATA EPS/0.1/,MKOUNT/10/                                               BOOM0490
C       G         ACCELERATION OF GRAVITY, FT/SEC**2                           BOOM0500
C       G2        HALF OF G                                                    BOOM0510
C       RETAO     DRAG COEF * AIR DENS * 7000 / (2*144)                        BOOM0520
        DATA G,G2,RETAO/32.17,16.085,2.3269/                                   BOOM0530
C       CL,CH     VALUE LIMITS FOR OUTPUT VALUES.. VEL., ANGLE, DENSITY        BOOM0540
        DATA CL/0.,-1.5708,0./,CH/1.E+38,+1.5708,1.E+38/                       BOOM0550
C                                                                              BOOM0560
        WHY(4) = WHY(7) .OR. WHY(3)                                            BOOM0570
        IF (NPH.LE.0 .OR. NPH.GT.MXPH .OR. NALF.LE.0 .OR. NALF.GT.MXRAY        BOOM0580
       * .OR. NRNG.LE.0 .OR. NRNG.GT.MXRAY .OR. NRNG.GT.MXRNG .OR.             BOOM0590
       * DRNG.LE.0.) GO TO 910                                                 BOOM0600
        KITEMP = MMORD*4                                                       BOOM0610
        IVA = PD                                                               BOOM0620
        IVF = PD                                                               BOOM0630
        CALL MOVE(IVA,1,1,IPOSOP,1)                                            BOOM0640
        CALL MOVE(IVF,1,1,IPOSOP,2)                                            BOOM0650
        IF (IVA.EQ.PS) IVA=PD                                                  BOOM0660
        IF (IVF.EQ.PS) IVF=PD                                                  BOOM0670
        DO 212 I=1,NWN                                                         BOOM0680
  212   IPOSOP(I) = IDIP(I)                                                    BOOM0690
        I = IFXO(IPOSOP,1,2,0,IDORD)                                           BOOM0700
        CALL MOVE(IPOSOP,3,1,IVO,1)                                            BOOM0710
        CALL MOVE(IPOSOP,4,1,IVA,1)                                            BOOM0720
        CALL MOVE(IPOSOP,5,1,IVF,1)                                            BOOM0730
        READ(5,210) TTLFRG                                                     BOOM0740
  210   FORMAT(12A6)                                                           BOOM0750
        WRITE (6,211) TTLFRG                                                   BOOM0760
  211   FORMAT(1H0,18X,12A6)                                                   BOOM0770
        WRITE (6,213) NPH,NALF,NRNG,DRNG,IPOSOP                                BOOM0780
  213   FORMAT(1H0,18X,5HNPH =,I3,5X,6HNALF =,I3,5X,6HNRNG =,I3,5X,            BOOM0790
       * 6HDRNG =,F6.1,5X,11HTAPE KEY = ,A6,A2)                                BOOM0800
```

Reproduced from best available copy.

```
C  READ BARRIER INFORMATION, IF INDICATED                              BOOM0810
      IF (NBAR.LE.0) GO TO 202                                          BOOM0820
      IF (NBAR.GT.1) GO TO 905                                          BOOM0830
      WRITE (6,224)                                                     BOOM0840
  224 FORMAT(1H0,18X,19HBARRIER INFORMATION)                            BOOM0850
      DO 229 I=1,NBAR                                                   BOOM0860
      READ(5,220) IBTYP(I),IDUM,BCPH(I),BWPH(I),BHAL(I)                 BOOM0870
  220 FORMAT(2I5,6F10.0)                                                BOOM0880
      IF(PRTDAT) WRITE(6,221) IBTYP(I),IDUM,BCPH(I),BWPH(I),BHAL(I)     BOOM0890
  221 FORMAT(19X,2I5,3F15.5)                                            BOOM0900
      BCPH(I) = BCPH(I)*RADEG                                           BOOM0910
      BWPH(I) = BWPH(I)*RADEG                                           BOOM0920
      BHAL(I) = BHAL(I)*RADEG                                           BOOM0930
      TBHAL(I) = TAN(BHAL(I))                                           BOOM0940
  229 CONTINUE                                                          BOOM0950
C  RETURN IF ERROR FLAGS ARE SET.                                       BOOM0960
  202 IF(WHY(1).OR.WHY(4).OR.WHY(6)) GO TO 100                          BOOM0970
C  BEGIN FRAGMENT FIELD OUTPUT                                          BOOM0980
      WRITE (IFRTAP) IDOB,IPOSOB                                        BOOM0990
      WRITE (IFRTAP) TTLOBD,TTLFDG,NM,XKORD,NRNG,DRNG,NPH               BOOM1000
      WRITE (IFRTAP) (XKORD(I),I=1,NM)                                  BOOM1010
      IMX = 6*NM                                                        BOOM1020
      IMY = 2*NM                                                        BOOM1030
      DAL = PI/(2.*FLOAT(NALF))                                         BOOM1040
      DPH = PI/FLOAT(NPH)                                               BOOM1050
      DPH2 = DPH/2.                                                     BOOM1060
      DAL2 = DAL/2.                                                     BOOM1070
      PHI = DPH2                                                        BOOM1080
C                                                                       BOOM1090
C  COMPUTE FRAGMENT FIELD FOR EVERY AZIMUTH ANGLE (NPH OF THEM)         BOOM1100
C                                                                       BOOM1110
      DO 401 IPH=1,NPH                                                  BOOM1120
      DO 404 IM=1,NM                                                    BOOM1130
  404 KPTA(IM) = 0                                                      BOOM1140
      DO 403 IM=1,IMY                                                   BOOM1150
  403 KPK1(IM) = 0                                                      BOOM1160
      SPHI = SIN(PHI)                                                   BOOM1170
      CPHI = COS(PHI)                                                   BOOM1180
C  TEST FOR BARRIER(S) ALONG CURRENT AZIMUTH                            BOOM1190
      ALFBAR = 0.                                                       BOOM1200
```

```
      IF(NBAR.LE.0) GO TO 249                                           BOOM1210
      DO 248 I=1,NBAR                                                   BOOM1220
      DUM = ABS(PHI-BCPH(I))                                            BOOM1230
      IF(DUM.GT.BKPH(I)) GO TO 248                                      BOOM1240
      IF(IBTYP(I).NE.1) GO TO 240                                       BOOM1250
      ALFBAR = AMAX1(ALFBAR,BHAL(I))                                    BOOM1260
      GO TO 248                                                         BOOM1270
  240 ALFBAR = AMAX1(ALFBAR,ATAN(BHAL(I)*COS(DUM)))                     BOOM1280
  248 CONTINUE                                                          BOOM1290
  249 QBARPH = ALFBAR*RT.C.                                             BOOM1300
C                                                                       BOOM1310
C  COMPUTE ASCENDING FRAGS AND DESC LOW-REG FRAGS, STEPPING RANGE       BOOM1320
C                                                                       BOOM1330
      DO 395 IM=1,NM                                                    BOOM1340
      ALF = 0.                                                          BOOM1350
      RNG = DRNG                                                        BOOM1360
      DO 390 IR=1,NRNG                                                  BOOM1370
      TEST2L = 1.E+35                                                   BOOM1380
      R2 = RNG*RNG                                                      BOOM1390
      U = 0.                                                            BOOM1400
      T = 0.                                                            BOOM1410
      H = Z                                                             BOOM1420
      DUM = SQRT(H*H+R2)                                                BOOM1430
      SAP = H/DUM                                                       BOOM1440
      CAP = RNG/DUM                                                     BOOM1450
C  1-STEP INTEGRATION METHOD -- ITERATE FOR SATISFACTORY INITIAL ALPHA  BOOM1460
      DO 315 ICOUNT=1,MKCNT                                             BOOM1470
      TH = ACOS(CAP*CPHI)                                               BOOM1480
      CALL INTSET(NVORD,DTH2,DTH,TH,ITH,PTH)                            BOOM1490
      VORDI = AMAX1(0.,FINET(VORD,ITH,PTH))                             BOOM1500
      WORDI = AMAX1(0.,FINET(WORD(1,IM),ITH,PTH))                       BOOM1510
      BETA = BETAO/(XKORD*XKORD*WORDI)**0.333333                        BOOM1520
      XB = -G2*T*T*SAP*(1.+U/3.)/(1.+U)                                 BOOM1530
      YB = -G2*T*T*CAP*(U*(1.+U*0.5)-ALOG(1.+U))/(U*U)                  BOOM1540
      XBO = RNG/CAP+YB*SAP/CAP-XB                                       BOOM1550
      DUM = BETA*XBO                                                    BOOM1560
C***  ABORT ITERATION IF EXP(DUM) WOULD BE VERY LARGE OR SMALL          BOOM1570
      IF (ABS(DUM).GT.12.) GO TO 392                                    BOOM1580
      U = EXP(DUM)-1.                                                   BOOM1590
      T=U/(BETA*VORDI)                                                  BOOM1600
```

Reproduced from best available copy.

```
      XB = -G2*T*T*SAP*(1.+U/3.)/(1.+U)                                 BOOM1610
      YB = -G2*T*T*CAP*(U*(1.+U*0.5)-ALOG(1.+U))/(U*U)                  BOOM1620
      H = (XBO+XB)*SAP+YB*CAP                                           BOOM1630
      DUM1 = Z-H                                                        BOOM1640
      DUM = SQRT(DUM1*DUM1+R2)                                          BOOM1650
      SDA = DUM1/DUM                                                    BOOM1660
      CDA = RNG/DUM                                                     BOOM1670
      DUM = SAP*CDA+CAP*SDA                                             BOOM1680
      CAP = CAP*CDA-SAP*SDA                                             BOOM1690
      SAP = DUM                                                         BOOM1700
      TEST2=ABS(DUM1)                                                   BOOM1710
C  DIVERGENCE INDICATES THAT RNG.GT.MAX RANGE FOR GIVEN INIT. CONDS.    BOOM1720
      IF(TEST2.GT.TEST2L) GO TO 392                                     BOOM1730
C  CONVERGENCE IF TERM. HEIGHT CLOSE ENOUGH TO Z.                       BOOM1740
      IF (TEST2.LT.EPS) GO TO 320                                       BOOM1750
      TEST2L=TEST2                                                      BOOM1760
  315 CONTINUE                                                          BOOM1770
      GO TO 392                                                         BOOM1780
  320 XDO = VORDI/(1.+U)                                                BOOM1790
      XDB = -G*T*SAP*(1.+U*(1.+U/3.))/(1.+U)**2                         BOOM1800
      YDB = -G*T*CAP*(1.+U*0.5)/(1.+U)                                  BOOM1810
      XD = (XDO+XDB)*CAP-YDB*SAP                                        BOOM1820
      YD = (XDO+XDB)*SAP+YDB*CAP                                        BOOM1830
      VELT = SQRT(XD*XD+YD*YD)                                          BOOM1840
      DUM1 = XDO+XDB                                                    BOOM1850
      DUM = SQRT(YDB*YDB+DUM1*DUM1)                                     BOOM1860
      SDA = YDB/DUM                                                     BOOM1870
      CDA = DUM1/DUM                                                    BOOM1880
      SAP = SAP*CDA+CAP*SDA                                             BOOM1890
      CAP = CAP*CDA-SAP*SDA                                             BOOM1900
      DUM = 2.*SAP*CAP                                                  BOOM1910
      DRDA = (2.*RNG-XBO*CAP)*((CAP*CAP-SAP*SAP)/DUM-CAP/SAP)           BOOM1920
     *       -XBO*CAP/DUM                                               BOOM1930
      DORDI = AMAX1(0.,FINET(DORD(1,IM),ITH,PTH))                       BOOM1940
      ALF = ATAN2(SAP,CAP)                                              BOOM1950
      QBAR = QBARPH .AND. ALF.LE.ALFBAR                                 BOOM1960
      I = KPTA(IM)+1                                                    BOOM1970
      KPTA(IM) = I                                                      BOOM1980
      TEMP(1,IM,I) = RNG                                                BOOM1990
      TEMP(2,IM,I) = VELT                                               BOOM2000
```

```
      TEMP(3,IM,I) = ATAN2(SAF,CAF)                                     BOOM2010
      TEMP(4,IM,I) = 0.                                                 BOOM2020
C***  NOTE THAT UNIFORM-SOURCE FORM OF DR/D ALPHA IS USED               BOOM2030
      IF (.NOT.QBAR) TEMP(4,IM,I)=DORDI*CAF/ABS(RNG*DRDA*SAF)           BOOM2040
  390 RNG = RNG+DRNG                                                    BOOM2050
  392 ALFHI(IM) = ALF                                                   BOOM2060
  395 CONTINUE                                                          BOOM2070
C                                                                       BOOM2080
C  COMPUTE HIGH-REC DESCENDING FRAGS, STEPPING ELEVATION ANGLE          BOOM2090
C                                                                       BOOM2100
      ALFLO = PI                                                        BOOM2110
      DO 405 IM=1,NM                                                    BOOM2120
      KPTD(IM) = KPTA(IM)                                               BOOM2130
      ALFLO = AMIN1(ALFLO,ALFHI(IM))                                    BOOM2140
  405 CONTINUE                                                          BOOM2150
      KAL = (ALFLO+DAL2)/DAL+1.                                         BOOM2160
      ALF = FLOAT(KAL)*DAL-DAL2                                         BOOM2170
      DO 402 IAL=KAL,NALF                                               BOOM2180
      SALF = SIN(ALF)                                                   BOOM2190
      CALF = COS(ALF)                                                   BOOM2200
      QBAR = QBARPH .AND. ALF.LE.ALFBAR                                 BOOM2210
C  THETA IS POLAR ANGLE ON SPHERE (NOSE=0, TAIL=180.)                   BOOM2220
      CTH = CALF*CPHI                                                   BOOM2230
      TH = ACOS(CTH)                                                    BOOM2240
      STH = SQRT(1.-CTH*CTH)                                            BOOM2250
      DTHDAL = SALF*CPHI/STH                                            BOOM2260
C  PSI IS LONGITUDINAL ANGLE ON SPHERE, OBSERVED LOOKING FROM TAIL TO   BOOM2270
C  NOSE,  (UP=0,  AT AZIM=90, PSI=90-ELEV)                              BOOM2280
      SPSI = CALF*SPHI/STH                                              BOOM2290
      CPSI = SALF/STH                                                   BOOM2300
      CALL INTSET(NVORD,DTH2,DTH,TH,ITH,PTH)                            BOOM2310
      VORDI = FINFT(VORD,ITH,PTH)                                       BOOM2320
      VORDI = AMAX1(0.,VORDI)                                           BOOM2330
      DVORDI = FINFD(VORD,ITH,PTH,DTH)                                  BOOM2340
C  COMPUTE FIELD FOR EACH MASS CATEGORY                                 BOOM2350
      DO 410 IM=1,NMORD                                                 BOOM2360
      IF (ALF.LE.ALFHI(IM)) GO TO 410                                   BOOM2370
      WORDI = FINFT(WORD(1,IM),ITH,PTH)                                 BOOM2380
      WORDI = AMAX1(0.,WORDI)                                           BOOM2390
      DORDI = FINFT(DORD(1,IM),ITH,PTH)                                 BOOM2400
```

Reproduced from best available copy.

```
      DORDI = AMAX1(0.,DORDI)                                           BOOM2410
      CALL NSTEP(XKORD,YORDI,VORDI,-7,ALF,RNGT,VELT,SALFT,CALFT)        BOOM2420
      ALFT = ATAN2(SALFT,CALFT)                                         BOOM2430
      I = KPTD(IM)+1                                                    BOOM2440
      IF (I.GT.MXRAY) GO TO 900                                         BOOM2450
      KPTD(IM) = I                                                      BOOM2460
      TEMP(1,IM,I) = RNGT                                               BOOM2470
      TEMP(2,IM,I) = VELT                                               BOOM2480
      TEMP(3,IM,I) = ALFT                                               BOOM2490
      TEMP(4,IM,I) = 0.                                                 BOOM2500
C***  NOTE THAT DR/DA TAKEN AS DELTA R/DELTA ALPHA                      BOOM2510
      IF(.NOT.QBAR) TEMP(4,IM,I) = DORDI*CALF*DAL/ABS(RNGT*SALFT)       BOOM2520
  410 CONTINUE                                                          BOOM2530
  402 ALF = ALF+DAL                                                     BOOM2540
C                                                                       BOOM2550
C  COMPLETE PROCESSING OF TERMINAL CHARACTERISTICS                      BOOM2560
C                                                                       BOOM2570
      IMM = 0                                                           BOOM2580
      JM = 0                                                            BOOM2590
      DO 420 IM=1,NMORR                                                 BOOM2600
C  COMPLETE FRAGMENT DENSITY COMPUTATION FOR DESCENDING LEG OF TRAJ.    BOOM2610
      I1 = KPTA(IM)+1                                                   BOOM2620
      I2 = KPTD(IM)                                                     BOOM2630
      IF (I1.GT.I2) GO TO 465                                           BOOM2640
      DUM1 = TEMP(1,IM,1)                                               BOOM2650
      DUM = DAL*DUM1                                                    BOOM2660
      DO 461 I=I1,I2                                                    BOOM2670
      IF (I.EQ.I2) GO TO 462                                            BOOM2680
      DUM2 = TEMP(1,IM,I+1)                                             BOOM2690
      DUM = ABS(DUM2-DUM1)                                              BOOM2700
      DUM1 = DUM2                                                       BOOM2710
  462 TEMP(4,IM,I) = TEMP(4,IM,I)/DUM                                   BOOM2720
  461 CONTINUE                                                          BOOM2730
  465 IE = 0                                                            BOOM2740
  421 IE = IE+1                                                         BOOM2750
      GO TO (422,423,420),IE                                            BOOM2760
C  FOR ASC. AND LOW-REG. DESC. FRAGMENTS, INSERT INFO INTO OUTPUT TABLESBOOM2770
  422 IN = KPTA(IM)                                                     BOOM2780
      IF (IN.LE.0) GO TO 490                                            BOOM2790
      JM = JM+1                                                         BOOM2800
```

```
      KPK1(JM) = 1                                                      BOOM2810
      KPK2(JM) = IN                                                     BOOM2820
      DO 424 I=1,3                                                      BOOM2830
      IMM = IMM+1                                                       BOOM2840
      DO 424 J=1,IN                                                     BOOM2850
  424 PACK(J,IMM) = TEMP(I+1,IM,J)                                      BOOM2860
      GO TO 421                                                         BOOM2870
C  FOR DESCENDING HIGH-REG. FRAGMENTS, INTERPOLATE FOR OUTPUT INFO.     BOOM2880
  423 IF (KPTD(IM).LE.IN) GO TO 490                                     BOOM2890
      IS = IN+1                                                         BOOM2900
      II = KPTD(IM)                                                     BOOM2910
C  FIRST, REORDER TRAJ'S TO ORDER OF ASC. RANGE (FOR THE MOST PART)     BOOM2920
      J = II                                                            BOOM2930
      IDUM = IS+(II-IS-1)/2                                             BOOM2940
      DO 426 I=IS,IDUM                                                  BOOM2950
      DO 425 K=1,4                                                      BOOM2960
      DUM = TEMP(K,IM,I)                                                BOOM2970
      TEMP(K,IM,I) = TEMP(K,IM,J)                                       BOOM2980
  425 TEMP(K,IM,J) = DUM                                                BOOM2990
  426 J = J-1                                                           BOOM3000
C  ELIMINATE ANY LOW-REG. TRAJ'S., I.E., MAKE RANGE INCR. MONOTONICALLY BOOM3010
      J = II-1                                                          BOOM3020
      DO 427 I=IS,J                                                     BOOM3030
      IF (TEMP(1,IM,I+1).LE.TEMP(1,IM,I)) GO TO 428                     BOOM3040
  427 CONTINUE                                                          BOOM3050
      I = II                                                            BOOM3060
  428 II = I                                                            BOOM3070
      IN = II-IN                                                        BOOM3080
      IDUM = MAX0(1,MIN0(NRNG,TEMP(1,IM,IS)/DRNG+0.999999))             BOOM3090
      JDUM = MAX0(IDUM,MIN0(NRNG,TEMP(1,IM,II)/DRNG))                   BOOM3100
      JM = JM+1                                                         BOOM3110
      KPK1(JM) = IDUM                                                   BOOM3120
      KPK2(JM) = JDUM                                                   BOOM3130
      DO 430 I=1,3                                                      BOOM3140
      IMM = IMM+1                                                       BOOM3150
      RNG = DRNG*FLOAT(IDUM)                                            BOOM3160
      DO 431 J=IDUM,JDUM                                                BOOM3170
      DUM = FLINT(TEMP(1,IM,IS),KITEMP,TEMP(I+1,IM,IS),KITEMP,IN,RNG)   BOOM3180
      DUM = AMAX1(CL(I),AMIN1(CH(I),DUM))                               BOOM3190
      PACK(J,IMM) = DUM                                                 BOOM3200
```

Reproduced from best available copy.

```
      RNG = RNG+DRNG                                                    B00M3210
  431 CONTINUE                                                          B00M3220
  430 CONTINUE                                                          B00M3230
      GO TO 421                                                         B00M3240
C  IF NO DATA FOR A TRAJ. TYPE, FILL IN ZEROES                          B00M3250
  490 DO 491 I=1,3                                                      B00M3260
      IMM = IMM+1                                                       B00M3270
      DO 492 J=1,NRNG                                                   B00M3280
  492 PACK(J,IMM) = 0.                                                  B00M3290
  491 CONTINUE                                                          B00M3300
      JM = JM+1                                                         B00M3310
      GO TO 421                                                         B00M3320
  420 CONTINUE                                                          B00M3330
C  OUTPUT SMOOTHED TERR. CHAR. FOR THIS AZ. RAY                         B00M3340
      WRITE (IFRTAP) (KPK1(J),KPK2(J),I=1,IMY),                         B00M3350
     * ((PACK(I,J),I=1,NRNG),J=1,IMX)                                   B00M3360
  401 PHI = PHI+DPH                                                     B00M3370
      GO TO 100                                                         B00M3380
C                                                                       B00M3390
C  ERRORS                                                               B00M3400
C                                                                       B00M3410
  900 IF (.NOT. QCCPRT) WRITE(6,FCC) PC,KTCD,KARD                       B00M3420
      QCCPRT = .TRUE.                                                   B00M3430
      WRITE (6,901) KPTA(IM),I,IPH,IM                                   B00M3440
  901 FORMAT(22H0******  RAY OVERFLOW,,I4,10H R-POINTS,,I3,             B00M3450
     * 20H TOTAL POINTS ON RAY,I3,10H, MASS CAT,I3)                     B00M3460
      GO TO 990                                                         B00M3470
  905 IF (.NOT.QCCPRT) WRITE(6,FCC) PC,KTCD,KARD                        B00M3480
      QCCPRT = .TRUE.                                                   B00M3490
      WRITE (6,906)                                                     B00M3500
  906 FORMAT(27H0******  TOO MANY BARRIERS.)                            B00M3510
      GO TO 990                                                         B00M3520
  910 IF (.NOT.QCCPRT) WRITE(6,FCC) PC,KTCD,KARD                        B00M3530
      QCCPRT = .TRUE.                                                   B00M3540
      WRITE (6,911)                                                     B00M3550
  911 FORMAT(21H0******  BAD VALUE(S))                                  B00M3560
  990 CALL FLUSH                                                        B00M3570
      WHY(4) = .TRUE.                                                   B00M3580
  100 PC = PD                                                           B00M3590
      RETURN                                                            B00M3600
      END                                                               B00M3610
```

```
      SUBROUTINE NSTEP(XK,XGR,VLO,HOR,ELVO,X,VL,SLV,CLV)
C
      DATA PI/3.141592653/
      DATA NDT/10/,BETAC/2.3060/
      DATA G/32.17/
C
      BT = BETAC/(XK*XK*XGR)**0.333333
      VST=SQRT(G/BT)
      TST=1./(BT*VST)
      VL = VLO
      X=0.
      Y=HOR
      T=0.
      DT=TST/FLOAT(NDT)
      SLV = SIN(ELVO)
      CLV = COS(ELVO)
    2 VL1=VL
      TH=BT*VL*DT
      TH1=1.+TH
      ALT=ALOG(TH1)
      UB=VL/TH1-G*SLV*DT*(TH1+TH*TH/3.)/TH1**2
      VB=-G*CLV*DT*(1.+TH/2.)/TH1
      DXB1=VL*DT
      IF(BT.GT.0.) DXB1=ALT/BT
      DXB=DXB1-G*SLV*DT*DT*(3.+TH)/(6.*TH1)
      ETHY=1.-TH/3.
      IF(TH.GT.1.E-04) ETHY=(TH*(2.+TH)-2.*ALT)/(2.*TH*TH)
      DYB=-G*CLV*DT*DT*ETHY/2.
      DX=DXB*CLV-DYB*SLV
      DY=DXB*SLV+DYB*CLV
      VL=SQRT(UB*UB+VB*VB)
      UB = UB/VL
      VB = VB/VL
      DUM = SLV*UB+CLV*VB
      CLV = CLV*UB-SLV*VB
      SLV = DUM
      X=X+DX
      Y=Y+DY
      T=T+DT
      IF (Y.GT.0. .OR. DY.GT.0.) GO TO 2
```

Reproduced from best available copy.

```
      VL=VL-Y*(VL-VLT)/DY                                               NSTE0410
      DUM = -Y*ATAN2(VB,UB)/DY                                          NSTE0420
      VB = SIN(DUM)                                                     NSTE0430
      UB = COS(DUM)                                                     NSTE0440
      DUM = SLV*UB+CLV*VB                                               NSTE0450
      CLV = CLV*UB-SLV*VB                                               NSTE0460
      SLV = DUM                                                         NSTE0470
      X=X-Y*DX/DY                                                       NSTE0480
      T=T-Y*DT/DY                                                       NSTE0490
      RETURN                                                            NSTE0500
      END                                                               NSTE0510
```

```
      SUBROUTINE INTSET(N,XL,DX,Y,IX,PX)                                INTS0010
C                                                                       INTS0020
C  FIND MAIN ENTRY NO. 'IX' AND DX-FRACT. 'PX', GIVEN TABLE OF 'N'      INTS0030
C  ENTRIES AT 'DX' INTERVAL, STARTING AT 'XL'.  SEARCH VALUE IS 'Y'     INTS0040
C                                                                       INTS0050
      IX = (Y-XL)/DX+1.0                                                INTS0060
      IX = MAX0(2,MIN0(IX,N-2))                                         INTS0070
      PX = (Y-XL)/DX-FLOAT(IX-1)                                        INTS0080
      RETURN                                                            INTS0090
      END                                                               INTS0100
```

```
      FUNCTION FINFT(Y,I,P)                                             FINF0010
      REAL Y(1)                                                         FINF0020
C                                                                       FINF0030
C     4-POINT (-1,0,+1,+2) LAGRANGE INTERPOLATION                       FINF0040
C                                                                       FINF0050
      P1 = P*P-1                                                        FINF0060
      P2 = P-2.                                                         FINF0070
      FINFT = -P*(P-1.)*P2*Y(I-1)/6.+P1*P2*Y(I)/2.-P*(P+1.)*P2*Y(I+1)/2.FINF0080
     * + P*P1*Y(I+2)/6.                                                 FINF0090
      RETURN                                                            FINF0100
      END                                                               FINF0110
```

Reproduced from best available copy.

```
      FUNCTION FINFD(Y,I,P,D)                                           FINF0010
      REAL Y(1)                                                         FINF0020
C                                                                       FINF0030
C     4-PT (-1,0,+1,+2) LAGRANGE DIFFERENTIATION                        FINF0040
C                                                                       FINF0050
      P1 = 3.*P*P                                                       FINF0060
      FINFD = ( (-P1+6.*P-2.)*Y(I-1)/6.+(P1-4.*P-1.)*Y(I)/2.            FINF0070
     * -(P1-2.*P-2.)*Y(I+1)/2.+(P1-1.)*Y(I+2)/6.)/D                     FINF0080
      RETURN                                                            FINF0090
      END                                                               FINF0100
```

```
      FUNCTION FLINT(XK,KX,YK,KY,N,X)                                   FLIN0010
      REAL XK(1),YK(1)                                                  FLIN0020
C                                                                       FLIN0030
C     LINEAR INTERPOLATION ROUTINE.                                     FLIN0040
C     THERE ARE N VALUE PAIRS (XK(IX),YK(IY)), WHERE IX=1,1+KX, ETC.,   FLIN0050
C     AND IY=1,1+KY, ETC.  THE ARGUMENT IS X.                           FLIN0060
C     RETURN 0 IF X IS OUT OF RANGE.  RETURN YK(1) IF N=1.              FLIN0070
C                                                                       FLIN0080
      JX = 1+(N-1)*KX                                                   FLIN0090
      IF (X.LT.XK(1).OR.X.GT.XK(JX)) GO TO 12                           FLIN0100
      IF (N.LT.2) GO TO 10                                              FLIN0110
      JX = 1                                                            FLIN0120
      DO 1 J=2,N                                                        FLIN0130
      JX = JX+KX                                                        FLIN0140
      IF (X.LE.XK(JX)) GO TO 2                                          FLIN0150
    1 CONTINUE                                                          FLIN0160
      J = N                                                             FLIN0170
    2 LX = JX                                                           FLIN0180
      JX = JX-KX                                                        FLIN0190
      JY = 1+(J-2)*KY                                                   FLIN0200
      LY = JY+KY                                                        FLIN0210
      FLINT = YK(JY)+(X-XK(JX))*(YK(LY)-YK(JY))/(XK(LX)-XK(JX))         FLIN0220
      RETURN                                                            FLIN0230
C  ONLY ONE VALUE IN TABLE.  RETURN IT.                                 FLIN0240
   10 FLINT = YK(1)                                                     FLIN0250
      RETURN                                                            FLIN0260
C  X IS OUT OF RANGE.  RETURN 0.                                        FLIN0270
   12 FLINT = 0.                                                        FLIN0280
      RETURN                                                            FLIN0290
      END                                                               FLIN0300
```

```
      SUBROUTINE LINK2                                                  LINK0010
C                                                                       LINK0020
C     GIVEN FRAGMENT FIELD DATA GENERATED BY 'PROGM', COMPUTE PROBABILITYLINK0030
C     FUNCTIONS AND OTHER FUNCTIONS OF THE FIELD PARAMETERS.            LINK0040
C     FRAGMENT FIELD INPUT IS READ FROM FILE 'INPTAP'                   LINK0050
C     FUNCTION VALUE TABLES ARE WRITTEN ON FILE 'PLTAP', FOR USE BY     LINK0060
C     A PLOT GENERATION PROGRAM.                                        LINK0070
C     FUNCTION VALUE CONTOUR PLOTS CAN ALSO BE GENERATED                LINK0080
C     ON THE PRINTER, FOR QUICK-LOOK RESULTS.                           LINK0090
C                                                                       LINK0100
C  CONTROL CARD TABLES AND GENERAL PROGRAM CONTROL VARIABLES            LINK0110
      COMMON /CNTRL/ GCCRD,GCCPRT,KTCD,FCC(8),KARD(14),PC,PS,PD,PF,     LINK0120
     * TXTIME(2),TXDATE(2),WHY(20),KKARD(800)                            LINK0130
      LOGICAL GCCRD,GCCPRT,WHY                                          LINK0140
      INTEGER TXTIME,TXDATE,FCC,PC,PS,PD,PF                             LINK0150
C                                                                       LINK0160
      LOGICAL PRTDAT,LIST,GWKPLT,EXEC                                   LINK0170
      EQUIVALENCE (KKARD(489),PRTDAT),(KKARD(377),LIST),                LINK0180
     * (KKARD(353),GWKPLT),(KKARD(385),EXEC)                            LINK0190
C                                                                       LINK0200
      IF(EXEC) GO TO 110                                                LINK0210
C                                                                       LINK0220
C  READ FUNCTION PARAMETERS, TABLES, ETC.                               LINK0230
C                                                                       LINK0240
      CALL OUTSET                                                       LINK0250
      GO TO 100                                                         LINK0260
C                                                                       LINK0270
C  COMPUTE AND OUTPUT FUNCTIONS                                         LINK0280
C                                                                       LINK0290
  110 CALL COMPUT                                                       LINK0300
      IF (WHY(5)) GO TO 100                                             LINK0310
      CALL OUTPUT                                                       LINK0320
C                                                                       LINK0330
  100 PC = PS                                                           LINK0340
      IF(LIST.OR.PRTDAT.OR.WHY(5)) PC=PD                                LINK0350
      IF (GWKPLT) PC=PF                                                 LINK0360
      RETURN                                                            LINK0370
      END                                                               LINK0380
```

Reproduced from best available copy.

```
      BLOCK DATA                                                        BLKC0010
C                                                                       BLKC0020
C  FUNCTION TABLES, PARAMETERS AND VALUE STORAGE                        BLKC0030
      COMMON /PLOTCM/ NRNG,DRNG,NPH,DPH,DPH2,IVO,IVA,IVF,IPOSOP(2),     BLKC0040
     * NSETS,IDORD,                                                     BLKC0050
     * TTLORD(12),TTLFRG(12),NPLOTS,NX,NY,XMIN,YMIN,DX,DY,NZ,CH(55),    BLKC0060
     * RECT(81,41),ITF(20),ISF(20),ITYF(20),IVPF(20),NF(20),F(14,20),   BLKC0070
     * ITTLF(12,20),KTF(20),DZMIN(20),DZMAX(20),XTB(14,10),YTB(14,10),  BLKC0080
     * C(7,10),NTB(10),KTB(10),OUT(20,36,8),ITTLP(12,8),KTPP(8),OLP(8), BLKC0090
     * OWP(8),ISP(8),ITP(8),DZMIN(8),DZMAX(8),ITYP(8),ITYS(8),IVPF(8)   BLKC0100
C                                                                       BLKC0110
      DATA KTF,DZMIN,DZMAX/20*0,20*-1.E+30,20*1.E+30/                   BLKC0120
      DATA NZ,CH/10,1H ,1H ,1H ,1H-,1H0,1H1,1H2,1H3,1H4,1H5,1H6,1H7,    BLKC0130
     * 1H8,1H9,1H+,40*B/                                                BLKC0140
C                                                                       BLKC0150
      END                                                               BLKC0160
```

```
      SUBROUTINE OUTSET                                                 OUTS0010
C                                                                       OUTS0020
C  CONTROL CARD TABLES AND GENERAL PROGRAM CONTROL VARIABLES            OUTS0030
      COMMON /CNTRL/ GCCRD,GCCPRT,KTCD,FCC(8),KARD(14),PC,PS,PD,PP,     OUTS0040
     * TXTIME(2),TXDATE(2),WHY(20),KKARD(600)                           OUTS0050
      LOGICAL GCCRD,GCCPRT,WHY                                          OUTS0060
      INTEGER TXTIME,TXDATE,FCC,PC,PS,PD,PP                             OUTS0070
C                                                                       OUTS0080
      LOGICAL PRTDAT,LIST,QSOJAC,RANGE,TABLES,LIMITS,PARAMS             OUTS0090
      EQUIVALENCE (KKARD(429),PRTDAT),(KKARD(377),LIST),                OUTS0100
     * (KKARD(401),QSOJAC),(KKARD(393),RANGE),                          OUTS0110
     * (KKARD(361),TABLES),(KKARD(337),LIMITS),(KKARD(345),PARAMS)      OUTS0120
C                                                                       OUTS0130
C  FUNCTION TABLES, PARAMETERS AND VALUE STORAGE                        OUTS0140
      COMMON /PLOTCM/ NRUG,DPNG,DPH,DPH2,IVO,IVA,IVF,IPOSOP(20),        OUTS0150
     * NSETS,IDORD,                                                     OUTS0160
     * TTLORD(12),TTLFRS(12),NPLOTS,NX,NY,XMIN,YMIN,DX,DY,NZ,CH(55),    OUTS0170
     * RECT(41,41),ITF(20),ISF(20),ITYF(20),IVPF(20),NF(20),F(14,20),   OUTS0180
     * ITTLF(12,20),KTF(20),DZMIN(20),DZMAX(20),XTB(14,10),YTB(14,10),  OUTS0190
     * C(7,10),NTB(10),KTB(10),OUT(20,36,8),ITTLP(12,8),KTPP(8),OLP(8), OUTS0200
     * OHP(8),ISP(8),ITP(8),PZMIN(8),PZMAX(8),ITYP(8),ITYS(8),IVPF(8)   OUTS0210
      INTEGER CH                                                        OUTS0220
C                                                                       OUTS0230
      INTEGER TXEXT(2),TXINT(2),TXXLL(2),TXYLL(2),TXFLL(2)              OUTS0240
      DATA MFCNS/20/,MF/14/                                             OUTS0250
      DATA TXEXT,TXINT,TXX,TXY/6HEXTERN,2HAL,6HINTERN,2HAL,1HX,1HY/     OUTS0260
      DATA TXXLL,TXYLL,TXFLL/5HX-LIN,5HX-LOG,5HY-LIN,5HY-LOG,6HLINEAR,  OUTS0270
     * 6HLOG   /                                                        OUTS0280
C                                                                       OUTS0290
      IF(RANGE) GO TO 120                                               OUTS0300
      IF(QSOJAC) GO TO 140                                              OUTS0310
      IF(TABLES) WHY(16)=.FALSE.                                        OUTS0320
      IF(LIMITS) WHY(17)=.FALSE.                                        OUTS0330
      IF(PARAMS) WHY(18)=.FALSE.                                        OUTS0340
      L = 0                                                             OUTS0350
C                                                                       OUTS0360
C  READ FUNCTION DEFINITIONS (TABLES, LIMITS, PARAMS)                   OUTS0370
C                                                                       OUTS0380
  300 READ (5,301) IT,IS,IVP,IDUM,JDUM,KDUM,ADUM,BDUM,CDUM,DDUM,EDUM    OUTS0390
  301 FORMAT(2I1,1X,A1,1X,3I5,5E10.0)                                   OUTS0400
```

Reproduced from best available copy.

```
      IF(IT.EQ.0 .AND. IS.EQ.0) GO TO 100                               OUTS0410
      IF (IT.LE.0) IT=1                                                 OUTS0420
      IF (IS.LE.0) IS=1                                                 OUTS0430
      L = L+1                                                           OUTS0440
      ITY = IT*10+IS                                                    OUTS0450
      IS = IS+1                                                         OUTS0460
      IF (IT.GT.0) GO TO 302                                            OUTS0470
      IT = IS+10                                                        OUTS0480
      IS = IT                                                           OUTS0490
  302 IF (TABLES) GO TO 320                                             OUTS0500
      IF(LIMITS) GO TO 340                                              OUTS0510
      IF(PARAMS) GO TO 360                                              OUTS0520
      IF(.NOT.OCCPRT) WRITE(6,FCC) PC,KTCD,KARD                         OUTS0530
      OCCPRT = .TRUE.                                                   OUTS0540
      WRITE(6,303)                                                      OUTS0550
  303 FORMAT(46H0****** REQUIRED FIELD(S) MISSING OR IN ERROR)          OUTS0560
      CALL FLUSH                                                        OUTS0570
      DO 306 I=16,20                                                    OUTS0580
  306 WHY(I) = .TRUE.                                                   OUTS0590
      GO TO 100                                                         OUTS0600
C                                                                       OUTS0610
C  READ TABLES AND/OR THOR COEFS.                                       OUTS0620
C                                                                       OUTS0630
  320 IT = (IT+2)/3                                                     OUTS0640
      IF(IT.GT.0 .AND. IT.LT.3) GO TO 323                               OUTS0650
      IF(.NOT.OCCPRT) WRITE(6,FCC) PC,KTCD,KARD                         OUTS0660
      OCCPRT = .TRUE.                                                   OUTS0670
      WRITE(6,324)                                                      OUTS0680
  324 FORMAT(47H0****** ILLEGAL VALUE IN DATA -- FUNCTION CODE)         OUTS0690
      CALL FLUSH                                                        OUTS0700
      WHY(16) = .TRUE.                                                  OUTS0710
      GO TO 100                                                         OUTS0720
  323 IF(IT.EQ.1) GO TO 322                                             OUTS0730
      READ(5,321) (C(I,IS),I=1,7)                                       OUTS0740
  321 FORMAT(7F10.0)                                                    OUTS0750
      IF (PRTDAT) WRITE(6,325) IS,TXEXT,(C(I,IS),I=1,7)                 OUTS0760
  325 FORMAT(1H0,18X,18HTHOR COEF, SET NO.,I3,2H (,A6,A2,9H FORM)       OUTS0770
     * 7F10.4)                                                          OUTS0780
      C(3,IS) = -0.666667*C(3,IS)                                       OUTS0790
      C(4,IS) = -C(3,IS)-C(4,IS)                                        OUTS0800
```

```
      C(5,IS) = -C(5,IS)                                                OUTS0810
      IF (LIST) WRITE(6,325) IS,TXINT,(C(I,IS),I=1,7)                   OUTS0820
      GO TO 300                                                         OUTS0830
  322 READ(5,321) (XTB(I,IS),I=1,IDUM)                                  OUTS0840
      NTB(IS) = IDUM                                                    OUTS0850
      KTB(IS) = JDUM                                                    OUTS0860
      READ(5,321) (YTB(I,IS),I=1,IDUM)                                  OUTS0870
      II = IS-1                                                         OUTS0880
      IX = JDUM/10+1                                                    OUTS0890
      IY = MOD(JDUM,10)+1                                               OUTS0900
      IF(.NOT.PRTDAT) GO TO 328                                         OUTS0910
      WRITE (6,327) II,IDUM,TXXLL(IX),TXYLL(IY)                         OUTS0920
  327 FORMAT(1H0,18X,9HTABLE NO.,I3,17,7H VALUES,5X,A5,1X,A5)           OUTS0930
      WRITE(6,326) TXX,TXEXT,(XTB(I,IS),I=1,IDUM)                       OUTS0940
  326 FORMAT(19X,A1,7H=VALUES (,A6,A7,8H FORM)  ,7E13.5/(45X,7E13.5))   OUTS0950
      WRITE(6,326) TXY,TXEXT,(YTB(I,IS),I=1,IDUM)                       OUTS0960
  328 IF (IX.EQ.1) GO TO 346                                            OUTS0970
      DO 345 I=1,IDUM                                                   OUTS0980
  345 XTB(I,IS) = ALOG10(XTB(I,IS))                                     OUTS0990
  346 IF (IY.EQ.1) GO TO 348                                            OUTS1000
      DO 347 I=1,IDUM                                                   OUTS1010
  347 YTB(I,IS) = ALOG10(YTB(I,IS))                                     OUTS1020
  348 CONTINUE                                                          OUTS1030
      IF(.NOT.LIST) GO TO 300                                           OUTS1040
      IF (.NOT.PRTDAT) WRITE (6,327) II,IDUM,TXXLL(IX),TXYLL(IY)        OUTS1050
      WRITE(6,326) TXX,TXINT,(XTB(I,IS),I=1,IDUM)                       OUTS1060
      WRITE(6,326) TXY,TXINT,(YTB(I,IS),I=1,IDUM)                       OUTS1070
      GO TO 300                                                         OUTS1080
C                                                                       OUTS1090
C  READ OUTPUT FCN TYPES (LOG OR LINEAR) AND BOUNDS (ALWAYS LINEAR)     OUTS1100
C                                                                       OUTS1110
  340 KTP(IT) = JDUM                                                    OUTS1120
      IX = JDUM/10                                                      OUTS1130
      IY = MOD(JDUM,10)+1                                               OUTS1140
      IF (PRTDAT) WRITE (6,342) IT,TYEXT,TXFLL(IY),IX,ADUM,BDUM         OUTS1150
  342 FORMAT(1H0,18X,8HFCN. NO.,I3,2H (,A6,A2,26H FORM)     OUTPUT MODE OUTS1160
     *IS ,A6,5X,17HSMOOTHING ORDER =,I2/19X,13HPLOT BOUNDS =,E12.4,      OUTS1170
     * 3H TO,E12.4)                                                     OUTS1180
      IF (IY.EQ.1 .OR. ADUM.GE.BDUM) GO TO 341                          OUTS1190
      ADUM = ALOG10(ADUM)                                               OUTS1200
```

Reproduced from best available copy.

```
      BDUM = ALOG10(BDUM)                                               OUTS1210
  341 DZMIN(IT) = ADUM                                                  OUTS1220
      DZMAX(IT) = BDUM                                                  OUTS1230
      IF (LIST) WRITE (6,342) IT,ITYPE,TXFLL(IY),IY,ADUM,BDUM           OUTS1240
      GO TO 300                                                         OUTS1250
C                                                                       OUTS1260
C  READ OUTPUT FCN TITLES AND PARAMETERS                                OUTS1270
C                                                                       OUTS1280
  360 IF(L.LE.MFCNS) GO TO 361                                          OUTS1290
      WHY(18) = .TRUE.                                                  OUTS1300
  363 IF(.NOT.OCCPRT) WRITE(6,FCC) PC,KTCD,KARD                         OUTS1310
      OCCPRT = .TRUE.                                                   OUTS1320
      WRITE(6,364)                                                      OUTS1330
  364 FORMAT(42H0******  TOO MANY SETS OF OUTPUT FUNCTIONS)             OUTS1340
      CALL FLUSH                                                        OUTS1350
      GO TO 100                                                         OUTS1360
  361 ITF(L) = IT                                                       OUTS1370
      ISF(L) = IS                                                       OUTS1380
      ITYF(L) = ITY                                                     OUTS1390
      IF (IVP.EQ.PS) IVP=PD                                             OUTS1400
      IVPF(L) = IVP                                                     OUTS1410
      NF(L) = IDUM                                                      OUTS1420
      NSETS = L                                                         OUTS1430
      READ(5,362) (ITTLE(I,L),I=1,12)                                   OUTS1440
  362 FORMAT(12A6)                                                      OUTS1450
      IF (PRTDAT.OR.LIST) WRITE(6,365) L,ITY,IVP,IDUM,                  OUTS1460
     * (ITTLE(I,L),I=1,12)                                              OUTS1470
  365 FORMAT(1H0,18X16HOUTPUT DECL. NO.,I3,10H, FCN. NO.,I3,1H=,A1,     OUTS1480
     * 1H,,I6,12H PARAMETERS,/19X,8HTITLE = ,12A6)                       OUTS1490
      DO 367 I=1,MF                                                     OUTS1500
  367 F(I,L) = 0.                                                       OUTS1510
      IF(IDUM.LE.0) GO TO 300                                           OUTS1520
      READ(5,321) (F(I,L),I=1,IDUM)                                     OUTS1530
      IF (PRTDAT.OR.LIST) WRITE(6,366) (F(I,L),I=1,IDUM)                OUTS1540
  366 FORMAT(19X,12HPARAMETERS =,7E13.5/(31X,7E13.5))                   OUTS1550
      GO TO 300                                                         OUTS1560
C                                                                       OUTS1570
C  READ NEW SET OF PLOT RANGE PARAMETERS                                OUTS1580
C                                                                       OUTS1590
  120 READ(5,121) NX,NY,XMIN,YMIN,DX,DY                                 OUTS1600
```

```
121 FORMAT(5X,2I5,5X,5F10.0)                                          OUTS1610
    IF (PRTDAT.OR.LIST) WRITE(6,122) NX,NY,XMIN,YMIN,DX,DY            OUTS1620
122 FORMAT(1H0,19X4HNX =,I3,5X,4HNY =,I3,5X,6HXMIN =,E12.4,5X,6HYMIN =OUTS1630
   *,E12.4,5X,4HDX =,E12.4,5X,4HDY =,E12.4)                           OUTS1640
    GO TO 100                                                         OUTS1650
C                                                                     OUTS1660
C  READ NEW SET OF SOLID PLOT CHARACTERS, AND SET NO. CONTOUR INTERVALS.OUTS1670
C                                                                     OUTS1680
140 READ(5,141) NZ,((CH(I),I=1,NZ),CH(NZ+1),CH(NZ+2),CH(NZ+3),CH(NZ+4) OUTS1690
   *,CH(NZ+5)                                                         OUTS1700
141 FORMAT(5X,I5,5X,15A1)                                             OUTS1710
    IDUM = NZ+4                                                       OUTS1720
    IF(LIST.OR.PRTDAT) WRITE(6,142) NZ,((CH(I),J=1,3),I=1,4),         OUTS1730
   * (CH(NZ+5),J=1,3),((CH(I),J=1,3),I=5,IDUM)                        OUTS1740
142 FORMAT(1H0,18X,I2,32H CONTOUR INTERVALS, BORDER CHARS, ARE ,3A1,  OUTS1750
   * 5H AND ,3A1,11H, UNDEF. = ,3A1,8H, LOW = ,3A1,9H, HIGH = ,3A1/   OUTS1760
   * 19X,10HIN-RANGE =,20(1X,3A1,1H,)/(29X,20(1X,3A1,1H,)))          OUTS1770
100 RETURN                                                            OUTS1780
    END                                                               OUTS1790
```

Reproduced from best available copy.

```
      SUBROUTINE COMPUT                                                 COMP0010
C                                                                       COMP0020
C  OUTPUT FUNCTION LOOP                                                 COMP0030
C  OUT(K,IPH,LL) IS A FUNCTION OF LL, PACK(K,IMO), PACK(K,IMO+1),       COMP0040
C  PACK(K,IMO+2), AMORD(IM), ETC., FOR ALL IDA AND IM.                  COMP0050
C                                                                       COMP0060
C  CONTROL CARD TABLES AND GENERAL PROGRAM CONTROL VARIABLES            COMP0070
      COMMON /CNTRL/ QCCRD,QCCPRT,KTCD,FCC(8),KARD(14),PC,PS,PD,PP,     COMP0080
     * TXTIME(2),TXDATE(2),WHY(20),KKARD(800)                           COMP0090
      LOGICAL QCCRD,QCCPRT,WHY                                          COMP0100
      INTEGER TXTIME,TXDATE,FCC,PC,PS,PD,PP                             COMP0110
      LOGICAL PRTDAT,LIST,PLOT,QWKPLT                                   COMP0120
      EQUIVALENCE (KKARD(489),PRTDAT),(KKARD(377),LIST),                COMP0130
     * (KKARD(369),PLOT),(KKARD(353),QWKPLT)                            COMP0140
C                                                                       COMP0150
C  FUNCTION TABLES, PARAMETERS AND VALUE STORAGE                        COMP0160
      COMMON /PLOTCM/ MRNG,DRNG,NPH,DPH,DPH2,IVO,IVA,IVF,IPOSOP(2),     COMP0170
     * NSETS,IDORD,                                                     COMP0180
     * TTLORD(12),TTLFRG(12),NPLOTS,NX,NY,XMIN,YMIN,DX,DY,NZ,CH(55),    COMP0190
     * RECT(81,41),ITF(20),ISF(20),ITYF(20),IVPF(20),NF(20),F(14,20),   COMP0200
     * ITTLF(12,20),KTP(20),DZMIN(20),DZMAX(20),XTB(14,10),YTB(14,10),  COMP0210
     * C(7,10),NTB(10),KTB(10),OUT(20,36,8),ITTLP(12,8),KTPP(8),DLP(8), COMP0220
     * DHP(8),ISP(8),ITP(8),PZMIN(8),PZMAX(8),ITYP(8),ITYS(8),IVPP(8)   COMP0230
C                                                                       COMP0240
      DIMENSION PACK(20,180),KPK1(60),KPK2(60)                          COMP0250
      DIMENSION SCA(2),AMORD(30)                                        COMP0260
      LOGICAL QTWO                                                      COMP0270
C                                                                       COMP0280
      DATA PI,ELN10,RADEG/3.14159265,2.30258509,57.2957795/             COMP0290
      DATA MPLOTS/8/,IFRTAP/4/                                          COMP0300
C     CONKE    KINETIC ENERGY CONVERSION FACTOR, =2*7000*32.17          COMP0310
C     CONM     GRAINS/LB, =7000                                         COMP0320
C     CONMV    MOMENTUM CONVERSION FACTOR, =7000*32.17                  COMP0330
      DATA CONKE,CONM/450380.,7000./,CONMV/225190./                     COMP0340
C                                                                       COMP0350
C  SELECT FUNCTIONS TO BE EVALUATED, AND PLOT                           COMP0360
  110 READ(5,111) (ITYS(LP),LP=1,MPLOTS)                                COMP0370
  111 FORMAT(14(I2,3X))                                                 COMP0380
      WHY(5) = .FALSE.                                                  COMP0390
      DO 112 NS=1,MPLOTS                                                COMP0400
```

```
      IF(ITYS(NS).LE.0) GO TO 113                                       COMP0410
  112 CONTINUE                                                          COMP0420
      NS = NPLOTS+1                                                     COMP0430
  113 NS = NS-1                                                         COMP0440
      IF (LIST.OR.PRTD:T) WRITE(6,116) NS,(ITYS(I),I=1,NS)              COMP0450
  116 FORMAT(1H0,13X12,16H PLOT REQUESTS,,/(29X,14I5))                  COMP0460
      IF (NS.GT.0) GO TO 119                                            COMP0470
      IF(.NOT.OCCPRT) WRITE(6,FCC) PC,KTCD,KARD                         COMP0480
      OCCPRT = .TRUE.                                                   COMP0490
      WRITE(6,115)                                                      COMP0500
  115 FORMAT(30H0******  NO FUNCTIONS SELECTED)                         COMP0510
      GO TO 117                                                         COMP0520
  119 IF(PLOT.OR.QWKPLT) GO TO 114                                      COMP0530
      IF(.NOT.OCCPRT) WRITE(6,FCC) PC,KTCD,KARD                         COMP0540
      OCCPRT = .TRUE.                                                   COMP0550
      WRITE(6,118)                                                      COMP0560
  118 FORMAT(47H0******  NEITHER 'PLOT' NOR 'KWIKPLOT' SELECTED)        COMP0570
  117 WHY(5) = .TRUE.                                                   COMP0580
      GO TO 100                                                         COMP0590
C  RETURN IF ERROR FLAGS ARE SET                                        COMP0600
  114 IF(WHY(1).OR.WHY(5).OR.WHY(6)) GO TO 100                          COMP0610
C  READ HEADER INFO FOR CURRENT FRAGMENT FIELD                          COMP0620
      READ (IFRTAP) JHDR,IPOSOP                                         COMP0630
      READ (IFRTAP) TTLORD,TTLFRG,NM,XK,ARNG,DRNG,NPH                   COMP0640
      READ (IFRTAP) (ANORD(I),I=1,NM)                                   COMP0650
      I = INTI(IPOSOP,1,2,0,IDORD)                                      COMP0660
      IVO = PS                                                          COMP0670
      IVA = PS                                                          COMP0680
      IVF = PS                                                          COMP0690
      CALL MOVE(IVO,1,1,IPOSOP,3)                                       COMP0700
      CALL MOVE(IVA,1,1,IPOSOP,4)                                       COMP0710
      CALL MOVE(IVF,1,1,IPOSOP,5)                                       COMP0720
      IMX = 6*NM                                                        COMP0730
      IMY = 2*NM                                                        COMP0740
      DPH = PI/FLOAT(NPH)                                               COMP0750
      DPH2 = DPH/2.                                                     COMP0760
      PHI = DPH2                                                        COMP0770
C  PROCESS FRAGMENT FIELD AN AZIMUTH RAY AT A TIME                      COMP0780
      DO 510 IPH=1,NPH                                                  COMP0790
      READ (IFRTAP) (KPK1(J),KPK2(J),J=1,IMY),                          COMP0800
```

Reproduced from best available copy.

```
     * ((PACK(I,J),I=1,NPLG),J=1,IMX)                                    COMP0810
      DO 515 L=1,MPLOTS                                                  COMP0820
      DO 515 K=1,NRUN                                                    COMP0830
  515 OUT(K,IPH,L) = 1.E-30                                              COMP0840
      LL = 0                                                             COMP0850
      DO 521 LP=1,NS                                                     COMP0860
      IK = 1                                                             COMP0870
      ITY = ITYS(LP)                                                     COMP0880
      IF(MOD(ITY,30).GE.10 .OR. ITY. EQ.10) GO TO 560                    COMP0890
      ITY = ITY-20                                                       COMP0900
      IK = 2                                                             COMP0910
  560 DO 550 L=1,NSETS                                                   COMP0920
      IF(ITY.EQ.ITYF(L)) GO TO 551                                       COMP0930
  550 CONTINUE                                                           COMP0940
      IF(.NOT.OCCPRT) WRITE(6,FOO) FO,KTCO,KARD                          COMP0950
      OCCPRT = .TRUE.                                                    COMP0960
      IF (IPH.GT.1) GO TO 521                                            COMP0970
      WRITE(6,552) ITY(LP)                                               COMP0980
  552 FORMAT(18H0******* PLOT CODE,I7,12H NOT DEFINED)                   COMP0990
      GO TO 521                                                          COMP1000
  551 IT = ITF(L)                                                        COMP1010
      IS = ISF(L)                                                        COMP1020
      IVP = IVPF(L)                                                      COMP1030
      STWO = IK.EQ.2 .OR. MOD(ITYS(LP)/10,3).EQ.2                        COMP1040
      DO 553 JK=1,IK                                                     COMP1050
      LL = LL+1                                                          COMP1060
      IF(LL.LE.MPLOTS) GO TO 512                                         COMP1070
      WHY(5) = .TRUE.                                                    COMP1080
      GO TO 363                                                          COMP1090
  512 IL = LL-1+IK                                                       COMP1100
      IF(IPH.GT.1) GO TO 554                                             COMP1110
      KTPP(LL) = KTP(IT)                                                 COMP1120
      OLP(LL) = OZMIN(IT)                                                COMP1130
      OHP(LL) = OZMAX(IT)                                                COMP1140
      IF (OZMIN(IT).LT.OZMAX(IT)) GO TO 563                              COMP1150
      OLP(LL) = -1.E+30                                                  COMP1160
      IF (MOD(KTP(IT),10).GT.0) OLP(LL)=1.E-30                           COMP1170
      OHP(LL) = 1.E+30                                                   COMP1180
  563 PZMIN(LL) = OZMIN(IT)                                              COMP1190
      PZMAX(LL) = OZMAX(IT)                                              COMP1200
```

```
      ITP(LL) = IT                                                      COMP1210
      ISP(LL) = IS                                                      COMP1220
      ITYP(LL) = ITY                                                    COMP1230
      IVPP(LL) = IVP                                                    COMP1240
      DO 556 I=1,12                                                     COMP1250
  556 ITTLP(I,LL) = ITTLF(I,L)                                          COMP1260
  554 IF(JK.EQ.2) GO TO 556                                             COMP1270
      IMO = 1                                                           COMP1280
      JM = 0                                                            COMP1290
      DO 511 IM=1,NM                                                    COMP1300
      W = AMORD(IM)                                                     COMP1310
C  LOOK UP THRESHOLD VELOCITY IF USING A DAMAGE THRESHOLD TABLE         COMP1320
      IF(IT.NE.1 .AND. IT.NE.2) GO TO 557                               COMP1330
      DUM = W                                                           COMP1340
      IF (KTB(IS).GE.10) DUM=LOG10(DUM)                                 COMP1350
      IDUM = NTB(IS)                                                    COMP1360
      DO 558 I=2,IDUM                                                   COMP1370
      IF(DUM.LE.XTB(I,IS)) GO TO 559                                    COMP1380
  558 CONTINUE                                                          COMP1390
      I = IDUM                                                          COMP1400
  559 VDUM = YTB(I-1,IS)+(YTB(I,IS)-YTB(I-1,IS))*(DUM-XTB(I-1,IS))/     COMP1410
     * (XTB(I,IS)-XTB(I-1,IS))                                          COMP1420
      IF (MOD(KTB(IS),10).GT.0) VDUM=EXP(VDUM*ELN10)                    COMP1430
  557 DO 519 IDA=1,2                                                    COMP1440
      JM = JM+1                                                         COMP1450
      K1 = KPK1(JM)                                                     COMP1460
      IF(K1.LE.0) GO TO 519                                             COMP1470
      K2 = KPK2(JM)                                                     COMP1480
      DO 520 K=K1,K2                                                    COMP1490
      V = PACK(K,IMO)                                                   COMP1500
      A = PACK(K,IMO+1)                                                 COMP1510
      SCA(1) = ABS(SIN(A))                                              COMP1520
      SCA(2) = ABS(COS(A))                                              COMP1530
      D = PACK(K,IMO+2)                                                 COMP1540
C  ELIMINATE CONSIDERATION OF FRAGMENTS FALLING BELOW A PRIORI T.HOLDS  COMP1550
      IF (D.LE.0.) GO TO 520                                            COMP1560
      IF (W.LT.F(1,L) .OR. V.LT.F(2,L)) GO TO 520                       COMP1570
      EMV = W*V/CONMV                                                   COMP1580
      EKE = W*V*V/CONKE                                                 COMP1590
      IF (EMV.LT.F(3,L) .OR. EKE.LT.F(4,L)) GO TO 520                   COMP1600
```

Reproduced from best available copy.

```
      GO TO (210,210,290,220,220,200,200,200,200,200,260,265,270,    COMP1610
     * 275,200,200,200,290,295),IT                                   COMP1620
  200 IF(IPH.GT.1) GO TO 521                                         COMP1630
      IF(.NOT.OCCPRT) WRITE(6,FCC) PC,KTCD,KARD                      COMP1640
      OCCPRT = .TRUE.                                                COMP1650
      WRITE(6,201) ITYP(LL)                                          COMP1660
  201 FORMAT(22H0******  FUNCTION TYPE,I3,14H HAS NO DRIVER)         COMP1670
      GO TO 521                                                      COMP1680
C                                                                    COMP1690
C  OUTPUT FUNCTIONS                                                  COMP1700
C                                                                    COMP1710
C  FUNCTION 01 -- NUMBER OF FRAGMENTS PER SQ FT                      COMP1720
  260 DUM = D*SCA(1)                                                 COMP1730
      GO TO 209                                                      COMP1740
C  FUNCTION 02 -- FRAGMENT WEIGHT (LB) PER SQ FT                     COMP1750
  265 DUM = D*SCA(1)*W/COMM                                          COMP1760
      GO TO 209                                                      COMP1770
C  FUNCTION 03 -- FRAGMENT MOMENTUM (LB-SEC) PER SQ FT               COMP1780
  270 DUM = D*SCA(1)*FMV                                             COMP1790
      GO TO 209                                                      COMP1800
C  FUNCTION 04 -- FRAGMENT K.E. (FT-LB) PER SQ FT                    COMP1810
  275 DUM = D*SCA(1)*FKE                                             COMP1820
      GO TO 209                                                      COMP1830
C  FUNCTION 08 AND 09 -- MUST BE AT END OF EXEC LIST.                COMP1840
C  LIST MUST IMPLY FEWER THAN MPLOTS PLOTS.                          COMP1850
C  PRODUCE PLOT OF AVG VELOCITY AND OF AVG STRIKING ANGLE            COMP1860
  290 OUT(K,IPH,LL) = OUT(K,IPH,LL)+V*D*W                            COMP1870
      GO TO 520                                                      COMP1880
  295 OUT(K,IPH,LL) = OUT(K,IPH,LL)+A*RADEG*D*W                      COMP1890
      OUT(K,IPH,LL+1) = OUT(K,IPH,LL+1)+D*W                          COMP1900
      GO TO 520                                                      COMP1910
C  FUNCTION 1X -- TOTAL NO. LETHAL FRAGMENTS (SEE TABLE NO. X)       COMP1920
C  FUNCTION 2X -- PROB. OF KILL              (SEE TABLE NO. X)       COMP1930
C  FUNCTION 3X -- (DO BOTH 1X AND 2X)                                COMP1940
C  (FIRST, FIND VELOCITY RELATIVE TO (MOVING) TARGET)                COMP1950
  210 IF (V.LE.VDUM) GO TO 520                                       COMP1960
      DUM = D                                                        COMP1970
      IF(.NOT.QTWO) GO TO 209                                        COMP1980
      ADUM = EXP(-D*(F(8,L)*SCA(1)+F(9,L)*SCA(2)))                   COMP1990
      IF(IT.EQ.2) GO TO 207                                          COMP2000
```

```
      GO TO 208                                                         COMP2010
C  FUNCTION 4X -- TOTAL NO. LETHAL FRAGMENTS (SEE TWOR COEF. SET IC, X) COMP2020
C  FUNCTION 5X -- PROB. OF KILL                (SEE TWOR COEF. SET IC, X) COMP2030
C  FUNCTION 6X -- (LO PLUS 4X AND 5X)                                   COMP2040
  220 BDUM = 1.                                                         COMP2050
      IF (F(11,L).GT.0.) BDUM=F(11,L)**C(7,IS)                          COMP2060
      CDUM = 10.**C(1,IS)*F(10,L)**C(2,IS)*XK**C(3,IS)*V**C(4,IS)       COMP2070
     * *V**C(6,IS)*BDUM                                                 COMP2080
      DO 221 I=1,2                                                      COMP2090
      IF (V.GT.SCA(I)**C(5,IS)*CDUM) GO TO 222                          COMP2100
  221 CONTINUE                                                          COMP2110
      GO TO 520                                                         COMP2120
  222 DUM = 0                                                           COMP2130
      IF(.NOT.STRC) GO TO 208                                           COMP2140
      ADUM = EXP(-D*(F(9,L)**SCA(1)-F(9,L)*SCA(2)))                     COMP2150
      IF(IT.EQ.5) GO TO 207                                             COMP2160
      GO TO 208                                                         COMP2170
C  CONTRIBUTE TO OUT(K,IPH,LL) AND/OR OUT(K,IPH,IL)                     COMP2180
C  DUM IS A 'PARTIAL NUMBER DENSITY'                                    COMP2190
C  ADUM IS A 'PARTIAL SURVIVABILITY'                                    COMP2200
  209 OUT(K,IPH,LL) = OUT(K,IPH,LL)+DUM                                 COMP2210
      GO TO 520                                                         COMP2220
  208 OUT(K,IPH,LL) = OUT(K,IPH,LL)+DUM                                 COMP2230
  207 OUT(K,IPH,IL) = 1.-(1.-OUT(K,IPH,IL))*ADUM                        COMP2240
  520 CONTINUE                                                          COMP2250
  519 IMO = IMO+3                                                       COMP2260
  511 CONTINUE                                                          COMP2270
  555 IT = IT+1                                                         COMP2280
      L = L+1                                                           COMP2290
  553 ITY = ITY+10                                                      COMP2300
  521 CONTINUE                                                          COMP2310
      NPLOTS = LL                                                       COMP2320
C  SCALE FUNCTION VALUES, TAKE LOG IF NEEDED.                           COMP2330
      DO 523 LL=1,NPLOTS                                                COMP2340
      IS = ISP(LL)                                                      COMP2350
      IT = ITP(LL)                                                      COMP2360
      DO 522 K=1,NRNG                                                   COMP2370
      DUM = OUT(K,IPH,LL)                                               COMP2380
      IF (ABS(DUM).LE.1.E-38) GO TO 524                                 COMP2390
      IF(IT.NE.19 .AND. IT.NE.20) GO TO 525                             COMP2400
```

Reproduced from best available copy.

```
C  NORMALIZE BY DIVIDING BY FRAG. NO. DENSITY, FOR FCNS 08 AND 09 ONLY  COMP2410
      ILL = LL+1                                                        COMP2420
      IF(IT.EQ.19) ILL = LL+2                                           COMP2430
      DUM1 = OUT(K,IPH,ILL)                                             COMP2440
      IF(DUM1.NE.0.) DUM=DUM/DUM1                                       COMP2450
  525 IF (DUM.LT.OLP(LL)) DUM=OLP(LL)                                   COMP2460
      IF (DUM.GT.OHP(LL)) DUM=OHP(LL)                                   COMP2470
      IF (MOD(KTPP(LL),10).GT.0) DUM=ALOG10(DUM)                        COMP2480
      GO TO 522                                                         COMP2490
C  IF NO VALUE HAS BEEN STORED, SET VALUE UNDEFINED (=1.E+38)           COMP2500
  524 DUM = 1.E+38                                                      COMP2510
  522 OUT(K,IPH,LL) = DUM                                               COMP2520
  523 CONTINUE                                                          COMP2530
  510 CONTINUE                                                          COMP2540
      GO TO 100                                                         COMP2550
  363 IF(.NOT.GCCPRT) WRITE(6,FCC) PC,KTCD,KARD                         COMP2560
      GCCPRT = .TRUE.                                                   COMP2570
      WRITE(6,364)                                                      COMP2580
  364 FORMAT(42H0******  TOO MANY SETS OF OUTPUT FUNCTIONS)             COMP2590
      CALL FLUSH                                                        COMP2600
  100 RETURN                                                            COMP2610
      END                                                               COMP2620
```

```
      SUBROUTINE OUTPUT                                                 OUTP0010
C                                                                       OUTP0020
C  GENERATE PLOTS USING PRINTER AND/OR WRITE FUNCTIONS TO TAPE          OUTP0030
C                                                                       OUTP0040
C  CONTROL CARD TABLES AND GENERAL PROGRAM CONTROL VARIABLES            OUTP0050
      COMMON /CNTRL/ QCCRD,QCCPRT,KTCD,FCC(8),KARD(14),PC,PS,PD,PP,     OUTP0060
     * TXTIME(2),TXDATE(2),WHY(20),KKARD(800)                            OUTP0070
      LOGICAL QCCRD,QCCPRT,WHY                                          OUTP0080
      INTEGER TXTIME,TXDATE,FCC,PC,PS,PD,PP                             OUTP0090
      LOGICAL LIST,PLOT,QWKPLT                                          OUTP0100
      EQUIVALENCE (KKARD(377),LIST),(KKARD(369),PLOT),                  OUTP0110
     * (KKARD(353),QWKPLT)                                              OUTP0120
C                                                                       OUTP0130
C  FUNCTION TABLES, PARAMETERS AND VALUE STORAGE                        OUTP0140
      COMMON /PLOTCM/ NRNG,DRNG,NPH,DPH,DPH2,IVC,IVA,IVF,IPOSOP(2),     OUTP0150
     * NSETS,IPOPD,                                                     OUTP0160
     * TTLORD(12),TTLFRG(12),NPLOTS,MX,NY,XMIN,YMIN,DX,DY,NZ,CH(55),    OUTP0170
     * RECT(81,41),ITF(20),ISF(20),ITYF(20),IVPF(20),NF(20),F(14,20),   OUTP0180
     * ITTLF(12,20),KTP(20),DZMIN(20),DZMAX(20),XTB(14,10),YTB(14,10),  OUTP0190
     * C(7,10),NTB(10),KTB(10),OUT(20,36,8),ITTLP(12,8),KTPP(8),OLP(8), OUTP0200
     * OHP(8),ISP(8),ITP(8),DZMIN(8),DZMAX(8),ITYP(8),ITYS(8),IVPP(8)   OUTP0210
      INTEGER CH                                                        OUTP0220
C                                                                       OUTP0230
      EXTERNAL PURPLE                                                   OUTP0240
      DIMENSION TT(41,4),TEMP(6,4),KTEMP(6,4),TEMQ(6,4)                 OUTP0250
      EQUIVALENCE (TT(1,1),TEMP(1,1)),(TT(1,3),KTEMP(1,1)),             OUTP0260
     * (TT(1,5),TEMQ(1,1))                                              OUTP0270
      DATA MXRNG/20/                                                    OUTP0280
      DATA ELN10/2.30258509/                                            OUTP0290
      DATA MXORDR/5/,IHDR/6H*HEAD*/                                     OUTP0300
      DATA IPLTAP/2/                                                    OUTP0310
C                                                                       OUTP0320
      DO 600 LL=1,NPLOTS                                                OUTP0330
      IF(.NOT.PLOT) GO TO 604                                           OUTP0340
      I = IFXO(IPOSOP,6,2,0,ITYP(LL))                                   OUTP0350
      CALL MOVE(IPOSOP,8,1,IVPP(LL))                                    OUTP0360
      WRITE (IPLTAP) IHDR,IPOSOP                                        OUTP0370
      WRITE (IPLTAP) TTLORD                                             OUTP0380
      WRITE (IPLTAP) TTLFRG                                             OUTP0390
      WRITE (IPLTAP) (ITTLP(I,LL),I=1,12)                               OUTP0400
```

Reproduced from best available copy.

```
  604 PC = PD                                                           OUTP0410
      IF(QWKPLT) PC=PP                                                  OUTP0420
      IF (.NOT. (LIST.OR.QWKPLT)) GO TO 605                             OUTP0430
      WRITE (6,602) PC,IDORD,IVO,TTLORD,IVA,IVF,TTLFRG,ITYP(LL),        OUTP0440
     * IVPP(LL),(ITTLP(I,LL),I=1,12)                                    OUTP0450
  602 FORMAT(A1,18X,19HDATA I.D.          =,I5,1H-,A1,5X,12A6/          OUTP0460
     * 19X,19HFRAG I.D.          =,4X,A1,1H-,A1,5X,12A6/                OUTP0470
     * 19X,19HPLOT I.D.          =,I5,1H-,A1,5X,12A6)                   OUTP0480
      IF (PLOT) WRITE(6,603) IPOSOP                                     OUTP0490
  603 FORMAT(20H PLOT TAPE KEY      =,1X,A6,A2)                         OUTP0500
      WRITE (6,607)                                                     OUTP0510
  607 FORMAT(1X)                                                        OUTP0520
C  FCN. VALUES COMPUTED ON POLAR GRID.  RE-EXPRESS ON A RECT. GRID.     OUTP0530
  605 ZMIN = 1.E38                                                      OUTP0540
      ZMAX = -1.E38                                                     OUTP0550
      DO 606 I=1,NRNG                                                   OUTP0560
      DO 606 J=1,NPH                                                    OUTP0570
      IF (OUT(I,J,LL).GE.1.E+38) GO TO 606                              OUTP0580
      ZMIN = AMIN1(ZMIN,OUT(I,J,LL))                                    OUTP0590
      ZMAX = AMAX1(ZMAX,OUT(I,J,LL))                                    OUTP0600
  606 CONTINUE                                                          OUTP0610
      Y = YMIN                                                          OUTP0620
      DO 610 IY=1,NY                                                    OUTP0630
      X = XMIN                                                          OUTP0640
      DO 611 IX=1,NX                                                    OUTP0650
      RECT(IX,IY) =                                                     OUTP0660
     * POLCAR(OUT(1,1,LL),MXRNG,NRNG,NPH,DRNG,DPH,DRNG,DPH2,X,Y)        OUTP0670
  611 X = X+DX                                                          OUTP0680
  610 Y = Y+DY                                                          OUTP0690
C  OPTIONALLY SMOOTH GRID.  (RECT(I,J)=AVG OF IT AND MANY NEIGHBORS)    OUTP0700
      IORDER = KTPP(LL)/10                                              OUTP0710
      IOR = MIN0(MXORDR,IORDER)                                         OUTP0720
      IF(IOR.LE.0) GO TO 613                                            OUTP0730
      IOR1 = IOR+1                                                      OUTP0740
      IDUM = NX+IOR1                                                    OUTP0750
      II = 0                                                            OUTP0760
      IXOR = 1-IOR1                                                     OUTP0770
      DO 640 IX=1,IDUM                                                  OUTP0780
      DO 641 IY=1,NY                                                    OUTP0790
      II = II+1                                                         OUTP0800
```

```
      IF(IX.GT.IOR1) RECT(IXOR,IY)=TT(IY,II)                            OUTP0810
      IF(IX.GT.NX) GO TO 641                                            OUTP0820
      IL = MAX0(1,IX-IOR)                                               OUTP0830
      IH = MIN0(NX,IX+IOR)                                              OUTP0840
      JL = MAX0(1,IY-IOR)                                               OUTP0850
      JH = MIN0(NY,IY+IOR)                                              OUTP0860
      DUM = 0.                                                          OUTP0870
      DO 642 I=IL,IH                                                    OUTP0880
      DO 643 J=JL,JH                                                    OUTP0890
  644 DUM = DUM+RECT(I,J)                                               OUTP0900
  643 CONTINUE                                                          OUTP0910
  642 CONTINUE                                                          OUTP0920
      TT(IY,II) = DUM/FLOAT((IH-IL+1)*(JH-JL+1))                        OUTP0930
  641 CONTINUE                                                          OUTP0940
  640 IXOR = IXOR+1                                                     OUTP0950
  613 IF (PZMIN(LL).GE.PZMAX(LL)) GO TO 612                             OUTP0960
      ZMIN = PZMIN(LL)                                                  OUTP0970
      ZMAX = PZMAX(LL)                                                  OUTP0980
  612 CONTINUE                                                          OUTP0990
C  IF REQUESTED, OUTPUT FUNCTION IN POLAR AND RECT. FORM               OUTP1000
      IF (.NOT.PLOT) GO TO 645                                          OUTP1010
      WRITE (IPLTAP) NRNG,NPH,DRNG,DPH2,DRNG,DPH,ZMIN,ZMAX              OUTP1020
      WRITE (IPLTAP) ((OUT(I,J,LL),I=1,NRNG),J=1,NPH)                   OUTP1030
      WRITE (IPLTAP) NX,NY,XMIN,YMIN,DX,DY,ZMIN,ZMAX                    OUTP1040
      WRITE (IPLTAP) ((RECT(I,J),I=1,NX),J=1,NY)                        OUTP1050
C  GEN. PRINTER CONTOUR PLOT OF FCN. (IF QWKPLT=T)                     OUTP1060
  645 IF (.NOT.QWKPLT) GO TO 600                                        OUTP1070
      CALL SOJAC(PURPLE,YMIN,YMIN+FLOAT(NY-1)*DY,XMIN,XMIN+FLOAT(NX-1)* OUTP1080
     * DX,ZMIN,ZMAX,NX,1,NZ,CH)                                         OUTP1090
      DZ = (ZMAX-ZMIN)/FLOAT(NZ)                                        OUTP1100
      Z = ZMIN                                                          OUTP1110
      NL = (NZ+5)/4                                                     OUTP1120
      NI = NL-(NL*4-(NZ+2))                                             OUTP1130
      K = 3                                                             OUTP1140
      DO 682 J=1,4                                                      OUTP1150
      DO 680 I=1,NL                                                     OUTP1160
      K = K+1                                                           OUTP1170
      TEMP(I,J) = Z                                                     OUTP1180
      IF (MOD(KTPP(LL),10).GT.0) TEMP(I,J)=EXP(TEMP(I,J)*ELN10)         OUTP1190
      KTEMP(I,J) = CH(K)                                                OUTP1200
```

```
680 Z = Z+DZ                                                          OUTP1210
682 CONTINUE                                                          OUTP1220
    NJ = NL-1                                                         OUTP1230
    DO 684 J=1,4                                                      OUTP1240
    IF(NJ.LE.0) GO TO 687                                             OUTP1250
    DO 686 I=1,NJ                                                     OUTP1260
686 TEMQ(I+1,J) = TEMP(I,J)                                           OUTP1270
687 IF(J.EQ.4) GO TO 684                                              OUTP1280
    TEMQ(1,J+1) = TEMP(NL,J)                                          OUTP1290
684 CONTINUE                                                          OUTP1300
    TEMQ(1,1) = -1.E38                                                OUTP1310
    TEMP(NI,4) = +1.E38                                               OUTP1320
    NJ = 4                                                            OUTP1330
    WRITE(6,685)                                                      OUTP1340
685 FORMAT(//57X,19HTHUS SCALES SOJAC.,//)                            OUTP1350
    DO 683 I=1,NL                                                     OUTP1360
    IF(I.GT.NI) NJ=3                                                  OUTP1370
    WRITE (6,681) (TEMQ(I,J),(KTEMP(I,J),K=1,3),TEMP(I,J),J=1,NJ)     OUTP1380
681 FORMAT(4(3X,1PE10.2,4H LF ,3A1,3H LT,E10.2))                      OUTP1390
683 CONTINUE                                                          OUTP1400
600 CONTINUE                                                          OUTP1410
C                                                                     OUTP1420
100 RETURN                                                            OUTP1430
    END                                                               OUTP1440
```

```
      FUNCTION PURPLE(X,Y)                                              PURP0010
C  FUNCTION TABLES, PARAMETERS AND VALUE STORAGE                        PURP0020
      COMMON /PLOTCM/ ARNG,DRNG,NPH,OPH,OPH2,IVO,IVA,IVF,IPOSOP(2),     PURP0030
     * NSETS,IPORD,                                                    PURP0040
     * TTLORD(12),TTLFRG(12),NPLOTS,NX,NY,XMIN,YMIN,DX,DY,NZ,CH(55),    PURP0050
     * RECT(81,41),ITF(20),ISF(20),ITYF(20),IVPF(20),NF(20),F(14,20),   PURP0060
     * ITTLF(12,20),KTF(20),OZMIN(20),OZMAX(20),XTB(14,10),YTB(14,10),  PURP0070
     * C(7,10),NTB(10),KTB(10),OUT(20,36,8),ITTLP(12,8),KTPP(8),OLP(8), PURP0080
     * OHP(8),ISP(8),ITP(8),PZMIN(8),PZMAX(8),ITYP(8),ITYS(8),IVPP(8)   PURP0090
C                                                                       PURP0100
C  COMMON BLOCK PASSED FROM 'SOJAC'                                     PURP0110
      COMMON /CSOJAC/ GX,GY,IXP,IYP,IX,IY                               PURP0120
C                                                                       PURP0130
C     SUPPLIES VALUES OF RECT TO 'SOJAC' (PRINTER CONTOUR PLOTTER)      PURP0140
C                                                                       PURP0150
      XX = (X-XMIN)/GY+1.                                               PURP0160
      IXX = XX                                                          PURP0170
      XX = 1.+AMOD(XX,1.)                                               PURP0180
      IXX = MIND(NY-1,IXX)                                              PURP0190
      IXX = NY-IXX                                                      PURP0200
      IF(RECT(IY,IXX).LT.1.E+38 .AND. RECT(IY,IXX+1).LT.1.E+38) GO TO 1 PURP0210
      PURPLE = 1.E+38                                                   PURP0220
      RETURN                                                            PURP0230
    1 PURPLE = RECT(IY,IXX)*(1.-XX)+RECT(IY,IXX+1)*XX                   PURP0240
      RETURN                                                            PURP0250
      END                                                               PURP0260
```

Reproduced from best available copy.

```
      FUNCTION POLCAR(P,MR,NR,NT,DR,DT,DRM,DTM,X,Y)                     POLC0010
      DIMENSION P(MR,1)                                                 POLC0020
C                                                                       POLC0030
C     GIVEN TABULAR FUNCTION P DEFINED ON A POLAR GRID OF NR RADII AND  POLC0040
C     NT AZIMUTHS,, RADIUS I = DR*(I-1)+DRM,  AZIMUTH J = DT*(J-1)+DTM, POLC0050
C     AND CARTESIAN COORDINATES (X,Y), RETURN VALUE OF P AT (X,Y).      POLC0060
C     A BILINEAR (IN R,T) INTERPOLATION IS USED.                        POLC0070
C     POLCAR IS SET 'UNDEFINED' (=1.E+38) IF ANY OF THE FOUR DATA POINTSPOLC0080
C     SURROUNDING (X,Y) ARE UNDEFINED, OR IF (X,Y) LIES WITHIN RADIUS   POLC0090
C     'DRM' OR BEYOND RADIUS 'DRM*NR'                                   POLC0100
C                                                                       POLC0110
      IF (X.EQ.0. .AND. Y.EQ.0.) GO TO 5                                POLC0120
      RS = SQRT(X*X+Y*Y)                                                POLC0130
      TS = ATAN2(Y,X)                                                   POLC0140
      I = (RS-DRM)/DR+1.                                                POLC0150
      J = (TS-DTM)/DT+1.                                                POLC0160
      IF(I.LT.1 .OR. I.GT.NR) GO TO 5                                   POLC0170
      J = MAX0(1,MIN0(NT-1,J))                                          POLC0180
      TH = (TS-DT*FLOAT(J-1)-DTM)/DT                                    POLC0190
      I = MAX0(1,MIN0(NR-1,I))                                          POLC0200
      IF(P(I,J).GE.1.E+38 .OR. P(I,J+1).GE.1.E+38 .OR.                  POLC0210
     * P(I+1,J).GE.1.E+38 .OR. P(I+1,J+1).GE.1.E+38) GO TO 5            POLC0220
      RH = (RS-DR*FLOAT(I-1)-DRM)/DR                                    POLC0230
      POLCAR = (P(I,J)*(1.-TH) + P(I,J+1)*TH) * (1.-RH)                 POLC0240
     *       + (P(I+1,J)*(1.-TH) + P(I+1,J+1)*TH) * RH                  POLC0250
      RETURN                                                            POLC0260
    5 POLCAR = 1.E+38                                                   POLC0270
      RETURN                                                            POLC0280
      END                                                               POLC0290
```

```
C     **********                                                        CCAR0010
C     * CCARD  *          JOHN HOUSE      IITRI       SEPT 1970          CCAR0020
C     **********                                                        CCAR0030
C                                                                       CCAR0040
      LOGICAL FUNCTION CCARD(S1,C1,L1,IACT,W)                           CCAR0050
C                                                                       CCAR0060
C     INTERPRET CONTROL CARD FIELDS IDENTIFIED BY SYNTAX.               CCAR0070
C                                                                       CCAR0080
C     S1        TEXT STRING.                                            CCAR0090
C     C1        POINTER TO START OF CURRENT FIELD                       CCAR0100
C     L1        LENGTH OF CURRENT FIELD                                 CCAR0110
C     IACT      ACTION CODE RETRIEVED BY SYNTAX FROM SYNTAX RULES       CCAR0120
C     W         GENERAL ARRAY FOR CONTROL CARDS.,                       CCAR0130
C               W(1)--W(10).. FLAGS AND COUNTERS FOR USE BY AND COMMUNI- CCAR0140
C                             CATION WITH SYNTAX                        CCAR0150
C               W(11).. NUMBER OF CONTROL CARD ENTRIES TIMES 7 PLUS 20, CCAR0160
C               W(12)--W(20).. HOLD OUTPUT VALUES FOR USE BY PROGRAM    CCAR0170
C                              CALLING SYNTAX                           CCAR0180
C               W(12).. CONTROL CARD TYPE CODE                          CCAR0190
C               W(13).. NON-SYNTAX ERROR COLUMN (=0 IF NONE)            CCAR0200
C               W(14),W(15).. POINT TO START AND END OF FIELD ORDINAL   CCAR0210
C                             LIST (NOT USED IF W(14).GT.W(15))         CCAR0220
C               W(16),W(17).. POINT TO START AND END OF POSITIONAL FIELD CCAR0230
C                             LIST (NOT USED IF W(16).GT.W(17))         CCAR0240
C                                                                       CCAR0250
C     UPON RETURN, CCARD=.TRUE. IF FIELD IS OK, OR CCARD=.FALSE. IF NOTCCAR0260
C                                                                       CCAR0270
C     CONTROL CARD TYPE 1 IS RESERVED FOR C.CDS. WHICH NEVER REQUIRE    CCAR0280
C     FURTHER USER ACTION.  (I.E., CARD MERELY SETS FLAGS AND VALUES)   CCAR0290
C     FIELD TYPE CODE 9 IS RESERVED FOR THE 'FIELD DEFAULT VALUE        CCAR0300
C     REDEFINITION' FLAG.  THE REMAINDER OF A C.CD. ON WHICH THIS FLAG  CCAR0310
C     APPEARS CONSISTS OF FIELD DEF. VAL. REDEFNS.  (NOTE,. FLAG DEFAULTCCAR0320
C     SETTINGS ARE ENTERED IN THE LOGICAL FORM (NAME=T. NAME=.FALSE.,   CCAR0330
C     ETC.)  DO NOT SIMPLY ENTER 'NAME' OR 'NO'+'NAME')                 CCAR0340
C                                                                       CCAR0350
      INTEGER S1(1)                                                     CCAR0360
      INTEGER C1,W(1),NULLV(5),INULL(3)                                 CCAR0370
      LOGICAL GRIPE,POSIT,EXIST,LNULL,DEFINE                            CCAR0380
      REAL FNULL                                                        CCAR0390
C     KV AND RV ARE EQUIVALENT NAMES FOR THE 1ST WORD OF KVA.           CCAR0400
```

Reproduced from best available copy.

```
C     KV IS ALWAYS USED FOR LOGICAL AND INTEGER VALUES.                 CCAR0410
C     RV IS ALWAYS USED FOR REAL VALUES.                                CCAR0420
      INTEGER KVA(2)                                                    CCAR0430
      EQUIVALENCE (KVA(1),KV,RV)                                        CCAR0440
      DATA IBL/1H /,INULL,LNULL,FNULL/0,0.1H ,.FALSE.,0.0/               CCAR0450
C***  VALUE OF NCHW IS MACHINE DEPENDENT -- MAX. NO CHAR. IN NAME OR VALCCAR0460
      DATA NCHW/12/                                                     CCAR0470
C***  KNL IS MACH. DEP. -- NUMBER OF WORDS ALLOWED FOR A NAME OR VALUECCAR0480
      DATA KNL/2/                                                       CCAR0490
      EQUIVALENCE (NULLV(1),INULL(1)),(NULLV(4),LNULL),(NULLV(5),FNULL),CCAR0500
     * (IEXIST,EXIST)                                                   CCAR0510
C                                                                       CCAR0520
      LMAX = MIN0(L1,NCHW)                                              CCAR0530
      GO TO (100,200,300,400,500,600,700,800,900,1000),IACT             CCAR0540
C                                                                       CCAR0550
C  CONTROL CARD NAME                                                    CCAR0560
C                                                                       CCAR0570
  100 N = W(11)                                                         CCAR0580
      W(12) = 0                                                         CCAR0590
      W(13) = 0                                                         CCAR0600
      DO 101 KK=21,N,7                                                  CCAR0610
      I = W(   KK+1)                                                    CCAR0620
  102 IF(I.LE.0) GO TO 101                                              CCAR0630
      II = I+KNL                                                        CCAR0640
      J = KOML(W(I),1,IABS(W(II)),S1,C1,LMAX)                           CCAR0650
      IF (J.EQ.0) GO TO 110                                             CCAR0660
      IF (J.NE.-1) GO TO 104                                            CCAR0670
  103 IF(W(II).LT.0) GO TO 110                                          CCAR0680
  104 I = W(II+1)                                                       CCAR0690
      GO TO 102                                                         CCAR0700
  101 CONTINUE                                                          CCAR0710
      GO TO 9                                                           CCAR0720
C  CONTROL CARD NAME FOUND                                              CCAR0730
  110 W(12)= W(KK)                                                      CCAR0740
      KPOS1 = W(KK+2)                                                   CCAR0750
      KPOS = KPOS1                                                      CCAR0760
      W(16)= KPOS1                                                      CCAR0770
      KPOS2 = W(KK+3)-1                                                 CCAR0780
      KEY1 = W(KK+3)                                                    CCAR0790
      KEY2 = W(KK+4)                                                    CCAR0800
```

```
      KOR1 = W(KK+5)                                                    CCAR0810
      KOR = KOR1                                                        CCAR0820
      W(14) = KOR1                                                      CCAR0830
      KOR2 = W(KK+6)                                                    CCAR0840
C  SET DEFAULT FIELDS                                                   CCAR0850
       IF(KPOS1.GT.KEY2) GO TO 120                                      CCAR0860
      DO 115 K=KPOS1,KEY2                                               CCAR0870
      I = W(K)                                                          CCAR0880
      W(I+2) = 0                                                        CCAR0890
      IF(W(I).GE.81000) GO TO 113                                       CCAR0900
      W(I+3) = W(I)/1000                                                CCAR0910
      II = I+KNL+4                                                      CCAR0920
      IF(MOD(W(I),10)-4) 113,116,114                                    CCAR0930
C***  LOADING OF DEFAULT VALUE MAY REQUIRE A DO LOOP                    CCAR0940
  113 W(I+4) = W(II)                                                    CCAR0950
      W(I+5) = W(II+1)                                                  CCAR0960
      GO TO 115                                                         CCAR0970
  114 W(I+4) = W(II)                                                    CCAR0980
      GO TO 115                                                         CCAR0990
  116 L = MIN0(W(I+4),W(II))                                            CCAR1000
      K1 = W(I+5)                                                       CCAR1010
      K2 = W(II+1)                                                      CCAR1020
      CALL MOVE(W(K1),1,L,W(K2),1)                                      CCAR1030
      J = W(I+4)-L                                                      CCAR1040
      IF(J.LE.0) GO TO 115                                              CCAR1050
      CALL MOVE(W(K1),L+1,1,1H ,1)                                      CCAR1060
      CALL MOVE(W(K1),L+2,J-1,W(K1),L+1)                                CCAR1070
  115 CONTINUE                                                          CCAR1080
  120 GRIPE = .FALSE.                                                   CCAR1090
      DEFINE = .FALSE.                                                  CCAR1100
      GO TO 1                                                           CCAR1110
C                                                                       CCAR1120
C  POSITIONAL VALUE OR FIELD NAME                                       CCAR1130
C                                                                       CCAR1140
  200 KS = C1                                                           CCAR1150
      LS = LMAX                                                         CCAR1160
      LS1 = L1                                                          CCAR1170
      GO TO 1                                                           CCAR1180
C                                                                       CCAR1190
C  PRIOR WORD WAS A FIELD NAME (FOLLOWED BY A VALUE)                    CCAR1200
```

Reproduced from best available copy.

```
C                                                                       CCAR1210
  300 POSIT = .FALSE.                                                   CCAR1220
C  ALL REG. POSIT. FIELDS MUST BE SUPPLIED PRIOR TO 1ST K=V TYPE FIELD  CCAR1230
      IF(KPOS.GT.KPOS2) GO TO 510                                       CCAR1240
      KW = W(KPOS)                                                      CCAR1250
      IF(W(KW).GE.82000) GO TO 5                                        CCAR1260
      GO TO 510                                                         CCAR1270
C                                                                       CCAR1280
C  FIELD VALUE.  INTERPRET AS INTEGER, ALPHA, LOGICAL, OR REAL.         CCAR1290
C                                                                       CCAR1300
  400 KS = C1                                                           CCAR1310
      LS = LMAX                                                         CCAR1320
      LS1 = L1                                                          CCAR1330
  410 K2 = K2+1                                                         CCAR1340
      IF(K2.LE.0) GO TO 1                                               CCAR1350
      KV = 0                                                            CCAR1360
      L = 0                                                             CCAR1370
      GO TO (440,440,420,430,440,450,460,470),K2                        CCAR1380
C  INTEGER                                                              CCAR1390
  420 IF (INTI(S1,KS,LS1,0,KV).NE.1) GO TO 495                          CCAR1400
      GO TO 405                                                         CCAR1410
C  ALPHAMERIC                                                           CCAR1420
C***  BLANK OUT ALL WORDS OF KVA BEFORE MOVING TEXT                     CCAR1430
  430 KV = IBL                                                          CCAR1440
      KVA(2) = IBL                                                      CCAR1450
      CALL MOVE(KV,1,LS,S1,KS)                                          CCAR1460
      L = LS                                                            CCAR1470
      GO TO 405                                                         CCAR1480
C  LOGICAL                                                              CCAR1490
  440 IF (ILGI(S1,KS,LS1,0,KV).NE.1) GO TO 405                          CCAR1500
      GO TO 405                                                         CCAR1510
C  REAL                                                                 CCAR1520
  450 IF (IRFI(S1,KS,LS1,0,0,RV).NE.1) GO TO 495                        CCAR1530
      GO TO 405                                                         CCAR1540
C  LONG-ALPHAMERIC                                                      CCAR1550
  460 IF(DEFINE) GO TO 465                                              CCAR1560
      LS1 = MIND(MAP,LS1)                                               CCAR1570
      CALL MOVE(W(JAP),1,LS1,S1,KS)                                     CCAR1580
      W(KW+2) = 1                                                       CCAR1590
      W(KW+3) = LS1                                                     CCAR1600
```

```
      GO TO 1                                                           CCAR1610
  465 IF(W(KW),GE,81000) GO TO 5                                        CCAR1620
      LS1 = MIN(MAP,LS1)                                                CCAR1630
      CALL MOVE(W(KW),1,LS1,S1,LS)                                      CCAR1640
      W(KW) = LS1*1000+MOD(W(KW),1000)                                  CCAR1650
      J = MAP-LS1                                                       CCAR1660
      IF(J,LE,0) GO TO 1                                                CCAR1670
      CALL MOVE(W(JAP),1,J,1   ,1)                                      CCAR1680
      CALL MOVE(W(JAP),2,J-1,W(JAP),1)                                  CCAR1690
      GO TO 1                                                           CCAR1700
C  SYMBOLIC -- SEARCH DICTIONARY FOR MATCH WITH SYMBOL                  CCAR1710
  470 I = W(KW+5)                                                       CCAR1720
      L = 0                                                             CCAR1730
      KV = 0                                                            CCAR1740
  471 IF(I,LE,0) GO TO 405                                              CCAR1750
      II = I+KNL                                                        CCAR1760
      KV = KV+1                                                         CCAR1770
      J = KOML(W(I),1,IABS(W(II)),S1,KC,LS)                             CCAR1780
      IF (J,EQ,0) GO TO 405                                             CCAR1790
      IF (J,NE,-1) GO TO 472                                            CCAR1800
  473 IF(W(II),LT,0) GO TO 405                                          CCAR1810
  472 I = W(II+1)                                                       CCAR1820
      GO TO 471                                                         CCAR1830
C  CONCLUSION                                                           CCAR1840
  405 IF(DEFINE) GO TO 407                                              CCAR1850
      W(KW+2) = 1                                                       CCAR1860
      W(KW+3) = L                                                       CCAR1870
C  A DO LOOP MAY BE REQUIRED TO MOVE ALL OF KVA                         CCAR1880
      W(KW+4) = KV                                                      CCAR1890
      IF(K2,GE,7) GO TO 1                                               CCAR1900
      W(KW+5) = KVA(2)                                                  CCAR1910
      GO TO 1                                                           CCAR1920
C  A DO LOOP MAY BE REQUIRED TO MOVE ALL OF KVA                         CCAR1930
  407 IF(W(KW),GE,81000) GO TO 5                                        CCAR1940
      II = KW+KNL+4                                                     CCAR1950
      W(II) = KV                                                        CCAR1960
      IF(K2,LT,7) W(II+1)=KVA(2)                                        CCAR1970
      W(KW) = L*1000+MOD(W(KW),1000)                                    CCAR1980
      GO TO 1                                                           CCAR1990
C  VALUE ERROR                                                          CCAR2000
```

Reproduced from best available copy.

```
  495 W(KW+2) =-1                                                       CCAR2010
      IF(DEFINE) GO TO 5                                                CCAR2020
      W(KW+3) = 0                                                       CCAR2030
      W(KW+4) = 0                                                       CCAR2040
      IF(K2.GE.7) GO TO 5                                               CCAR2050
      W(KW+5) = 0                                                       CCAR2060
      GO TO 5                                                           CCAR2070
C                                                                       CCAR2080
C  PRIOR WORD WAS EITHER A FIELD NAME (W/O A VALUE) OR A POSIT. VALUE   CCAR2090
C                                                                       CCAR2100
  500 POSIT = .TRUE.                                                    CCAR2110
      IF(DEFINE) GO TO 5                                                CCAR2120
C  IF ANY REQ. POSIT. FIELDS REMAIN UNSATISFIED, DO NOT SRC FOR KEYWORD CCAR2130
      IF(KPOS.GT.KPOS2) GO TO 515                                       CCAR2140
      KW = W(KPOS)                                                      CCAR2150
      IF(W(KW).GE.R2000) GO TO 550                                      CCAR2160
C                                                                       CCAR2170
C  SEARCH FOR POSSIBLE FIELD NAME                                       CCAR2180
C                                                                       CCAR2190
  510 IF(.NOT.DEFINE) GO TO 515                                         CCAR2200
      IF (INTI(S1,KS,LS,0,K).NE.1) GO TO 515                            CCAR2210
      IF(K.LE.0) GO TO 5                                                CCAR2220
      K = KPOS1-1+K                                                     CCAR2230
      IF(K.GT.KPOS2) GO TO 5                                            CCAR2240
      KW = W(K)                                                         CCAR2250
      K2 = MOD(W(KW),10)                                                CCAR2260
      IF(KOR.GT.KOR2) GO TO 1                                           CCAR2270
      W(KOR) = KW                                                       CCAR2280
      KOR = KOR+1                                                       CCAR2290
      GO TO 1                                                           CCAR2300
  515 DO 511 KEY=KEY1,KEY2                                              CCAR2310
      KW = W(KEY)                                                       CCAR2320
      IF(W(KW+2).NE.0) GO TO 511                                        CCAR2330
      K2 = MOD(W(KW),10)                                                CCAR2340
      K = W(KW+1)                                                       CCAR2350
C  IF K=V TYPE FIELD ENCOUNTERED, DO NOT SEARCH K OR V TYPE FIELD ENTR. CCAR2360
C  IF K OR V TYPE FIELD ENCOUNTERED, DO NOT SRC K=V TYPE ENTRIES        CCAR2370
      EXIST = K.LE.0 .OR. K2.LE.1 .OR. K2.GE.9                          CCAR2380
      IF(POSIT .AND. .NOT.EXIST .OR. .NOT.POSIT .AND. EXIST) GO TO 511  CCAR2390
  512 IF(K.LE.0) GO TO    511                                           CCAR2400
```

```
      II = K+K*L                                                        CCAR2410
      K1 = IABS(W(II))                                                  CCAR2420
      I = KOMLP (K),1,K1,S1, S,LS)                                      CCAR2430
      IF (IABS(I).GT.1) GO TO 513                                       CCAR2440
      IF (I) 514,520,519                                                CCAR2450
  513 IF(K2.NE.1) GO TO 519                                             CCAR2460
      IF(LS.LE.2) GO TO 519                                             CCAR2470
      IF(KOR (2*KOMC,1,2,S1,KS),NE,0) GO TO 519                         CCAR2480
      J = KOMLP(W(K),1,K1,S1,KS+2,LS-2)                                 CCAR2490
      IF (I.EQ.0) GO TO 521                                             CCAR2500
      IF (I.NE.-1) GO TO 519                                            CCAR2510
  514 IF(W(II).LT.0) GO TO 521                                          CCAR2520
      GO TO 519                                                         CCAR2530
  518 IF(W(II).LT.0) GO TO 521                                          CCAR2540
  519 K = W(II+1)                                                       CCAR2550
      GO TO 512                                                         CCAR2560
  511 CONTINUE                                                          CCAR2570
C  NO MATCH FOUND -- OK IF THIS IS POSITIONAL                           CCAR2580
      IF(KPOS.GT.KPOS2 ,OR, .NOT.POSIT) GO TO 5                         CCAR2590
  550 KW = W(KPOS)                                                      CCAR2600
      KPOS = KPOS+1                                                     CCAR2610
      K2 = MOD(W(KW),10)                                                CCAR2620
      IF(KOR.GT.KOR2) GO TO 551                                         CCAR2630
      W(KOR) =   KW                                                     CCAR2640
      KOR = KOR+1                                                       CCAR2650
  551 IF(POSIT) GO TO 560                                               CCAR2660
      K2 = MOD(W(KW),10)                                                CCAR2670
      GO TO 523                                                         CCAR2680
C  PREPARE FIELD (IF LONG-ALPHA)                                        CCAR2690
  560 IF(MOD(W(KW),10).NE.6) GO TO 410                                  CCAR2700
      ASSIGN 410 TO KGO                                                 CCAR2710
      GO TO 546                                                         CCAR2720
C  FIELD NAME (OR NO-NAME) FOUND                                        CCAR2730
  520 EXIST = .TRUE.                                                    CCAR2740
      GO TO 522                                                         CCAR2750
  521 EXIST = .FALSE.                                                   CCAR2760
  522 IF(DEFINE) GO TO 523                                              CCAR2770
      IF(K2.EQ.9) GO TO 540                                             CCAR2780
  523 IF(KOR.GT.KOR2) GO TO 530                                         CCAR2790
      W(KOR) = KW                                                       CCAR2800
```

Reproduced from best available copy.

```
      KOR = KOR+1                                                       CCAR2810
C  SET EXISTANCE OR NULL VALUES                                         CCAR2820
  530 IF(K2.GT.1) GO TO 535                                             CCAR2830
      W(KW+2) = 1                                                       CCAR2840
      W(KW+4) = IEXIST                                                  CCAR2850
      GO TO 1                                                           CCAR2860
  535 IF(K2-6) 536,545,555                                              CCAR2870
  536 W(KW+4) = NULLV(K2)                                               CCAR2880
      IF(K2.EQ.3) W(KW+5)=NULLV(K2)                                     CCAR2890
      W(KW+3) = 0                                                       CCAR2900
      JAP = KW+4                                                        CCAR2910
      LAP = NCHW                                                        CCAR2920
      MAP = NCHW                                                        CCAR2930
      KAP = 1                                                           CCAR2940
      GO TO 1                                                           CCAR2950
C  FIELD IS LONG-ALPHABETIC                                             CCAR2960
  545 ASSIGN 1 TO KGO                                                   CCAR2970
  546 II = KW+4                                                         CCAR2980
      IF (DEFINE) II=II+KAL                                             CCAR2990
      JAP = W(II+1)                                                     CCAR3000
      LAP = W(II)                                                       CCAR3010
      MAP = LAP                                                         CCAR3020
      KAP = 1                                                           CCAR3030
      IF(LAP.LE.0) GO TO 549                                            CCAR3040
      CALL MOVE(W(JAP),1,1,1H ,1)                                       CCAR3050
      CALL MOVE(W(JAP),2,LAP-1,W(JAP),1)                                CCAR3060
      IF (DEFINE) GO TO 547                                             CCAR3070
      W(KW+3) = 0                                                       CCAR3080
      GO TO 549                                                         CCAR3090
  547 W(KW) = MOD(W(KW),1000)                                           CCAR3100
  549 GO TO KGO                                                         CCAR3110
C  SYMBOLIC -- COLLECT ARGUMENT FOR A SEARCH                            CCAR3120
  555 W(KW+4) = 0                                                       CCAR3130
      GO TO 1                                                           CCAR3140
C  IF A FIELD OF TYPE 9 IS ENCOUNTERED, THEN REST OF CARD SETS DEFAULTS CCAR3150
  540 DEFINE = .TRUE.                                                   CCAR3160
      W(12) = 1                                                         CCAR3170
      GO TO 1                                                           CCAR3180
C                                                                       CCAR3190
C  END OF CONTROL CARD -- CHECK FOR MISSING REQUIRED FIELDS             CCAR3200
```

```
C                                                                       CCAR3210
  600 IF(KPOS1.GT.KEY2) GO TO 6[illegible]                              CCAR3220
      DO 619 K=KPOS1,KEY2                                               CCAR3230
      I = W(K)                                                          CCAR3240
      IF(W(I).LT.82000) GO TO 619                                       CCAR3250
      IF(W(I+2).LE.0) GO TO 620                                         CCAR3260
  619 CONTINUE                                                          CCAR3270
  620 IF(GRIPE) GO TO 635                                               CCAR3280
      CCARD = .TRUE.                                                    CCAR3290
      GO TO 640                                                         CCAR3300
  630 IF (GRIPE) GO TO 635                                              CCAR3310
      IGRIPE = 01                                                       CCAR3320
  635 W(13) = IGRIPE                                                    CCAR3330
      CCARD = .FALSE.                                                   CCAR3340
  640 W(15) = KOR-1                                                     CCAR3350
      W(17) = KPOS-1                                                    CCAR3360
      RETURN                                                            CCAR3370
C                                                                       CCAR3380
C  ADD TO I-TYPE STRING (PROHIBITED FOR NON-ALPHANUMERIC FIELDS)        CCAR3390
C                                                                       CCAR3400
  700 IF(K2.NE.3 .AND. K2.NE.6) GO TO 4                                 CCAR3410
      CALL CAT(W(JAP),KAP,LAP,51,01,11)                                 CCAR3420
      IF(DEFINE) GO TO 705                                              CCAR3430
      W(KW+2) = 2                                                       CCAR3440
      W(KW+3) = MAP-LAP                                                 CCAR3450
      GO TO 1                                                           CCAR3460
  705 W(KW) = (MAP-LAP)*1000+MOD(W(KW),1000)                            CCAR3470
      GO TO 1                                                           CCAR3480
C                                                                       CCAR3490
C  START OF POSITIONAL I-TYPE ALPHA VALUE (RESERVE POSIT. FIELD ENTRY)  CCAR3500
C                                                                       CCAR3510
  800 IF(DEFINE) GO TO 4                                                CCAR3520
      POSIT = .FALSE.                                                   CCAR3530
      IF(KPOS.GT.KPOS2) GO TO 4                                         CCAR3540
      GO TO 550                                                         CCAR3550
C                                                                       CCAR3560
C  RETURNS                                                              CCAR3570
C                                                                       CCAR3580
  900 CONTINUE                                                          CCAR3590
 1000 CONTINUE                                                          CCAR3600
```

Reproduced from best available copy.

```
9 CCARD = .FALSE.                                          CCAR3610
  RETURN                                                   CCAR3620
6 IF(GRIPE) GO TO 1                                        CCAR3630
  IGRIPE = C1+L1                                           CCAR3640
  GO TO 7                                                  CCAR3650
5 IF(GRIPE) GO TO 1                                        CCAR3660
  IGRIPE = KS+LS1                                          CCAR3670
7 GRIPE = .TRUE.                                           CCAR3680
1 CCARD = .TRUE.                                           CCAR3690
  RETURN                                                   CCAR3700
  END                                                      CCAR3710
```

```
C     **********                                                        CAT 0010
C     *  CAT   *          JOHN ROWE      IITRI      SEPT 1970            CAT 0020
C     **********                                                        CAT 0030
C                                                                       CAT 0040
      SUBROUTINE CAT (S1,C1,L1,S2,C2,L2)                                CAT 0050
      INTEGER S1(1),C1,L1,S2(1),C2,L2                                   CAT 0060
C                                                                       CAT 0070
C     CONCATENATE L CHAR., STARTING WITH CHAR. C2 OF STRING S2, ONTO    CAT 0080
C     STRING S1, WHICH HAS ROOM FOR L1 MORE CHAR., STARTING AT CHAR.    CAT 0090
C     POS. C1.  L=MIN(L1,L2)                                            CAT 0100
C     UPON RETURN, C1 IS INCREASED BY L, AND L1 IS DECREASED BY L.      CAT 0110
C     BOTH STRINGS S1 AND S2 ARE PACKED (I.E., 6 CHAR. PER WORD)        CAT 0120
C                                                                       CAT 0130
      L = MIND(L1,L2)                                                   CAT 0140
      IF(L.LE.0) RETURN                                                 CAT 0150
      CALL RCHXTK(S1,C1)                                                CAT 0160
      CALL RCHXTK(S2,C2)                                                CAT 0170
      DO 1 I=1,L                                                        CAT 0180
    1 CALL SCHXTK(S1,W1,ICHXTK(S2,W2))                                  CAT 0190
      C1 = C1+L                                                         CAT 0200
      L1 = L1-L                                                         CAT 0210
      RETURN                                                            CAT 0220
      END                                                               CAT 0230
```

Reproduced from best available copy.

```
C     **********                                                        IFX00010
C     *  IFX0  *          JOHN BOUSE      IITRI       MAY 1971           IFX00020
C     **********                                                        IFX00030
C                                                                       IFX00040
      FUNCTION IFX0(S,C,L,N,I)                                          IFX00050
      INTEGER S(1),D                                                    IFX00060
      INTEGER C,X,N,M                                                   IFX00070
      LOGICAL OS                                                        IFX00080
C***  M,D, INTEGER VALUE OF CHAR, ZERO                                  IFX00090
      DATA D/060/                                                       IFX00100
C                                                                       IFX00110
C     CONVERT THE ABSOLUTE VALUE OF INTEGER 'N' TO CHAR, FORM AND PLACE IFX00120
C     IT IN THE OUTPUT FIELD.                                           IFX00130
C     IF N LE 0, THE OUTPUT FIELD IS THE 'L' CHAR. OF STRING S, STARTINGIFX00140
C             AT CHAR, 'C'.  RESULT IS RIGHT-JUSTIFIED W/BLANK FILL.    IFX00150
C     IF N GT 0, THE OUTPUT FIELD IS THE UP TO N CHAR. (OR UP TO 'L'    IFX00160
C             CHAR, IF L LT N) OF STRING S, STARTING AT CHAR, 'C'.      IFX00170
C             RESULT TAKES ONLY AS MANY CHAR. AS REQUIRED.              IFX00180
C     THE OUTPUT FIELD IS COMPLETELY FILLED.  LEADING ZEROES ARE        IFX00190
C     SUPPLIED IF NECESSARY.  IF FIELD IS TOO NARROW, LEADING DIGITS    IFX00200
C     ARE TRUNCATED.                                                    IFX00210
C     NOTE.. IF N GT 0, C AND L MUST BOTH BE VARIABLES, NOT CONSTANTS.  IFX00220
C     UPON RETURN, IFX0 IS SET TO..                                     IFX00230
C         =-1  IF OUTPUT FIELD IS TOO NARROW FOR THE CONVERTED VALUE.   IFX00240
C              C AND L ARE RETURNED UNCHANGED.                          IFX00250
C              IF N LE 0, THE FIELD IS FILLED WITH ASTERISKS.           IFX00260
C         =+1  IF OUTPUT FIELD IS WIDE ENOUGH.  IF N GT 0, C IS SET TO  IFX00270
C              REFER TO THE 1ST CHAR. AFTER THE OUTPUT FIELD, AND       IFX00280
C              L IS A COUNT OF THE REMAINING CHARS IN S, AFTER THE FLD. IFX00290
C                                                                       IFX00300
      OS = N.GT.0                                                       IFX00310
      LL = L                                                            IFX00320
      IF (OS) LL=MIN0(L,N)                                              IFX00330
      IF (LL.LE.0) GO TO 30                                             IFX00340
      CALL RCHXTK(U,C+LL-1)                                             IFX00350
      M = IABS(N)                                                       IFX00360
      DO 1 I=1,LL                                                       IFX00370
      J=M/10                                                            IFX00380
      X=M-J*10                                                          IFX00390
      CALL PCHXTK(S,U,X+D)                                              IFX00400
```

```
 1 M=J                                                    IFX00410
   IF(.NOT.OS) RETURN                                     IFX00420
   C = C+LL                                               IFX00430
   L = L-LL                                               IFX00440
   RETURN                                                 IFX00450
30 IFXO = -1                                              IFX00460
   RETURN                                                 IFX00470
   END                                                    IFX00480
```

Reproduced from best available copy.

```
C     **********                                                        ILGI0010
C     *  ILGI  *          JOHN LOUGH      IITRI      MAY 1971           ILGI0020
C     **********                                                        ILGI0030
C                                                                       ILGI0040
      FUNCTION ILGI(S,C,L,W,Q)                                          ILGI0050
      INTEGER S(1),C,W,Z                                                ILGI0060
      LOGICAL QS,G                                                      ILGI0070
C***  M,D, CHAR, F, T                                                   ILGI0080
      DATA IF,IT/013,031/                                               ILGI0090
C                                                                       ILGI0100
C     CONVERT THE LL CHAR, OF STRING 'S', STARTING AT CHAR, 'C', TO     ILGI0110
C     THE LOGICAL VALUE 'G'.                                            ILGI0120
C     IF W LE 0, LL=L.  IF W GT 0, LL=W, OR LL=L IF L LT W.             ILGI0130
C     BLANKS ARE IGNORED.  AN ALL BLANK OR A NULL STRING IS OK.         ILGI0140
C     G IS SET FALSE UNLESS A 'T' IS FOUND BEFORE THE 1ST 'F' (IF ANY)  ILGI0150
C     NOTE.. IF W GT 0, C AND L MUST BOTH BE VARIABLES, NOT CONSTANTS.  ILGI0160
C     UPON RETURN, ILGI IS SET TO 1.  NO ERROR CONDITIONS ARE CHECKED.  ILGI0170
C                                                                       ILGI0180
      QS = W.GT.0                                                       ILGI0190
      LL = L                                                            ILGI0200
      IF (QS) LL=MIN0(L,W)                                              ILGI0210
      ILGI = 1                                                          ILGI0220
      G = .FALSE.                                                       ILGI0230
      IF(LL.LE.0) RETURN                                                ILGI0240
      CALL RCHXTK(Z,C)                                                  ILGI0250
      DO 8 I=1,LL                                                       ILGI0260
      J = ICHXTK(S,Z)                                                   ILGI0270
      IF(J.EQ.IT) GO TO 7                                               ILGI0280
      IF(J.EQ.IF) GO TO 9                                               ILGI0290
    8 CONTINUE                                                          ILGI0300
      I = LL                                                            ILGI0310
    9 GO TO 10                                                          ILGI0320
    7 G = .TRUE.                                                        ILGI0330
   10 IF(.NOT.QS) RETURN                                                ILGI0340
      L = L-I                                                           ILGI0350
      C = C+I                                                           ILGI0360
      RETURN                                                            ILGI0370
      END                                                               ILGI0380
```

```
C     **********                                                        INTI0010
C     *  INTI  *          JOHN HOUSE      IITRI       MAY 1971           INTI0020
C     **********                                                        INTI0030
C                                                                       INTI0040
      FUNCTION INTI(S,C,L,W,I)                                          INTI0050
      INTEGER S(1),C,W                                                  INTI0060
      LOGICAL QS                                                        INTI0070
C***  M, D, CHAR, BLANK,PLUS,[illegible],[illegible],[illegible],EQUIV PLUS  INTI0080
      DATA IB,IP,IM,ID/[illegible],[illegible],[illegible],[illegible],[illegible],[illegible]/,IPL/043/  INTI0090
C                                                                       INTI0100
C     CONVERT THE LL CHAR. OF STRING 'S', STARTING AT CHAR. 'C', TO     INTI0110
C     A SIGNED INTEGER 'I'.                                             INTI0120
C     IF W LE 0, LL=L.  IF W GT 0, LL=W, OR LL=L IF L LT W.             INTI0130
C     BLANKS ARE IGNORED.  AN ALL BLANK OR A NULL STRING IS OK.         INTI0140
C     NOTE,, IF W GT 0, C AND L MUST BOTH BE VARIABLES, NOT CONSTANTS.  INTI0150
C     UPON RETURN, INTI IS SET TO,.                                     INTI0160
C         =-1  IF NO LEGAL CHAR. IS FOUND (EXCEPT POSSIBLY BLANKS)      INTI0170
C         = 0  IF AN ILLEGAL CHAR. IS FOUND, BUT AFTER AT LEAST 1 LEGAL.INTI0180
C         IF ONE OF THESE CODES IS RETURNED, AND IF W GT 0,             INTI0190
C         THEN THE ILLEGAL CHAR. IS CHAR. C OF STRING S, AND            INTI0200
C         L IS A COUNT OF REMAINING CHARS IN S, INCL. THE ILL. CHAR.    INTI0210
C         =+1  IF THERE ARE NO ILLEGAL CHARS IN THE INPUT FIELD.        INTI0220
C              IF W GT 0, C IS SET TO REFER TO THE 1ST CHAR AFTER THE   INTI0230
C              INPUT FIELD, AND L IS A COUNT OF THE REMAINING CHAR. IN SINTI0240
C                                                                       INTI0250
      QS = W.GT.0                                                       INTI0260
      LL = L                                                            INTI0270
      IF (QS) LL=MIN0(L,W)                                              INTI0280
C  CONVERT FIELD                                                        INTI0290
    1 N = 0                                                             INTI0300
      NS = 0                                                            INTI0310
      INTI = 1                                                          INTI0320
      IF (LL.LE.0) RETURN                                               INTI0330
      CALL RCHXTK(U,C)                                                  INTI0340
      K2 = C+LL-1                                                       INTI0350
      DO 10 K=C,K2                                                      INTI0360
      I = ICHXTK(S,U)                                                   INTI0370
      IF(I.EQ.IB) GO TO 10                                              INTI0380
      IF (NS.NE.0) GO TO 18                                             INTI0390
      IF(I.EQ.IM) GO TO 15                                              INTI0400
```

Reproduced from best available copy.

```
      IF(I.EQ.IP .OR. I.EQ.IPE) GO TO 16                                INTI0410
   18 IF(I.LT.IDD .OR. I.GT.IDD) GO TO 20                               INTI0420
      N = N*10+I-IDD                                                    INTI0430
      IF (NS.EQ.0) GO TO 16                                             INTI0440
      GO TO 10                                                          INTI0450
   15 NS = -1                                                           INTI0460
      GO TO 10                                                          INTI0470
   16 NS = +1                                                           INTI0480
   10 CONTINUE                                                          INTI0490
      K = C+LL                                                          INTI0500
      GO TO 25                                                          INTI0510
   20 IF (NS.EQ.0) GO TO 30                                             INTI0520
      INTI = 0                                                          INTI0530
   25 IF(NS.LT.0) N=-N                                                  INTI0540
      GO TO 35                                                          INTI0550
   30 INTI = -1                                                         INTI0560
   35 IF(.NOT.QS) RETURN                                                INTI0570
      L = L-K+C                                                         INTI0580
      C = K                                                             INTI0590
      RETURN                                                            INTI0600
      END                                                               INTI0610
```

```
C     **********                                                        IREI0010
C     *  IREI  *          JOHN ROUSE      IITRI       MAY 1971           IREI0020
C     **********                                                        IREI0030
C                                                                       IREI0040
      FUNCTION IREI(S,C,L,W,DAV)                                        IREI0050
      DOUBLE PRECISION FV,VV,BIG,SMALL,T(10),TT(5)                      IREI0060
      INTEGER S(1),C,A,D,ECOD,SC(2),U                                   IREI0070
      LOGICAL GS,GX                                                     IREI0080
      EQUIVALENCE(BIG,ECOD(1)),(SMALL,SC(1))                            IREI0090
C***  M.D. CHAR. BLANK, ZERO, NINE, PLUS, 'E', MINUS.  MXI=MAX INT DGTS IREI0100
      DATA IB,ID0,ID9,IP,IE,IM/005,060,071,1H.,1HE,041/,MXI/10/         IREI0110
C***  M.D. LARGEST AND SMALLEST REAL VALUES                             IREI0120
      DATA BG,SM/021777777777,077777777777,016000000000,0/               IREI0130
      DATA T/1.D0,1.D1,1.D2,1.D3,1.D4,1.D5,1.D6,1.D7,1.D8,1.D9/         IREI0140
      DATA TT/1.D-40,1.D-30,1.D-20,1.D-10,1.D+00,1.D+10,1.D+20,1.D+30/  IREI0150
C                                                                       IREI0160
C     CONVERT THE LL CHAR. OF STRING 'S', STARTING AT CHAR. 'C', TO     IREI0170
C     A REAL VALUE 'V'.                                                 IREI0180
C     IF W LE 0, LL=L.  IF W GT 0, LL=W, OR LL=L IF L LT W.             IREI0190
C     BLANKS ARE IGNORED.  AN ALL BLANK OR A NULL STRING IS OK.         IREI0200
C     NO SIGN OR DECIMAL POINT IS REQUIRED.  HOWEVER, IF THERE IS NO    IREI0210
C     DECIMAL POINT, ONE IS ASSUMED TO EXIST JUST TO THE LEFT OF THE    IREI0220
C     'D'TH DIGIT (COUNTING FROM THE RIGHT.)                            IREI0230
C     AN EXPONENT FIELD IS OPTIONAL.  IF PRESENT, IT MUST BE IN ONE OF  IREI0240
C     THE FOLLOWING FORMS,.  +N, -N, E+N, E-N, E, WHERE 'N' IS ONE      IREI0250
C     OR MORE DIGITS.                                                   IREI0260
C     NOTE.. IF W GT 0, C AND L MUST BOTH BE VARIABLES, NOT CONSTANTS.  IREI0270
C     UPON RETURN, IREI IS SET TO..                                     IREI0280
C         =-1  IF NO LEGAL CHAR. IS FOUND (EXCEPT POSSIBLY BLANKS)      IREI0290
C              OR IF THE MAGNITUDE OF THE INPUT VALUE IS OUT OF RANGE.  IREI0300
C         = 0  IF AN ILLEGAL CHAR. IS FOUND, BUT AFTER AT LEAST 1 LEGAL.IREI0310
C         IF ONE OF THESE CODES IS RETURNED, AND IF W GT 0,             IREI0320
C         THEN THE ILLEGAL CHAR. IS CHAR. C OF STRING S, AND            IREI0330
C         L IS A COUNT OF REMAINING CHARS IN S, INCL. THE ILL. CHAR.    IREI0340
C         =+1  IF THERE ARE NO ILLEGAL CHARS IN THE INPUT FIELD.        IREI0350
C              IF W GT 0, C IS SET TO REFER TO THE 1ST CHAR AFTER THE   IREI0360
C              INPUT FIELD, AND L IS A COUNT OF THE REMAINING CHAR. IN SIREI0370
C                                                                       IREI0380
      GS = W.GT.0                                                       IREI0390
      IL = L                                                            IREI0400
```

Reproduced from best available copy.

```
      IF (OS) IL=MIN0(L, )                                              IREI0410
C  SET UP CONVERSION                                                    IREI0420
      IK = 0                                                            IREI0430
      IF(IL.LE.0) GO TO 41                                              IREI0440
      IREI = 1                                                          IREI0450
      IXV = 0                                                           IREI0460
      IX = 1                                                            IREI0470
      ID = 0                                                            IREI0480
      FV = 0.0                                                          IREI0490
      OX = .FALSE.                                                      IREI0500
C  SEARCH FOR MINUS SIGN (TO ALLOW PROPER HANDLING OF -0.XXX NUMBERS)   IREI0510
      IIS = +1                                                          IREI0520
      CALL BCHXTK(U,IK)                                                 IREI0530
      DO 11 I=1,IL                                                      IREI0540
      J = ICHXTK(S,U)                                                   IREI0550
      IF (J.EQ.IM) GO TO 13                                             IREI0560
      IF (J.NE.IB) GO TO 12                                             IREI0570
   11 CONTINUE                                                          IREI0580
      I = IL                                                            IREI0590
      GO TO 14                                                          IREI0600
   12 I = I-1                                                           IREI0610
      GO TO 14                                                          IREI0620
   13 IIS = -1                                                          IREI0630
   14 IK = IK+I                                                         IREI0640
      IL = IL-I                                                         IREI0650
C  CONVERT INTEGER PART, LESS SIGN                                      IREI0660
   15 LL = IL                                                           IREI0670
      II = INTI(S,IK,IL,LL,IIV)                                         IREI0680
      IF(IIV.LT.0) GO TO 60                                             IREI0690
      VV = IIV*IIS                                                      IREI0700
      IF(II.LE.0) GO TO 20                                              IREI0710
      ID = 0                                                            IREI0720
      GO TO 50                                                          IREI0730
C  SEARCH FOR DECIMAL POINT                                             IREI0740
   20 IF(KOM(S,IK,1,IP,1).NE.0) GO TO 38                                IREI0750
   25 IK = IK+1                                                         IREI0760
      IL = IL-1                                                         IREI0770
      IF(IL.LE.0) GO TO 50                                              IREI0780
C  CONVERT FRACTION(IF PRESENT)                                         IREI0790
      IF = 0                                                            IREI0800
```

```
      IFV = 0                                                           IREI0810
      LL = 0                                                            IREI0820
      CALL RCHXTK(U,IK)                                                 IREI0830
      DO 31 I=1,IL                                                      IREI0840
      J = ICHXTK(S,U)                                                   IREI0850
      IF(J.EQ.IB) GO TO 31                                              IREI0860
      IF(J.LT.'00 .OR. J.GT.'09') GO TO 35                              IREI0870
      IF (LL.GE.MXI) GO TO 31                                           IREI0880
      IFV = IFV*10+J-IC0                                                IREI0890
      LL = LL+1                                                         IREI0900
   31 CONTINUE                                                          IREI0910
      IF = 1                                                            IREI0920
      I = IL+1                                                          IREI0930
   35 IF(LL.LE.0) IF=-1                                                 IREI0940
      I = I-1                                                           IREI0950
      IK = IK+1                                                         IREI0960
      IL = IL-I                                                         IREI0970
      FV = IFV                                                          IREI0980
      L1 = MOD(LL+50,10)+1                                              IREI0990
      L2 = (LL+50)/10                                                   IREI1000
      FV = FV/T(L1)/TT(L2)                                              IREI1010
      IF(IIS.LT.0) FV=-FV                                               IREI1020
   34 IF(IF.GT.0) GO TO 50                                              IREI1030
      GO TO 40                                                          IREI1040
   38 IF(II.LT.0) GO TO 60                                              IREI1050
      ID = 0                                                            IREI1060
C  SEARCH FOR 'E'                                                       IREI1070
   40 IF(KOM(S,IK,1,IE,1).NE.0) GO TO 45                                IREI1080
      GX = .TRUE.                                                       IREI1090
   41 IK = IK+1                                                         IREI1100
      IL = IL-1                                                         IREI1110
      IF(IL.LE.0) GO TO 60                                              IREI1120
C  CONVERT EXPONENT                                                     IREI1130
   45 LL = IL                                                           IREI1140
      IX = INTI(S,IK,IL,LL,IXV)                                         IREI1150
      IF (IX.LT.0 .AND. (GX .OR. .NOT.GS)) GO TO 60                     IREI1160
C  GENERATE FINAL VALUE AND CHECK SIZE                                  IREI1170
   50 VV = VV+FV                                                        IREI1180
      IF(DABS(VV).LE.0.D0) GO TO 55                                     IREI1190
      IXV = IXV-ID                                                      IREI1200
```

Reproduced from best available copy.

```
      IF(IXV.LT.-40 .OR. IXV.GE.+40) GO TO 60                           IREI1210
      L1 = MOD(IXV+50,10)+1                                             IREI1220
      L2 = (IXV+50)/10                                                  IREI1230
      VV = VV*T(L1)*TT(L2)                                              IREI1240
      IF(DABS(VV).LT.SMALL .OR. DABS(VV).GT.BIG) GO TO 60               IREI1250
   55 V = VV                                                            IREI1260
      IF(IX.LE.0) IREI=0                                                IREI1270
      GO TO 70                                                          IREI1280
C  ERROR DETECTED                                                       IREI1290
   60 IREI = -1                                                         IREI1300
   61 V = 0.                                                            IREI1310
   70 IF(.NOT.OS) RETURN                                                IREI1320
      C = IK                                                            IREI1330
      L = IL                                                            IREI1340
      RETURN                                                            IREI1350
      END                                                               IREI1360
```

```
C     **********                                                        ISCA0010
C     * ISCAN  *          JOHN ROUSE      IITRI       SEPT 1970          ISCA0020
C     **********                                                        ISCA0030
C                                                                       ISCA0040
      FUNCTION ISCAN(S1,C1,L1,TRT,C2,L2)                                ISCA0050
C                                                                       ISCA0060
C     SCANS UP TO L1 CH., STARTING W/CH.C1 OF STRING S1.  PICKS         ISCA0070
C     UP TOKENS, DEFINED BY USER, VIA TRT TABLE.                        ISCA0080
C     A TOKEN IS 1) 1 CH. WHOSE CH. W/A ZERO ENTRY IN TRT TABLE, OR     ISCA0090
C                2) 1 CH., W/A NONZERO TRT ENTRY                        ISCA0100
C     TOKEN TYPE IS INDICATED BY RETURNED VALUE OF ISCAN                ISCA0110
C                                                                       ISCA0120
C     ISCAN    = 0  TEXT (1 OF MORE ZERO-TRT-ENTRY CH.)                 ISCA0130
C     ISCAN    = 1  END OF STRING  (L1=0 ON INPUT.  C1 OUT=C1 IN)       ISCA0140
C     ISCAN    .GT.1  DELIMITER  (NON-ZERO-TRT-ENTRY CH.)               ISCA0150
C     C1       1ST CH. SCANNED.  ON OUTPUT, IS NEXT CH. TO BE SCANNED.  ISCA0160
C     L1       NO. CH. TO BE SCANNED.  ON OUTPUT, IS NO.CH. LEFT.       ISCA0170
C     TRT      64 WORD INTEGER TABLE, 1 ENTRY FOR EACH POSSIBLE CH.     ISCA0180
C     C2       = INPUT VALUE OF C1                                      ISCA0190
C     L2       = LEN. OF TOKEN.  (=0 IF END OF STRING)                  ISCA0200
C     STRING S IS PACKED (6 CHAR/WORD)                                  ISCA0210
C                                                                       ISCA0220
      INTEGER S1(1),TRT(64)                                             ISCA0230
      INTEGER C1,C2,CN                                                  ISCA0240
C                                                                       ISCA0250
C  L=0 SIGNALS END-OF-STRING                                            ISCA0260
      IF(L1.GT.0) GO TO 1                                               ISCA0270
      L2 = 0                                                            ISCA0280
      CN = C1                                                           ISCA0290
      ISCAN = 1                                                         ISCA0300
      GO TO 6                                                           ISCA0310
C  TRANSLATE AND TEST S1, USING PROVIDED TRT TABLE                      ISCA0320
    1 I = ITRT(S1,C1,L1,TRT,CN)                                         ISCA0330
      IF(I) 4,5,4                                                       ISCA0340
    4 IF(CN.GT.C1+1) GO TO 7                                            ISCA0350
C  DELIMITER FOUND                                                      ISCA0360
      L2 = 1                                                            ISCA0370
      ISCAN = IABS(I)                                                   ISCA0380
      GO TO 6                                                           ISCA0390
C  TEXT FOUND                                                           ISCA0400
```

Reproduced from best available copy.

```
    7 CN = CN-1                                                         ISCA0410
    5 L2 = CN-C1                                                        ISCA0420
      ISCAN =0                                                          ISCA0430
C   SET UP FOR NEXT SCAN, AND SET TOKEN LOCATION                        ISCA0440
    6 L1 = L1+L2                                                        ISCA0450
      C2 = C1                                                           ISCA0460
      C1 = CN                                                           ISCA0470
      RETURN                                                            ISCA0480
      END                                                               ISCA0490
```

```
C     **********                                                        ITRT0010
C     *  ITRT  *          [illegible]      ITRT      SEPT 1970           ITRT0020
C     **********                                                        ITRT0030
C                                                                       ITRT0040
      FUNCTION ITRT(S1,C1,L1,TRT,CN)                                    ITRT0050
C                                                                       ITRT0060
C     EXAMINE UP TO L1 CHS. STARTING W/CH.C1 OF STRING S1,              ITRT0070
C     STOP WITH 1ST CH. WHOSE TRT ENTRY IS NONZERO.                     ITRT0080
C     RETURN WITH CN = NEXT CH. LOC., AND                               ITRT0090
C     ITRT   =-J   TOKEN OF TYPE J FOUND AT END OF STRING               ITRT0100
C            = 0   NO NON-ZERO CH. ENCOUNTERED                          ITRT0110
C            = N   TOKEN OF TYPE N FOUND, MORE WILL FOLLOW              ITRT0120
C     STRING S1 IS PACKED (6 CHAR/WORD)                                 ITRT0130
C                                                                       ITRT0140
      INTEGER S1(1),TRT(64),J                                           ITRT0150
      INTEGER C1,CN                                                     ITRT0160
C                                                                       ITRT0170
      CALL BCHXTK(W,C1)                                                 ITRT0180
      IC = C1-1+L1                                                      ITRT0190
      DO 1 I=C1,IC                                                      ITRT0200
      J = ICHXTK(S1,I)+1                                                ITRT0210
      ITRT = TRT(J)                                                     ITRT0220
      IF(ITRT.NE.0) GO TO 2                                             ITRT0230
    1 CONTINUE                                                          ITRT0240
      CN = IC+1                                                         ITRT0250
      ITRT = 0                                                          ITRT0260
      RETURN                                                            ITRT0270
    2 CN = I+1                                                          ITRT0280
      IF(I.EQ.IC) ITRT=-ITRT                                            ITRT0290
      RETURN                                                            ITRT0300
      END                                                               ITRT0310
```

Reproduced from best available copy.

```
C     **********                                                        KOM 0010
C     *  KOM   *          JOHN ROUSE      IITRI       SEPT 1970          KOM 0020
C     **********                                                        KOM 0030
C                                                                       KOM 0040
      FUNCTION KOM (S1,C1,L,S2,C2)                                      KOM 0050
      INTEGER S1(1),C1,W1,S2(1),C2,W2                                   KOM 0060
C                                                                       KOM 0070
C     COMPARE TWO STRINGS OVER L CHAR. (IF L.LE.0, RETURN A ZERO)       KOM 0080
C     THE STRINGS ARE,, STRING S1, STARTING AT CHAR. C1, AND STRING S2, KOM 0090
C     STARTING AT CHAR. C2.  A VALUE BETWEEN -1 AND +1 IS RETURNED,,    KOM 0100
C          -1) S1.LT.S2      0) S1=S2      +1) S1.GT.S2                 KOM 0110
C     BOTH STRINGS S1 AND S2 ARE PACKED (I.E., 6 CHAR. PER WORD)        KOM 0120
C                                                                       KOM 0130
      KOM = 0                                                           KOM 0140
      IF(L.LE.0) RETURN                                                 KOM 0150
      CALL RCHXTK(W1,C1)                                                KOM 0160
      CALL RCHXTK(W2,C2)                                                KOM 0170
      DO 6 I=1,L                                                        KOM 0180
      IF(ICHXTK(S1,W1)-ICHXTK(S2,W2)) 1,6,2                             KOM 0190
    6 CONTINUE                                                          KOM 0200
      RETURN                                                            KOM 0210
    1 KOM = -1                                                          KOM 0220
      RETURN                                                            KOM 0230
    2 KOM = +1                                                          KOM 0240
      RETURN                                                            KOM 0250
      END                                                               KOM 0260
```

```
C     **********                                                        KOML0010
C     *  KOML  *          KOMP RPOSL      IITRI      MAY 1971            KOML0020
C     **********                                                        KOML0030
C                                                                       KOML0040
      FUNCTION KOML(S1,C1,L1,S2,C2,L2)                                  KOML0050
      INTEGER S1(1),C1,L1,S2(1),C2,L2                                   KOML0060
C                                                                       KOML0070
C     COMPARE TWO STRINGS OVER THE LENGTH OF THE SHORTER STRING.        KOML0080
C     THE TWO STRINGS ARE..                                             KOML0090
C         THE L1 CHAR. OF STRING S1, STARTING WITH CHAR. C1, AND        KOML0100
C         THE L2 CHAR. OF STRING S2, STARTING WITH CHAR. C2.            KOML0110
C     UPON RETURN, KOML IS SET TO..                                     KOML0120
C         =-2  IF S1 LT S2                                              KOML0130
C         =-1  IF S1 EQ S2 BUT L1 LT L2                                 KOML0140
C         = 0  IF S1 EQ S2 AND L1 EQ L2                                 KOML0150
C         =+1  IF S1 EQ S2 BUT L1 GT L2                                 KOML0160
C         =+2  IF S1 GT S2                                              KOML0170
C                                                                       KOML0180
      L = L1                                                            KOML0190
      KOML = 0                                                          KOML0200
      IF (L1-L2) 4,5,3                                                  KOML0210
    3 L = L2                                                            KOML0220
      KOML = +1                                                         KOML0230
      GO TO 5                                                           KOML0240
    4 KOML = -1                                                         KOML0250
    5 IF (L.LE.0) RETURN                                                KOML0260
      CALL RCHXTK(W1,C1)                                                KOML0270
      CALL RCHXTK(W2,C2)                                                KOML0280
      DO 6 I=1,L                                                        KOML0290
      IF (ICHXTK(S1,W1).NE.ICHXTK(S2,W2)) 2,6,1                         KOML0300
    6 CONTINUE                                                          KOML0310
      RETURN                                                            KOML0320
    2 KOML = -2                                                         KOML0330
      RETURN                                                            KOML0340
    1 KOML = +2                                                         KOML0350
      RETURN                                                            KOML0360
      END                                                               KOML0370
```

Reproduced from best available copy.

```
      FUNCTION KOMW(S1,L,S2)                                            KOMW0010
C                                                                       KOMW0020
C     **********                                                        KOMW0030
C     *  KOMW  *        JOHN ROUSE      IITRI      OCT 1970            KOMW0040
C     **********                                                        KOMW0050
C                                                                       KOMW0060
      INTEGER S1(1),S2(1)                                               KOMW0070
C     COMPARE TWO ALPHANUMERIC STRINGS, S1 AND S2, EACH L WORDS LONG,   KOMW0080
C     UPON RETURN, KOMW IS SET TO..                                     KOMW0090
C         -1   IF S1 LT S2                                              KOMW0100
C          0   IF S1 EQ S2 (OR IF L LE 0, IE NULL STRINGS)              KOMW0110
C         +1   IF S1 GT S2                                              KOMW0120
C                                                                       KOMW0130
      KOMW = 0                                                          KOMW0140
      IF (L.LE.0) RETURN                                                KOMW0150
      DO 30 I=1,L                                                       KOMW0160
      IF (S1(I)) 10,5,15                                                KOMW0170
    5 IF (S2(I).EQ.0) GO TO 35                                          KOMW0180
      GO TO 35                                                          KOMW0190
   10 IF (S2(I).GE.0) GO TO 40                                          KOMW0200
      GO TO 20                                                          KOMW0210
   15 IF (S2(I)) 35,40,20                                               KOMW0220
C***  THIS STEP COUNTS ON NEG. VALUES IN 1- OR 2-COMPL FORM             KOMW0230
   20 IF (S1(I)-S2(I)) 35,30,40                                         KOMW0240
   30 CONTINUE                                                          KOMW0250
      RETURN                                                            KOMW0260
   35 KOMW = -1                                                         KOMW0270
      RETURN                                                            KOMW0280
   40 KOMW = +1                                                         KOMW0290
      RETURN                                                            KOMW0300
      END                                                               KOMW0310
```

```
C     **********                                                        MOVE0010
C     *  MOVE  *          JOHN [illegible]      DATE:      SEPT 1970     MOVE0020
C     **********                                                        MOVE0030
      SUBROUTINE MOVE (S1,C1,L,S2,C2)                                   MOVE0040
      INTEGER S1(1),S2(1),C1,C2,L                                       MOVE0050
C                                                                       MOVE0060
C     MOVE L CHAR. FROM STRING S2, STARTING WITH CHAR. C2, TO STRING S1,MOVE0070
C     STARTING AT CHAR. C1.  IF L.LE.0, NO MOVE TAKES PLACE.            MOVE0080
C     BOTH STRINGS S1 AND S2 ARE PACKED (I.E., 6 CHAR. PER WORD)        MOVE0090
C                                                                       MOVE0100
      IF(L.LE.0) RETURN                                                 MOVE0110
      CALL SCHXTK(S1,C1)                                                MOVE0120
      CALL SCHXTK(S2,C2)                                                MOVE0130
      DO 1 I=1,L                                                        MOVE0140
    1 CALL SCHXTK(S1,C1,[illegible]                                     MOVE0150
      RETURN                                                            MOVE0160
      END                                                               MOVE0170
```

Reproduced from best available copy.

```
C     **********                                                        NOTA0010
C     * NOTARY *          JOHN ROUSE      IITRI       SEPT 1970          NOTA0020
C     **********                                                        NOTA0030
C                                                                       NOTA0040
      SUBROUTINE NOTARY(FCN,N,M,XS,XE,MYY,YS,YE,CHAR)                   NOTA0050
C                                                                       NOTA0060
C     PLOTS N FUNCTIONS (N.LE.20) ON PRINTER                            NOTA0070
C     LIST 'CHAR' HOLDS PLOT CHARACTERS,. FCN NO. I IS PLOTTED USING    NOTA0080
C     CHAR(I).                                                          NOTA0090
C     FCN MUST BE OF THE FORM 'FCN(X,N,VAL)', AND MUST BE DECLARED      NOTA0100
C     EXTERNAL IN THE PROGRAM WHICH CALLS NOTARY.                       NOTA0110
C     FUNCTION VALUES ARE GENERATED BY FCN, AND STORED IN VECTOR        NOTA0120
C     VAL,  FCN VALUE I IS STORED IN VAL(I).  FCN IS CALLED M TIMES,    NOTA0130
C     AT X = XS, XS+DX, ...., XE.  (DX=(XE-XS)/(M-1))                   NOTA0140
C     THE Y AXIS INCREASES ALONG THE WIDTH OF THE PAGE, AND THE X AXIS  NOTA0150
C     INCREASES DOWN THE PAGE,  THE Y RANGE IS YS THRU YE, IN MYY       NOTA0160
C     (MYY.LE.121) INCREMENTS,  OFF-SCALE FUNCTION VALUES ARE NOT PLOT- NOTA0170
C     TED.                                                              NOTA0180
      INTEGER POT(121),CHAR(N),DOT,BL                                   NOTA0190
      REAL VAL(20)                                                      NOTA0200
      DIMENSION IG(30)                                                  NOTA0210
      LOGICAL GGRID                                                     NOTA0220
      DATA MG/30/                                                       NOTA0230
      DATA DOT/1H./,BL/1H /                                             NOTA0240
      IF(N.LE.20 .AND. N.GT.0 .AND. M.GT.0 .AND. MYY.GT.0) GO TO 201    NOTA0250
      WRITE(6,200) N,M,MYY                                              NOTA0260
  200 FORMAT(57H0******  ERROR FROM NOTARY.  20 FUNCTIONS ALLOWED. BUT NNOTA0270
     *>,I5/24X,16HOR,. NO, LINES (,I5,16H) OR NO. COLS. (,I5,10H) IN ERRNOTA0280
     *OR)                                                               NOTA0290
      RETURN                                                            NOTA0300
  201 MY = MIN0(MYY,121)                                                NOTA0310
      DX = (XE-XS)/FLOAT(M-1)                                           NOTA0320
      DY = (YE-YS)/FLOAT(MY-1)                                          NOTA0330
      MGJ = MAX0(2,MIN0(MY/20+1,MG-2))                                  NOTA0340
      MGI = MG-MGJ                                                      NOTA0350
      CALL PPGRID(YS,YE,MY,NGJ,MGJ,IG,20)                               NOTA0360
      CALL PPGRID(XS,XE,M,NGI,MGI,IG(NGJ+1),12)                         NOTA0370
      NGI = NGI+NGJ                                                     NOTA0380
      KGI = NGJ+1                                                       NOTA0390
      SC = AMAX1(ABS(XS),ABS(XE))                                       NOTA0400
```

```
      IF(SC.GT.0.) SC=ALOG10(SC)                                        NOTA0410
      IF(SC.LT.0.) SC=SC-3.                                             NOTA0420
      SC = AINT(SC/3.)*3.                                               NOTA0430
      SC = 10.**(-AINT(SC))                                             NOTA0440
      WRITE(6,202) YS,SC,YE                                             NOTA0450
  202 FORMAT(11X,6HYMIN =,E12.4,31X,10HX MULT. BY,E12.4,32X,6HYMAX =,   NOTA0460
     * E12.4/)                                                          NOTA0470
      DSC = DX*SC                                                       NOTA0480
      XSC = XS*SC                                                       NOTA0490
      X=XS                                                              NOTA0500
      DO 102 I=1,M                                                      NOTA0510
      GGRID=.FALSE.                                                     NOTA0520
      IF(KGI.GT.NGI) GO TO 110                                          NOTA0530
      IF(I.LT.IG(KGI)) GO TO 110                                        NOTA0540
      GGRID = .TRUE.                                                    NOTA0550
      KGI = KGI+1                                                       NOTA0560
  110 KGJ = 1                                                           NOTA0570
      DO 101 J=1,MY                                                     NOTA0580
      IF(GGRID) GO TO 111                                               NOTA0590
      IF(KGJ.GT.NGJ) GO TO 112                                          NOTA0600
      IF(J.LT.JG(KGJ)) GO TO 112                                        NOTA0610
      KGJ = KGJ+1                                                       NOTA0620
  111 POT(J) = DOT                                                      NOTA0630
      GO TO 101                                                         NOTA0640
  112 POT(J) = BL                                                       NOTA0650
  101 CONTINUE                                                          NOTA0660
      CALL FCN(X,N,VAL)                                                 NOTA0670
      DO 103 J=1,N                                                      NOTA0680
      IY = (VAL(J)-YS)/DY+1.                                            NOTA0690
      IF(IY.GE.1 .AND. IY.LE.MY) POT(IY)=CHAR(J)                        NOTA0700
  103 CONTINUE                                                          NOTA0710
      WRITE(6,151) XSC,(POT(J),J=1,MY)                                  NOTA0720
  151 FORMAT(1X,F8.2,2X,121A1)                                          NOTA0730
      XSC = XSC+DSC                                                     NOTA0740
  102 X=X+DX                                                            NOTA0750
      RETURN                                                            NOTA0760
      END                                                               NOTA0770
```

```
C     **********                                                        PPGR0010
C     * PPGRID *          JOHN ROUSE      IITRI       SEPT 1970          PPGR0020
C     **********                                                        PPGR0030
C                                                                       PPGR0040
      SUBROUTINE PPGRID(QMIN,QMAX,NG,MG,MG,IG,JG)                       PPGR0050
      DIMENSION IG(1)                                                   PPGR0060
C                                                                       PPGR0070
C     COMPUTE GRID LINE POSITIONS FOR PRINTER PLOTTER.                  PPGR0080
C     AXIS RANGES FROM QMIN TO QMAX.  UP TO MG GRID LINES MAY BE PRO-   PPGR0090
C     DUCED, THE ACTUAL NUMBER IS RETURNED IN NG.  VECTOR IG HOLDS THE  PPGR0100
C     GRID LINE INDICES.  IG(1) WILL ALWAYS BE 1, AND IG(NG) WILL ALWAYSPPGR0110
C     EQUAL NG, THE NUMBER OF CHAR. POS, OR LINES USED BY THE AXIS.     PPGR0120
C     GRID LINES ARE CHOSEN TO BE AT LEAST JG/4 COLS. OR LINES FROM     PPGR0130
C     EITHER END OF THE AXIS, AND FROM JG TO 2*JG COLS/LINES APART FROM PPGR0140
C     EACH OTHER.  GRID LINES ARE CHOSEN AT 'ROUND' VALUES OF AXIS      PPGR0150
C     VARIABLE                                                          PPGR0160
C                                                                       PPGR0170
      NG = 1                                                            PPGR0180
      IG(1) = 1                                                         PPGR0190
      IF(MG-2) 54,50,43                                                 PPGR0200
   43 QR = QMAX-QMIN                                                    PPGR0210
      IDUM = NG/JG+1                                                    PPGR0220
      DG = QR/FLOAT(IDUM)                                               PPGR0230
      DQ = QR/FLOAT(NG-1)                                               PPGR0240
      DG = ALOG10(DG)                                                   PPGR0250
      IF(DG.LT.0.) DG=DG-1.                                             PPGR0260
      DG = 10.**AINT(DG)                                                PPGR0270
      DUM = FLOAT(NG)*DG/QR                                             PPGR0280
   41 IF(DUM.GE.FLOAT(JG)) GO TO 40                                     PPGR0290
      DUM = DUM*2.                                                      PPGR0300
      DG = DG*2.                                                        PPGR0310
      GO TO 41                                                          PPGR0320
   40 IF (DUM.LE.FLOAT(2*JG)) GO TO 42                                  PPGR0330
      DUM = DUM*0.5                                                     PPGR0340
      DG = DG*0.5                                                       PPGR0350
      GO TO 40                                                          PPGR0360
   42 JDUM = JG/4                                                       PPGR0370
      IDUM = NG-JDUM                                                    PPGR0380
      JDUM = JDUM+1                                                     PPGR0390
      DUM = AINT(QMIN/DG)*DG                                            PPGR0400
```

```
      IF(QMIN.LT.0.) DUMSUM=DUMSUM-DQ                                   PPGR0410
      IF(DUM.LT.QMIN) GO TO 52                                          PPGR0420
   51 J = (DUM-QMIN)/DQ+1.                                              PPGR0430
      IF(J.LE.JDUM) GO TO 52                                            PPGR0440
      IF(J.GE.IDUM) GO TO 52                                            PPGR0450
      NG = NG+1                                                         PPGR0460
      IF(NG.GE.MG) GO TO 53                                             PPGR0470
      IG(NG) = J                                                        PPGR0480
   52 DUM = DUM+DQ                                                      PPGR0490
      GO TO 51                                                          PPGR0500
   50 NG = NG+1                                                         PPGR0510
   53 IG(NG) = NP                                                       PPGR0520
   54 RETURN                                                            PPGR0530
      END                                                               PPGR0540
```

Reproduced from best available copy.

```
.          CHARACTER STORE AND EXTRACT ROUTINES                          RCHX0010
.                                                                        RCHX0020
           RETURN                                                        RCHX0030
.                                                                        RCHX0040
$(R),                                                                    RCHX0050
.                                                                        RCHX0060
.          SET POINTER 'W' TO CHAR. NO. OF STRING.                       RCHX0070
.                                                                        RCHX0080
.          FORTRAN CALL IS..                                             RCHX0090
.                    CALL RCHXTK( ,R)                                    RCHX0100
.                                                                        RCHX0110
RCHXTK*    LA,C16    A0,0            . CLEAR TOP OF DIVIDEND            RCHX0120
           LA        A1,*1,R11       . LOAD BOTTOM (ST. CHAR. COUNT)    RCHX0130
           AA,C16    A1,5            . PLUS 5                           RCHX0140
           DI,C16    A0,6            . DIVIDE BY 6 TO GET..             RCHX0150
           SA,C1     A0,*2,R11       . ST. WORD NO. (QUOTIENT), AND     RCHX0160
           SA,C2     A1,*2,R11       . ST. CHAR. NO. WITHIN WORD (REM.) RCHX0170
           J         3,R11                                              RCHX0180
.                                                                        RCHX0190
.          SET NEXT CHAR. IN STRING 'S' TO BIT PATTERN IN THE LOWER 6    RCHX0200
.          BITS OF 'K'.  'W' IS STRING PTR.                              RCHX0210
.                                                                        RCHX0220
.          FORTRAN CALL IS..                                             RCHX0230
.                    CALL SCHXTK(S,W,K)                                  RCHX0240
.                                                                        RCHX0250
SCHXTK*    LA,C1     A3,*1,R11       . GET CURR. WORD NO.               RCHX0260
           LA,C2     A2,*1,R11       . GET CURR. CHAR. NO.              RCHX0270
           LA        A1,C,B11        . GET ADDR OF START OF STRING      RCHX0280
           AA        A1,A3           . COMPUTE ADDR OF CURRENT WORD..   RCHX0290
           AMA,C16   A1,1            .     IS START-1+WORD NO.          RCHX0300
           LA        A0,*2,R11       . GET CHARACTER TO BE STORED.      RCHX0310
           EX        STORCH,A2       . EXECUTE PROPER CHAR. STORE INSTR. RCHX0320
           LMU       A1,NEXTCH       . JUMP TO PTR UPDATE ROUTINE       RCHX0330
           J         4,R11           . AND EXIT                         RCHX0340
.                                                                        RCHX0350
.         RETRIEVE NEXT CHAR. FROM STRING 'S'.  'W' IS STRING PTR.       RCHX0360
.                                                                        RCHX0370
.          FORTRAN CALL IS..                                             RCHX0380
.                    K = ICHXTK(S,W)                                     RCHX0390
.                                                                        RCHX0400
```

```
ICHXTK*   LA,01     A3,*1,B11      . GET CURR. WORD NO.                  RCHX0410
          LA,02     A2,*1,B11      . AND CURR. CHAR. NO.                 RCHX0420
          LA        A1,0,B11       . GET ADDR OF START OF STRING         RCHX0430
          AA        A1,A3          . COMPUTE ADDR OF CURRENT WORD..      RCHX0440
          ANA,016   A1,1           .    IS START-1+WORD NO.              RCHX0450
          EX        LOADCH,A2      . EXECUTE PROPER CHAR. LOAD INSTR.    RCHX0460
          LMJ       A1,NEXTCH      . JUMP TO PTR UPDATE ROUTINE          RCHX0470
          J         3,B11          . AND EXIT                            RCHX0480
.                                                                        RCHX0490
NEXTCH    AA,016    A2,1           . ADD 1 TO CHAR NO                    RCHX0500
          TLE,016   A2,6           . SKIP NI IF NEW CHAR NO GE 6         RCHX0510
          J         SAMEWD         . ELSE ARE IN SAME WORD AS BEFORE     RCHX0520
          AA,016    A3,1           . NEW WORD.  SET CHAR NO TO ZERO, ANDRCHX0530
          SA        A3,*1,B11      . UP WORD NO BY 1                     RCHX0540
          J         0,A1                                                 RCHX0550
SAMEWD    SA,02     A2,*1,B11      . SAME WORD.  STORE NEW CHAR NO       RCHX0560
          J          ,A1                                                 RCHX0570
.                                                                        RCHX0580
.         SIMILAR TO SCHXTK, BUT CHARACTERS ARE STORED FROM RIGHT TO     RCHX0590
.         LEFT.                                                          RCHX0600
.                                                                        RCHX0610
.         FORTRAN CALL IS..                                              RCHX0620
.                   CALL PCHXTK(S,W,K)                                   RCHX0630
.                                                                        RCHX0640
PCHXTK*   LA,01     A3,*1,B11      . GET CURR. WORD NO.                  RCHX0650
          LA,02     A2,*1,B11      . GET CURR. CHAR. NO.                 RCHX0660
          LA        A1,0,B11       . GET ADDR OF START OF STRING         RCHX0670
          AA        A1,A3          . COMPUTE ADDR OF CURRENT WORD..      RCHX0680
          ANA,016   A1,1           .    IS START-1+WORD NO.              RCHX0690
          LA        A0,*2,B11      . GET CHARACTER TO BE STORED.         RCHX0700
          EX        STORCH,A2      . EXECUTE PROPER CHAR. STORE INSTR.   RCHX0710
          LMJ       A1,PREVCH      . JUMP TO PTR UPDATE ROUTINE          RCHX0720
          J         4,B11          . AND EXIT                            RCHX0730
.                                                                        RCHX0740
.         SIMILAR TO ICHXTK, BUT CHARACTERS ARE RETRIEVED FROM RIGHT TO  RCHX0750
.         LEFT.                                                          RCHX0760
.                                                                        RCHX0770
.         FORTRAN CALL IS..                                              RCHX0780
.                   K = JCHXTK(S,W)                                      RCHX0790
.                                                                        RCHX0800
```

Reproduced from best available copy.

```
JCHXTK*   LA,D1     A3,*1,B11     . GET CURR. WORD NO.                RCHX0810
          LA,D2     A2,*1,B11     . AND CURR. CHAR. NO.               RCHX0820
          LA        A1,S,B11      . GET ADDR OF START OF STRING       RCHX0830
          AA        A1,A3         . COMPUTE ADDR OF CURRENT WORD.     RCHX0840
          ANA,D16   A1,1          .    IS START+1+WORD NO.            RCHX0850
          EX        LCHCH,A2      . EXECUTE PROPER CHAR. LOAD INSTR.  RCHX0860
          LJ        A1,PREVCH     . JUMP TO PTR UPDATE ROUTINE        RCHX0870
          J         B,B11                                             RCHX0880
.                                                                     RCHX0890
PREVCH    A A,D16   A2,1          . SUBTRACT 1 FROM CHAR. NO.         RCHX0900
          TLE,D16   A2,0          . SKIP NI IF NEW CHAR. NO. GE 0.    RCHX0910
          J         PREVL         . ELSE ARE STARTING PREVIOUS WORD   RCHX0920
          SA,D2     A2,*1,B11     . SAME WORD.  STORE NEW CHAR. NO.   RCHX0930
          J         B,A1          . AND EXIT                          RCHX0940
PREVLD    ANA,D16   A3,1          . PREVIOUS WORD.  SET CHAR. NO. TO 5.RCHX0950
          LXI,D16   A3,5          . PACK INTO ONE REGISTER            RCHX0960
          SA        A3,*1,B11     . AND STORE INTO PTR WORD           RCHX0970
          J         B,A1          . AND EXIT                          RCHX0980
.                                                                     RCHX0990
$(*).                                                                 RCHX1000
.                                                                     RCHX1010
.                                 STORE CHAR 0 THRU CHAR 5            RCHX1020
STORCH    SA,D15    A0,0,A1                                           RCHX1030
          SA,D14    A0,0,A1                                           RCHX1040
          SA,D13    A0,0,A1                                           RCHX1050
          SA,D12    A0,0,A1                                           RCHX1060
          SA,D11    A0,0,A1                                           RCHX1070
          SA,D10    A0,0,A1                                           RCHX1080
.                                 LOAD CHAR 0 THRU CHAR 5             RCHX1090
LOADCH    LA,D15    A0,0,A1                                           RCHX1100
          LA,D14    A0,0,A1                                           RCHX1110
          LA,D13    A0,0,A1                                           RCHX1120
          LA,D12    A0,0,A1                                           RCHX1130
          LA,D11    A0,0,A1                                           RCHX1140
          LA,D10    A0,0,A1                                           RCHX1150
          END                                                         RCHX1160
```

```
      SUBROUTINE SOJAC(FCN,XMIN,XMAX,YMIN,YMAX,FMIN,FMAX,IYR,INT,NF,L)  SOJA0010
C                                                                       SOJA0020
C     ***********                                                       SOJA0030
C     * SOJAC   *          JOHN HOUSE      11181       SEPT 1970        SOJA0040
C     ***********                                                       SOJA0050
C                                                                       SOJA0060
C***  X-VALUES ARE SCALED BY APPROPRIATE POWER OF TEN.                  SOJA0070
C     PLOTS CONTOUR MAP OF FCN(X,Y) ON PRINTER.                         SOJA0080
C                                                                       SOJA0090
C     NOTE.. IN CALLING ROUTINE 'FCN' MUST APPEAR IN AN 'EXTERNAL'      SOJA0100
C     STATEMENT.                                                        SOJA0110
C                                                                       SOJA0120
C     X RANGES BETWEEN XMIN AND XMAX.  X INCREASES DOWN THE PAGE.       SOJA0130
C                                                                       SOJA0140
C     Y RANGES BETWEEN YMIN AND YMAX.  Y INCREASES ACROSS THE PAGE.     SOJA0150
C                                                                       SOJA0160
C     F(X,Y) IS EXPECTED TO RANGE BETWEEN FMIN AND FMAX.                SOJA0170
C     L(3) IS PLOTTED IF A FUNCTION VALUE IS .GE. 1.E+38 (I.E. UNDE-    SOJA0180
C     FINED.)  L(4) IS PLOTTED FOR VALUES BELOW FMIN.  L(NF+4) FOR ABOVESOJA0190
C     FMAX.  THE REGION BETWEEN FMIN AND FMAX IS DIVIDED INTO NF EQUAL  SOJA0200
C     INTERVALS.  INTERVAL K IS PLOTTED WITH SYMBOL L(K+3).             SOJA0210
C                                                                       SOJA0220
C     TO HELP SMOOTH THE CONTOURS, THIS PROCEDURE IS FOLLOWED..  IF THE SOJA0230
C     FUNCTION VALUE AT A POINT BE WITHIN 0.05 INTERVAL WIDTH FROM THE  SOJA0240
C     EDGE OF AN INTERVAL, IT WILL BE PLOTTED WITH A BORDER CHARACTER.  SOJA0250
C     IF THE PLOT POSITION ABOVE THE CURRENT POSITION  (I.E., SAME Y,   SOJA0260
C     LOWER X) BE BLANK, THE BORDER CHARACTER IS L(2).  OTHERWISE IT IS SOJA0270
C     L(1).  THIS PROCEDURE IS NOT USED IF L(1) AND L(2) ARE BLANK.     SOJA0280
C                                                                       SOJA0290
C     THE PLOT IS IYR PRINT POSITIONS WIDE.  THE LENGTH OF THE PLOT IS  SOJA0300
C     WHATEVER IT HAS TO BE TO KEEP THE UNIT OF LENGTH THE SAME IN THE  SOJA0310
C     X AND Y DIRECTIONS.                                               SOJA0320
C     IF INT = 0, SOJAC RETURNS AFTER COMPUTING THESE QUANTITIES, FOUND SOJA0330
C     IN COMMON /CSOJAC/..  DX, DY THE AMOUNT X AND Y ARE STEPPED       SOJA0340
C     BETWEEN EVALUATION POINTS, AND IXR, THE NUMBER OF POINTS IN THE   SOJA0350
C     X DIRECTION.  THERE ARE IYR*IXR POINTS PLOTTED IN ALL.            SOJA0360
C     IF INT.GT.0, A PLOT IS GENERATED.                                 SOJA0370
C                                                                       SOJA0380
C                                THE Y RANGE AND THE WIDTH OF AN INTERVAL SOJA0390
C     ARE PLOTTED.  THE VALUE OF X IS PRINTED ALONG THE LEFT MARGIN,    SOJA0400
```

Reproduced from best available copy.

```
C                                                                       SOJA0410
      COMMON /CSPOJAC/D),PX,IXR,IYY,I,J                                 SOJA0420
C                                                                       SOJA0430
      INTEGER POT,DOT,BL,IG(30)                                         SOJA0440
      LOGICAL OFILL,OGRID                                               SOJA0450
      DIMENSION POT(125,2),L(1)                                         SOJA0460
      DATA DOT,BL,MG/1H.,1H ,30/                                        SOJA0470
C                                                                       SOJA0480
      IYY = IYR                                                         SOJA0490
      IF(IYR.LE.0.OR.IYR.GT.121) IYY = 121                              SOJA0500
      OFILL = L(1).NE.BL .OR. L(2).NE.BL                                SOJA0510
      MGJ = MAX0(2,MIN0(IYY/20+1,MG-2))                                 SOJA0520
      MGI = MG-MGJ                                                      SOJA0530
      CALL PPGRID(YMIN,YMAX,IYY,MGI,MGJ,IG,20)                          SOJA0540
C                     SET UP X AND Y SCALES, GRID LINE INTERVALS        SOJA0550
      YR = YMAX-YMIN                                                    SOJA0560
      XR = XMAX-XMIN                                                    SOJA0570
      DY = YR/FLOAT(IYY-1)                                              SOJA0580
      DX=DY*1.666667                                                    SOJA0590
      DX2 = DX/2.                                                       SOJA0600
      IXR = XR/DX+1.5                                                   SOJA0610
      CALL PPGRID(XMIN,XMAX,IXR,MGI,MGI,IG(MGJ+1),12)                   SOJA0620
      MGI = MGI+MGJ                                                     SOJA0630
      MGI = MGJ+1                                                       SOJA0640
C                     IF INT .LE.0, ASSUME THAT ONLY SCALE DATA WAS WANTEDSOJA0650
      IF(INT.LE.0) RETURN                                               SOJA0660
C                     SET UP FUNCTION RANGE, AND CONTOUR INTERVAL       SOJA0670
      FR = FMAX-FMIN                                                    SOJA0680
      FNF = NF                                                          SOJA0690
      DF = FR/FNF                                                       SOJA0700
      XSC = AMAX1(ABS(XMIN),ABS(XMAX))                                  SOJA0710
      IF(FR.GT.0. .AND. XSC.GT.0.) GO TO 130                            SOJA0720
      WRITE(6,131) FMIN,FMAX                                            SOJA0730
  131 FORMAT(19HONULL PLOT,  FMIN =,E15.7,8H, FMAX =,E15.7)             SOJA0740
      RETURN                                                            SOJA0750
  130 XSC = ALOG10(XSC)                                                 SOJA0760
      IF(XSC.LT.0.) XSC=XSC-3.                                          SOJA0770
      XSC = AINT(XSC/3.)*3.                                             SOJA0780
      XSC = -AINT(XSC)                                                  SOJA0790
      XSC = 10.**XSC                                                    SOJA0800
```

```
      DXS = DX*XSC                                                      SOJA0810
C                     PRINT PLOT LEGEND                                 SOJA0820
      WRITE(6,152) YMIN,DF,XSC,YMAX                                     SOJA0830
  152 FORMAT(11X,6HYMIN =,E12.4,12X,18HCONTOUR INTERVAL =,E12.4,12X,    SOJA0840
     * 10HX MULT. FY,E8.2,12X,6HYMAX =,E12.4/)                          SOJA0850
      Y = YMIN                                                          SOJA0860
      YS = Y*YSC                                                        SOJA0870
      II = 2                                                            SOJA0880
      DO 102 I=1,IYY                                                    SOJA0890
      KK  = II                                                          SOJA0900
      II = 3-II                                                         SOJA0910
      OGRID = .FALSE.                                                   SOJA0920
      IF(KGI.GT.NGI) GO TO 106                                          SOJA0930
      IF(I.LT.IG(KGI)) GO TO 106                                        SOJA0940
      OGRID = .TRUE.                                                    SOJA0950
      KGI = KGI+1                                                       SOJA0960
  106 KGJ = 1                                                           SOJA0970
      X = XMIN                                                          SOJA0980
      DO 103 J=1,IXX                                                    SOJA0990
      POT(J,II) = DOT                                                   SOJA1000
      IF (OGRID) GO TO 103                                              SOJA1010
      IF(KGJ.GT.NGJ) GO TO 101                                          SOJA1020
      IF(J.LT.IG(KGJ)) GO TO 101                                        SOJA1030
      KGJ = KGJ+1                                                       SOJA1040
      GO TO 103                                                         SOJA1050
  101 VAL = FCN(X,Y)                                                    SOJA1060
C                     CONVERT FUNCTION VALUE INTO PLOT SYMBOL           SOJA1070
      IF(VAL.GE.1.E+38) GO TO 114                                       SOJA1080
      TV = (VAL-FMIN)/DF+5.                                             SOJA1090
      ATV = AINT(TV+0.05)                                               SOJA1100
      IX = TV                                                           SOJA1110
      IF(OFILL .AND. IX.GE.5 .AND. IX.LE.NF+4 .AND. ABS(TV-ATV).LE.0.05)SOJA1120
     * GO TO 110                                                        SOJA1130
      IX = MIND(NF+5,MAX0(4,IFIX(TV)))                                  SOJA1140
      POT(J,II) = L(IX)                                                 SOJA1150
      GO TO 111                                                         SOJA1160
  110 POT(J,II) = L(1)                                                  SOJA1170
      IF(I.EQ.1) GO TO 111                                              SOJA1180
      IF(POT(J,KK).NE.BL) POT(J,II)=L(2)                                SOJA1190
      GO TO 111                                                         SOJA1200
```

Reproduced from best available copy.

```
114 POT(J,II) = L(3)                                                    SOJA1210
111 CONTINUE                                                            SOJA1220
103 Y=Y+DY                                                              SOJA1230
C                    PRINT A LINE OF PLOT SYMBOLS, AND CORRES. X VALUE  SOJA1240
161 WRITE(6,151) XS,(POT(J,II),J=1,IYY)                                 SOJA1250
151 FORMAT(1X,F8.2,2X,121A1)                                            SOJA1260
    GO TO 112                                                           SOJA1270
112 XS = XS+DXS                                                         SOJA1280
102 X=X+DX                                                              SOJA1290
    RETURN                                                              SOJA1300
    END                                                                 SOJA1310
```

```
      LOGICAL FUNCTION SYNTAX(S1,C1,L1,X,IX,TRT,U,STK,NS,W)              SYNT0010
C                                                                        SYNT0020
C     **********                                                         SYNT0030
C     * SYNTAX *          JOHN ROUSE      IITRI       SEPT 1970          SYNT0040
C     **********                                                         SYNT0050
C                                                                        SYNT0060
C     SCAN STRING, USING A LIST OF SYNTAX RULES.  A USER EXIT IS CALLED SYNT0070
C     AS THE RULES DICTATE.  UPON RETURN, SYNTAX = .TRUE. IF STRING IS  SYNT0080
C     CONSISTENT WITH RULES, OR SYNTAX=.FALSE. IF NOT.                   SYNT0090
C                                                                        SYNT0100
C     S1        STRING TO BE EXAMINED.                                   SYNT0110
C     C1        STARTING CHAR. TO BE EXAMINED.  S1 MUST BE A VARIABLE,   SYNT0120
C               BECAUSE UPON RETURN C1 = 1ST CHAR. BEYOND END OF SCAN.   SYNT0130
C     L1        NO. OF CHARS. TO BE EXAMINED.  L1 MUST BE A VARIABLE,    SYNT0140
C               BECAUSE UPON RETURN L1 = NO. CHAR. LEFT UNEXAMINED.      SYNT0150
C               (IF THE STRING IS OK, C1 OUT = C1 IN + L1 IN, AND        SYNT0160
C               L1 OUT = 0                                               SYNT0170
C     X         LIST OF RULES WHICH DESCRIBE SYNTAX                      SYNT0180
C     IX        1ST RULE NUMBER -- I.E., THE STARTING POINT              SYNT0190
C     TRT       ADDRESS OF CHAR. ATTRIBUTE TABLE -- ONE ENTRY FOR EACH   SYNT0200
C               CHAR.  VALUE OF EACH ENTRY IS CLASS TO WHICH CH. BELONGS,SYNT0210
C     U         USER-PROVIDED ROUTINE, CALLED AS DIRECTED BY THE RULES,  SYNT0220
C               MUST BE OF THE FORM,,                                    SYNT0230
C                    LOGICAL FUNCTION U(S1,C2,L2,K,W)                    SYNT0240
C               WHERE C2 AND L2 DELIMIT A PARTICULAR FIELD AND K IS A    SYNT0250
C               RULES-DETERMINED ACTION CODE.                            SYNT0260
C     STK       TEMPORARY STORAGE VECTOR.                                SYNT0270
C     NS        LENGTH OF STK.                                           SYNT0280
C     W         GENERAL VECTOR FOR USE BY USER-SUPPLIED ROUTINE.  FIRST  SYNT0290
C               "SEVERAL" CELLS ARE FOR STORAGE AS DIRECTED BY THE RULES,SYNT0300
C                                                                        SYNT0310
C     SUBPROGRAMS CALLED,,                                               SYNT0320
C     ISCAN     ISOLATES CHAR. GROUPS (TOKENS) IN INPUT STREAM           SYNT0330
C     U         USER-SUPPLIED ROUTINE (SEE ABOVE)                        SYNT0340
C     SYNTXF    SPECIAL ROUTINE TO RETRIEVE FIELDS OF RULES              SYNT0350
C     RCHXTK, ICHXTK   BASIC CHARACTER EXTRACTION ROUTINES               SYNT0360
C                                                                        SYNT0370
C***  TRT IS A TRANSLATE TABLE (SEE ABOVE), AS REQUIRED BY ISCAN.        SYNT0380
      INTEGER S1(1),TRT(1),X(1),W(1),C1,STK(1)                           SYNT0390
      INTEGER V(8),S,B,C2,C,C0,A,A1,A2,T,F                               SYNT0400
```

Reproduced from best available copy.

```
      LOGICAL U                                                         SYNT0410
      LOGICAL OPF5(3,6)                                                 SYNT0420
      EQUIVALENCE (Y(1),O),(Y(2),OO),(Y(3),A1),(Y(4),A2),(Y(5),A),      SYNT0430
     *            (Y(6),K),(Y(7),T),(Y(8),F)                            SYNT0440
      DATA OPF5/.TRUE.,3*.FALSE.,.TRUE.,.FALSE.,2*.TRUE.,3*.FALSE.,     SYNT0450
     * 2*.TRUE.,.FALSE.,.TRUE.,.FALSE.,2*.TRUE./                        SYNT0460
C***  NWDR IS NO. 'WORDS' PER SYNTAX RULE                               SYNT0470
      DATA NWDR/2/                                                      SYNT0480
C                                                                       SYNT0490
C  EMPTY STACK, SET UP SCAN, POINT TO INITIAL RULE                      SYNT0500
C                                                                       SYNT0510
      S = 0                                                             SYNT0520
      LL = L1+1                                                         SYNT0530
      B = (IX-1)*NWDR+1                                                 SYNT0540
      I = -1                                                            SYNT0550
C                                                                       SYNT0560
C  SCAN -- PICK UP ONE TOKEN (GROUP) FROM STRING.  EITHER TEXT OR DELIM.SYNT0570
C                                                                       SYNT0580
    1 ISAVE = I                                                         SYNT0590
      I = ISCAN(S1,C1,L1,TRT,C2,L2)                                     SYNT0600
C                                                                       SYNT0610
C  APPLY A RULE TO LATEST TOKEN                                         SYNT0620
C                                                                       SYNT0630
    2 CALL SYNTXF(9,Y,X(B))                                             SYNT0640
      GO TO (100,200,300,400,500,505,600,601,700,800,810,850,860),O     SYNT0650
    9 SYNTAX = .FALSE.                                                  SYNT0660
    8 IF(O.EQ.2) RETURN                                                 SYNT0670
      C1 = C1-L2                                                        SYNT0680
      L1 = L1+L2                                                        SYNT0690
      RETURN                                                            SYNT0700
C                                                                       SYNT0710
C  O=1 -- PROCEDURE CALL                                                SYNT0720
C                                                                       SYNT0730
  100 S = S+2                                                           SYNT0740
      IF(S.GT.NS) GO TO 9                                               SYNT0750
C  STACK STRING INFO AND RETURN ADDR.                                   SYNT0760
      STK(S-1) = C1-L2                                                  SYNT0770
      STK(S) = B                                                        SYNT0780
C  RULE=PROCEDURE ADDRESS IS KEPT AS ARGUMENT OF CALL                   SYNT0790
      B = A                                                             SYNT0800
```

```
      GO TO 2                                                           SYNT0810
C                                                                       SYNT0820
C  O=2 -- EXAMINE A TOKEN FOR (NO) MEMBERSHIP IN A SET OF TOKEN TYPES   SYNT0830
C                                                                       SYNT0840
  200 GO TO (300,210,220,230,240),OO                                    SYNT0850
  210 SYNTAX = I .EQ. A1                                                SYNT0860
      GO TO 209                                                         SYNT0870
  220 SYNTAX = I .NE. A1                                                SYNT0880
      GO TO 209                                                         SYNT0890
  230 SYNTAX = A1 .LE. I .AND. I.LE. A2                                 SYNT0900
      GO TO 209                                                         SYNT0910
  240 SYNTAX = I .LT. A1 .OR. I .GT. A2                                 SYNT0920
C                                                                       SYNT0930
C  CHECK VALUE OF 'SYNTAX' -- DETERMINES SUCCESS/FAILURE                SYNT0940
C                                                                       SYNT0950
  209 IF(SYNTAX) GO TO 290                                              SYNT0960
C                                                                       SYNT0970
C  SYNTAX EXAMINATION (OR OTHER TEST) FAILS                             SYNT0980
C                                                                       SYNT0990
  280 CALL SYNTXF(8,Y,X(B))                                             SYNT1000
      B = F                                                             SYNT1010
      IF (B.GT.0) GO TO 2                                               SYNT1020
C  IF NO F-ADDR., POP STACK (RETURN FROM RULE-PROC.) IF STACK NON-EMPTY SYNT1030
  282 IF(S.LE.0) GO TO 9                                                SYNT1040
      B = STK(S)                                                        SYNT1050
      C1 = STK(S-1)                                                     SYNT1060
      L1 = LL-C1                                                        SYNT1070
  283 S = S-2                                                           SYNT1080
      CALL SYNTXF(8,Y,X(B))                                             SYNT1090
      B = F                                                             SYNT1100
      IF(B.LE.0) GO TO 282                                              SYNT1110
      GO TO 1                                                           SYNT1120
C                                                                       SYNT1130
C  SYNTAX EXAMINATION (OR OTHER TEST) SUCCEEDS                          SYNT1140
C                                                                       SYNT1150
  290 CALL SYNTXF(10,Y,X(B))                                            SYNT1160
      IF (K.LE.0) GO TO 291                                             SYNT1170
      IF (.NOT. U(S1,C2,L2,K,W)) GO TO 280                              SYNT1180
  291 B = T                                                             SYNT1190
C  B IS A SUCCESSFUL BRANCH ADDRESS                                     SYNT1200
```

```
  295 IF(B.LE.0) GO TO 292                                              SYNT1210
      IF(O.EQ.2) GO TO 1                                                SYNT1220
      GO TO 2                                                           SYNT1230
C  IF NO T-ADDR., POP STACK IF NON-EMPTY (I.E. RETURN FROM RULE-PROC.)  SYNT1240
  292 IF(S.LE.0) GO TO 8                                                SYNT1250
      B = STK(S)                                                        SYNT1260
      S = S-2                                                           SYNT1270
      GO TO 290                                                         SYNT1280
C                                                                       SYNT1290
C  O=3 -- FORCE GOOD-SYNTAX RESPONSE (UNLESS USER ROUTINE RETURNS FALSE)SYNT1300
C                                                                       SYNT1310
  300 SYNTAX = .TRUE.                                                   SYNT1320
      GO TO 290                                                         SYNT1330
C                                                                       SYNT1340
C  O=4 -- FORCE BAD-SYNTAX RESPONSE                                     SYNT1350
C                                                                       SYNT1360
  400 SYNTAX = .FALSE.                                                  SYNT1370
      GO TO 280                                                         SYNT1380
C                                                                       SYNT1390
C  O=5 -- COMPARE W(A1) WITH A2                                         SYNT1400
C  O=6 -- COMPARE W(A1) WITH W(A2)                                      SYNT1410
C                                                                       SYNT1420
  505 A2 = W(A2)                                                        SYNT1430
  500 IF(W(A1)-A2) 501,502,503                                          SYNT1440
  501 J = 1                                                             SYNT1450
      GO TO 510                                                         SYNT1460
  502 J = 2                                                             SYNT1470
      GO TO 510                                                         SYNT1480
  503 J = 3                                                             SYNT1490
  510 IF (OPE5(J,O)) GO TO 300                                          SYNT1500
      GO TO 400                                                         SYNT1510
C                                                                       SYNT1520
C  O=7 -- SET W(A1) TO SOME FCN OF W(A1) AND A2                         SYNT1530
C  O=8 -- SET W(A1) TO SOME FCN OF W(A1) AND W(A2)                      SYNT1540
C                                                                       SYNT1550
  601 A2 = W(A2)                                                        SYNT1560
  600 GO TO (611,613,615,617,619,621,623,625,627,629,631,633,635,637,   SYNT1570
     *       639,641,643,645),O0                                        SYNT1580
  611 W(A1) = A2                                                        SYNT1590
      GO TO 300                                                         SYNT1600
```

Reproduced from best available copy.

```
613 W(A1) = -A2                                                    SYNT1610
    GO TO 300                                                      SYNT1620
615 W(A1) = ABS(A2)                                                SYNT1630
    GO TO 300                                                      SYNT1640
617 W(A1) = -ABS(A2)                                               SYNT1650
    GO TO 300                                                      SYNT1660
619 W(A1) = W(A1)+A2                                               SYNT1670
    GO TO 300                                                      SYNT1680
621 W(A1) = W(A1)-A2                                               SYNT1690
    GO TO 300                                                      SYNT1700
623 W(A1) = W(A1)*A2                                               SYNT1710
    GO TO 300                                                      SYNT1720
625 IF(A2.EQ.0) GO TO 400                                          SYNT1730
    W(A1) = W(A1)/A2                                               SYNT1740
    GO TO 300                                                      SYNT1750
627 IF (A2.EQ.0) GO TO 400                                         SYNT1760
    W(A1) = (W(A1)-1*A2)/A2                                        SYNT1770
    GO TO 300                                                      SYNT1780
629 IF (A2.LE.0) GO TO 400                                         SYNT1790
    W(A1) = MOD(W(A1),A2)                                          SYNT1800
    GO TO 300                                                      SYNT1810
631 IF (A2.LT.0) GO TO 400                                         SYNT1820
    W(A1) = W(A1)**A2                                              SYNT1830
    GO TO 300                                                      SYNT1840
633 W(A1) = MIND(W(A1),A2)                                         SYNT1850
    GO TO 300                                                      SYNT1860
635 W(A1) = MAXD(W(A1),A2)                                         SYNT1870
    GO TO 300                                                      SYNT1880
637 W(A1) = ISIGN(W(A1),A2)                                        SYNT1890
    GO TO 300                                                      SYNT1900
639 SYNTAX = A2.LE.0                                               SYNT1910
    GO TO 209                                                      SYNT1920
641 SYNTAX = W(A1).GT.0 .OR. A2.GT.0                               SYNT1930
    GO TO 209                                                      SYNT1940
643 SYNTAX = W(A1).GT.0 .AND. A2.GT.0                              SYNT1950
    GO TO 209                                                      SYNT1960
645 SYNTAX = W(A1).GT.0 .AND. A2.LE.0 .OR. W(A1).LE.0 .AND. A2.GT.0 SYNT1970
    GO TO 209                                                      SYNT1980
C                                                                  SYNT1990
C  0=9 -- SET W(A1) TO SOME 'SPECIAL VALUE'                        SYNT2000
```

Reproduced from best available copy.

```
C
  700 GO TO (705,710,720,730,740,750,760,770),00
C  SET W(A1) TO VALUE OF NEXT CHAR.  (FALSE RETURN IF L1.LE.0)
  705 IF(L1.LE.0) GO TO 706
      CALL RCHXTK(J,C1)
      W(A1) = ICHXTK(S1,J)
      C1 = C1+1
      L1 = L1-1
      GO TO 300
  706 W(A1) = 0
      GO TO 400
C  SET W(A1) TO TYPE-CODE OF PRIOR TOKEN
C  (TOKEN PRESUMABLY FOUND VIA TEST FOR TYPE IN/OUT-OF-RANGE)
  710 W(A1) = ISAVE
      GO TO 300
C  SET W(A1) TO START OF REMAINDER OF STRING (AFTER THIS TOKEN)
  720 W(A1) = C1
      GO TO 300
C  SET W(A1) TO LENGTH OF REMAINDER OF STRING
  730 W(A1) = L1
      GO TO 300
C  SET W(A1) TO START OF CURRENT TOKEN
  740 W(A1) = C2
      GO TO 300
C  SET W(A1) TO LENGTH OF CURRENT TOKEN
  750 W(A1) = L2
      GO TO 300
C  SET W(A1) TO ADDRESS OF RULE FOLLOWING CURRENT RULE
  760 W(A1) = B+NWDR
      GO TO 300
C  SET W(A1) TO F=ADDR OF CURRENT RULE
C***  (ANY USER EXIT USED WITH THIS RULE MUST RETURN WITH U=.TRUE.)
  770 CALL SYNTXF(B,Y,)(P))
      W(A1) = F
      GO TO 300
C
C  O=10 -- GO TO RULE AT (T)+W(A1)
C  O=11 -- GO TO RULE AT *+1+W(A1)
C  O=12 -- GO TO RULE WHOSE NUMBER IS AT (T)+W(A1)
C  O=13 -- GO TO RULE WHOSE NUMBER IS AT *+1+W(A1)
```

```
C***  NOTE.. FOR ALL OF THE ABOVE, W(A1) MUST RANGE BETW 0 AND A2        SYNT2410
C                                                                         SYNT2420
  800 CALL SYNTXF(7,Y,X(B))                                               SYNT2430
      B = T                                                               SYNT2440
      GO TO 840                                                           SYNT2450
  810 B = B+NWDR                                                          SYNT2460
  840 IF (W(A1).LT.0 .OR. W(A1).GT.A2) GO TO 400                          SYNT2470
      B = B+W(A1)*NWDR                                                    SYNT2480
      GO TO 295                                                           SYNT2490
  850 CALL SYNTXF(7,Y,X(B))                                               SYNT2500
      B = T                                                               SYNT2510
      GO TO 890                                                           SYNT2520
  860 B = B+NWDR                                                          SYNT2530
  890 IF (W(A1).LT.0 .OR. W(A1).GT.A2) GO TO 400                          SYNT2540
      B = B+W(A1)                                                         SYNT2550
      B = (X(B)-1)*NWDR+1                                                 SYNT2560
      GO TO 295                                                           SYNT2570
      END                                                                 SYNT2580
```

Reproduced from best available copy.

```
        REGNAM                                                          SYNT0010
.                                                                       SYNT0020
.       SPECIAL FIELD UNPACKING ROUTINE DESIGNED SOLELY FOR USE BY      SYNT0030
.       FUNCTION 'SYNTAX' TO DECODE SYNTAX RULES.                       SYNT0040
.                                                                       SYNT0050
.       THE FIELDS ARE DEFINED AS FOLLOWS..                             SYNT0060
.                                                                       SYNT0070
.       FLD   WORD  BIT   LEN   CODE  DESCRIPTION                       SYNT0080
.                                                                       SYNT0090
.       1     1     06    06    C     OPERATION CODE                    SYNT0100
.       2     1     12    06    CO    OPERATION MODIFIER                SYNT0110
.       3     1           06    A1    ARGUMENT OR REGISTER NO. 1        SYNT0120
.       4     1     30    06    A2    ARGUMENT OR REGISTER NO. 2        SYNT0130
.       5     1     18    18    A     PROCEDURE ADDRESS (O=1 ONLY)      SYNT0140
.                                     (NOTE THAT A OVERLAPS A1 AND A2)  SYNT0150
.       6     2     00    12    K     ACTION CODE (USED IF TRUE TEST)   SYNT0160
.       7     2     12    12    T     TRUE ADDRESS                      SYNT0170
.       8     2     24    12    F     FALSE ADDRESS                     SYNT0180
.                                                                       SYNT0190
.       FIELDS OF RULE 'X' ARE STORED IN LIST 'T' AS DIRECTED BY        SYNT0200
.       CONTROL CODE 'I'.                                               SYNT0210
.       IF I=1-8,                                                       SYNT0220
.               SET T(I) TO CONTENTS OF FIELD 'I' IN 'X'.               SYNT0230
.       IF I=9  SET T(1) THRU T(5)                                      SYNT0240
.       IF I=10 SET T(6) AND T(7)                                       SYNT0250
.                                                                       SYNT0260
.       FORTRAN CALL IS..                                               SYNT0270
.               CALL SYNTXF(I,T,X)                                      SYNT0280
.                                                                       SYNT0290
*(0)                                                                    SYNT0300
SYNTXF* LA      A2,*0,B11          . (A2) = I                           SYNT0310
        LA      A3,2,B11           . A3 POINTS TO X                     SYNT0320
        LA      A0,1,B11           .                                    SYNT0330
        ANA,14  A0,1               . A0 POINTS TO T-1                   SYNT0340
        SA,01   A0,ST1             . SAVE ADDR OF T-1                   SYNT0350
        EX      RL-1,A2            . EXECUTE UNPACK INSTR (DEP ON I)    SYNT0360
ST1     SA      A0,0,A2            . STORE 1 FIELD (FOR I=1-5 ONLY)     SYNT0370
        J       4,B11              .                                    SYNT0380
RL      LA,12   A0,0,A3            . IF I=1-5, UNPACK JUST 1 FIELD      SYNT0390
        LA,11   A0,0,A3            .                                    SYNT0400
```

```
        LA,9    A0,0,A3          .                                  SYNT0410
        LA,8    A0,0,A3          .                                  SYNT0420
        LA,1    A0,0,A3          .                                  SYNT0430
        J       T6               . GET FIELD 6 UNSIGNED             SYNT0440
        J       T7               . GET FIELD 7 UNSIGNED             SYNT0450
        J       T8               . GET FIELD 8 UNSIGNED             SYNT0460
        J       T15              . GET FIELDS 1-5 AT ONCE           SYNT0470
        J       T67              . GET FIELDS 6 AND 7 AT ONCE       SYNT0480
T6      LMJ     A1,F6            .                                  SYNT0490
        J       4,B11            .                                  SYNT0500
T7      LMJ     A1,F7            .                                  SYNT0510
        J       4,B11            .                                  SYNT0520
T8      LA      A5,1,A3          . GET FIELD 8                      SYNT0530
        LSSL    A5,24            .                                  SYNT0540
        SZ      A4               .                                  SYNT0550
        LDSL    A4,12            .                                  SYNT0560
        LSSL    A4,1             .                                  SYNT0570
        ANA,14  A4,1             . RETURN FIELD 8 TIMES 2 MINUS 1   SYNT0580
        SA      A4,8,A0          .                                  SYNT0590
        J       4,B11            .                                  SYNT0600
F6      LA      A5,1,A3          . GET FIELD 6 AS IS                SYNT0610
        SZ      A4               .                                  SYNT0620
        LDSL    A4,12            .                                  SYNT0630
        SA      A4,6,A0          .                                  SYNT0640
        J       0,A1             .                                  SYNT0650
F7      LA      A5,1,A3          . GET FIELD 7 TIMES 2 MINUS 1      SYNT0660
        LSSL    A5,12            .                                  SYNT0670
        SZ      A4               .                                  SYNT0680
        LDSL    A4,12            .                                  SYNT0690
        LSSL    A4,1             .                                  SYNT0700
        ANA,14  A4,1             .                                  SYNT0710
        SA      A4,7,A0          .                                  SYNT0720
        J       0,A1             .                                  SYNT0730
T15     LA,12   A4,0,A3          . GET FIELDS 1-5                   SYNT0740
        SA      A4,1,A0          .                                  SYNT0750
        LA,11   A4,0,A3          .                                  SYNT0760
        SA      A4,2,A0          .                                  SYNT0770
        LA,9    A4,0,A3          .                                  SYNT0780
        SA      A4,3,A0          .                                  SYNT0790
        LA,8    A4,0,A3          .                                  SYNT0800
```


Reproduced from best available copy.

```
        SA      A4,A4,10        .                       SYNT0810
        LA,1    A4,0,13         .                       SYNT0820
        SA      A4,5,0          .                       SYNT0830
        J       4,B11           .                       SYNT0840
T67     LMJ     A1,F6           .                       SYNT0850
        LMJ     A1,F7           .                       SYNT0860
        J       4,B11           .                       SYNT0870
        END                                             SYNT0880
```