

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 this document is unlimited.

BEST

AVAILABLE

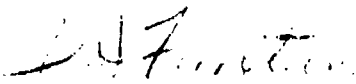
COPY

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		8b. ORIGINATOR'S REPORT NUMBER(S) J-6176	
b. PROJECT NO			
c.			
d.		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.			

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

<u>Section</u>		<u>Page</u>
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
	<u>APPENDIX</u>	
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

<u>Figure</u>		<u>Page</u>
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 (Tritonal 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 (Tritonal 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

<u>Table</u>		<u>Page</u>
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 (Tritonal 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 environmental protection. The model has been programmed for digital computation, 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 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.

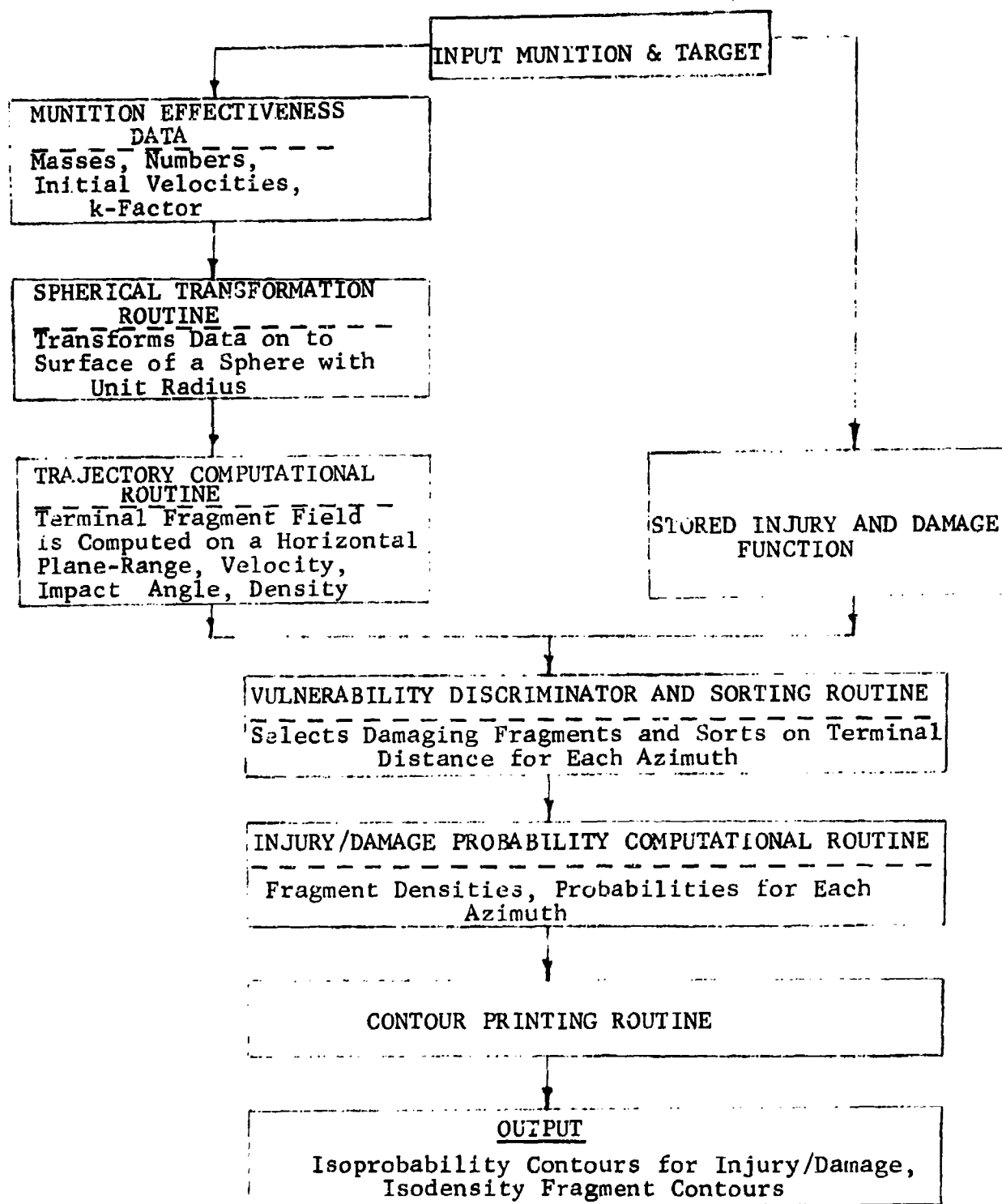


Fig. 1 FLOW CHART FRAGMENT HAZARD MODEL

- (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:

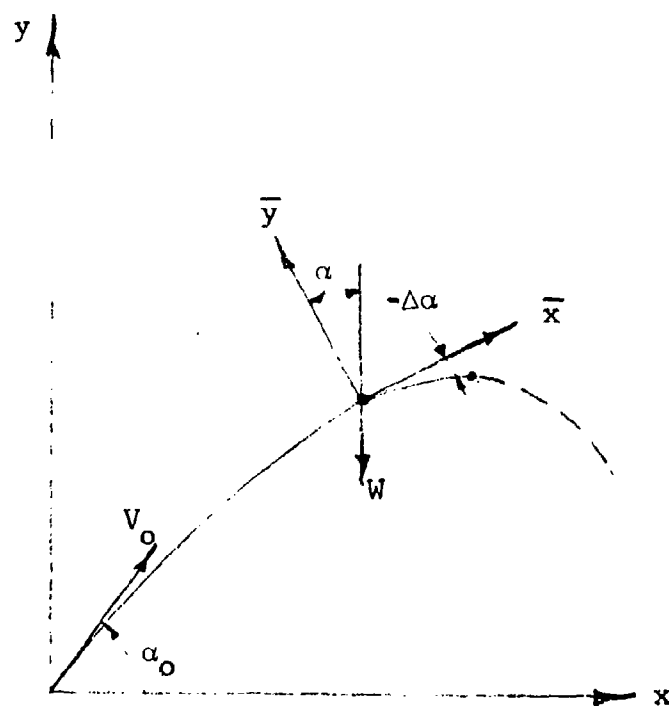


Fig. 2 COORDINATE SYSTEMS AND TRAJECTORY GEOMETRY

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

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

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

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

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

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

In terms of k , the aerodynamic coefficient becomes

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

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

$$\bar{x} = \bar{x}_0 + \bar{x}_p \quad (6)$$

where the basic solution $\bar{x}_0$ satisfies

$$\ddot{\bar{x}}_0 + \beta \dot{\bar{x}}_0^2 = 0 \quad (7)$$

and the initial condition $\dot{\bar{x}}_0 = v$, 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

$$\bar{x}_0 = [\log(1+u)] / \beta \quad (8)$$

$$\dot{\bar{x}}_0 = v_0 / (1+u) \quad (9)$$

where

$$u \equiv \beta v_0 t \quad (10)$$

In these expressions, t is measured from the time at which the fragment is at the local coordinate origin in Fig. 1, and v_0 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:

$$\ddot{\bar{x}}_p + 2\beta \dot{\bar{x}}_0 \dot{\bar{x}}_p + g \sin \alpha = 0 \quad (11)$$

$$\ddot{\bar{y}} + \beta \dot{\bar{x}}_0 \dot{\bar{y}} + g \cos \alpha = 0 \quad (12)$$

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

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

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

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

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

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 $v = V_0$ and $\alpha = \alpha_0$ 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

$$\Delta x = \bar{x} \cos \alpha - \bar{y} \sin \alpha \quad (17)$$

$$\Delta y = \bar{x} \sin \alpha + \bar{y} \cos \alpha \quad (18)$$

while the rotation of the trajectory tangent is given by

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

For low register trajectories, (i.e., those launched at angles less than that corresponding to maximum range) the above analytic perturbation equations with $\alpha = \alpha_0$ 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:

$$x = (\bar{x}_0 + \bar{x}_p) \cos \alpha_0 - \bar{y} \sin \alpha_0 \quad (20)$$

$$y = (\bar{x}_0 + \bar{x}_p) \sin \alpha_0 + \bar{y} \cos \alpha_0 = 0 \quad (21)$$

where $\bar{x}_0$, $\bar{x}_p$, 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_0 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_0 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)	0 - m ₁		m ₁ - m ₂				m ₈ - m ₉		Above m ₉		Total Number of Fragments	Initial Velocity
	W _{AV}	N	W _{AV}	N			W _{AV}	N	W _{AV}	N		
0 - 5	-	-	-	-			-	-	-	-	-	-
5 - 10	-	-	-	-			-	-	-	-	-	-
10 - 15	-	-	-	-			-	-	-	-	-	-
.	.	.	.	.			.	.	.	.	.	.
.	.	.	.	.			.	.	.	.	.	.
.	.	.	.	.			.	.	.	.	.	.
.	.	.	.	.			.	.	.	.	.	.
170 - 175	-	-	-	-			-	-	-	-	-	-
175 - 180	-	-	-	-			-	-	-	-	-	-

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

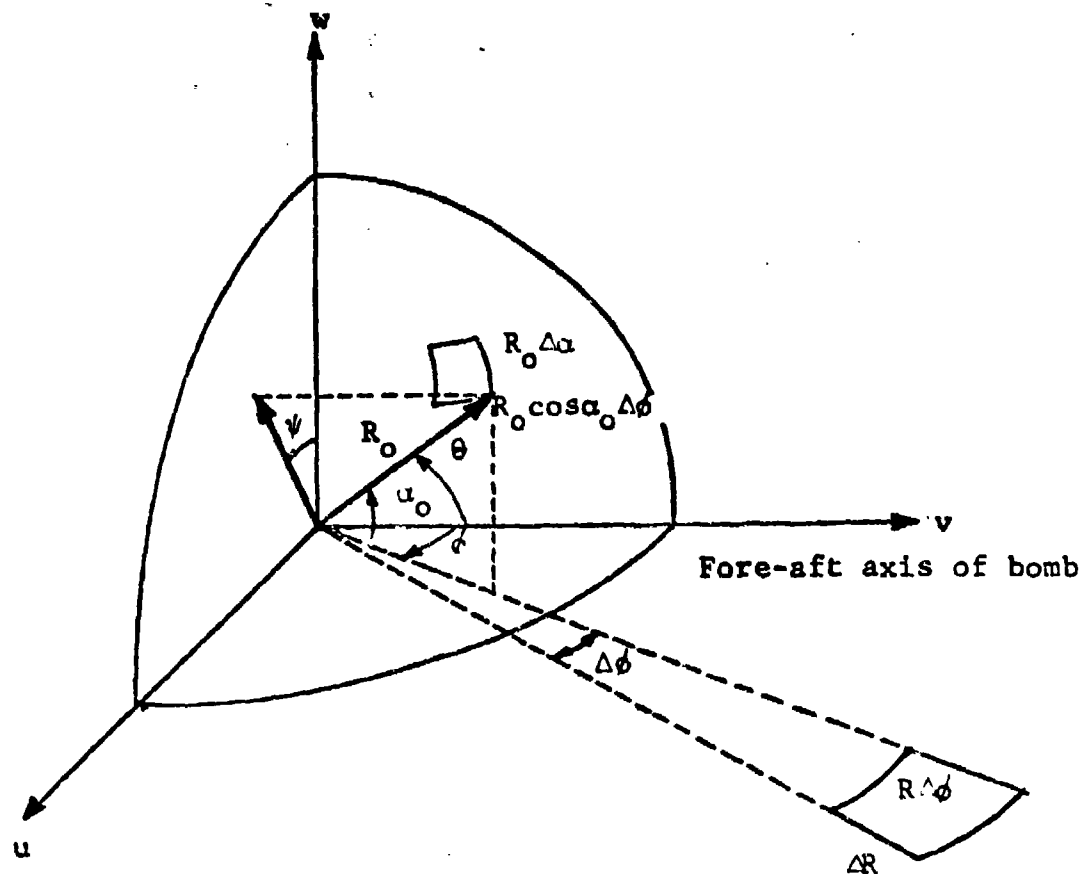


Fig. 3 FRAGMENT SOURCE GEOMETRY

Transformation formulas between the spherical coordinates (θ, ψ) with the munition nose at the pole and the spherical coordinates (α_0, ϕ) with the pole vertical give the following relations:

$$\cos \theta = \cos \alpha_0 \cos \phi \quad (22)$$

$$\cos \psi = \sin \alpha_0 / \sin \theta \quad (23)$$

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_0 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_0, 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 $R_0^2 \cos \alpha_0 \Delta \alpha_0 \Delta \phi$ on the hemisphere are projected into the element of area $R \Delta \phi \Delta R$ on the ground plane. Thus if n_i^0 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

$$n_i = n_i^0 \frac{R_0^2 \cos \alpha_0}{R |dR/d\alpha_0|} \quad (24)$$

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

$$n_i = n_i^0 \frac{R_o^2 \cos \alpha_o}{R |-\sin \alpha_f \frac{dR}{d\alpha_o}|} \quad (25)$$

where α_f is the final trajectory elevation angle.

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

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

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

$$\frac{\partial \theta}{\partial \alpha_o} = \sin \alpha_o \cos \phi / \sin \theta \quad (27)$$

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

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

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

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

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

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

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

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

$$q = 1 - \exp(-\sum \bar{n}_i A_i) \quad (32)$$

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 ± 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

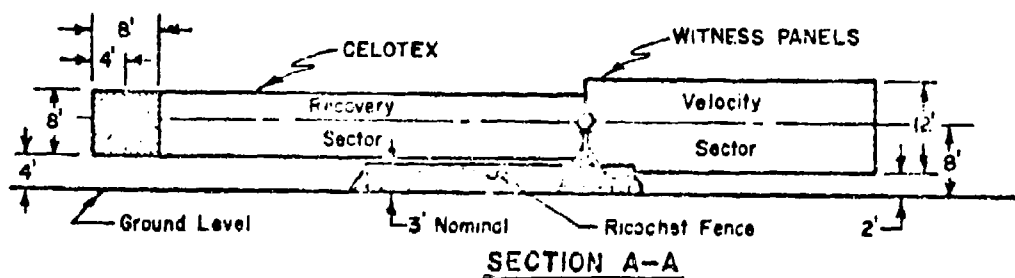
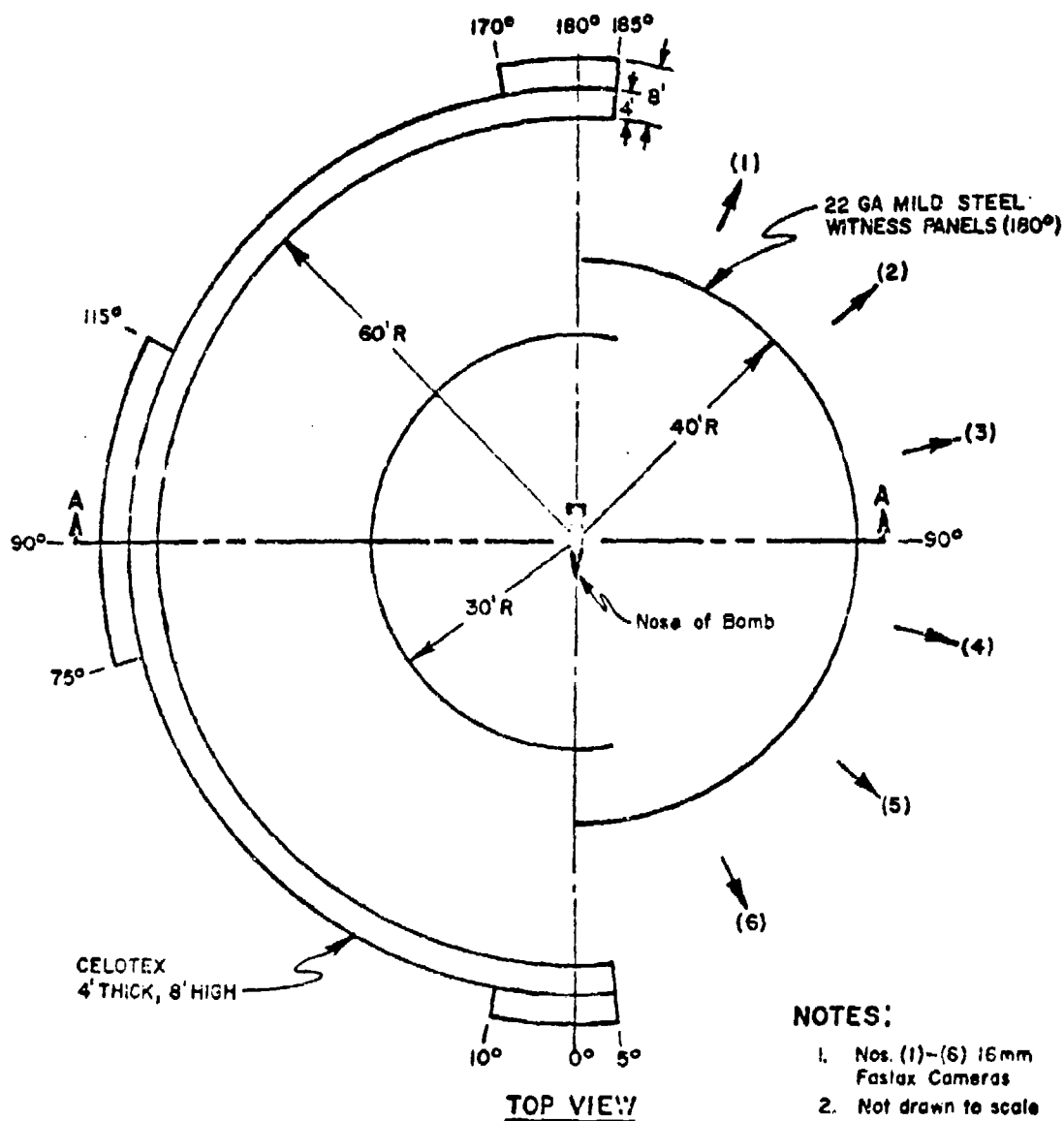


Fig. 4 FRAGMENTATION ARENA DIAGRAM
(500 LB MK 82-1 BOMB)

TABLE 2 ESTIMATED FRAGMENT SAMPLE SIZE DATA FOR GIVEN ARENA TESTS

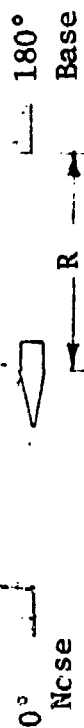
Munition Type	Arena Radius (R) (ft)	Recover Box Height (h) (ft)	90 Degree Polar Zone Region (Side Spray)		
			Zone Angle β (deg)	Sample Size (%)	Mag. Factor (μ)
105 mm Projectile	20	8	22	6.1	16.4
155 mm Projectile	28	8	16	4.4	22.8
175 mm Projectile	31	8	14.5	4.0	25.0
5 in./38 Projectile	17	8	25	2.0	14.3
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

90°

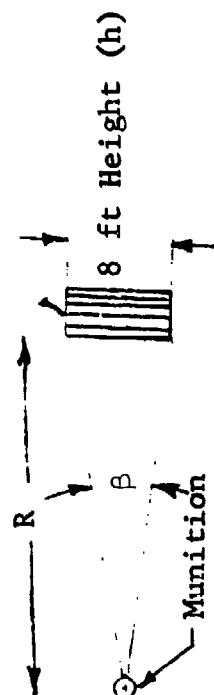
30°

Recovery Boxes

Recovery Box



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 $W = \sum \bar{m}_{ij} f_{ij}$, and the cumulative distribution of weight

$$F_k = \sum_{i=1}^k \frac{\sum \bar{m}_{ij} f_{ij}}{W} \quad k = 1, 2, \dots, N \quad (33)$$

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. The 155 mm Munition

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

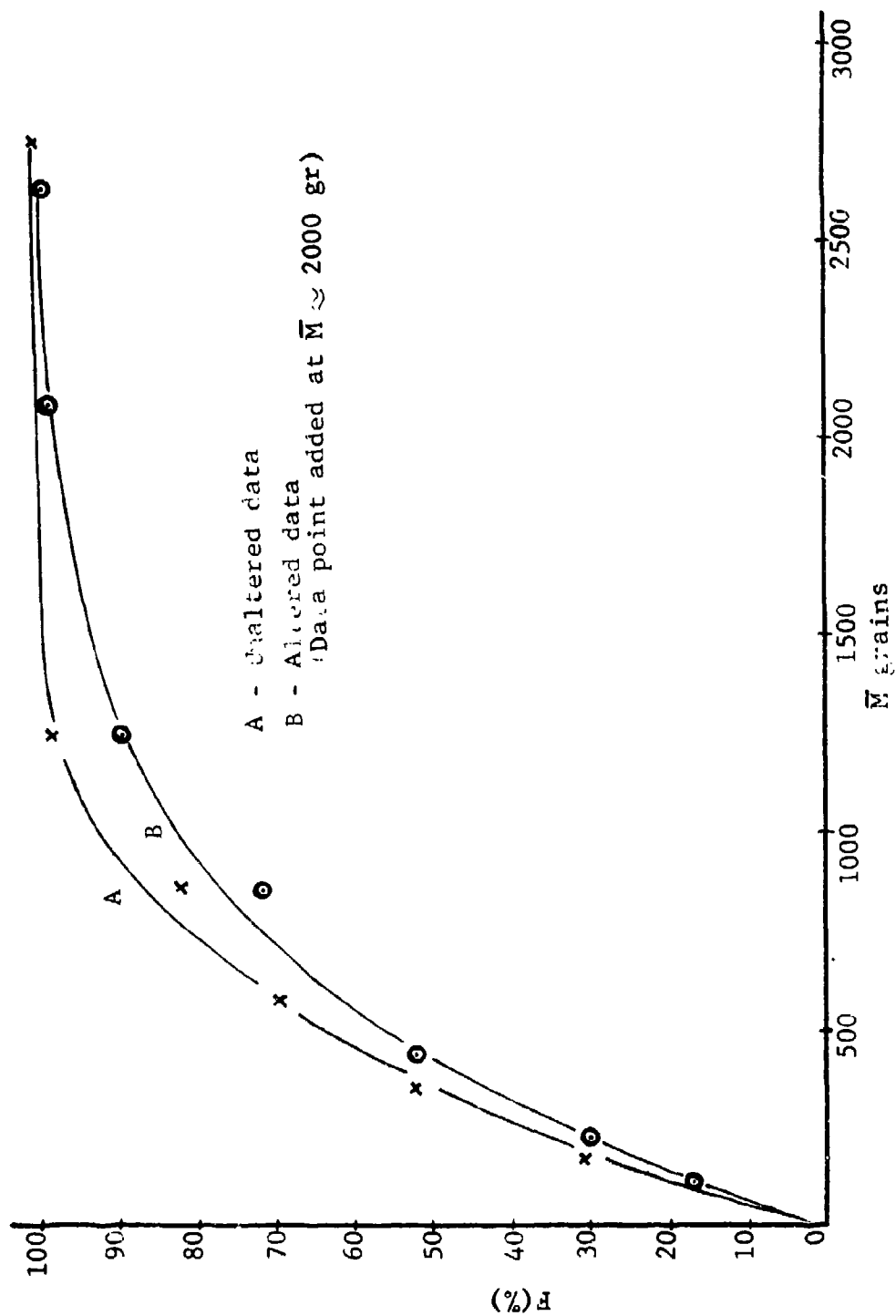


Fig. 5 CUMULATIVE DISTRIBUTION OF MASS FOR THE 155 MM MUNITION

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 sparsely 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 $\sigma_1/\bar{m}_1$ 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 $\bar{m} = 2000$, the dispersion measure is

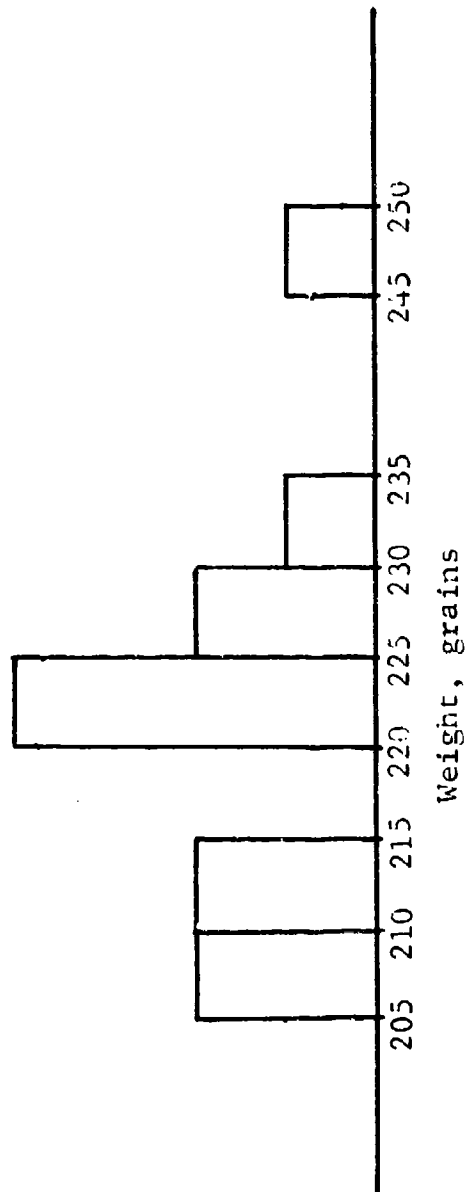


Fig. 6 DISTRIBUTION OF MASS IN THE CATEGORY 200 - 250 GRAINS FOR THE 155 MM MUNITION

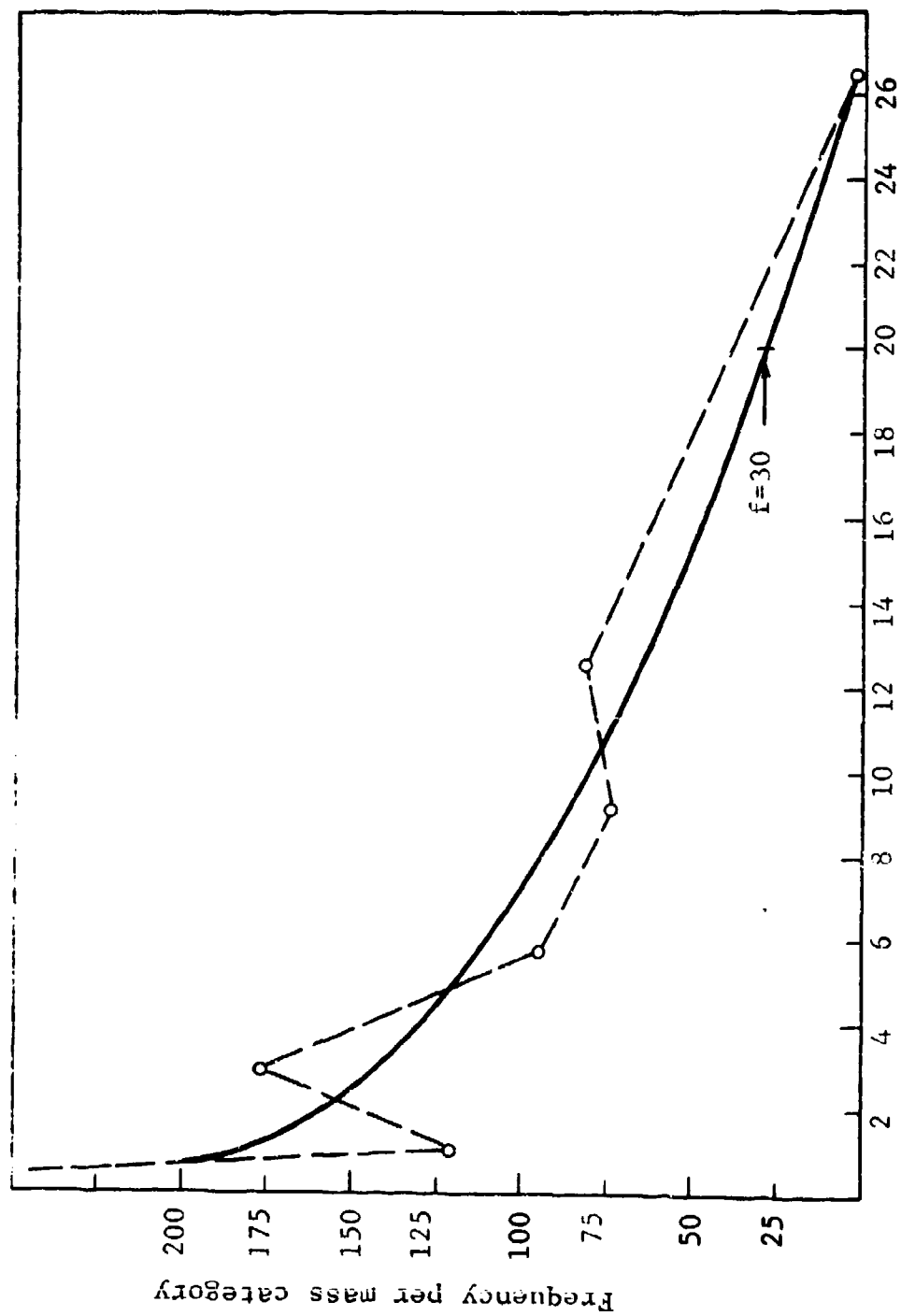


Fig. 7 POLAR ZONE FREQUENCY VERSUS MASS CATEGORY

estimated as $.061 \times 2000 = 122$. 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:

$$\begin{array}{ll}
 \text{Mass Category } \bar{m} & = 2000 \text{ grains} \\
 \text{Frequency } f & = 30 \\
 \text{Polar zones receiving entries} & = 7 \text{ and } 11 \\
 \left. \begin{array}{l} \text{The frequency in previous} \\ \text{mass category} \\ \bar{m} = 1230 \end{array} \right\} & = 79 \\
 \left. \begin{array}{l} \text{Distribution of mass in} \\ \text{zones 7 and 11 for} \\ \bar{m} = 1230 \end{array} \right\} & = \begin{cases} 11/79 \text{ for zone 7} \\ 68/79 \text{ for zone 11} \end{cases} \\
 \left. \begin{array}{l} \text{Distribution of mass in} \\ \text{zones 7 and 11 for} \\ \bar{m} \approx 2000 \end{array} \right\} & = \begin{cases} 11/79 \times 30 \approx 4, \text{ zone 7} \\ 68/79 \times 30 \approx 26, \text{ zone 11} \end{cases}
 \end{array}$$

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

$$\begin{array}{l}
 \bar{m} + \sigma = 2122 \\
 \bar{m} - \sigma = 1878
 \end{array}$$

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

$$\sum f_{ij} m_{ij} = T_1 \quad (34)$$

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

$$\sum f_{ij} m_{ij} + \sum f_{ij}^* m_{ij}^* = T_1 + T_2 \quad (35)$$

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} ,

$$\sum f_{ij} m_{ij} + \sum f_{ij}^* m_{ij}^* = T. \quad (36)$$

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} ,

$$\sum_{i=1}^k p_{ij} f_{ij} m_{ij} + \sum_{i=k}^N p_{ij} f_{ij}^* m_{ij}^* = T_1 \quad (37)$$

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

$$\frac{\sum p_{ij} f_{ij} m_{ij}}{\sum f_{ij} m_{ij}} + \frac{\sum p_{ij} f_{ij}^* m_{ij}^*}{\sum f_{ij} m_{ij}} = 1 \quad (38)$$

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

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

$$\bar{p} + \frac{\sum_{i=k}^N p_{ij} f_{ij}^* m_{ij}^*}{\sum_{i=1}^k f_{ij} m_{ij}} = 1 \quad (39)$$

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

$$\bar{p} + \bar{p} \frac{\sum \sum f_{ij}^* m_{ij}^*}{\sum \sum f_{ij} m_{ij}} = 1 \quad (40)$$

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

and

$$\bar{p} = \frac{T_1}{T_1 + T_2} \quad (41)$$

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. The 500 lb Bomb

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 this 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.

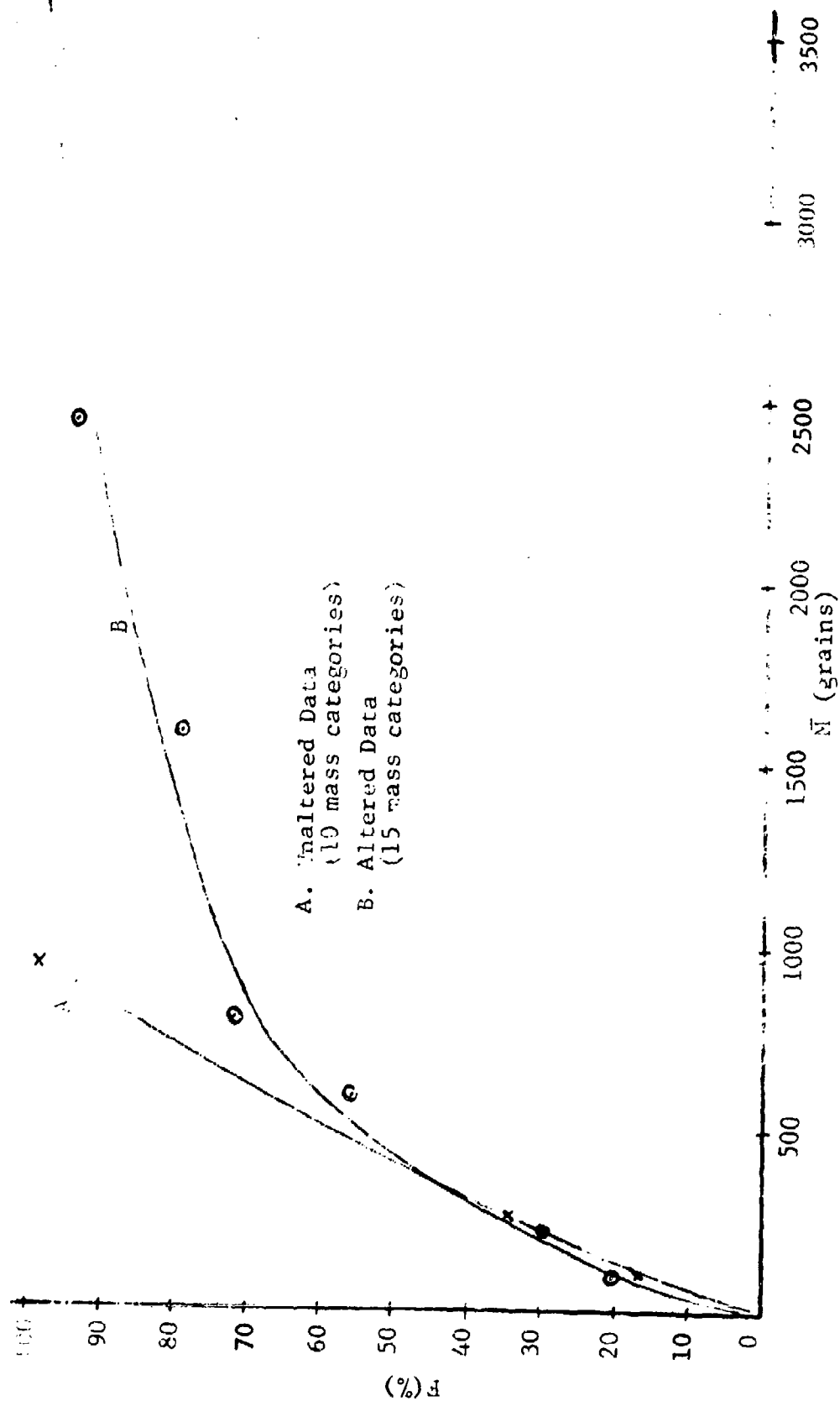


Fig. 8 CUMULATIVE DISTRIBUTION FOR 500 LB BOMB

<u>Category</u>	<u>Range</u>
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 $\sigma/\bar{m}$ 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 without 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 +

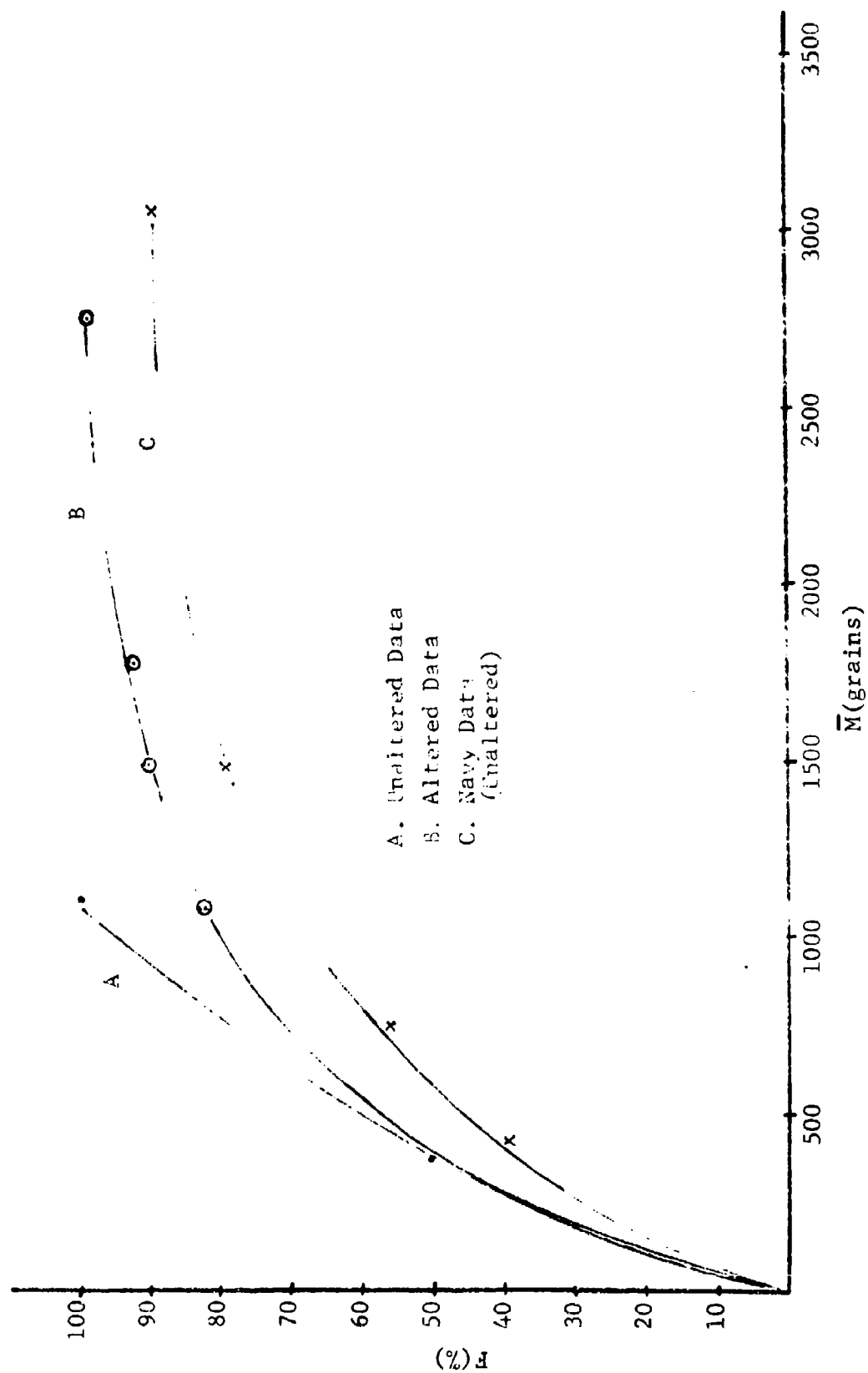


Fig. 9 CUMULATIVE DISTRIBUTION FOR 750 LB BOMB

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.

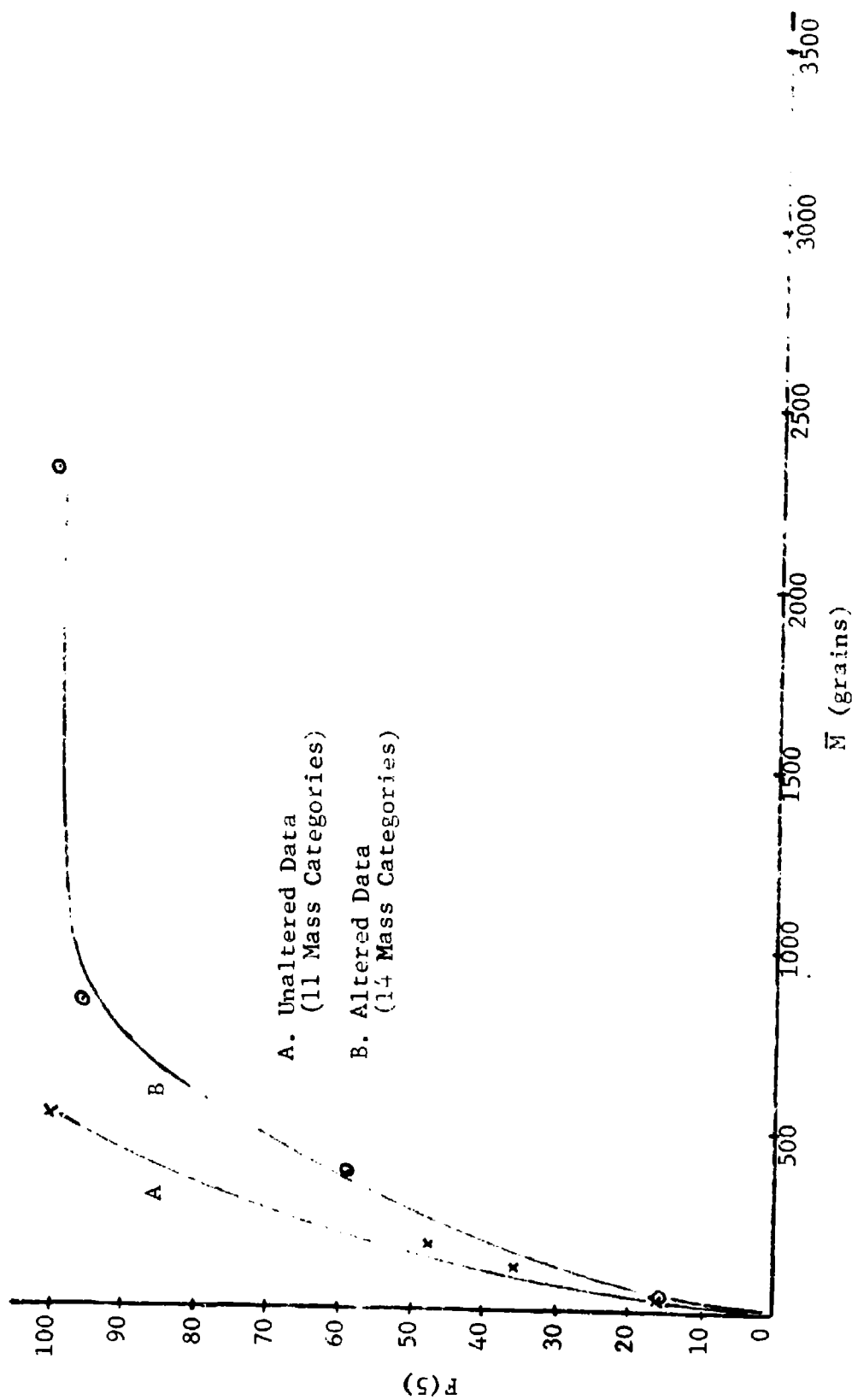


Fig. 10 CUMULATIVE DISTRIBUTION OF MASS FOR THE 5 IN. MUNITIONS

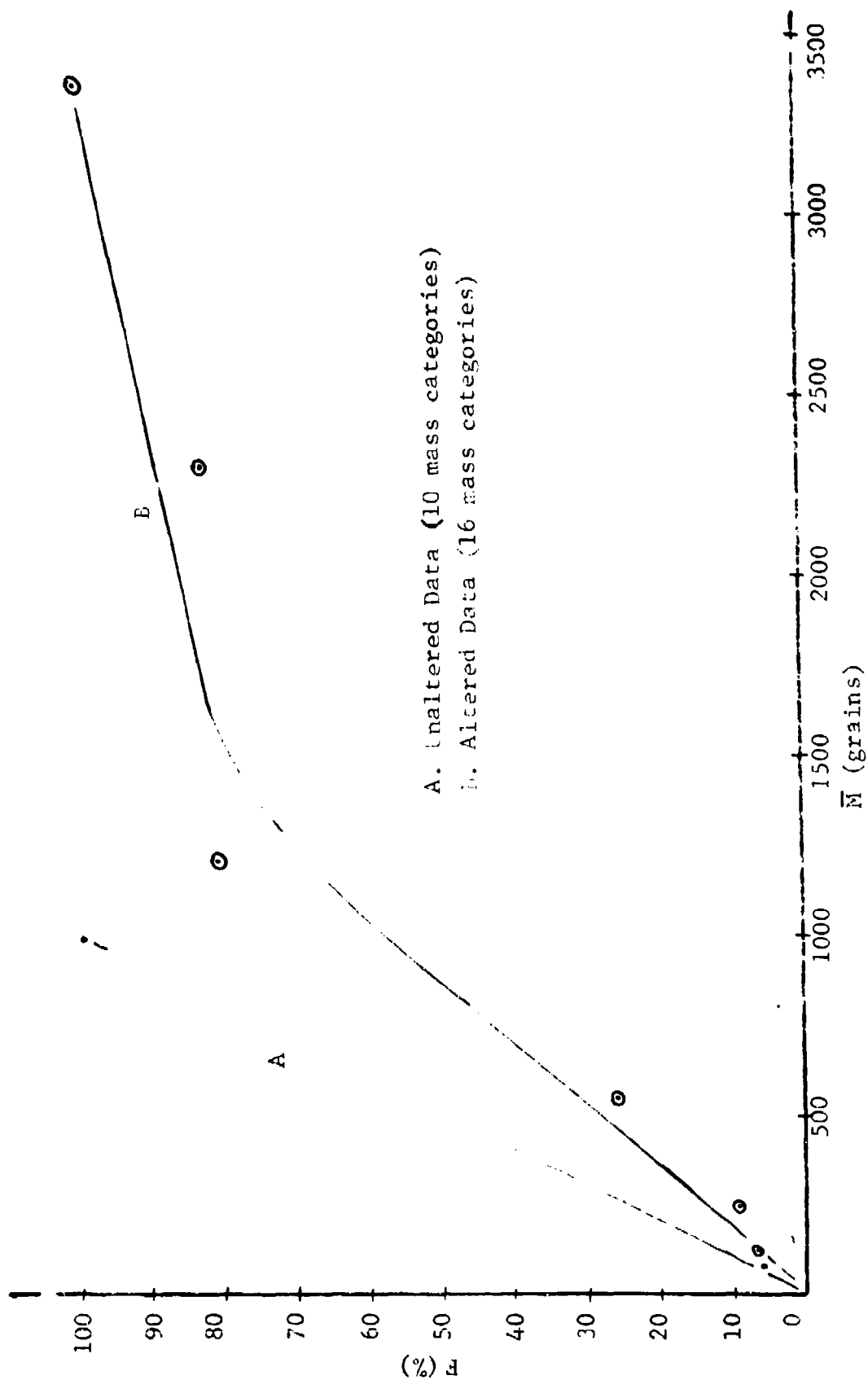


Fig. 11 CUMULATIVE DISTRIBUTIONS OF MASS FOR THE 8 IN. MUNITIONS

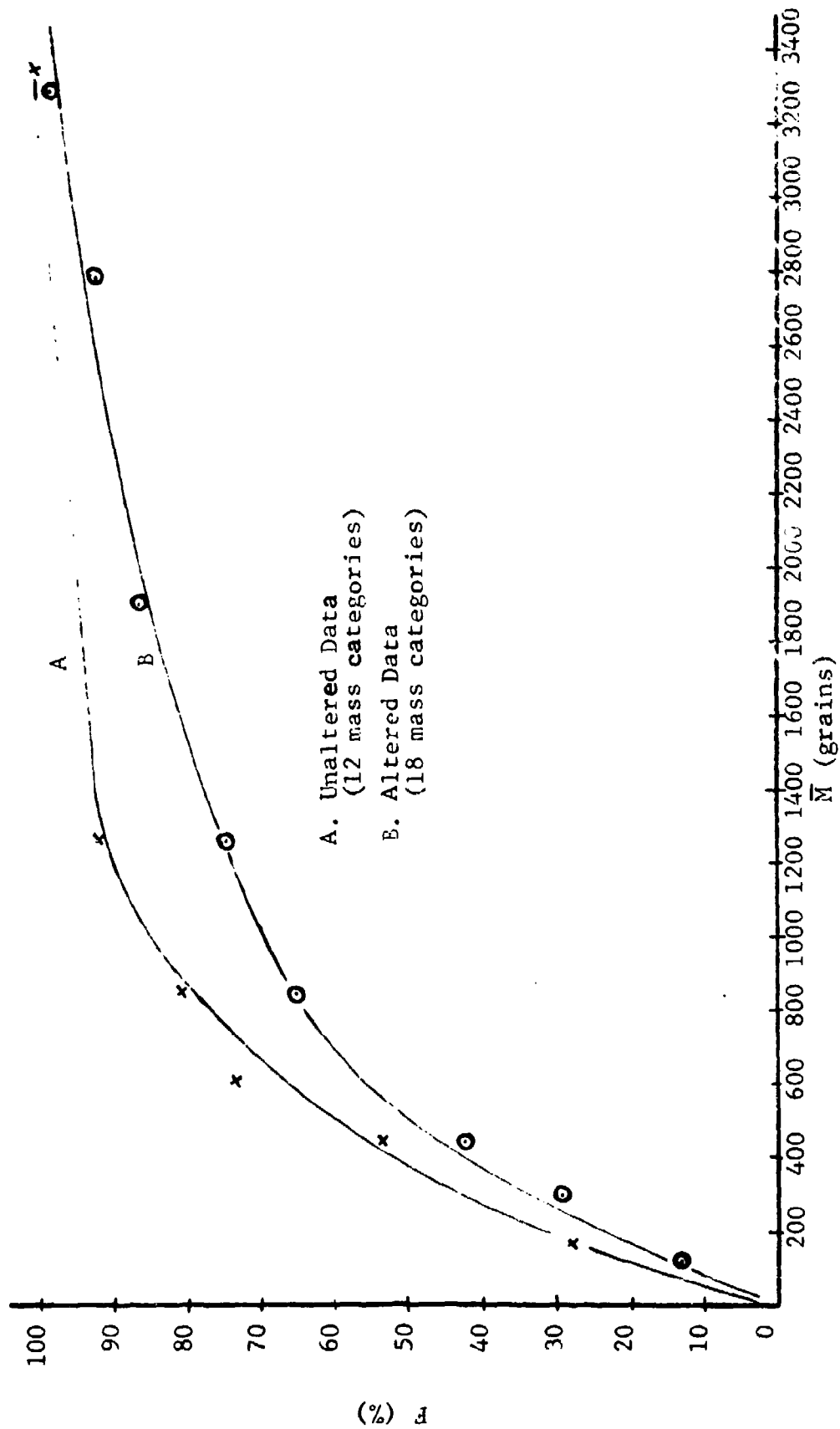


Fig. 12 CUMULATIVE DISTRIBUTION OF MASS FOR THE 175 MM MUNITIONS

Reproduced from
best available copy.

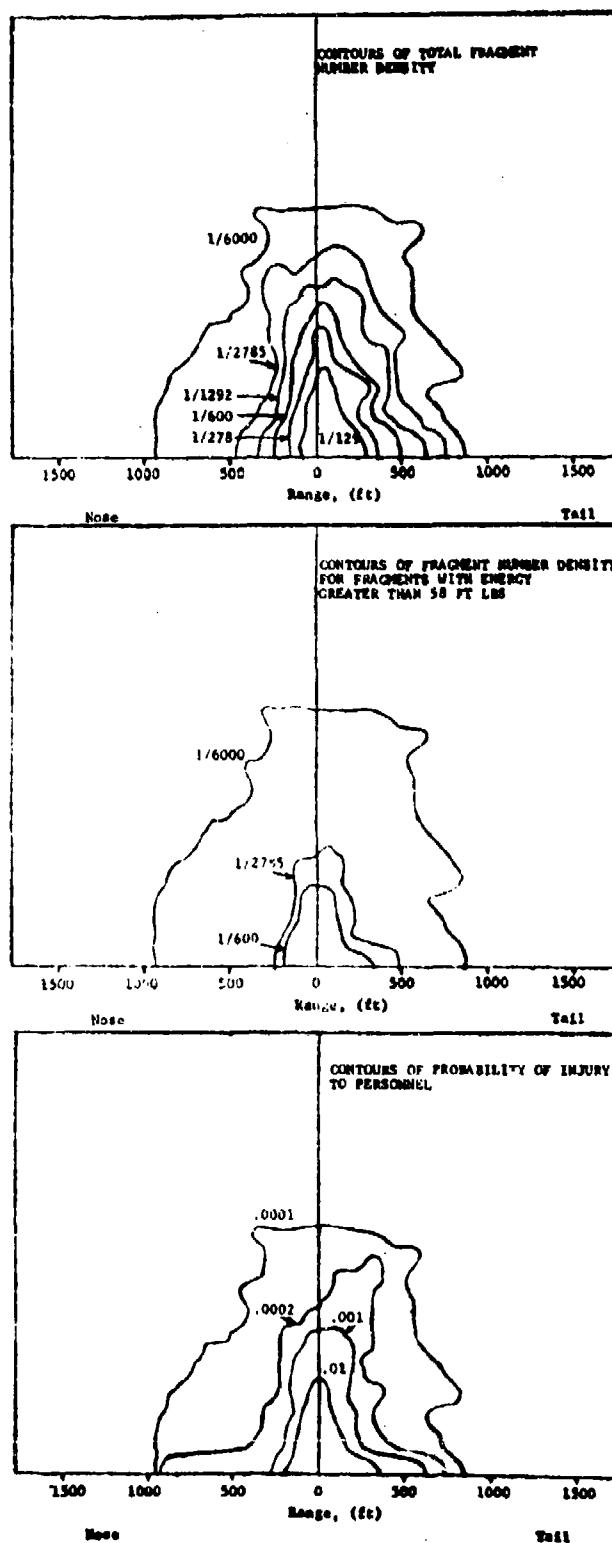


Fig. 13 175 MM GUN SHELL M437A2 (COMP. B LOAD)
(UNALTERED DATA)

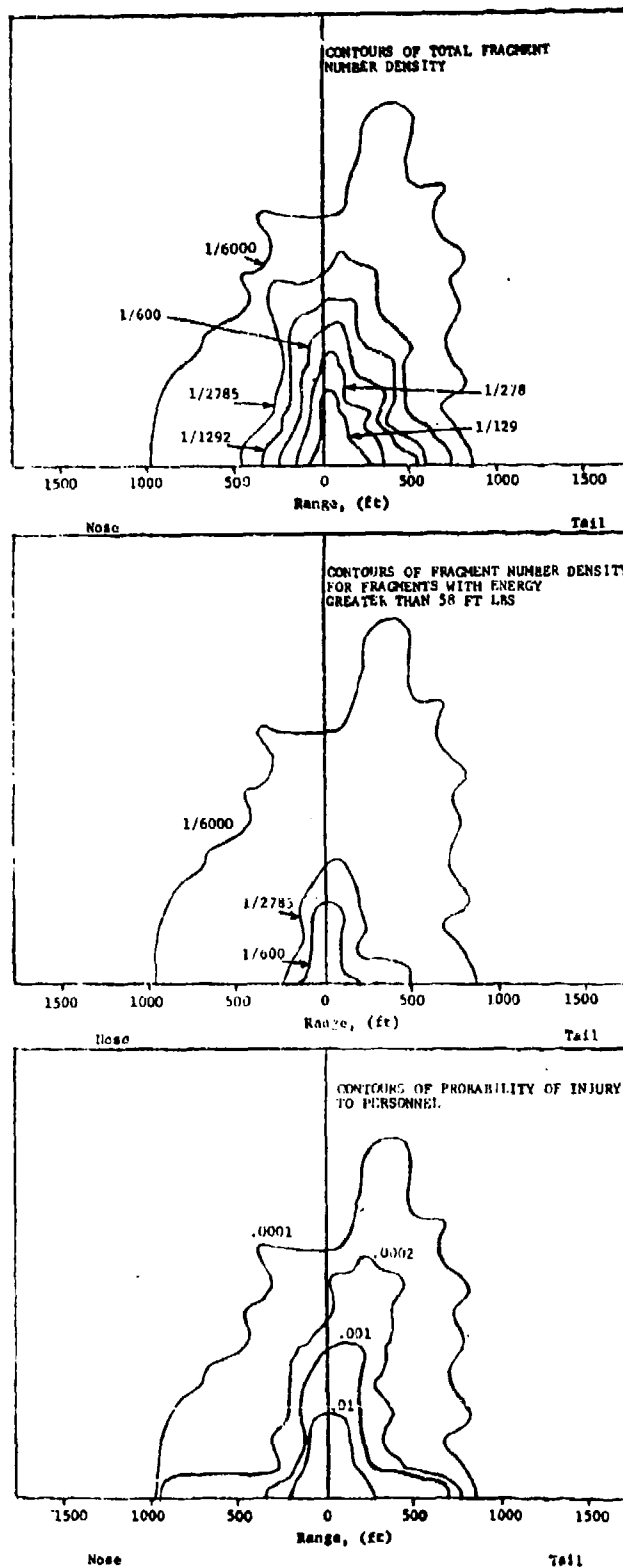


Fig. 14 175 MM GUN SHELL M437A2 (COMP. B LOAD)
(ALTERED DATA)

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 ft² 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	<u>All Fragments</u>			<u>Hazardous Fragments</u> **		
	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:

\$TAPE n {BIN, BCD} [,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 } [,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 XSRCH, 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 sentinel 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 (fuz., 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.
11-16	I5	IDORD	Ordnance I.D. number 0 ≤ IDORD ≤ 99. (ignored)
17	A1	IVO	Ordnance modification code. If left blank, IVO='0' is assumed. (ignored)
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

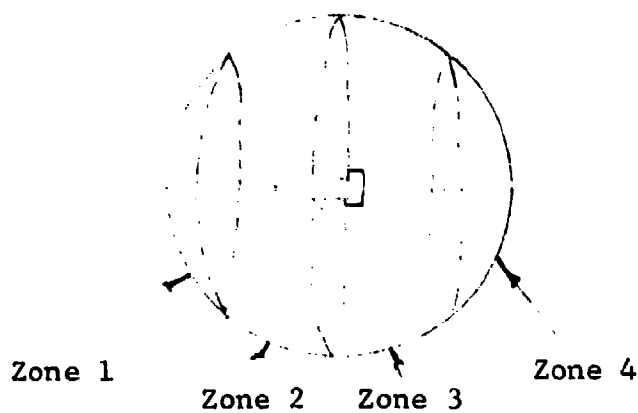
Data of format 2 is split into two parts.

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.

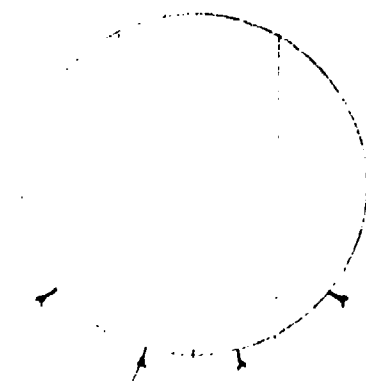
*[X] means the smallest integer not less than X.

Examples: [3.14]=4, [14]=14, [7.00001]=[7.9999]=8.

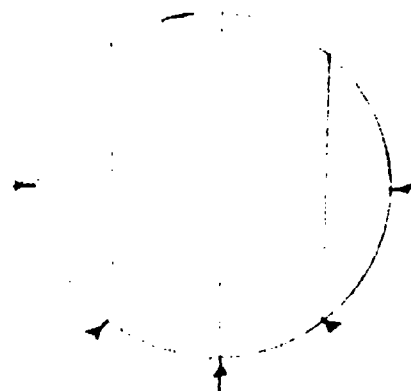
**Fragment number information in the form of a count of the total number of fragments in a polar zone may be used, provided that the NORM parameter is specified on the \$ORDNANCE statement.



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.

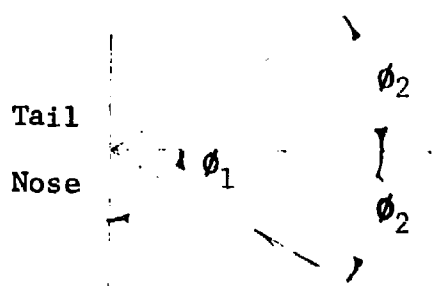
NEL= n Number of elevation angles to be considered (high register fragments), NEL $\leq$ 18.

NAX= n number of azimuth rays in fragment field (a semi-circle), NAX $\leq$ 36.

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

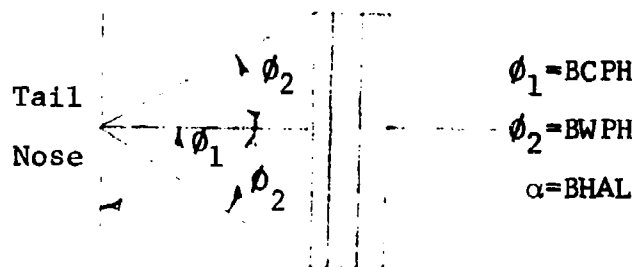
- A title card. Columns 1-72 contain a description of the fragment field.
- If NBAR $>$ 0, NBAR barrier description cards are read. The format of each card is:

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.



α

Circular Barrier



α

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/ft²)
- 04 total fragment kinetic energy (ft-lb/ft²)
- 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
<u>Heading Card</u>			
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
<u>Heading Card</u>			
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 $NF \leq 14$, parameters not read are set to zero.

COLS.	FORMAT	NAME	DESCRIPTION
-------	--------	------	-------------

Title Card

1-72	12A6	(ITTLE(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) minimum 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 (ft²)
- F(9) average target elevation area (ft²)
- 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. $NZ \leq 50$
------	----	----	--

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). $NX \leq 81$.
------	----	----	---

11-15	I5	NY	number of resolution increments in Y direction (laterally from waist of munition). Normally $NY = (NX - 1) / 2 + 1$. $NX \leq 41$.
-------	----	----	--

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.)

\$BODM AAZ=18,ORNG=250.,NRNG=8 3 2000 FOOT RADIUS, 10 DEG AZ SECTORS
 GROUND LEVEL FRAGMENT FIELD
 \$TIME
 \$TAPE 4,WEOF
 \$TAPE 4,REWIND
 \$OUTPUT SQUAC 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.F. 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 $R_i = \text{DRNG} * i$, out to $R_i = R_{\text{max}}$, 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 $\Delta\alpha = 90^\circ / \text{NEL}$, and where $\alpha_{\text{max}} < \alpha_i < 90^\circ$ 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

$$T = A * \text{NM} * \text{NAZ} * (1 + B / \text{DRNG} + C * \text{NEL})$$

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)

$$T = 7 \times \text{NM}$$

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.

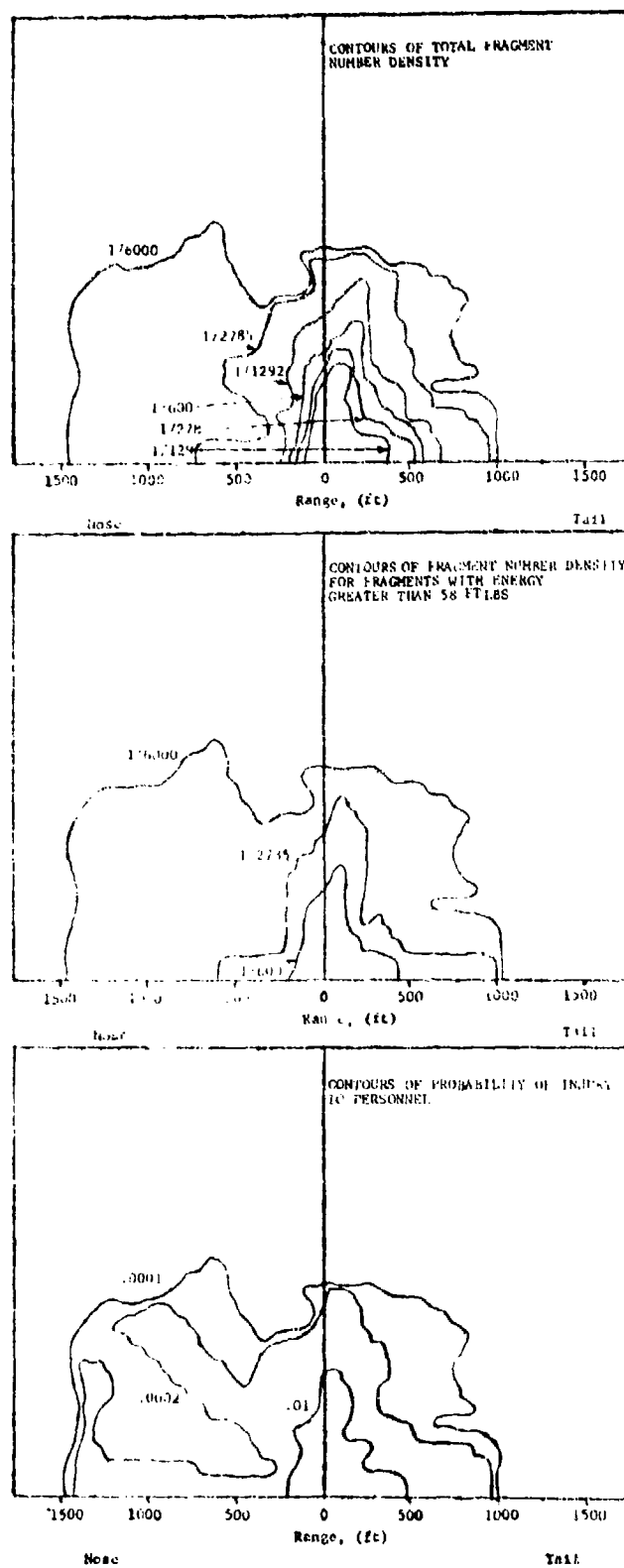


Fig. B1 500 LB LOW DRAG BOMB MARK 82 MOD 1
(H-6 LOAD)

Reproduced from
best available copy.

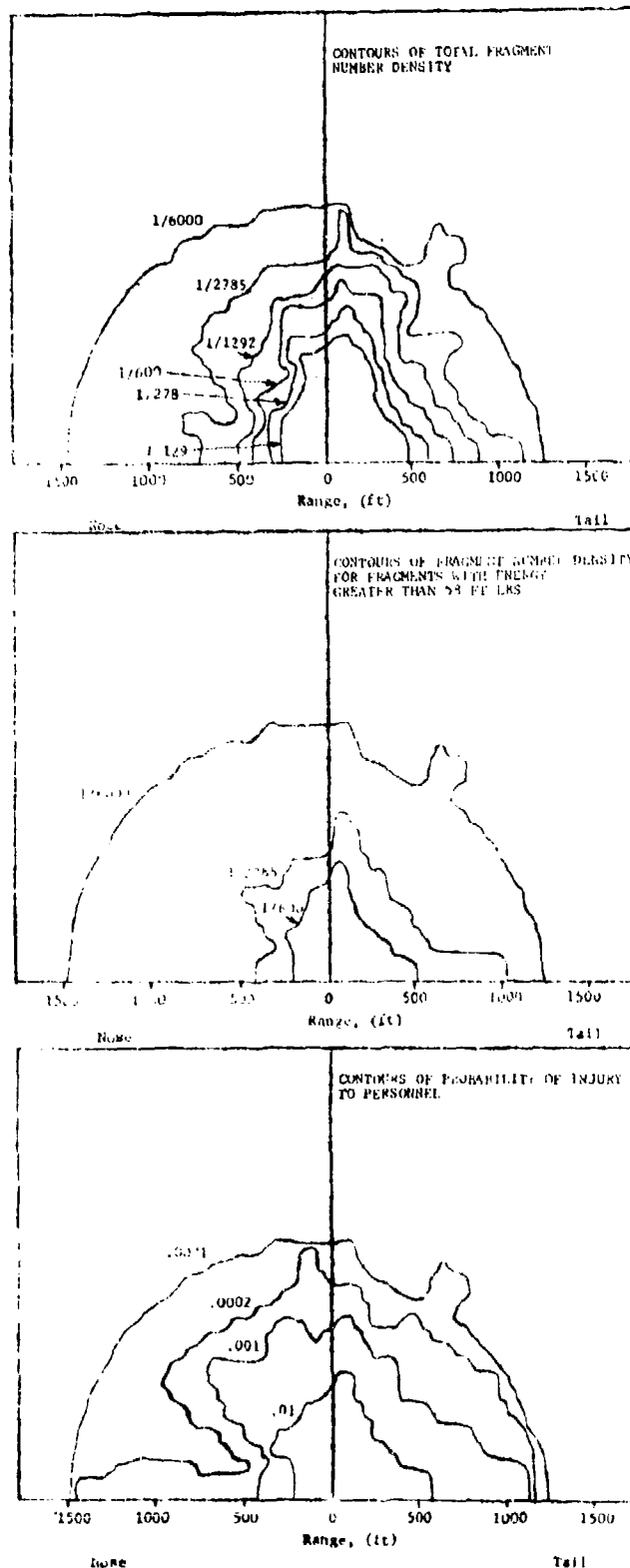


Fig. B2 750 LB BOMB M117A2
(TRITITIONAL LOAD)

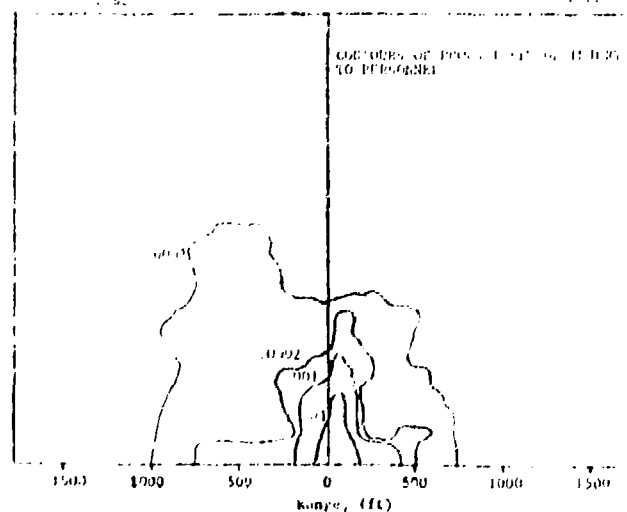
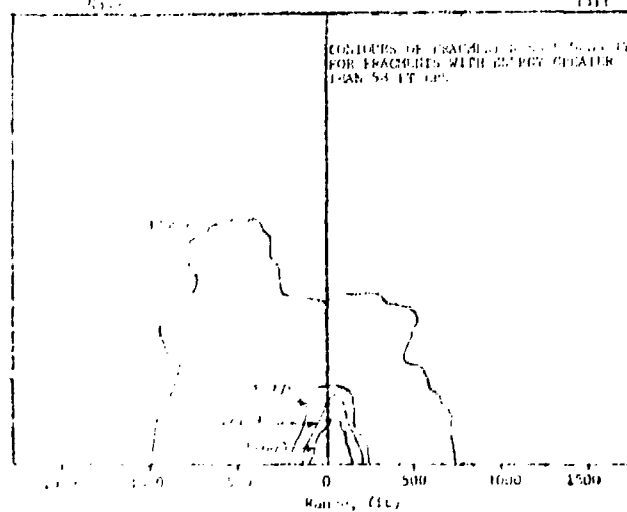
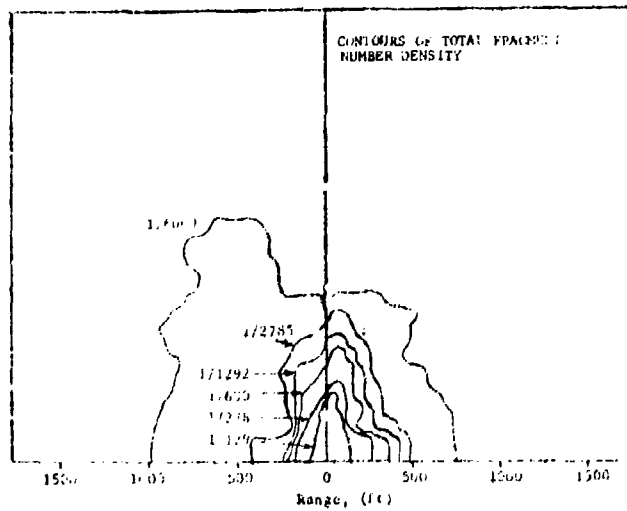


Fig. B3 105 MM HOWITZER SHELL M1
(COMP. B LOAD)

Reproduced from
best available copy.

Reproduced from
best available copy.

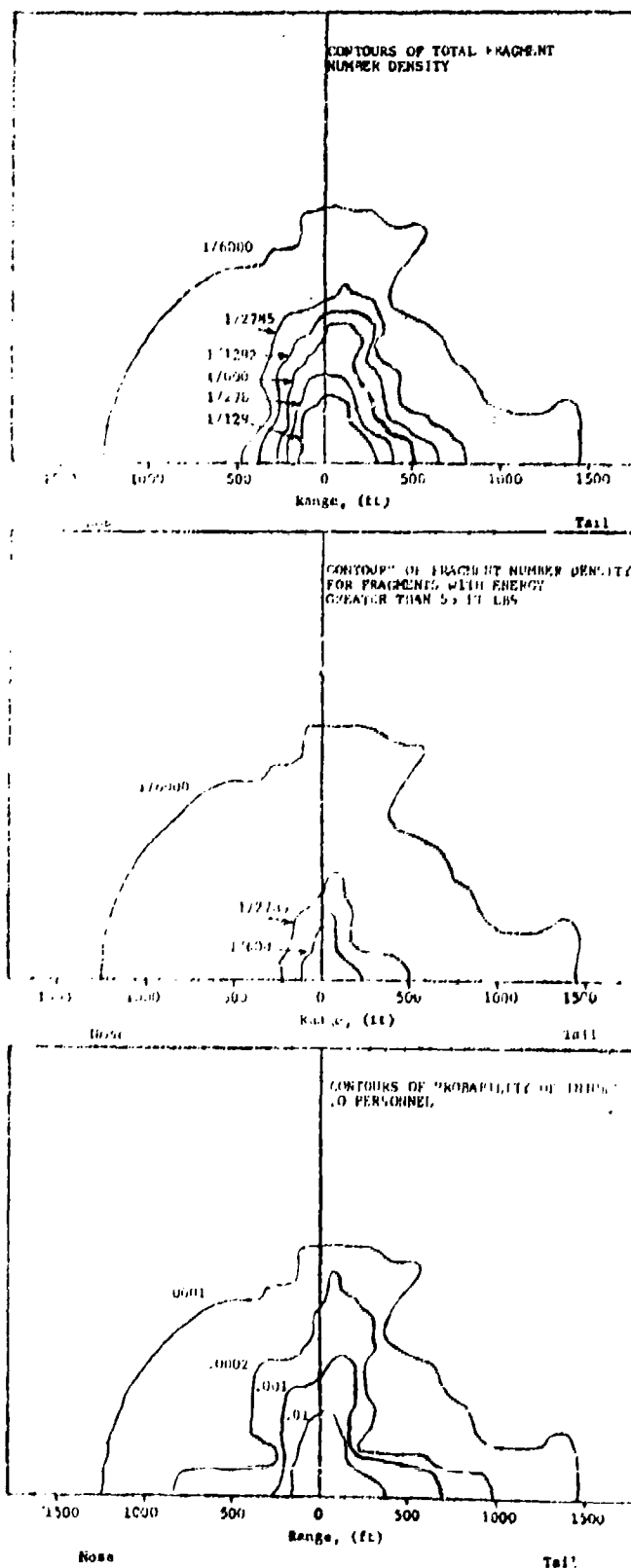


Fig. B4 155 MM HOWITZER SHELL M107
(COMP. B LOAD)

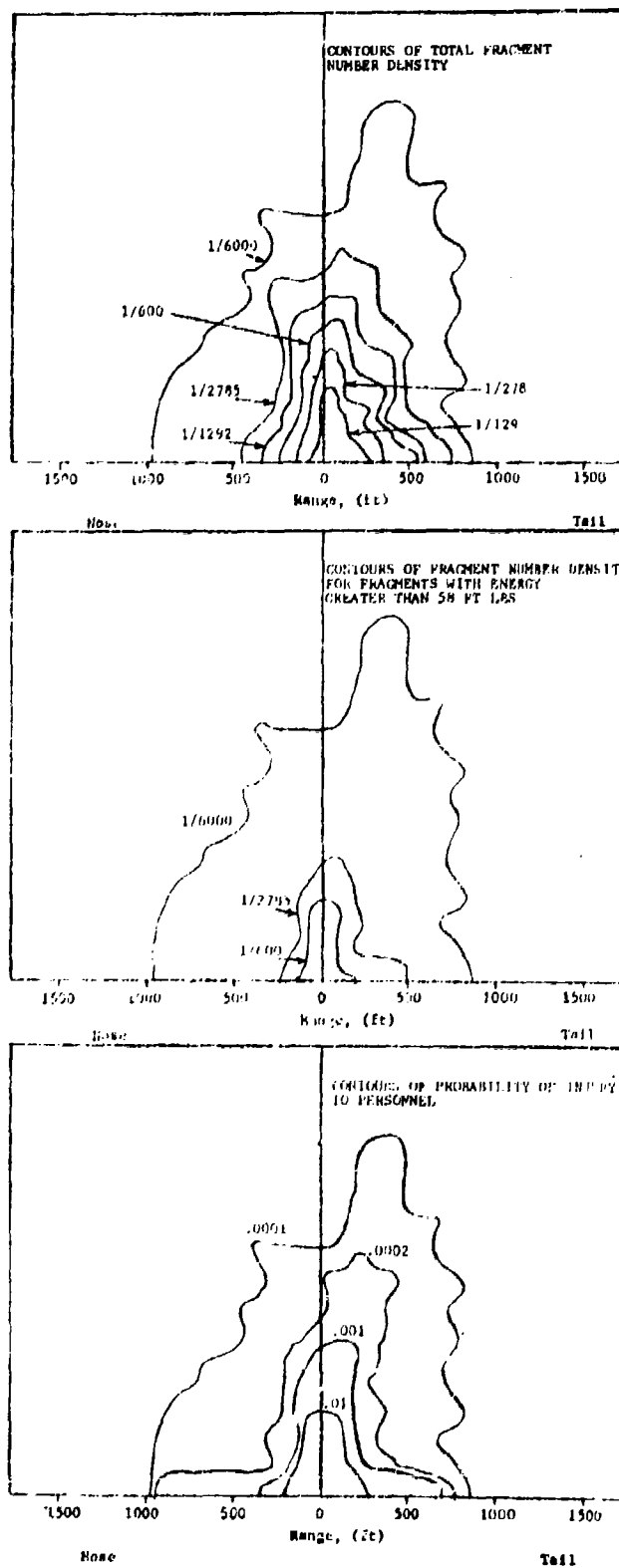


Fig. B5 175 MM GUN SHELL M437A2
(COMP. B LOAD)

Reproduced from
best available copy.

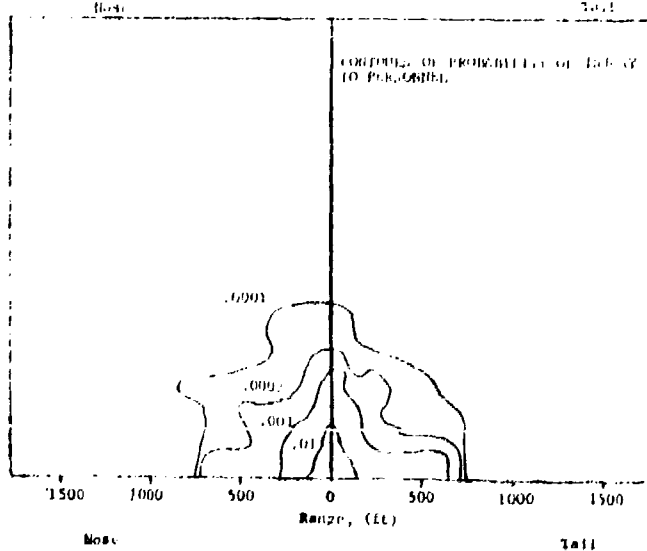
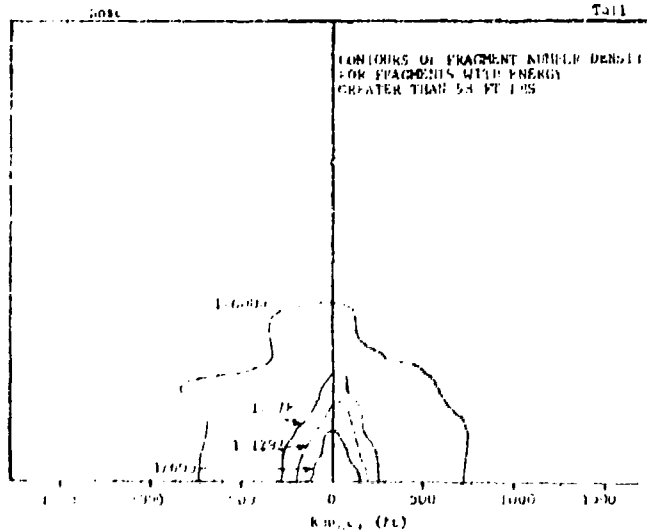
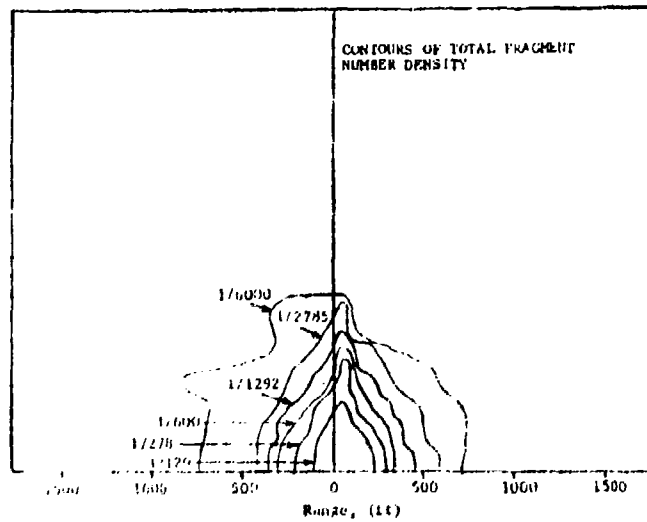


Fig. B6 5"/38 PROJECTILE MARK 49
(COMP. A3 LOAD)

Reproduced from
best available copy.

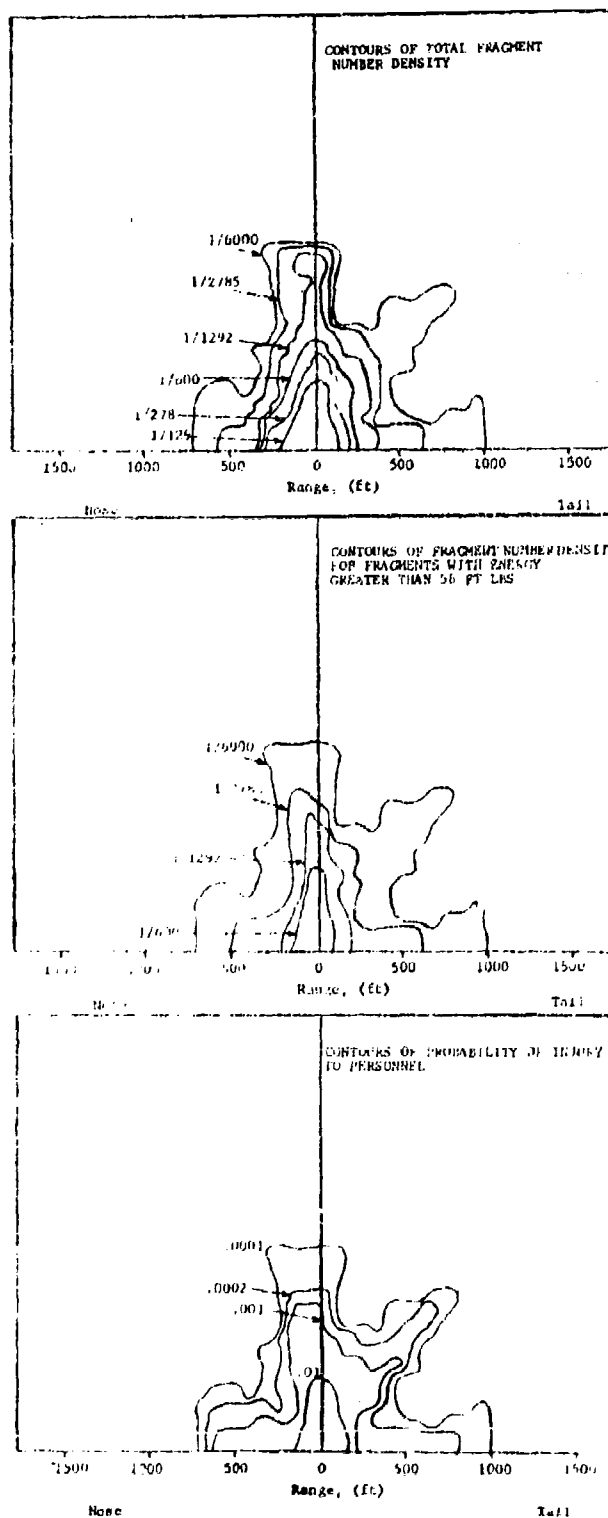


Fig. B7 8"/55 PROJECTILE MARK 25
(EXPLOSIVE D LOAD)

APPENDIX C

AVERAGE CURVES OF FRAGMENT NUMBER DENSITIES
AND INJURY PROBABILITIES

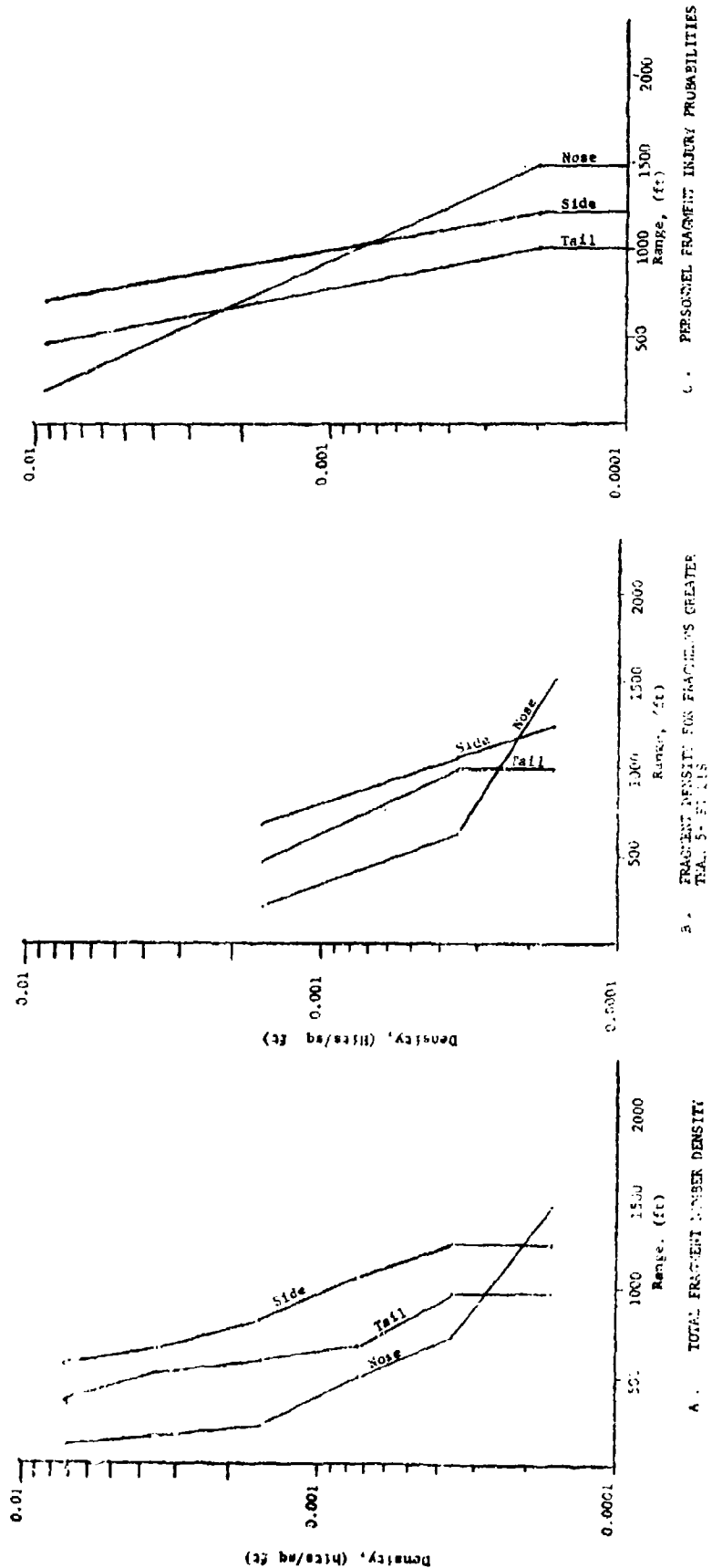


Fig. C1 500 LB LOW DRAG BOMB MARK 82 MOD 1 (H-6 LOAD)

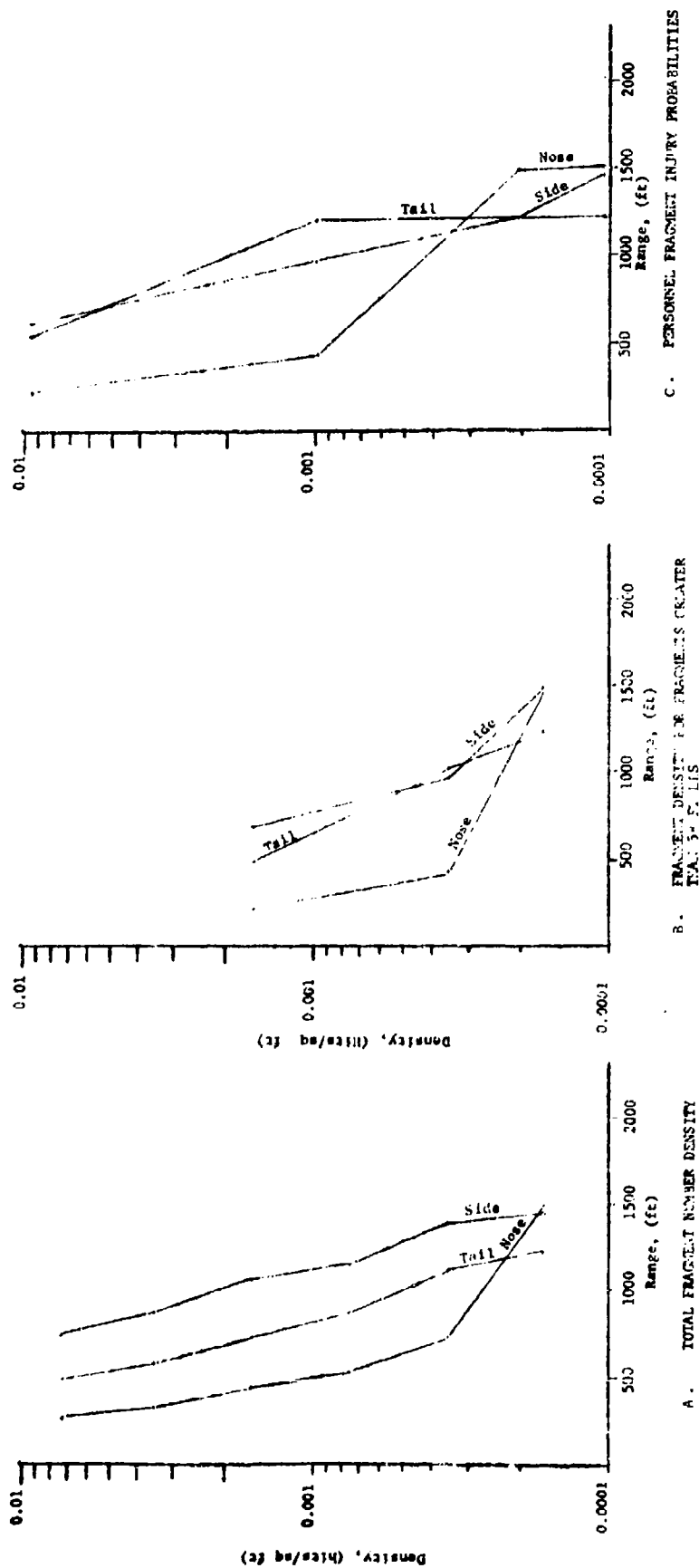


Fig. C2 750 LB BOMB M117A2 (TRITONAL LOAD)

Reproduced from
best available copy.

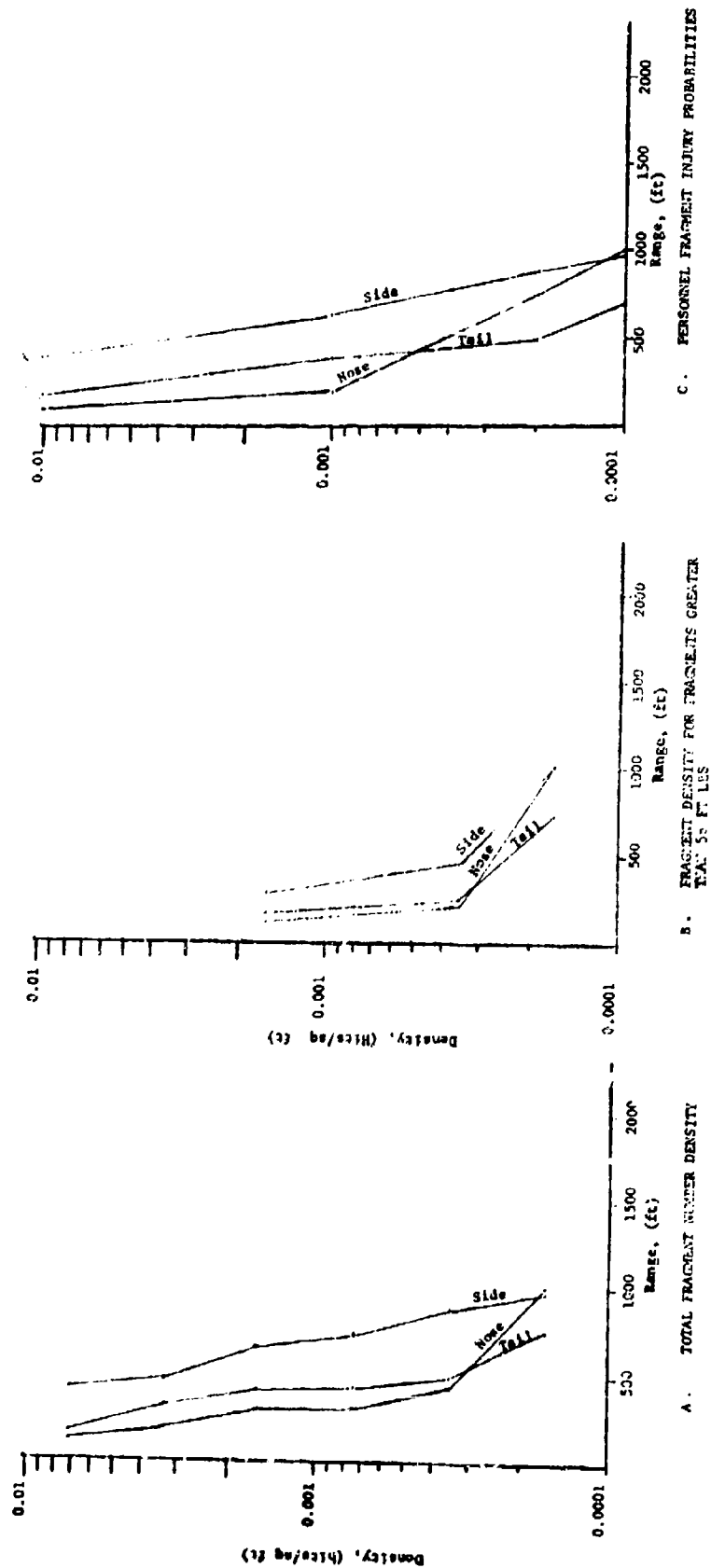


Fig. C3 105 MM HOWITZER SHELL M1 (COMP. B LOAD)

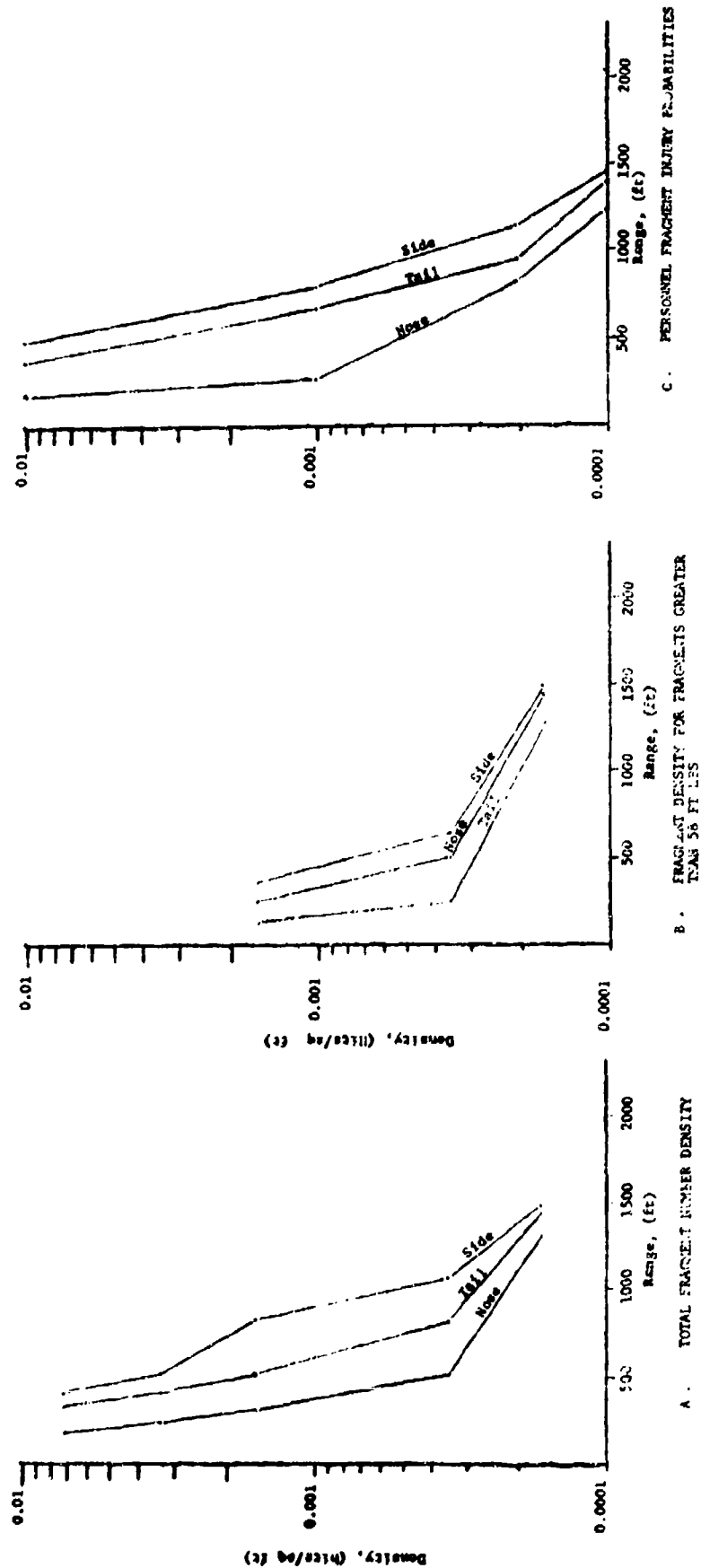


Fig. C4 155 MM HOWITZER SHELL M1C7 (CONF. B LOAD)

Reproduced from
best available copy.

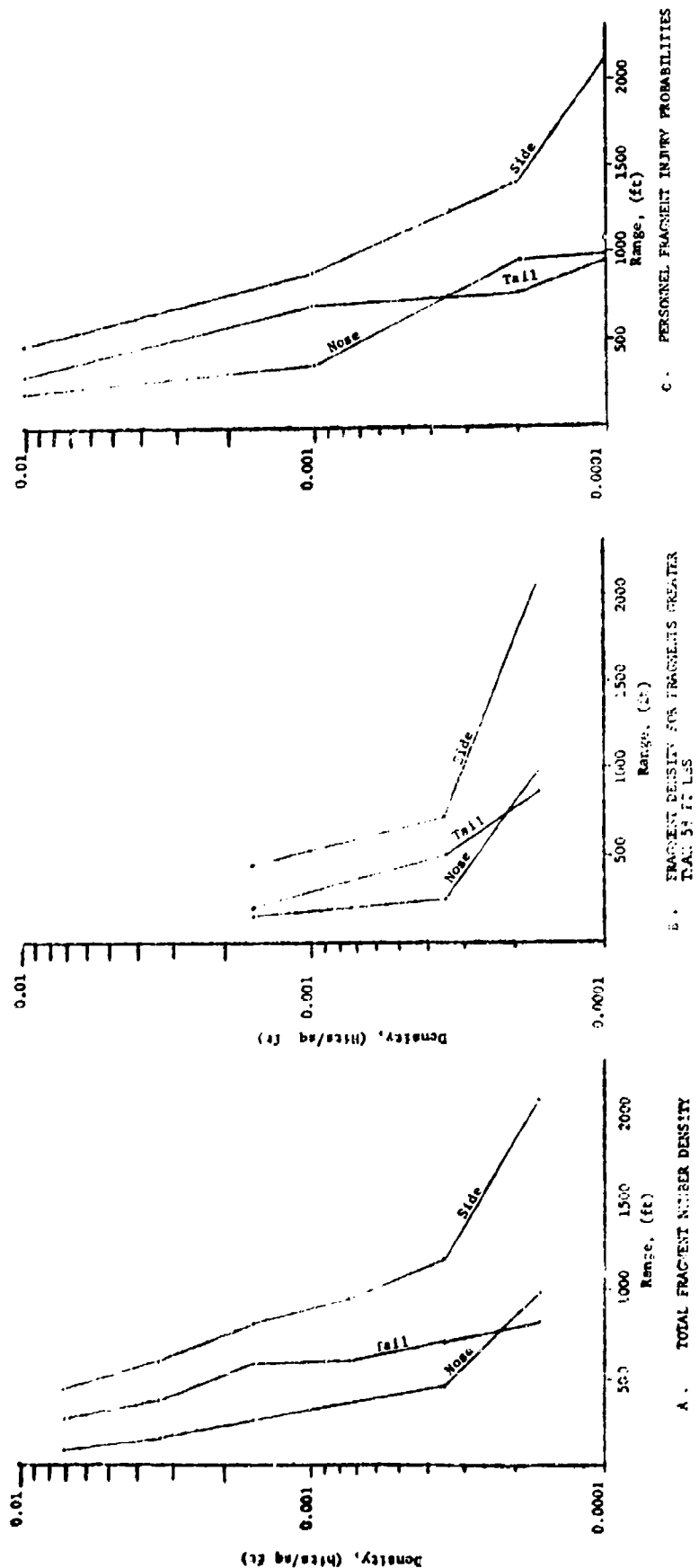


Fig. C5 175 MM GUN SHELL M437A2 (COMP. B LOAD)

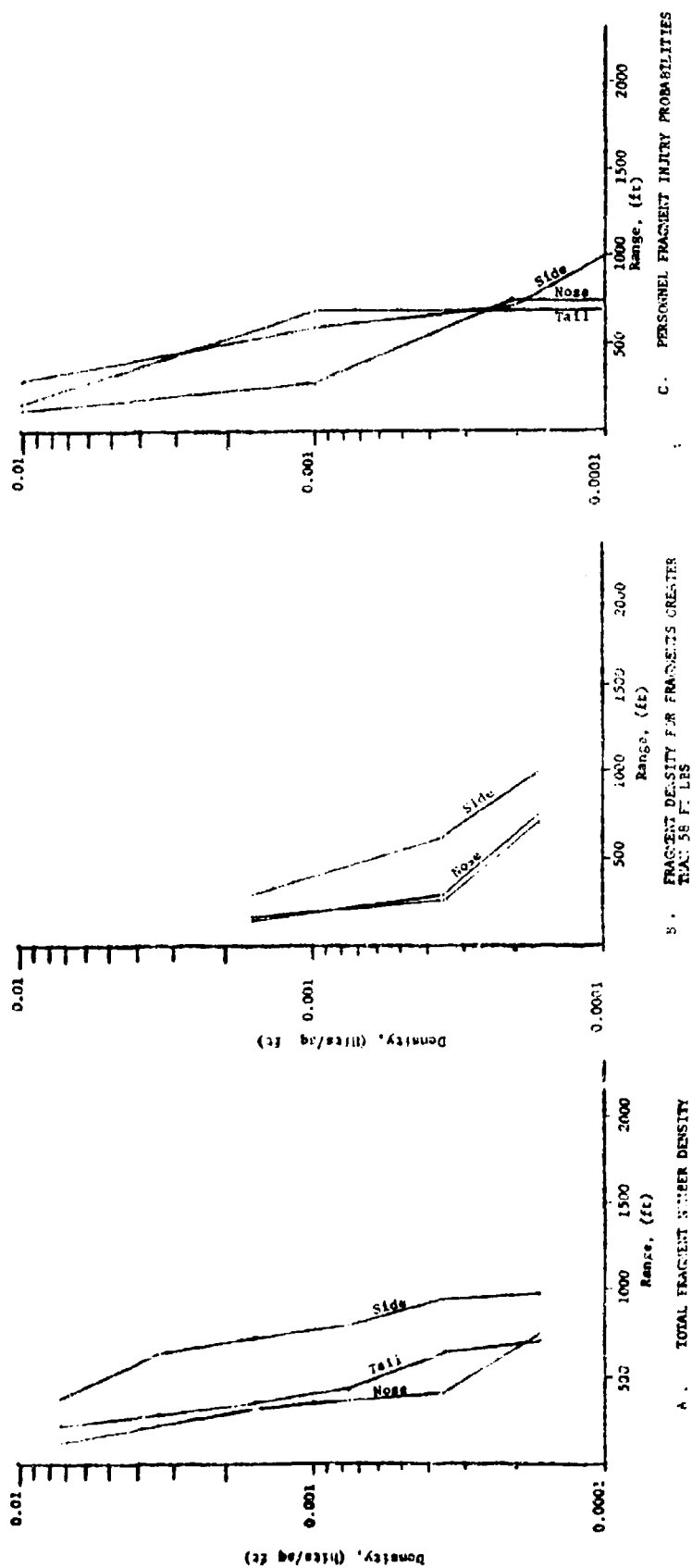


Fig. C6 8"/55 PROJECTILE MARK 25 (EXPLOSIVE D LOAD)

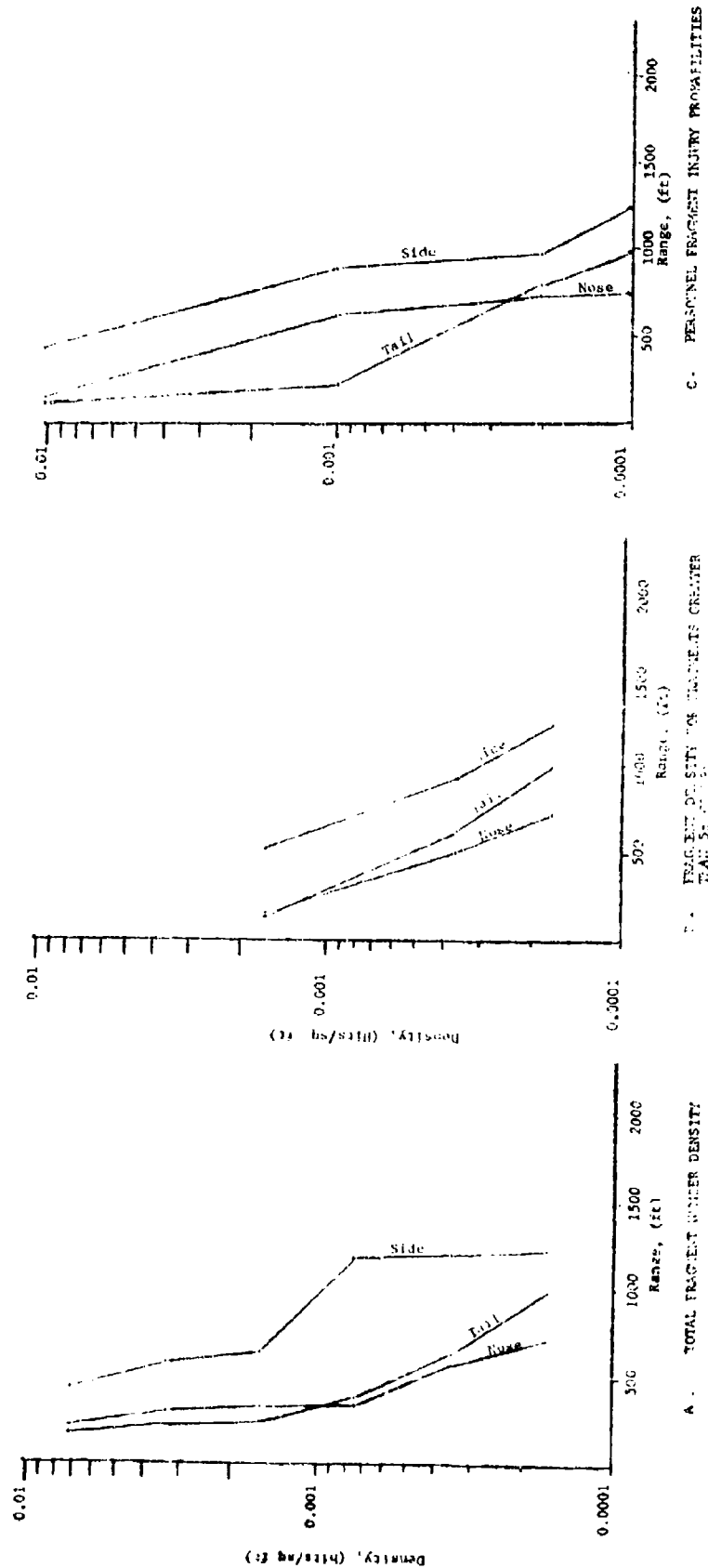


Fig. C7 8"/55 PROJECTILE MARK 25 (EXPLOSIVE D LOAD)

APPENDIX D
COMPUTER PROGRAM LISTINGS

```

C      GIVEN STD TRAJECTORY TABLES (TAPE1) AND ORONANCE 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 FUNCTION TRAG0040
C      I.D. CODES AND VALUES (TAPE2) TRAG0050
C      TRAG0060
C      TRAG0070
C      TRAG0080
C      TRAG0090
C      TRAG0100
C      TRAG0110
C      TRAG0120
C      TRAG0130
C      TRAG0140
C      TRAG0150
C      TRAG0160
C      TRAG0170
C      TRAG0180
C      TRAG0190
C      TRAG0200
C      TRAG0210
C      TRAG0220
C      TRAG0230
C      TRAG0240
C      TRAG0250
C      TRAG0260
C      TRAG0270
C      TRAG0280
C      TRAG0290
C      TRAG0300
C      TRAG0310
C      TRAG0320
C      TRAG0330
C      TRAG0340
C      TRAG0350
C      TRAG0360
C      TRAG0370
C      TRAG0380
C      TRAG0390
C      TRAG0400

C      CONTROL CARD TABLES AND GENERAL PROGRAM CONTROL VARIABLES
C      COMMON /CNTRL/ GCRD,GCCPRT,KTRD,FCC(8),KARD(14),PC,PS,PD,PF,
C      * TXTIME(2),TXDATE(2),WHY(20),KWARD(800)
C      LOGICAL GCCRD,GCCPRT,WHY
C      INTEGER TXTIME,TXDATE,FCC,PC,PS,PD,PP

C      EQUIVALENCE (KKARD(12),KTYPE)

C      LOGICAL PRTCDD
C      EQUIVALENCE (KKAFD(497),PRTCDD)

C      SYNTAX TABLES
C      INTEGER XSYN(64),TSYN(64),SSYN(4),SYND,SYNE
C      LOGICAL SYNTAX
C      EXTERNAL CCARD
C      SYND IS A MARKING CHARACTER TO POINT AT SYNTAX ERRORS (11-7-8 PCH)
C      DATA NSSYN,SYND,SYNB/4,14A,14 /,
C      * TSYN/500,4,30*0.5,0.2,3*0.6,0.0,3.11*0.6,5*0 /
C      STD CONTROL-CARD SYNTAX RULES, MODIFIED FOR $END OF STATEMENT
C      KKARD(4)*2 (NEW SYNTAX ON SAME CARD) SHOULD THEN NEVER APPEAR
C      DATA XSYN/
C      * 00202000200,0000000020000, 00202000400,0000000020011,
C      * 00202000200,0000000050000, 00202000200,0000000070000,
C      * 00202000400,0000000050006, 00202000300,0000000140000,
C      * 00202000400,0000000070010, 00202000300,0000000260000,
C      * 00202000000,0000100140012, 00202000200,0000000000013,
C      * 00202000100,0000000000000, 00202000400,0000000140015,
C      * 00202000300,0000000160022, 00202000400,0000000160017,
C      * 00204000102,0000000200024, 00701000000,0000000000000,
C      * 00701000402,0000000000000, 00202000200,0000000000023,
C      * 00202000100,0000600000024, 00202000000,0000000330025,
C      * 00202000600,0001000026000, 00202000300,0000000270030,
C      * 00701000404,0000000000000, 00202000000,0000000310032,
C      * 00202000600,0000700260014, 00202000000,0000700260000,

```

Reproduced from
best available copy.

```

* 00202000406,0000000330034, 002020000500,0000300360035,
* 00300000000,0000500150000, 00202000400,00000000360037,
* 00202000000,0000400140040, 002020000600,00000000260000/

C MISCELLANEOUS DECLARATIONS
  LOGICAL COMMT
  DATA KNCOMM/1H#/

C
C
C ONCE-ONLY CODE
  CALL DATE(9,TXDATE)
  CALL TOD(8,TXTIME)
  CALL KLOCK
  OPTIME = 0.
  WRITE(6,8) TXTIME,TXDATE
  A FORMAT(1H1,30X,61TTR ,16,A2,5X,6HDATE ,16,A3//)
  KKARD(4) = 1

C
C
C READ CONTROL CARD AND INTERPRET
  100 IF(GCCPD) READ(5,101,END=200) KARD
  101 FORMAT(14A5)
  GCCPD = .TRUE.
  IF (KKARD(4).EQ.2) GO TO 112
  KTCD = KTCD+1
  GCCPRT = PRTCOD

C * IN CC1 MEANS COMMENT
  COMMT = KOM(KARF,1,1,KNCOMM,1).F0.0
  IF (COMMT) GO TO 106
  IF(.NOT,PRTCOD) GO TO 110
C LIST CONTROL CARD IF SWITCH IS ON
  106 WRITE(6,FCC) PC,KTCD,KARD
  PC = PS
  IF (COMMT) GO TO 100
C SEARCH FOR C.C. NAME, AND INTERPRET CARD
  110 KSYN = 1
  LSYN = 72
  112 I = KKARD(4)
  KKARD(4) = 1
  IF(SYNTEX(KARD,KEYN,LSYN,XSYN,1,TSYN,CCARD,SSYN,NSSYN,KKARD))
    * GO TO 119

```

```

TRAG0410
TRAG0420
TRAG0430
TRAG0440
TRAG0450
TRAG0460
TRAG0470
TRAG0480
TRAG0490
TRAG0500
TRAG0510
TRAG0520
TRAG0530
TRAG0540
TRAG0550
TRAG0560
TRAG0570
TRAG0580
TRAG0590
TRAG0600
TRAG0610
TRAG0620
TRAG0630
TRAG0640
TRAG0650
TRAG0660
TRAG0670
TRAG0680
TRAG0690
TRAG0700
TRAG0710
TRAG0720
TRAG0730
TRAG0740
TRAG0750
TRAG0760
TRAG0770
TRAG0780
TRAG0790
TRAG0800

```

```

IF(KTYPE,LE,0) WLY(1)=.TRUE,
IF(KTYPE,GT,0) WLY(KTYPE)=.TRUE,
IF(.NOT.OCCPRT) WRITE(5,FCC) PC,KTCO,KARD
OCCPRT = .TRUE,
IF(KKARD(13),GT,0) KSYN=KKARD(13)
WRITE(6,111) (SYN,I=1,KSYN),SYND
111 FORMAT(17X,100A1)
WRITE (6,113)
113 FORMAT(44H ***** ERROR IN ABOVE CONTROL CARD ***** )
CALL FLUSH
GO TO 100
119 IF(KKARD(4),GE,3) GO TO 100
IF (KKARD(4),EQ,2) OCCRD=.FALSE,
GO TO (100,200,200,400,500,600,700,800,900
*),KTYPE
C
C END OF PROCESSING
C
C 200 STOP
C
C COMPUTE BASIC FRAGMENTATION INFO. TABLE
C
C 400 CALL LINK1(3)
GO TO 9000
C
C DEFINE TYPES OF OUTPUT FUNCTIONS. ALL MUST YIELD VALUES AT (R,PHI)
C DEPENDENT ONLY ON R, PHI, AND FRAG, INFO AT (R,PHI).
C
C 500 CALL LINK2
GO TO 9000
C
C POSITION A FILE
C
C 600 CALL TAPE
GO TO 9000
C
C READ, NORMALIZE, SMOOTH (AND LIST) ORDNANCE DATA
C
C 700 CALL LINK1(1)
GO TO 9000

```

```

TRAG0810
TRAG0820
TRAG0830
TRAG0840
TRAG0850
TRAG0860
TRAG0870
TRAG0880
TRAG0890
TRAG0900
TRAG0910
TRAG0920
TRAG0930
TRAG0940
TRAG0950
TRAG0960
TRAG0970
TRAG0980
TRAG0990
TRAG1000
TRAG1010
TRAG1020
TRAG1030
TRAG1040
TRAG1050
TRAG1060
TRAG1070
TRAG1080
TRAG1090
TRAG1100
TRAG1110
TRAG1120
TRAG1130
TRAG1140
TRAG1150
TRAG1160
TRAG1170
TRAG1180
TRAG1190
TRAG1200

```

```

C
C READ AND PRINT CLOCK
C
  ADD CALL KLOCK(DUM)
  DUM1 = DUM-CPTIME
  IF (PC,E3,PS) PC=PD
  WRITE (6,801) PC:DUM,DUM1
  801 FORMAT(A1,33X,27H***** C.P. TIME TOTAL =,F8.3,9H SEC (UP,
    * F7.3,13H SEC) *****
  CPTIME = DUM
  PC = PD
  GO TO 100
C
C UNUSED STATEMENTS
C
  907 CONTINUE
  GO TO 200
C
C AFTER EVERY COMMAND, TEST ABORT FLAG BEFORE READING NEXT COMMAND
C
  9000 IF(WHY(2)) GO TO 200
  GO TO 100
C
  END

```

```

TRAG1210
TRAG1220
TRAG1230
TRAG1240
TRAG1250
TRAG1260
TRAG1270
TRAG1280
TRAG1290
TRAG1300
TRAG1310
TRAG1320
TRAG1330
TRAG1340
TRAG1350
TRAG1360
TRAG1370
TRAG1380
TRAG1390
TRAG1400
TRAG1410
TRAG1420
TRAG1430
TRAG1440

```

BLOCK DATA

COMMON /CNTRL/ GCRD, GCRRT, VTCO, ECC(8), KAPD(14), PC, PS, PD, PP,

* YTIME(2), TXDATE(2), WHY(20), KVARP(800)

DATA GCRD, KTCO, ECC, PC, PS, PD, PP, WHY, .TRUE., .,

* 26H(A1, AHCARD NO., 15, 5X, 14A6), 3*0,

* 1H, 1H, 1H, 1H, 1H, 2C*, FALSE, /

DATA (KKARD (1), I=1, 124)/10*0, 76.9*0,

* 1, 797, 501, 505, 504, 1, 0, 2, 777, 505, 505, 504, 1,

* 2, 777, 50, 0, 0, 0, 4, 761, 505, 505, 512, 1,

* 5, 737, 513, 513, 525, 1, 0, 6, 685, 526, 527, 537, 1,

* 7, 637, 538, 538, 551, 1, 0, 8, 589, 552, 552, 551, 1,

* 48*0/

DATA (KKARD (1), I=125, 276)/

* 81000, 593.0, 0.0, 0,

* 81000, 597.0, 0.0, 0,

* 81000, 601.0, 0.0, 0,

* 1, 605.0, 0.0, 0,

* 2, 609.0, 0.0, 0,

* 1, 613.0, 0.0, 0,

* 1, 617.0, 0.0, 0,

* 1, 621.0, 0.0, 0,

* 1, 625.0, 0.0, 0,

* 82002, 629.0, 0.0, 0,

* 2, 633.0, 0.0, 0,

* 81000, 641.0, 0.0, 0,

* 81000, 645.0, 0.0, 0,

* 81000, 649.0, 0.0, 0,

* 1, 653.0, 0.0, 0,

* 1, 657.0, 0.0, 0,

* 3, 661.0, 0.0, 0, 6H

* 3, 665.0, 0.0, 0, 6H

* 1, 669.0, 0.0, 0, /

DATA (KKARD (1), I=277, 428)/

* 1, 673.0, 0.0, 0,

* 1, 677.0, 0.0, 0,

* 2, 681.0, 0.0, 0,

* 82002, 0.0, 0.0, 0,

* 81000, 689.0, 0.0, 0,

* 81000, 693.0, 0.0, 0,

* 81000, 697.0, 0.0, 0,

RLK 0010
RLK 0020
RLK 0030
RLK 0040
RLK 0050
RLK 0060
RLK 0070
RLK 0080
RLK 0090
RLK 0100
RLK 0110
RLK 0120
RLK 0130
RLK 0140
RLK 0150
RLK 0160
RLK 0170
RLK 0180
RLK 0190
RLK 0200
RLK 0210
RLK 0220
RLK 0230
RLK 0240
RLK 0250
RLK 0260
RLK 0270
RLK 0280
RLK 0290
RLK 0300
RLK 0310
RLK 0320
RLK 0330
RLK 0340
RLK 0350
RLK 0360
RLK 0370
RLK 0380
RLK 0390
RLK 0400

Reproduced from
best available copy.

```

* 1, 701,0, 0,0,0, .FALSE, .0,
* 1, 705,0, 0,0,0, .FALSE, .0,
* 1, 709,0, 0,0,0, .FALSE, .0,
* 1, 713,0, 0,0,0, .FALSE, .0,
* 1, 717,0, 0,0,0, .FALSE, .0,
* 1, 721,0, 0,0,0, .FALSE, .0,
* 1, 725,0, 0,0,0, .FALSE, .0,
* 1, 729,0, 0,0,0, .FALSE, .0,
* 1, 733,0, 0,0,0, .FALSE, .0,
* 2, 741,0, 0,0,0, 76, .0,
* 2, 745,0, 0,0,0, 14, .0,
* 2, 749,0, 0,0,0, 0, .0,
DATA (KKARD (I),I= 429, 500)/
* 5, 753,0, 0,0,0, 0, .0,
* 2, 757,0, 0,0,0, 0, .0,
* 2003, 773,0, 0,0,0, 6000, .64,
* 5, 765,0, 0,0,0, 250, .0,
* 2, 769,0, 0,0,0, 16, .0,
* 8100, 781,0, 0,0,0, 0,0, .0,
* 8100, 785,0, 0,0,0, 0,0, .0,
* 1, 789,0, 0,0,0, .FALSE, .0,
* 1, 793,0, 0,0,0, .FALSE, .0,
DATA (KKARD (I),I= 501, 588)/
* 493, 485, 477, 469, 461, 453, 445, 437, 429, 421, 413, 405,
* 397, 389, 381, 373, 365, 357, 349, 341, 333, 325, 317,
* 309, 301, 293, 285, 277, 269, 261, 253, 245, 237, 229, 221,
* 213, 205, 197, 189, 181, 173, 165, 157, 149, 141, 133, 125,
* 37*0/
DATA (KKARD (I),I= 589, 740)/
* 64TIM, .64, 0, -3, 0, 64ZZ7, .64, 3, 0, 64ZZ7, .64,
* 64YY7, .64, 0, 3, 0, 64XX7, .64, 3, 0, 64XX7, .64,
* 64SUM, .64, 0, -3, 0, 64REAR, .64, -4, 0, 64REAR, .64,
* 64SMOOTH, .64, 0, -6, 0, 64NORM, .64, -4, 0, 64NORM, .64,
* 64REZONE, .64, 0, -6, 0, 64REDUCE, .64, -6, 0, 64REDUCE, .64,
* 64FOR, .64, 0, -3, 0, 64NSFT, .64, -4, 0, 64NSFT, .64,
* 64ORN, .64, 0, 3, 0, 64ZZ6, .64, 3, 0, 64ZZ6, .64,
* 64YYA, .64, 0, 3, 0, 64XX6, .64, 3, 0, 64XX6, .64,
* 64WE, .64, 0, -2, 0, 64BK, .64, -2, 0, 64BK, .64,
* 64SR, .64, 0, -2, 0, 64XS, .64, -2, 0, 64XS, .64

```

```

ALK 0410
ALK 0420
ALK 0430
ALK 0440
ALK 0450
ALK 0460
ALK 0470
ALK 0480
ALK 0490
ALK 0500
ALK 0510
ALK 0520
ALK 0530
ALK 0540
ALK 0550
ALK 0560
ALK 0570
ALK 0580
ALK 0590
ALK 0600
ALK 0610
ALK 0620
ALK 0630
ALK 0640
ALK 0650
ALK 0660
ALK 0670
ALK 0680
ALK 0690
ALK 0700
ALK 0710
ALK 0720
ALK 0730
ALK 0740
ALK 0750
ALK 0760
ALK 0770
ALK 0780
ALK 0790
ALK 0800

```


DATA NAN,NCN/2,6/

C

```

IF(KFILE,LE,1) GO TO 640
IF(IFILE,LF,0) GO TO 640
IF(IFILE,EO,5) .OR. IFILE,FO,4) GO TO 640
IF(IFMODE .AND. IFLAR) GO TO 640
IF(KSRC*KXSRC,GT,0) GO TO 640
WHY(6) = 'FALSE'
C REWIND IF REQUESTED.
IF (IFRWD) REWIN IFILE
IF (IFSKIP,LE,0) GO TO 620
C SKIP RECORDS IF REQUESTED.
IF (IFMODE) GO TO 610
GO 601 I=1,IFSKIP
601 READ (IFILE) IDUM
GO TO 620
610 DO 611 I=1,IFSKIP
611 READ (IFILE,612) IDUM
612 FORMAT(41)
C SEARCH FOR SENTINEL RECORD
620 IF(KSRC,LE,0) .AND. KXSRC,LE,0) GO TO 650
IF(KSRC,GT,0) CALL MOVE(IFGATG,1,NCN,ISRC,1)
IF(KXSRC,GT,0) CALL MOVE(IFSARC,1,NCN,IXSRC,1)
IFSRWD = 'FALSE'
621 IF(IFMODE) GO TO 622
READ (IFILE,END=630) IDUM,IFLAR
GO TO 625
622 READ (IFILE,624,END=630) IDUM,IFLAR
624 FORMAT(3A6)
625 IF(IDUM,NE,IFHEAD) GO TO 621
IF (KONW(IFSARC,'X')IFLAR).NE,0) GO TO 621
630 IF(IFSKIP) BACK SPACE IFILE
IF(.NOT. IFWECP) GO TO 100
IF(IFMODE) GO TO 652
WRITE(IFILE) IFHEAD,IHEND
GO TO 652
651 WRITE(IFILE,624) IFHEAD,IHEND
652 END FILE IFILE
GO TO 100
630 IF(IFSRWD .OR. KXSRC,LE,0) GO TO 635

```

TAPE0410
 TAPE0420
 TAPE0430
 TAPE0440
 TAPE0450
 TAPE0460
 TAPE0470
 TAPE0480
 TAPE0490
 TAPE0500
 TAPE0510
 TAPE0520
 TAPE0530
 TAPE0540
 TAPE0550
 TAPE0560
 TAPE0570
 TAPE0580
 TAPE0590
 TAPE0600
 TAPE0610
 TAPE0620
 TAPE0630
 TAPE0640
 TAPE0650
 TAPE0660
 TAPE0670
 TAPE0680
 TAPE0690
 TAPE0700
 TAPE0710
 TAPE0720
 TAPE0730
 TAPE0740
 TAPE0750
 TAPE0760
 TAPE0770
 TAPE0780
 TAPE0790
 TAPE0800

TAPE0810
 TAPE0820
 TAPE0830
 TAPE0840
 TAPE0850
 TAPE0860
 TAPE0870
 TAPE0880
 TAPE0890
 TAPE0900
 TAPE0910
 TAPE0920
 TAPE0930
 TAPE0940
 TAPE0950
 TAPE0960

```

IFSRWD = ,TRUE,
REWIND IFILE
GO TO 421
635 IF(.NOT.OCCPRT) WRITE(6,FOC) PC,KTCO,KARD
OCCPRT = ,TRUE,
WRITE (6,636) IFSRWD
636 FORMAT(10H***** SENTINAL (/6,12,12H) NOT FOUND./)
GO TO 645
640 IF(.NOT.OCCPRT) WRITE(6,FOC) PC,KTCO,KARD
OCCPRT = ,TRUE,
WRITE(6,641)
641 FORMAT(48H0***** INCONSISTENT OR INCORRECT PARAMETER(S)./)
645 WHY(6) = ,TRUE.
PC = PC
100 RETURN
END
  
```

Reproduced from
 best available copy.

LINK0010
 LINK0020
 LINK0030
 LINK0040
 LINK0050
 LINK0060
 LINK0070
 LINK0080
 LINK0090
 LINK0100
 LINK0110
 LINK0120

SUBROUTINE LINK1(L)
 C
 C CALL LINK 1 ROUTINES
 C
 IF (L-2) 10,20,30
 10 CALL ORDNAN
 RETURN
 20 CALL EVIT
 RETURN
 30 CALL BOOM
 RETURN
 END

ALKB0010
 ALKB0020
 ALKB0030
 ALKB0040
 ALKB0050
 ALKB0060
 ALKB0070
 ALKB0080

BLOCK DATA
 C ORDNANCE TABLES
 COMMON /ORDCM/ NVORD, NVORD, I*ORD, IVO, XKORD,
 * TTLOD, NVORD, NVORD, NVORD, NVORD,
 * VORD(37), ANORD(30), DOED(37,30), WORD(37,30)
 INTEGER TTLOD(10)
 DATA NVORD, NVORD/37,30/
 END

Reproduced from
 best available copy.

```

C
C SUPROUTINE ORDNAME
C
C CONTROL CARD TABLES AND GENERAL PROGRAM CONTROL VARIABLES
COMMON /CNTRL/ GCCRD, GCCPRT, KTCD, FCC(8), KARD(14), PC, PS, PD, PP,
* TXTIME(2), TXDATE(2), WHY(20), KKARD(800)
LOGICAL GCCRD, GCCPRT, WHY
INTEGER TXTIME, TXDATE, FCC, PC, PS, PD, PP

C
C INTEGER AFFORD
LOGICAL PRIDAT, GCCRD, REZONE, NMWORD, SMORD, LIST
LOGICAL GWKPLT, SUMRY
EQUIVALENCE (KKARD(449), PRIDAT), (KKARD(209), NSETS),
* (KKARD(201), IFGRD), (KKARD(191), GCBORD), (KKARD(185), RFZONE),
* (KKARD(177), NMWORD), (KKARD(169), SMORD), (KKARD(161), INU),
* (KKARD(177), LIST), (KKARD(441), AFFORD), (KKARD(353), GWKPLT),
* (KKARD(153), SUMRY)

C
C ORDNAME TABLES
COMMON /ORDCH/ MVORD, NMWORD, IDORD, IVO, XKORD,
* TTLORD, NVORD, NMORD, DTH, FTH2,
* VORD(37), AMORD(30), DORD(37,30), XORD(37,30)
INTEGER TTLORD(12)

C
C REAL COFORD(21)
EXTERNAL ORDPLT
REAL ZAREA(36), DTZ(36), HTZ(36), WAZ(36), WTH(30), DTM(30), DAM(30),
* SDW(30), SDD(30), CRG(4,30), ORGXX(122)
INTEGER CHOP(9)
DATA CHOP/1MV,1HD,1HW,1HC,1HV,1HD,1HW,1HD,1HW/
DATA PI/3.14159265/, MXFORD/10/
DATA TYWGT, TXNO/6HWGT, (.5H) NO./
EQUIVALENCE (ORGXX(1), CRG(1,1)), (ORGXX(121), VMN),
* (ORGXX(122), VMX)

C
C WHY(7) = .FALSE.
IF (INU, NE, 5 .AND. (WHY(1), OR, WHY(6))) RETURN
SUMRY=SUMRY, OR, LIST, OR, GWKPLT
READ (5,160) TTLORD
160 FORMAT(12A6)
READ (INU,161) NMWORD, NVORD, IDORD, IVO, XKORD
ORDN0010
ORDN0020
ORDN0030
ORDN0040
ORDN0050
ORDN0060
ORDN0070
ORDN0080
ORDN0090
ORDN0100
ORDN0110
ORDN0120
ORDN0130
ORDN0140
ORDN0150
ORDN0160
ORDN0170
ORDN0180
ORDN0190
ORDN0200
ORDN0210
ORDN0220
ORDN0230
ORDN0240
ORDN0250
ORDN0260
ORDN0270
ORDN0280
ORDN0290
ORDN0300
ORDN0310
ORDN0320
ORDN0330
ORDN0340
ORDN0350
ORDN0360
ORDN0370
ORDN0380
ORDN0390
ORDN0400

```

ORDN0410
ORDN0420
ORDN0430
ORDN0440
ORDN0450
ORDN0460
ORDN0470
ORDN0480
ORDN0490
ORDN0500
ORDN0510
ORDN0520
ORDN0530
ORDN0540
ORDN0550
ORDN0560
ORDN0570
ORDN0580
ORDN0590
ORDN0600
ORDN0610
ORDN0620
ORDN0630
ORDN0640
ORDN0650
ORDN0660
ORDN0670
ORDN0680
ORDN0690
ORDN0700
ORDN0710
ORDN0720
ORDN0730
ORDN0740
ORDN0750
ORDN0760
ORDN0770
ORDN0780
ORDN0790
ORDN0800

```

161 FORMAT(3I5,1X,A1,3X,F10.0)
    IF(IVO.EQ,PS) IVC=PD
    WRITE(6,201) TILORD,NMORD,NVORD,ICORD,IVC,YK,PC
201 FORMAT(1H0,1PX,1PA6//19X,7MNWORD =,I3,5X,7MNWORD =,I3,5X,
    * 6H1.D.,=,I6,1H--,A1,5X,12HARC CR/IN2 =,F10.2)
    IF (NMORD,LE,0 .OR. NVORD,LE,0 .OR. NVORD,GT,MVORD .OR.
    * XKORD,LE,0.) STOP
    IF(NSETS,LE,0) NSETS=1
    IF(ICORD,EG,1) CALL ORDIN1(NMORD,NVORD,NSETS,XORD,DORD,VORD,
    * GCBORD,PRTDAT,KTCO,INU,MVORD)
    IF(ICORD,EG,2) CALL ORDIN2(NMORD,NVORD,NSETS,XORD,DORD,VORD,
    * GCBORD,PRTDAT,KTCO,INU,MVORD)
    PC = PD
    IF (PRTDAT) PC=PF
    IF(.NOT.REZONE) GO TO 210
C IF REDUCTION OF ZONES IS REQUESTED
    IDUM = NVORD-1
    DO 230 I=1,NMORD
        NORD(1,I) = DORD(1,I)*2.
        DORD(NVORD,I) = DORD(NVORD,I)*2.
        DO 231 J=1,IVC
            DUM = DORD(J,I)+DORD(J+1,I)
            IF (DUM,LE,0.) GO TO 232
            WORD(J,I) = (WORD(J,I)*DORD(J,I)+WORD(J+1,I)*DORD(J+1,I))/DUM
            GO TO 231
232 WORD(J,I) = 0.
231 DORD(J,I) = DUM/2.
230 CONTINUE
        NVORD = IDUM
210 DTH = PI/FLOAT(NVORD)
        DTH2=DTH/2.
        OPHP = 180./FLOAT(NVORD)
C GENERATE POLAR ZONE AREA (ON UNIT SPHERE)
        DUM = DTH2
        PDUM = 4.*PI*SIN(DUM)
        DO 211 I=1,NVORD
            ZAREA(I) = SIN(DUM)*PDUM
211 DUM = DUM+DTH
            IF(.NOT.NRMORD) GO TO 220
C NORMALIZE

```

Reproduced from
best available copy.

```

DO 215 I=1,NWORD
  NO 215 J=1,NWORD
  215 WORD(I,J) = WORD(I,J)/7ABREA(I)
C CLEAR STD. DEVIATION CELLS
220 SDV = 0.
  NO 249 I=1,NWORD
  SDV(I) = 0.
  249 SDV(I) = 0.
  IF(.NOT.SWORD) GO TO 271
C SMOOTH WORD, WORD, AND WORD
C 1) TRY TO *FILL IN* WORD(*,*)=0 ENTRIES
  NO 221 I=1,NWORD
  DUM = 0.
  J = 1
  22A IF(WORD(J,I).GT.0.) GO TO 222
  IF(J.GT.1) DUM=WORD(J-1,I)
  IF(J.GE.NWORD) GO TO 241
  NO 223 K=J,NWORD
  IF(WORD(K,I).GT.0.) GO TO 224
  223 CONTINUE
C A) TRAILING ZEROES -- SET TO LAST NON-ZERO VALUE (IF ANY)
  241 DO 225 K=J,NWORD
  225 WORD(K,I)=DUM
  GO TO 221
  224 K1 = K-1
  IF(J.GT.1) GO TO 226
C B) LEADING ZEROES -- SET TO 1ST NON-ZERO VALUE
  DO 227 L=1,K1
  227 WORD(L,I) = WORD(K,I)
  J = K
  GO TO 222
C C) INTERMEDIATE ZEROES -- FILL IN VIA LINEAR INTERPOLATION
  226 DUM = WORD(J-1,I)
  DK = (WORD(K,I)-DUM)/FLOAT(K-J+1)
  NO 242 L=J,K1
  DUM = DUM+DK
  242 WORD(L,I) = DUM
  J = K
  222 J = J+1
  229 IF(J.LE.NWORD) GO TO 226

```

```

ORDN0810
ORDN0820
ORDN0830
ORDN0840
ORDN0850
ORDN0860
ORDN0870
ORDN0880
ORDN0890
ORDN0900
ORDN0910
ORDN0920
ORDN0930
ORDN0940
ORDN0950
ORDN0960
ORDN0970
ORDN0980
ORDN0990
ORDN1000
ORDN1010
ORDN1020
ORDN1030
ORDN1040
ORDN1050
ORDN1060
ORDN1070
ORDN1080
ORDN1090
ORDN1100
ORDN1110
ORDN1120
ORDN1130
ORDN1140
ORDN1150
ORDN1160
ORDN1170
ORDN1180
ORDN1190
ORDN1200

```

221 CONTINUE

C 2) SMOOTH ALL CURVES USING FOURIER SERIES

C NOTE.. MASS AND NUMBER NOT CONSERVED EXACTLY BY THESE FITS

IF (AFFORD,LE,0) GO TO 270

AFFORD = MIN(AFFORD,MXFORD)

CALL FOUR(VORD,COFORD,NVORD,AFFORD)

CALL FORGET(WT7,COFORD,NVORD,AFFORD)

DO 246 J=1,NVORD

SDV = SDV+(WTZ(J)-VORD(J))**2

246 VORD(J) = WTZ(J)

SDV = SQRT(SDV/FLOAT(NVORD))

DO 243 I=1,NMORD

CALL FOUR(WORD(1,I),COFORD,NVORD,AFFORD)

CALL FORGET(WT7,COFORD,NVORD,AFFORD)

DO 244 J=1,NVORD

SDW(I) = SDW(I)+(WTZ(J)-WORD(J,I))**2

244 WORD(J,I) = WTZ(J)

SDW(I) = SQRT(SDW(I)/FLOAT(NVORD))

CALL FOUR(DORD(1,I),COFORD,NVORD,AFFORD)

CALL FORGET(WT7,COFORD,NVORD,AFFORD)

DO 245 J=1,NVORD

SDD(I) = SDD(I)+(WTZ(J)-DORD(J,I))**2

245 DORD(J,I) = WTZ(J)

SDD(I) = SQRT(SDD(I)/FLOAT(NVORD))

243 CONTINUE

C DETERMINE VARIOUS AVERAGES AND TOTALS

270 VA = 0.

WT = 0.

DT = 0.

VMN = +1.E+3#

VMX = -VMN

DO 273 I=1,NMORD

ORG(1,I) = VMN

ORG(2,I) = VMX

ORG(3,I) = VMN

ORG(4,I) = VMX

DO 271 J=1,NVORD

DUM = AMAX1(VORD(J),0.)

VMN = AMIN1(VMN,DUM)

VMX = AMAX1(VMX,DUM)

273

ORDN1210
ORDN1220
ORDN1230
ORDN1240
ORDN1250
ORDN1260
ORDN1270
ORDN1280
ORDN1290
ORDN1300
ORDN1310
ORDN1320
ORDN1330
ORDN1340
ORDN1350
ORDN1360
ORDN1370
ORDN1380
ORDN1390
ORDN1400
ORDN1410
ORDN1420
ORDN1430
ORDN1440
ORDN1450
ORDN1460
ORDN1470
ORDN1480
ORDN1490
ORDN1500
ORDN1510
ORDN1520
ORDN1530
ORDN1540
ORDN1550
ORDN1560
ORDN1570
ORDN1580
ORDN1590
ORDN1600

Reproduced from
best available copy.

```

WT7(J) = 0.
DTZ(J) = 0.
DO 272 I=1,NVORD
  UNIT = AMAX1(DCR(J,I),0.)
  COUNT = UNIT*ZAREA(J)
  WEIGHT = AMAX1(WORD(J,I),0.)
  WTZ(J) = WTZ(J)+(COUNT*WEIGHT
  DTZ(J) = DTZ(J)+COUNT
  ORG(1,I) = AMIN1(ORG(1,I),WEIGHT)
  ORG(2,I) = AMAX1(ORG(2,I),WEIGHT)
  ORG(3,I) = AMIN1(ORG(3,I),UNIT)
  ORG(4,I) = AMAX1(ORG(4,I),UNIT)
272 WTZ(J) = WTZ(J)/DTZ(J)
  VA = VA+WORD(J)*WTZ(J)
  WT = WT+WTZ(J)
274 DT = DT+DTZ(J)
  VA = VA/WT
  WA = WT/DT
DO 274 I=1,NVORD
  DAM(I) = 0.
  WTM(I) = 0.
  RTM(I) = 0.
DO 275 J=1,NVORD
  WEIGHT = AMAX1(WORD(J,I),0.)
  UNIT = AMAX1(DCR(J,I),0.)
  COUNT = UNIT*ZAREA(J)
  WTM(I) = WTM(I)+WEIGHT*COUNT
  RTM(I) = RTM(I)+COUNT
  DAM(I) = DAM(I)/(4.*PI)
274 AMORD(I) = WTM(I)/RTM(I)
C LIST FINAL DATA IN COLUMNS IF REQUESTED
280 IF(.NOT.LIST) GO TO 250
W2 = WIND(5,NVORD)
WRITE (6,2P1) PC,(TXWGT,I,TXND,I=1,M2)
281 FORMAT(A1,63X,15HMASS CATEGORIES/
* 12H0 POLAR ZONE,12X,8HVELOCITY,5(7X,A6,12,A5))
PHP1 = G.
PC = PD
DO 282 J=1,NVORD
  PHP2 = PHP1+PHP

```

```

ORDN1610
ORDN1620
ORDN1630
ORDN1640
ORDN1650
ORDN1660
ORDN1670
ORDN1680
ORDN1690
ORDN1700
ORDN1710
ORDN1720
ORDN1730
ORDN1740
ORDN1750
ORDN1760
ORDN1770
ORDN1780
ORDN1790
ORDN1800
ORDN1810
ORDN1820
ORDN1830
ORDN1840
ORDN1850
ORDN1860
ORDN1870
ORDN1880
ORDN1890
ORDN1900
ORDN1910
ORDN1920
ORDN1930
ORDN1940
ORDN1950
ORDN1960
ORDN1970
ORDN1980
ORDN1990
ORDN2000

```

```

283 WRITE(6,283) PC,PHP1,PHP2,WORD(J),(WORD(J,1),I=1,M2)
   FORMAT(A1,F5.1,10F5.1,10F5.1,10F5.1)
284 PHP1 = PHP2
285 PC = PS
286 M1 = M2+1
   IF(M2,GE,NMORD) GO TO 289
M2 = M2+1
PC = PC+1
   WRITE(6,285) PC,(TX,GT,1,1,1,1,M1,M2)
285 FORMAT(A1,63X,15H MASS CATEGORIES/
* 12H POLAR ZONE,6(7X,A6,12,A5))
   PHP1 = PC
   PC = PC
   DO 286 J=1,NWORD
   PHP2 = PHP1+PHP
   WRITE(6,287) PC,PHP1,PHP2,(WORD(J,1),I=1,M2)
287 FORMAT(A1,F5.1,10F5.1,10F5.1,10F5.1)
   PHP1 = PHP2
288 PC = PS
   GO TO 284
289 PC = PP
C LIST SUMMARY OF DATA IF REQUESTED
250 IF (.NOT.SUMRY) GO TO 260
   WRITE(6,257)
257 FORMAT(1H1,57X,15H SUMMARY OF DATA//
* 12H POLAR ZONE,5X,10HTOTAL MASS,7X,8HNO. FRAG,7X,8HAVG MASS/)
   PHP1 = 0
   PC = PP
   DO 256 J=1,NWORD
   PHP2 = PHP1+PHP
   WRITE(6,251) PHP1,PHP2,17Z(J),17(J),WAZ(J)
251 FORMAT(1X,F5.1,10F5.1,10F5.1,10F5.1)
256 PHP1 = PHP2
   WRITE(6,252) WT,CT,VA
252 FORMAT(12H0 ALL ZONES,3F15.1//)
   WRITE(6,253) VMN,VA,VMX,SDV
253 FORMAT(1H1,53X,23H SUMMARY OF DATA (CONT.))//
* 34X,3HLOW,8X,7H AVERAGE,11X,4H HIGH,8X,7H STD DEV,10X,5H TOTAL//
* 14X,8H VELOCITY,4F15.1)
   DO 254 I=1,NMORD

```



```

254 WRITE(6,255) I,ORG(3,I),DAM(I),ORG(4,I),SDR(I),DTM(I),
*
255 FORMAT(11HONASS CAT, 12,9H NO. FRAG,5F15.1/16X,4HMASS,5F15.1)
260 IF(.NOT.(3*KEPL)) GO TO 100
  NPLOT = NMORD/4+1
  AZ1 = NPWP/2.
  AZ2 = 180.-AZ1
  CALL ORDPLT(1,F+38,PC,ORG)
  PC = PC
  M1 = 1
  KC = 1
  DO 261 I=1,NPLOT
    M2 = MIN0(NMORD,M1+3)
    IF (1.EQ.1) M2=M1+3
  262 WRITE(6,263) PC,M1,M2
  263 FORMAT(A1,10X,47NPLOT OF FINAL DENSITY AND WEIGHT TABLES, 4SSFS,
* I3,5H THRU,I3/)
    NF = 2*(M2-M1+1)
    IF(1.EQ.1) NF=NF+1
    CALL NOTARY(ORDPLT, F, NMORD, AZ1, AZ2, 121, 0.18, CHOP(KC))
    KC = 2
    M1 = M2+1
    IF(1.NE.NPLOT) CALL ORDPLT(-1,0.01UM)
  261 CONTINUE
100 PC = PC
  IF(PRTAT .OR. SURVEY) PC=PC
  RETURN
  END
ORDN2410
ORDN2420
ORDN2430
ORDN2440
ORDN2450
ORDN2460
ORDN2470
ORDN2480
ORDN2490
ORDN2500
ORDN2510
ORDN2520
ORDN2530
ORDN2540
ORDN2550
ORDN2560
ORDN2570
ORDN2580
ORDN2590
ORDN2600
ORDN2610
ORDN2620
ORDN2630
ORDN2640
ORDN2650
ORDN2660
ORDN2670
ORDN2680

```



```

SUBROUTINE ORDING(MWORD,NWORD,NSETS,WORD,DORD,VORD,QCBORD,PRTDAT,
* KTCO,INU,MVORD)
ORD10010
ORD10020
ORD10030
ORD10040
ORD10050
ORD10060
ORD10070
ORD10080
ORD10090
ORD10100
ORD10110
ORD10120
ORD10130
ORD10140
ORD10150
ORD10160
ORD10170
ORD10180
ORD10190
ORD10200
ORD10210
ORD10220
ORD10230
ORD10240
ORD10250
ORD10260
ORD10270
ORD10280
ORD10290
ORD10300
ORD10310
ORD10320
ORD10330
ORD10340
ORD10350
ORD10360
ORD10370
ORD10380
ORD10390
ORD10400

C
C
C
C*** NOTE THAT WZ AND WZL SHOULD BE DIMENSIONED FOR COL.LEN=MWORD.
REAL MORD(MWORD,1),DORD(MWORD,1),VORD(MWORD,1),WZ(37,2),WZL(37),
* WT(4),DT(4)
C
LOGICAL QCBORD,QMORD,PRTDAT

NUM = MWORD
IF (QCBORD) NUM=(NUM+1)/2
QNMORD = QCBORD .AND. NMORD.E.MWORD/2*2
C
CLEAR ORD, ARRAYS
DO 2 J=1,NWORD
WZL(J)=0.
VORD(J) = 0.
DO 3 I=1,2
3 WZ(J,I) = 0.
DO 4 I=1,NUM
WORD(J,I) = 0.
4 DORD(J,I) = 0.
2 CONTINUE
C
READ ORD, MASS AND NUMBER TABLES
DO 11 N=1,NSETS
IF(PRTAT) WRITE(6,62) N
82 FORMAT(1X,23MWORD,DORD FOR COMPONENT,12)
11 = 1
16 12 = 11+1
IF(.NOT.QCBORD) 12=12+2
12 = MIN0(NUM,12)
DO 12 J=1,NWORD
READ (INU,13) (WT(K),DT(K),K=1,4)
13 FORMAT(4(2F5.0,5X))
IF (PRTDAT) WRITE(6,81) (WT(K),DT(K),K=1,4)
81 FORMAT(20X,4(2F10.3,4X))
K = 0
DO 14 I=1,12
K = K+1
IF(.NOT.QCBORD) GO TO 15

```

```

K = K+1
WORD(J,I) = WORD(J,I)+T(K-1)*T(K-1)
DORD(J,I) = DORD(J,I)+T(K-1)
WZ(J,N) = WZ(J,N)+T(K-1)*T(K-1)
IF(I.EQ.NUM.AND(.NOT.NVORD)) GO TO 14
15 WORD(J,I) = WORD(J,I)+I(K)*T(K)
DORD(J,I) = DORD(J,I)+T(K)
WZ(J,N) = WZ(J,N)+I(K)*T(K)
14 CONTINUE
12 CONTINUE
I1 = I2+1
IF(I2.LT.NUM) GO TO 16
11 CONTINUE
NMORD = NUM
C SET AVG MASS ARRAYS (DIVIDE TOTAL MASS(J,I) BY TOT. NO. (J,I) )
DO 20 J=1,NVORD
DO 20 I=1,NMORD
IF(DORD(J,I).LE.0.) GO TO 22
WORD(J,I) = WORD(J,I)/DORD(J,I)
GO TO 20
22 WORD(J,I) = 0.
20 CONTINUE
C READ AVG VEL'S(J,N); WEIGHT BY TOTAL MASSES
DO 23 N=1,NSETS
IF (PRTOAT) WRITE(6,93) N
83 FORMAT(10X,18MOVED FOR COMPONENT,I2)
DO 21 J=1,NVORD
READ(15,24) VT
24 FORMAT(50X,F10.0)
84 FORMAT(20X,F10.1)
WZ(J) = WZ(J)+Z(J,N)
21 VORD(J) = VT+WZ(J,N)+VORD(J)
23 CONTINUE
C STORE OVERALL AVG V'S -- CONSERVE MOMENTUM
DO 30 J=1,NVORD
IF(WZD(J).LE.0.) GO TO 31
VORD(J) = VORD(J)/WZD(J)
GO TO 30
31 VORD(J) = 0.
30 CONTINUE
RETURN
END

```

Reproduced from
best available copy.


```

K = 4*MMORD
VA = V(K+1)
VM = V(K+2)
VM = DP/(VM-VA)
VA = VMN-VA*VM
Y = VMN+DY
DO 14 J=1,MMORD
  PA(J) = DP/(PA(J)-VA(J))
  NA(J) = Y-NA(J)*PA(J)
  WA(J) = DP/(WA(J)-NA(J))
  WA(J) = Y-WA(J)*NA(J)
  Y = Y+DY
IF(Y.GE.YMAX) Y=VMN
14 CONTINUE
I = 0
RETURN
END

```

Reproduced from copy.
best available

```

OROP0410
OROP0420
OROP0430
OROP0440
OROP0450
OROP0460
OROP0470
OROP0480
OROP0490
OROP0500
OROP0510
OROP0520
OROP0530
OROP0540
OROP0550
OROP0560
OROP0570

```

[illegible]


```

C      MXPB      MAX NO. ZIMUTH RAYS ALLOWED
C      MXPB      MAX NO. RAYS ALLOWED
C      MXPB      MAX ALLOWED NO. TRAJ. CALCS PER A2 RAY
C      DATA MXPB,MXRN,MXRV/16,20,40/
C      EPN1, EPN2      FITTING TOLERANCES FOR 'CEIT'
C      DATA EPN1,EPN2/0.,.015/
C      EPS      HEIGHT TOLERANCE FOR CONVERGENCE, 1-STEP INTEGRATION
C      MKCOUNT      MAX NO. ITERATIONS ALLOWED, 1-STEP INTEGRATION
C      DATA EPS/C.17,MKCUNT/10/
C      G      ACCELERATION OF GRAVITY, FT/SEC**2
C      G2      HALF OF G
C      RETAN      DRAG COEF * AIR DENS * 7000 / (2*144)
C      DATA G,G2,RETAN/12.17,16.085,2.3269/
C      CL,CH      VALUE LIMITS FOR OUTPUT VALUES.. VEL., ANGLE, DENSITY
C      DATA CL/C.,-1.57,8.0./,CH/1.E+38,1.5708,1.E+38/
C
      WHY(4) = WHY(7) .OR. WHY(3)
      IF (NPH.LE.0 .OR. NPH.GT.MXPB .OR. NALF.LE.0 .OR. NALF.GT.MXRAY
      * .OR. NRNG.LE.0 .OR. NRNG.GT.MXRAY .OR. NRNG.GT.MXRN .OR.
      * DRNG.LE.0.) GO TO 910
      KTEMP = MMORD**4
      IVA = PD
      IVF = PD
      CALL MOVE(IVA,1,1,IPOSOP,1)
      CALL MOVE(IVF,1,1,IPOSOP,2)
      IF (IVA.EQ.PS) IVA=PD
      IF (IVF.EQ.PS) IVF=PD
      DO 212 I=1,NRN
      212 IPOSOP(1) = IDIP(1)
      I = IFX(IPOSOP,1,2,0,IDORN)
      CALL MOVE(IPOSOP,3,2,IVO,1)
      CALL MOVE(IPOSOP,4,2,IVA,1)
      CALL MOVE(IPOSOP,5,2,IVF,1)
      READ(5,210) TTLEFG
      210 FORMAT(12A6)
      WRITE(6,211) TTLEFG
      211 FORMAT(1H0,18X,12A6)
      WRITE(6,213) NPH,NALF,NRNG,DRNG,IPOSOP
      213 FORMAT(1H0,18X,5NHPR =,13,5X,6HNALF =,13,5X,6HNRNG =,13,5X,
      * 6HDRNG =,13,5X,11HTAPE KEY =,A6,A2)

```

Reproduced from
best available copy.

```

C READ BARRIER INFORMATION, IF INDICATED
    IF (NBAR,LE,0) GO TO 202
    IF (NBAR,GT,1) GO TO 905
    WRITE (6,224)
    224 FORMAT(1H0,1X,1CHP/BARRIER INFORMATION)
    DO 229 I=1,NBAR
        READ(5,220) IRTYP(I),IDUM,ICOR(I),RCPH(I),RHAL(I)
    220 FORMAT(2I5,6F17,0)
    IF (IRTPAT) WRITE(6,221) IRTYP(I),IDUM,RCPH(I),RCPH(I),RHAL(I)
    221 FORMAT(19X,2I5,3F15,5)
        RCPH(I) = RCPH(I)*RANDEG
        RCPH(I) = RCPH(I)*RANDEG
        RHAL(I) = RHAL(I)*RANDEG
        TBHAL(I) = TBHAL(I)
    229 CONTINUE
C RETURN IF ERROR FLAGS ARE SET.
    202 IF(WHY(1),OR,WHY(6),OR,WHY(6)) GO TO 100
C BEGIN FRAGMENT FIELD OUTPUT
    WRITE (IFRTAP) IIDE,IBOSAP
    WRITE (IFRTAP) ITLCE,ITTEDG,IV,XACRD,NRNG,DBRC,NPH
    WRITE (IFRTAP) (XORD(I),I=1,N)
    IMX = 6*NM
    IMY = 2*NM
    DAL = PI/(2.*FLOAT(NALF))
    NPH = PI/FLOAT(NPH)
    NPH2 = DPH/2.
    DAL2 = DAL/2.
    PHI = DPH2
C COMPUTE FRAGMENT FIELD FOR EVERY AZIMUTH ANGLE (NPH OF THEM)
C
    DO 401 IPH=1,NPH
        DO 404 IM=1,NV
            404 KPTA(IM) = 0
        DO 403 IM=1,IMY
            403 KPK1(IM) = 0
            SPHI = SIN(PHI)
            CPHI = COS(PHI)
C TEST FOR BARRIER(S) ALONG CURRENT AZIMUTH
        ALFBA = 0.

```

ROOM0810
 ROOM0820
 ROOM0830
 ROOM0840
 ROOM0850
 ROOM0860
 ROOM0870
 ROOM0880
 ROOM0890
 ROOM0900
 ROOM0910
 ROOM0920
 ROOM0930
 ROOM0940
 ROOM0950
 ROOM0960
 ROOM0970
 ROOM0980
 ROOM0990
 ROOM1000
 ROOM1010
 ROOM1020
 ROOM1030
 ROOM1040
 ROOM1050
 ROOM1060
 ROOM1070
 ROOM1080
 ROOM1090
 ROOM1100
 ROOM1110
 ROOM1120
 ROOM1130
 ROOM1140
 ROOM1150
 ROOM1160
 ROOM1170
 ROOM1180
 ROOM1190
 ROOM1200

```

IF(NRAB,LE,0) GO TO 240
DO 240 I=1,NRAB
  DUM = ABS(DMI-RCF-1)
  IF(DUM,GT,RCFPH(I)) GO TO 240
  IF(IRTP(I),NE,1) GO TO 240
  ALFAB = AMAX1(ALFOPR,SHALL(I))
  GO TO 240
240 ALFAB = AMAX1(ALFOPR,1) (CFM(I))*COS(" " )
240 CONTINUE
240 GBARPH = ALFAB*PI,
C
C COMPUTE ASCENDING FRABS AND OFSC LOW-REG FRABS, STEPPING RANGE
C
DO 305 IM=1,NR
  ALF = 0.
  RNG = CRNG
  DO 390 IR=1,NRNG
    TEST2L = 1.E+35
    R2 = RNG*RNG
    U = 0.
    T = 0.
    W = Z
    DUM = SQRT(H*W+R2)
    SAP = H/DUM
    CAP = RNG/DUM
    C 1-STEP INTEGRATION METHOD -- ITERATE FOR SATISFACTORY INITIAL ALPHA
    DO 315 ICOUNT=1,IKOUNT
      TW = ACOS(CAP*CP1)
      CALL INTSET(NVORD,DIM2,DTH,TM,ITH,PTH)
      VORDI = AMAX1(0.,F1NET(VORD,ITH,PTH))
      WORDI = AMAX1(0.,F1NET(WORD,ITH,PTH))
      BETA = BETA0/(XKORD*XKORD*VORDI)*0.33333
      XB = -G2*T*CAP*(1.+U/3.)/(1.+U)
      YB = -G2*T*CAP*(U*(1.+U+0.5)-ALOG(1.+U))/(1.+U)
      XBO = RNG/CAP+YB*SAP/CAP-XR
      DUM = BETA*XB0
      C*** ABORT ITERATION IF EXP(DUM) WOULDN BE VERY LARGE OR SMALL
      IF (ABS(DUM).GT.32.) GO TO 392
      U = EXP(DUM)-1.
      T=U/(BETA*VORDI)

```

Reproduced from
best available copy.

```

ROOM1610
ROOM1620
ROOM1630
ROOM1640
ROOM1650
ROOM1660
ROOM1670
ROOM1680
ROOM1690
ROOM1700
ROOM1710
ROOM1720
ROOM1730
ROOM1740
ROOM1750
ROOM1760
ROOM1770
ROOM1780
ROOM1790
ROOM1800
ROOM1810
ROOM1820
ROOM1830
ROOM1840
ROOM1850
ROOM1860
ROOM1870
ROOM1880
ROOM1890
ROOM1900
ROOM1910
ROOM1920
ROOM1930
ROOM1940
ROOM1950
ROOM1960
ROOM1970
ROOM1980
ROOM1990
ROOM2000

XB = -G2*I*T*SAP*(1.+U/3.)/(1.+U)
YB = -G2*I*T*CAP*(1.*(1.+U/3.))-ALOG(1.+U)/(1.+U)
H = (XRO+XR)*SAP+YB*CAP
DUM1 = Z-H
DUM = SQRT(DUM1*DUM1+R2)
SDA = DUM1/DUM
CDA = ENG/DUM
CUM = SAP*CDA+CAP*SDA
CAP = CAP*CDA-SAP*SDA
SAP = DUM
TEST2=ABS(DUM1)
C DIVERGENCE INDICATES THAT ENG.GT.MAX RANGE FOR GIVEN INIT. COMLS.
IF(TEST2.GT.TEST2L) GO TO 390
C CONVERGENCE IF TERM. HEIGHT CLOSE ENOUGH TO Z.
IF (TEST2.LT.EPS) GO TO 320
TEST2L=TEST2
315 CONTINUE
GO TO 392
320 XDO = VORD1/(1.+U)
XDB = -G2*I*SAP*(1.+U*(1.+U/3.))/(1.+U)**2
YDB = -G2*I*CAP*(1.+U*0.5)/(1.+U)
XD = (XDO+XDB)*CAP-YDB*SAP
YD = (XDO+XDB)*SAP+YDB*CAP
VELT = SQRT(XD*XD+YD*YD)
DUM1 = XDO+XDB
DUM = SQRT(YDB*YDB+(DUM1*DUM1))
SDA = YDB/DUM
CDA = DUM1/DUM
SAF = SAP*CDA+CAP*SDA
CAF = CAP*CDA-SAP*SDA
DUM = 2.*SAP*CAP
DRDA = (2.*ENG-XRO*CAP)*(CAP*CAP-SAP*SAP)/DUM-CAF/SAF
* XBO*CAP/DUM
CORD1 = AMAX1(0.,FINET(DORD(1,1),1TH,PTH))
ALF = ATAN2(SAP,CAP)
GBAR = GBARPH .AND. ALF.LE.ALFFAR
I = KPTA(IM)+1
KPTA(IM) = I
TEMP(1,IM,I) = RIG
TEMP(2,IM,I) = VELT

```

ROOM2010
ROOM2020
ROOM2030
ROOM2040
ROOM2050
ROOM2060
ROOM2070
ROOM2080
ROOM2090
ROOM2100
ROOM2110
ROOM2120
ROOM2130
ROOM2140
ROOM2150
ROOM2160
ROOM2170
ROOM2180
ROOM2190
ROOM2200
ROOM2210
ROOM2220
ROOM2230
ROOM2240
ROOM2250
ROOM2260
ROOM2270
ROOM2280
ROOM2290
ROOM2300
ROOM2310
ROOM2320
ROOM2330
ROOM2340
ROOM2350
ROOM2360
ROOM2370
ROOM2380
ROOM2390
ROOM2400

```

TEMP(3,IM,I) = A*AN2(SAF,CAP)
TEMP(4,IM,I) = 0.
C** NOTE THAT UNIFORM-SOURCE FORM OF DR/D ALPHA IS USED
IF (.NOT.GBAR) TEMP(4,IM,I)=DODI*CAP/ABS(RNC*ORDA*SAF)
390 RNC = RNC+DRNG
392 ALPHI(IM) = ALF
395 CONTINUE
C
C COMPUTE HIGH-REC DESCRIBING FRACS. STEPPING ELEVATION ANGLE
C
ALFO = PI
DO 405 IM=1,NM
  KPTD(IM) = KPTA(IM)
  ALFO = ANINI(ALFO,ALPHI(IM))
405 CONTINUE
  KAL = (ALFO+DAL2)/CAL+1.
  ALF = FLOAT(KAL)*DAL-DAL2
  DO 402 IAL=KAL,NALE
    SALT = SIN(ALF)
    CALF = COS(ALF)
    GBAR = GRAPW .AND. ALF.LE.ALFRAR
    CTH = CALF*CPHI
    TH = ACOS(CTH)
    STH = SQRT(1.-CTH*CTH)
    DTHDAL = SALT*CPH1/STH
  C PSI IS LONGITUDINAL ANGLE ON SPHERE; OBSERVED LOOKING FROM TAIL TO
  C NOSE. (UP=0, AT A7IM=90, PSI=90-ELEV)
  SPSI = CALF*SPHI/STH
  CPSI = SALT/STH
  CALL INTSET(NWORD,DM2,DTH,TH,ITH,PTH)
  VORDI = FINFT(VORD,ITH,PTH)
  VORDI = AMAX1(0.,VORDI)
  DWORDI = FINFT(VORD,ITH,PTH,PTH)
  C COMPUTE FIELD FOR EACH MASS CATEGORY
  DO 410 IM=1,NMORT
    IF (ALF.LE.ALPHI(IM)) GO TO 410
    WORDI = FINFT(WORD(1,IM),ITH,PTH)
    WORDI = AMAX1(0.,WORDI)
    DORDI = FINFT(WORD(1,IM),ITH,PTH)

```

Reproduced from
best available copy.

```

DORDI = AMAX(0.,DORDI)
CALL XSTEP(XKORD,DORDI,VORDI,-7,ALF,RNGT,VELT,SALEF,CALEF)
ALF = ATAN2(SALEF,CALEF)
I = KPTD(IM)+1
IF (I,GT,IXRAY) GO TO 900
KPTD(IM) = I
TEMP(1,IM,I) = RNGT
TEMP(2,IM,I) = VELT
TEMP(3,IM,I) = ALF+
TEMP(4,IM,I) = 0.
C** NOTE THAT DR/DA TAKEN AS DELTA R/DELTA ALPHA
IF(.NOT.OBAR) TEMP(4,IM,I) = DORDI*CALF/DAL/ANG(RNGT*SALEF)
410 CONTINUE
402 ALF = ALF+DAL
C
C COMPLETE PROCESSING OF TERMINAL CHARACTERISTICS
IMM = 0
JM = 0
DO 423 IM=1,NIMPR
  JM = 0
  C COMPLETE FRAGMENT DENSITY COMPUTATION FOR DESCENDING LEG OF TRAJ.
  I1 = KPTA(IM)+1
  I2 = KPTR(IM)
  IF (I1,GT,I2) GO TO 465
  DUM1 = TEMP(1,IM,I)
  DUM = DAL*DUM1
  DO 461 I=I1,I2
    IF (I,EQ,I2) GO TO 462
    NUM2 = TEMP(1,IM,I+1)
    NUM = ABS(DUM2-DUM1)
    DUM1 = DUM2
  462 TEMP(4,IM,I) = TEMP(4,IM,I)/NUM
  461 CONTINUE
  465 IE = 0
  421 IE = IF+1
  GO TO (422,423,420),IE
C FOR ASC. AND LOW-REG. DESC. FRAGMENTS: INSERT INFO INTO OUTPUT TABLES
  422 IN = KPTA(IM)
  IF (IN,LE,0) GO TO 490
  JM = JM+1

```

```

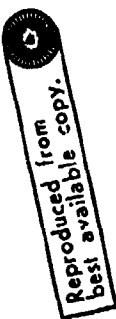
ROOM2410
ROOM2420
ROOM2430
ROOM2440
ROOM2450
ROOM2460
ROOM2470
ROOM2480
ROOM2490
ROOM2500
ROOM2510
ROOM2520
ROOM2530
ROOM2540
ROOM2550
ROOM2560
ROOM2570
ROOM2580
ROOM2590
ROOM2600
ROOM2610
ROOM2620
ROOM2630
ROOM2640
ROOM2650
ROOM2660
ROOM2670
ROOM2680
ROOM2690
ROOM2700
ROOM2710
ROOM2720
ROOM2730
ROOM2740
ROOM2750
ROOM2760
ROOM2770
ROOM2780
ROOM2790
ROOM2800

```

```

KPK1(JM) = 1
KPK2(JM) = IN
DO 424 I=1,3
  IMM = IMM+1
DO 424 J=1,IN
  424 PACK(J,IMM) = TEMP(I+1,IM,J)
GO TO 421
C FOR DESCENDING HIGH-REF. FRAGMENTS, INTERPOLATE FOR OUTPUT INFC.
423 IF (KPTC(IM).LE.IN) GO TO 490
IS = IN+1
II = KPTC(IM)
C FIRST, REORDER TRAJ'S TO ORDER OF ASC. RANGE (FOR THE MOST PART)
J = II
IDUM = IS+(II-IS-1)/2
DO 426 I=IS,IDUM
  DO 425 K=1,4
    DUM = TEMP(K,IM,I)
    TEMP(K,IM,I) = TEMP(K,IM,J)
  425 TEMP(K,IM,J) = DUM
  426 J = J+1
C ELIMINATE ANY LOW-REF. TRAJ'S.. I.E., MAKE RANGE INCR. MONOTONICALLY
J = II-1
DO 427 I=IS,J
  IF (TEMP(1,IM,I+1).LE.TEMP(1,IM,I)) GO TO 428
  427 CONTINUE
  428 I = II
  II = I
  IN = II+IN
  IDUM = MAX0(1,INQ(NRNG,TEMP(1,IM,IS)/DRNG+0.999999))
  JDUM = MAX0(IDUM,MINQ(NRNG,TEMP(1,IM,II)/DPNG))
  JM = JM+1
  KPK1(JM) = IDUM
  KPK2(JM) = JDUM
  DO 430 I=1,3
    IMM = IMM+1
    RNG = DRNG*FLOAT(IDUM)
  431 J=IDUM,JDUM
  DUM = FLINT(TEMP(1,IM,IS),KITEMP,IN,PNG)
  DUM = AMAX1(CL(I),AMIN1(CH(I),DUM))
  PACK(J,IMM) = DUM

```



```

RNG = RNG+DRNG
431 CONTINUE
432 CONTINUE
GO TO 421
C IF NO DATA FOR A TRIJ. TYPE, FILL IN ZEROES
490 DO 491 I=1,3
IIM = IIM+1
DO 492 J=1,NRNG
492 PACK(J,IIM) = 0.
491 CONTINUE
JM = JM+1
GO TO 421
420 CONTINUE
C OUTPUT SMOOTHED TERM, CHAR. FOR I-IF AZ. RAY
WRITE (IFRTAP) (KPR1(J),KPR2(J),I=1,IIM),
* ((PACK(I,J),I=1,NRNG),J=1,IIM)
401 PHI = PHI+DPH
GO TO 100
C
C ERRORS
C
900 IF (.NOT. GCCPRT) WRITE(6,FCC) PC,KTCO,KARD
GCCPRT = .TRUE.
WRITE(6,901) KPA(1M),I,IPRH,IM
901 FORMAT(27H0***** RAY OVERFLOW,,14,10H R-POINTS,,13,
* 20H TOTAL POINTS ON RAY,13,10H, MASS CAT,13)
GO TO 990
905 IF (.NOT. GCCPRT) WRITE(6,FCC) PC,KTCO,KARD
GCCPRT = .TRUE.
WRITE(6,906)
906 FORMAT(27H0***** TOO MANY BARRIERS.)
GO TO 990
910 IF (.NOT. GCCPRT) WRITE(6,FCC) PC,KTCO,KARD
GCCPRT = .TRUE.
WRITE(6,911)
911 FORMAT(21H0***** BAD VALUE(S))
990 CALL FLUSH
WHY(4) = .TRUE.
100 PC = PD
RETURN
END

```

```

ROOM3210
ROOM3220
ROOM3230
ROOM3240
ROOM3250
ROOM3260
ROOM3270
ROOM3280
ROOM3290
ROOM3300
ROOM3310
ROOM3320
ROOM3330
ROOM3340
ROOM3350
ROOM3360
ROOM3370
ROOM3380
ROOM3390
ROOM3400
ROOM3410
ROOM3420
ROOM3430
ROOM3440
ROOM3450
ROOM3460
ROOM3470
ROOM3480
ROOM3490
ROOM3500
ROOM3510
ROOM3520
ROOM3530
ROOM3540
ROOM3550
ROOM3560
ROOM3570
ROOM3580
ROOM3590
ROOM3600
ROOM3610

```

SUBROUTINE NSTEP(X,XV,XG,VLO,VLO,HOF,FLVO,X,VL,SLV,CLV)

DATA PI/3.141592/3/
DATA HOF/10/HEX/2.3069/
DATA G/32.17/

RT = HFTAG/(X*XV*(G))**0.333333
VST=SGRT(G/RT)
TST=1./(RT*VST)
VL = VLO
X=0.
V=HOF
T=0.
DT=TST/FLOAT(HOF)
SLV = SIN(FLVO)
CLV = COS(FLVO)
2 VLI=VL

Reproduced from
best available copy.

TH=BT*VL*DT
THI=1.+TH
ALT=ALOG(THI)
UB=VL/THI-G*SLV*DT*(THI+T.*TH/3.)/THI**2
VB=G*CLV*DT*(1.+TH/2.)/THI
DXBI=VL*DT
IF(BT,GT,0.) DXB=ALT/47
DXR=DXBI-G*SLV*DT*TH*(3.+TH)/(4.*TH)
FTHY=1.*TH/3.
IF(TH,GT,1.E-04) FTHY=(TH*(2.+TH)-2.*ALT)/(2.*TH*TH)
DYR=G*CLV*DT*FTHY/2.
DX=DXR*CLV-DYR*SLV
DY=DXB*SLV+DYB*CLV
VL=SGRT(UB*UB+VB*VB)
UB = UB/VL
VB = VB/VL
NUM = SLV*UB+CLV*VB
CLV = CLV*UB-SLV*VB
SLV = NUM
X=X+DX
V=V+DY
T=T+DT
IF (Y,GT,0. .OR. DY,GT,0.) GO TO 2

NSTEP010
NSTEP020
NSTEP030
NSTEP040
NSTEP050
NSTEP060
NSTEP070
NSTEP080
NSTEP090
NSTEP100
NSTEP110
NSTEP120
NSTEP130
NSTEP140
NSTEP150
NSTEP160
NSTEP170
NSTEP180
NSTEP190
NSTEP200
NSTEP210
NSTEP220
NSTEP230
NSTEP240
NSTEP250
NSTEP260
NSTEP270
NSTEP280
NSTEP290
NSTEP300
NSTEP310
NSTEP320
NSTEP330
NSTEP340
NSTEP350
NSTEP360
NSTEP370
NSTEP380
NSTEP390
NSTEP400

```

VL=VL-Y*(VL-VLT)/DY
DUM = -Y*ATAN2(VF,VF)/DY
VB = SIN(DUM)
UB = COS(DUM)
DUM = SLV*UB+CLV*VB
CLV = CLV*UB-SLV*VB
SLV = DUM
X=X-Y*DX/DY
Y=Y-V*DY/DV
RETURN
END

```

```

NSTE0410
NSTE0420
NSTE0430
NSTE0440
NSTE0450
NSTE0460
NSTE0470
NSTE0480
NSTE0490
NSTE0500
NSTE0510

```

```

SUBROUTINE INTSE(XL,DX,Y,X,PX)

```

```

C
C FIND MAIN ENTRY NO. IX AND EX-EXACT. IPX. GIVEN TABLE OF
C ENTRIES AT DX INTERVAL, STARTING AT XL. SEARCH VALUE IS Y.
C
      IX = (Y-VL)/DX+1.0
      IX = MAX(2,MIN(IX,N-2))
      PX = (Y+XL)/DX-FLOAT(IX-1)
      RETURN
      END

```

```

INTS0010
INTS0020
INTS0030
INTS0040
INTS0050
INTS0060
INTS0070
INTS0080
INTS0090
INTS0100

```

```

C
C
C
FUNCTION FINET(Y,I,P)
  REAL Y(1)
  4=POINT (-1,0,+1,+2) LAGRANGE INTERPOLATION
  P1 = P*P-1
  P2 = P-2
  FINET = -P*(P-1)*P*(P-1)/6 + P*(P-2)*Y(1)/2 + P*(P+1)*P*(P+1)/2
  * + P*(P+1)*Y(1)/6
  RETURN
END
FIN0010
FIN0020
FIN0030
FIN0040
FIN0050
FIN0060
FIN0070
FIN0080
FIN0090
FIN0100
FIN0110

```

Reproduced from
best available copy.

```

C
C
C
FUNCTION FINED(Y,I,P,D)
  REAL Y(1)
  4=PT (-1,0,+1,+2) LAGRANGE DIFFERENTIATION
  P1 = 3,*P*P
  FINED = ( (-P1+6.*P-2)*Y(1-1)/6 + (P1-4.*P-1)*Y(1)/2
  * -(P1-2.*P-2)*Y(1+1)/2 + (P1-1)*Y(1+2)/6 )/D
  RETURN
END
FIN0010
FIN0020
FIN0030
FIN0040
FIN0050
FIN0060
FIN0070
FIN0080
FIN0090
FIN0100

```



```

SUBROUTINE L102
C
C GIVEN FRAGMENT FIELD DATA GENERATED BY 'BIOGEM', COMPUTE PROBABILITY
C FUNCTIONS AND OTHER FUNCTIONS OF THE FIELD PARAMETERS.
C FRAGMENT FIELD INPUT IS READ FROM FILE 'FRTAB'.
C FUNCTION VALUE TABLES ARE WRITTEN ON FILE 'FRTAB', FOR USE BY
C A PLOT GENERATION PROGRAM.
C FUNCTION VALUE CONTROL PLOTS CAN ALSO BE GENERATED
C ON THE PRINTER, FOR QUICK-LOOK RESULTS.
C
C CONTROL CARD TABLES AND GENERAL PROGRAM CONTROL VARIABLES.
C COMMON /CNTRL/ GOCRD,SCCRPT,WTCO,FCC(8),KASD(14),PC,PS,PD,PF,
C * TXXIME(2),TXDATE(2),WHY(20),KKARD(800)
C LOGICAL RCRD,SCCRPT,WHY
C INTEGER TXXIME,TXDATE,FCC,PC,PS,PD,PP
C
C LOGICAL PRIDAT,LIST,OWKPLT,EXEC
C EQUIVALENCE (KKARD(489),PRIDAT),(KKARD(377),LIST),
C * (KKARD(353),SKELT),(KKARD(345),EXEC)
C
C IF(EXEC) GO TO 100
C
C READ FUNCTION PARAMETERS, TABLES, ETC.
C
C CALL OUTSET
C GO TO 100
C
C COMPUTE AND OUTPUT FUNCTIONS
C
C 110 CALL COMPUT
C IF (WHY(5)) GO TO 100
C CALL OUTPUT
C
C 100 PC = PS
C IF(LIST.OR.PRIDAT.OR.WHY(5)) PC=PD
C IF (OWKPLT) PC=PF
C RETURN
C END
LINK0010
LINK0020
LINK0030
LINK0040
LINK0050
LINK0060
LINK0070
LINK0080
LINK0090
LINK0100
LINK0110
LINK0120
LINK0130
LINK0140
LINK0150
LINK0160
LINK0170
LINK0180
LINK0190
LINK0200
LINK0210
LINK0220
LINK0230
LINK0240
LINK0250
LINK0260
LINK0270
LINK0280
LINK0290
LINK0300
LINK0310
LINK0320
LINK0330
LINK0340
LINK0350
LINK0360
LINK0370
LINK0380

```

Reproduced from
best available copy.

010
020
030
040
050
060
070
080
090
100
110
120
130
140
150
160

BLACK DATA

2

FUNCTION TABLES, PARAMETERS AND VALUE STORAGE

COMMON /PLOTCH/ NREG,ORNG,NPH,CPH,CPH2,IVC,IVA,IVF,IPOSOP(2),

* NSETS,IPORD,

* YTLORD(12),YTLFPG(12),NPLDTC,NX,NY,XNIN,YMIN,DX,DY,NZ,CH(55),

* RECT(R1,41),ITF(20),ISF(20),ITVF(20),IVPF(20),NF(20),F(14,20),

* ITYLF(12,20),KTF(20),CZM(20),CZMAX(20),XTR(14,12),YTB(14,10),

* C(7,10),NTR(20),KTR(10),OUT(20,16,8),ITTLR(12,8),KTRP(8),OLP(8),

* OWP(8),ISP(9),ITP(8),EZML(8),ITVP(8),ITVS(6),IVPF(8)

12

[illegible]

4.

22

```

SUBROUTINE OUTSET
C
C CONTROL CARD TABLES ARE GENERAL PROGRAM CONTROL VARIABLES
COMMON /CTRL/ GCODE, GCODEPRT, KICD, FCC(8), KAPD(14), PC, PS, PD, PP,
* TXTIME(2), TXDATE(2), WHY(20), KKARD(500)
LOGICAL GCODE, GCODEPRT, WHY
INTEGER TXTIME, TXDATE, FCC, PC, PS, PD, PP
C
LOGICAL PRICAT, LIST, GSOJAC, RANGE, TABLES, LIMITS, PARAMS
EQUIVALENCE (KKARD(400), PRICAT), (TXDATE(1), PRICAT),
* (KKARD(401), GSOJAC), (KKARD(393), RANGE),
* (KKARD(361), TABLES), (KKARD(337), LIMITS), (KKARD(345), PARAMS)
C
FUNCTION TABLES, PARAMETERS AND VALUE STORAGE
COMMON /PLOTCH/ PRIC, DPH2, DPH, DPH2, IVC, IVA, IVE, IPOSOP(2),
* NSETS, IORD,
* TILORD(12), ITLFS(12), NPLOTS, IX, IY, XMIN, XMAX, IY, IZ, CH(55),
* RECT(81, 41), IVE(20), IVE(20), IVE(20), NF(20), F(14, 20),
* ITLFS(12, 20), KTR(20), IZMIN(20), IZMAX(20), XTR(14, 10), YTB(14, 10),
* C(7, 10), NTR(10), KTR(10), OUT(20, 36, 8), ITLFS(12, 8), KTRP(8), KTRP(8),
* OHP(8), ISP(8), ITP(8), PZMIN(8), PZMAX(8), ITYP(4), ITYS(8), IVEF(8)
INTEGER CH
C
INTEGER TXEXT(2), TXINT(2), TXYL(2), TXYLL(2), TXFLL(2)
DATA MEONS/20, MF/14/
DATA TXEXT, TXINT, IY, IX, IY/6HEXTEN, 2HAL, 6HINTER, 2HAL, 1HX, 1HY/
DATA TXYLL, TXYLL, TXFLL/5HX-LIN, 5HX-LOG, 5HY-LIN, 5HY-LOG, 6HLINEAR,
* 6HLOG /
C
IF(RANGE) GO TO 120
IF(GSOJAC) GO TO 140
IF(TABLES) WHY(14) = .FALSE.
IF(LIMITS) WHY(17) = .FALSE.
IF(PARAMS) WHY(16) = .FALSE.
L = 0
C
READ FUNCTION DEFINITIONS (TABLES, LIMITS, PARAMS)
C
C 300 READ (5, 301) IT, IS, IVP, IDUM, ADUM, BDUM, CDUM, DDUM, EDUM
C 301 FORMAT(2I1, 1X, A1, 1X, 3I5, 5E10, 0)

```

Reproduced from
best available copy.

```
IF(RANGE) GO TO 120.  
IF(GSQJAC) GO TO 140.  
IF(TABLES) WHY(14)=.FALSE.  
IF(LIMITS) WHY(17)=.FALSE.  
IF(PARAMS) WHY(19)=.FALSE.  
L = 0
```

READ FUNCTION DEFINITIONS (TABLES, LIMITS, PARAMS)

```

300 READ (5,301) IT,IS,IYP,IPUR,JDUM,ADUM,BDUM,CDUM,DDUM,EDUM
301 FORMAT(2I1,1X,A1,1X,3I5,5F10.0)

```

```

IF(IT,EG,C,AND,IS,EN,0) GO TO 100
IF (IT,LE,C) IT=
IF (IS,LE,C) IS=
L = L+1
IIV = IT*10+IS
IS = IS+1
IF (IT,GT,C) GO TO 322
IT = IC+1C
IS = IT
302 IF (TABLES) GO TO 320
IF (LIMITS) GO TO 340
IF (PARAMS) GO TO 340
IF (NOT,ACCPRT) WRITE(6,FCC) PC,KICD,KARD
ACCPRT = ,TRUE.
WRITE(6,303)
303 FORMAT(46HC***** REQUIRED FIELD(S) MISSING OR IN ERROR)
CALL FLUSH
DO 306 I=16,20
306 WHY(I) = ,TRUE.
GO TO 100

C
C READ TABLES AND/OR TACK COEFS.
C
320 IT = (IT+2)/3
IF(IT,GT,0,AND,IT,LT,3) GO TO 323
IF(,NOT,ACCPRT) WRITE(6,FCC) PC,KICD,KARD
ACCPRT = ,TRUE.
WRITE(6,324)
324 FORMAT(47HC***** ILLEGAL VALUE IN DATA -- FUNCTION CODE)
CALL FLUSH
WHY(16) = ,TRUE.
GO TO 100
321 IF(IT,EG,1) GO TO 322
READ(5,321) (C(I,IS),I=1,7)
321 FORMAT(7F10,7)
IF (PRIDAT) WRITE(6,325) IS,TEXT,(C(I,IS),I=1,7)
325 FORMAT(140,14X,14HTHON COEF, SET NO.,13,24 (.A6,A2,9H FORM)
* 7F10,4)
C(3,IS) = -0.646467*C(3,IS)
C(4,IS) = -C(3,IS)-C(4,IS)

```

```

OUTS0410
OUTS0420
OUTS0430
OUTS0440
OUTS0450
OUTS0460
OUTS0470
OUTS0480
OUTS0490
OUTS0500
OUTS0510
OUTS0520
OUTS0530
OUTS0540
OUTS0550
OUTS0560
OUTS0570
OUTS0580
OUTS0590
OUTS0600
OUTS0610
OUTS0620
OUTS0630
OUTS0640
OUTS0650
OUTS0660
OUTS0670
OUTS0680
OUTS0690
OUTS0700
OUTS0710
OUTS0720
OUTS0730
OUTS0740
OUTS0750
OUTS0760
OUTS0770
OUTS0780
OUTS0790
OUTS0800

```

```

C(5,IS) = -C(5,IV)
IF (LIST) WRITE(6,325) IS, TXINT, C(1,IS), I=1,7)
GO TO 300
322 READ(5,321) (XTR(I,IS), I=1, JUM)
NTR(IS) = IDUM
VTR(IS) = JUM
READ(5,321) (VTR(I,IS), I=1, JUM)
II = IS-1
IX = JUM/10+1
IV = MOD(JUM,10)+1
IF (.NOT. PRDPT) GO TO 328
WRITE (6,327) II, IDUM, TXLL (IA), TXLL (IV)
327 FORMAT(1H0,1A,9) TABLE NO., I, 7, 7- VALUES, 5X, A5, IV, A5)
WRITE(6,326) TX, TXEXT, (VTR(I,IS), I=1, JUM)
326 FORMAT(19X, A1, 7- VALUES (A6, A7, 8- FORV) 7F17.5/(45X, 7F13.5))
WRITE(6,326) TX, TXEXT, (VTR(I,IS), I=1, JUM)
328 IF (IX.EQ.1) GO TO 346
DO 345 I=1, JUM
345 XTR(I,IS) = ALOG10(XTR(I,IS))
346 IF (IV.EQ.1) GO TO 340
DO 347 I=1, JUM
347 VTR(I,IS) = ALOG10(VTR(I,IS))
348 CONTINUE
IF (.NOT. LIST) GO TO 300
IF (.NOT. PRDPT) WRITE (6,327) II, IDUM, TXLL (IV), TXLL (IV)
WRITE(6,326) TX, TXINT, (XTR(I,IS), I=1, JUM)
WRITE(6,326) TX, TXINT, (VTR(I,IS), I=1, JUM)
GO TO 300

C
C READ OUTPUT FOR TYPES (LOG OR LINEAR) AND BOUNDS (ALWAYS LINEAR)
C

340 KTR(IT) = JUM
IX = JUM/10
IV = MOD(JUM,10)+1
IF (PRDPT) WRITE (6,342) IT, TXEXT, TXLL (IV), IX, ADUM, BRUM
342 FORMAT(1H0,1A,8HFCN, '0,13,24 (A6,A2,26H FORM) OUTPUT MODE
- IS , A6, 5X, 17HSMOOTHING ORDER =, I2/19X, 13HPIOT ROUNDS =, E12.4,
* 34 TO, E12.4)
IF (IV.EQ.1 .OR. ADUM.GE.BRUM) GO TO 341
ADUM = ALOG10(ADUM)

```

Reproduced from
best available copy.

```

      RDUM = ALOG10(RDUM)
341  NZMIN(IT) = ADIM
      NZMAX(IT) = RDUM
      IF (LIST) WRITE (6,342) IT,ITY,IY,IXFL(IY),IX,ADIM,BDUM
      GO TO 300

C      READ OUTPUT FOR TITLES AND PARAMETERS
C
360  IF(L,LE,NFCNS) GO TO 341
      WHY(18) = ,TRUE.
363  IF(.NOT.ACCEPT) WRITE(6,FCC) PC,KTCO,KARN
      ACCEPT = ,TRUE.
      WRITE(6,364)
364  FORMAT(42H***** TOO MANY SETS OF OUTPUT FUNCTIONS)
      CALL FLUSH
      GO TO 100
361  IYF(L) = IY
      ISF(L) = IS
      IYF(L) = IY
      IF (IVP.EG,PS) IVP=PN
      IVPF(L) = IVP
      NF(L) = IDUM
      NSETS = L
      READ(5,362) (ITITLE(I,L),I=1,12)
362  FORMAT(12A6)
      IF (PRIDAT.OR,LIST) WRITE(6,365) L,ITY,IVP,IDUM,
      * (ITITLE(I,L),I=1,12)
365  FORMAT(1H0,1A16HOUTPUT DECL. NO.,13,10H, FCC, NO.,13,1H-,A1,
      * 1H.,16,12H PARAMETERS./19X,9HTITLE = ,12A6)
      NO 367 I=1,MF
367  F(I,L) = 0.
      IF(IDUM.LE,0) GO TO 300
      READ(5,321) (F(I,L),I=1,10000)
      IF (PRIDAT.OR,LIST) WRITE(6,366) (F(I,L),I=1,IDUM)
366  FORMAT(19X,12HPARAMETERS =.7F13.5/(31X,7E13.5))
      GO TO 300

C      READ NEW SET OF PLOT RANGE PARAMETERS
C
320  READ(5,121) NX,NY,XMIN,YMIN,DX,DY

```

OUTS1210
 OUTS1220
 OUTS1230
 OUTS1240
 OUTS1250
 OUTS1260
 OUTS1270
 OUTS1280
 OUTS1290
 OUTS1300
 OUTS1310
 OUTS1320
 OUTS1330
 OUTS1340
 OUTS1350
 OUTS1360
 OUTS1370
 OUTS1380
 OUTS1390
 OUTS1400
 OUTS1410
 OUTS1420
 OUTS1430
 OUTS1440
 OUTS1450
 OUTS1460
 OUTS1470
 OUTS1480
 OUTS1490
 OUTS1500
 OUTS1510
 OUTS1520
 OUTS1530
 OUTS1540
 OUTS1550
 OUTS1560
 OUTS1570
 OUTS1580
 OUTS1590
 OUTS1600

```

121 FORMAT(5X,2I5,5X,5F10.4)
      IF (PRTDAT.OR.ELIST) WRITE(6,122) NX,NY,XUT,VMTN,DX,DY
122 FORMAT(140,14X4MXX =,13,5X,4MYY =,13,5X,6MVMN =,E12.4,5X,6MVMN
      * ,E12.4,5X,4MXX =,E12.4,5X,4MYY =,E12.4)
      GO TO 100
      C READ NEW SET OF SOURCE PLOT CHARACTERS, AND SET NO. CONTOUR INTERVALS.
      C
      C
140 READ(5,141) *Z,((CH(I),I=1,Z),CH(N7+1),CH(Z+2),CH(N7+3),CH(NZ+4)
      * ,CH(NZ+5)
141 FORMAT(5X,15,5X,15A1)
      IDUM = N7+4
      IF (LIST.OR.PRTPAT) WRITE(6,142) *Z,((CH(I),I=1,3),I=1,4),
      * (CH(N7+5),I=1,3),((CH(I),I=1,3),I=1,5,10M)
142 FORMAT(140,14X12,3M CONTOUR INTERVALS, 10M ORDER CHARS, 4E, 3A1,
      * 5H AND ,3A1,14H, 10DEF = ,3A1,8H, LOW = ,3A1,9H, HIGH = ,3A1/
      * 19X,10MIN-RANGE =,20(1X,3A1,14,)/(29X,20(1X,3A1,14,)))
100 RETURN
      END

```

Reproduced from
best available copy.

SUBROUTINE COMPU"

```

C
C
C      OUTPUT FUNCTION LOOP
C      OUT(K,IPH,LL) IS A FUNCTION OF LL, PACK(K,IMG), PACK(Y,IMO+1),
C      PACK(K,IMO+2), AMORD(IM), ETC., FOR ALL IDA AND IM.
C
C      CONTROL CARD TABLES AND GENERAL PROGRAM CONTROL VARIABLES
C      COMMON /CNTRL/ GICRD,GCPCRT,KTCO,FCC(B),KAPD(14),PC,PS,PD,PP,
C      * TXIME(2),TXDATE(2),WHY(20),KAPD(800)
C      LOGICAL GCGRD,GCPCRT,WHY
C      INTEGER TXIME,TXDATE,FCC,PC,PS,PD,PP
C      LOGICAL PRYDAT,LIST,PLOT,GKPKLT
C      EQUIVALENCE (KKARD(489),PRYDAT),(KKARD(377),LIST),
C      * (KKARD(369),PLOT),(KKARD(353),GKPKLT)
C
C      FUNCTION TABLES, PARAMETERS AND VALUE STORAGE
C      COMMON /PLOTCH/ BRNG,DRNG,DPH,CPH,DPH2,IVO,IYA,IVF,IPOSOP(2),
C      * NSETS,IDORD,
C      * ITLORD(12),ITLFRG(12),NPLOTS,XMIN,XV,XMIN,YMIN,DX,DY,NZ,CH(55),
C      * RECT(81,41),ITF(20),ISF(20),ITVF(20),IVPF(20),NF(20),F(14,20),
C      * ITTLF(12,20),KTP(20),CZMIN(20),CZMAX(20),YTB(14,10),YTB(14,10),
C      * C(7,17),NTB(10),KTS(10),OUT(20,36,8),ITTL(12,8),KTPP(8),CLP(8),
C      * OHP(8),ISP(8),ITP(8),PZMIN(8),PZMAX(8),ITVP(8),ITVS(8),IVPI(8)
C
C      DIMENSION PACK(20,180),KPKI(60),KPK2(60)
C      DIMENSION SCA(2),A"ORD(30)
C      LOGICAL OTWO
C
C      DATA PI,ELR10,PANEG/3.14159265,2.30258509,57.2957795/
C      DATA MPLOTS/A,IFRT,P/A/
C      CONKE KINETIC ENERGY CONVERSION FACTOR, =2*7000*32.17
C      CONM GRAYS/LB, =7000
C      CONMV MOMENTUM CONVERSION FACTOR, =7000*32.17
C      DATA CONKE,CONM/450380.,7000./,CONMV/225190./
C
C      SELECT FUNCTIONS TO BE EVALUATED, AND PLOT
C      I10 READ(5,111) (ITYS(LP),LP=1,MPLOTS)
C      I11 FORMAT(14(I2,3X))
C      WHY(5) = .FALSE.
C      DO 112 NS=1,MPLOTS

```

COMP0010
 COMP0020
 COMP0030
 COMP0040
 COMP0050
 COMP0060
 COMP0070
 COMP0080
 COMP0090
 COMP0100
 COMP0110
 COMP0120
 COMP0130
 COMP0140
 COMP0150
 COMP0160
 COMP0170
 COMP0180
 COMP0190
 COMP0200
 COMP0210
 COMP0220
 COMP0230
 COMP0240
 COMP0250
 COMP0260
 COMP0270
 COMP0280
 COMP0290
 COMP0300
 COMP0310
 COMP0320
 COMP0330
 COMP0340
 COMP0350
 COMP0360
 COMP0370
 COMP0380
 COMP0390
 COMP0400

COMP0410
COMP0420
COMP0430
COMP0440
COMP0450
COMP0460
COMP0470
COMP0480
COMP0490
COMP0500
COMP0510
COMP0520
COMP0530
COMP0540
COMP0550
COMP0560
COMP0570
COMP0580
COMP0590
COMP0600
COMP0610
COMP0620
COMP0630
COMP0640
COMP0650
COMP0660
COMP0670
COMP0680
COMP0690
COMP0700
COMP0710
COMP0720
COMP0730
COMP0740
COMP0750
COMP0760
COMP0770
COMP0780
COMP0790
COMP0800

```

IF(ITVS(NS).LE.0) GO TO 113
112 CONTINUE
NS = NPLOTIS+1
113 NS = NS-1
IF (LIST.OR.PRTOT) WRITE(6,116) NS,(ITVS(I),I=1,NS)
114 FORMAT(1H0,12X12,16H PLOT REQUESTS.../(29X,14I5))
IF (NS.GT.0) GO TO 119
IF(.NOT.CCCPRT) WRITE(6,500) P0,X100,KARP
CCCRT = .TRUE.
WRITE(6,115)
115 FORMAT(30H0***** NO FUNCTIONAL SELECTED)
GO TO 117
119 IF(PLOT.OR.QUKPLT) GO TO 114
IF(.NOT.CCCPRT) WRITE(6,500) P0,X100,KARP
CCCRT = .TRUE.
WRITE(6,118)
118 FORMAT(47H0***** NEITHER 'PLOT' NOR 'KWIPLOT' SELECTED)
117 WHY(5) = .TRUE.
GO TO 100
C RETURN IF ERROR FLAGS ARE SET
114 IF(WHY(1).OR.WHY(5).OR.WHY(6)) GO TO 100
C READ HEADER INFO FOR CURRENT FRAGMENT FIELD
READ (IFRTAP) JHDR,IPOSOP
READ (IFRTAP) TTLORD,TTLFRG,MN,XK,NRNG,DRNG,NPH
READ (IFRTAP) (ANORD(I),I=1,MN)
I = INT((IPOSOP-1)/2)+1,10000
IVO = PS
IVA = PS
IVF = PS
CALL MOVE(IVO,1,1,IPOSOP,3)
CALL MOVE(IVA,1,1,IPOSOP,4)
CALL MOVE(IVE,1,1,IPOSOP,5)
IMX = 6*MX
IMY = 2*MY
NPH = PI/FLOAT(NPH)
DPH2 = DPH/2.
PHI = DPH2
C PROCESS FRAGMENT FIELD AN AZIMUTH RAY AT A TIME
DO 510 IPH=1,NPH
READ (IFRTAP) (KPK1(J),KPK2(J),J=1,IMY),

```

Reproduced from
best available copy.

```

* ((PACK(I,J),I=1,NSIG),J=1,I * )
DO 515 L=1,NOLPTS
DO 515 K=1,NRIG
515 OUT(K,IPH,LL) = 1.E-30
LL = 0
DO 521 LP=1,NS
IK = 1
ITY = ITVS(LP)
IF (MOD(ITY,3).GT.0) GO TO 542
ITY = ITV-20
IK = 2
DO 550 L=1,NSETS
IF (ITY.EQ.ITVE(LL)) GO TO 561
550 CONTINUE
IF (.NOT.(GOCPRT)) WRITE(4,FOUR) PH,K*CO,KARD
GOCPRT = .TRUE.
IF (IPH.GT.1) GO TO 521
WRITE(4,552) ITV(ITY)
552 FORMAT(1A)***** PLOT CODES:IT,324 NOT DEFINED)
GO TO 521
551 IT = ITF(LL)
IS = ISF(LL)
IVP = IVPE(LL)
STWO = IK.EQ.2 .OR. .NOT.(ITVS(LP)/10,3).EQ.2
DO 553 JK=1,IK
LL = LL+1
IF (LL.LE.NPLOTS) GO TO 512
WHY(5) = .TRUE.
GO TO 363
512 IL = LL+1+IK
IF (IPH.GT.1) GO TO 554
KTPP(LL) = KTP(IT)
CLP(LL) = CZM(IN(IT))
OWP(LL) = OZMAX(IT)
IF (OZMIN(IT).LT.OZMAX(IT)) GO TO 563
OLP(LL) = -1.E+30
IF (MOD(KTP(IT),2).GT.0) CLP(LL)=1.E-30
OWP(LL) = 1.E+30
563 PZMIN(LL) = OZM(IN(IT))
PZMAX(LL) = OZMAX(IT)

```

```

COMP0810
COMP0820
COMP0830
COMP0840
COMP0850
COMP0860
COMP0870
COMP0880
COMP0890
COMP0900
COMP0910
COMP0920
COMP0930
COMP0940
COMP0950
COMP0960
COMP0970
COMP0980
COMP0990
COMP1000
COMP1010
COMP1020
COMP1030
COMP1040
COMP1050
COMP1060
COMP1070
COMP1080
COMP1090
COMP1100
COMP1110
COMP1120
COMP1130
COMP1140
COMP1150
COMP1160
COMP1170
COMP1180
COMP1190
COMP1200

```



```

ITP(LL) = IT
ISP(LL) = IS
ITYP(LL) = ITY
IVPP(LL) = IVP
DO 556 I=1,12
556 ITLP(I,LL) = ITP(I,IS)
554 IF(JK,50,2) GO TO 548
IMO = 1
JM = 0
DO 511 IM=1,IM
W = AMORD(IM)
C LOOK UP THRESHOLD VELOCITY IF USING A DAMAGE THRESHOLD TABLE
IF(IT,5E-1,AND,ITP(2)) GO TO 557
NUM = N
IF (KTR(IS),GE,10) GOTO 558(JM)
IDNA = NTR(IS)
DO 558 I=2,1000
IF(DUM,LF,XTR(I,IS)) GO TO 559
558 CONTINUE
I = 1000
559 VDUM = VTB(I-1,IS)+(VTR(I,IS)-VTR(I-1,IS))/
* (XTR(I,IS)-XTR(I-1,IS))
IF (MDP(KTR(IS),FC),GT,0) VDUM=EXP(VDUM*ELN10)
557 DO 519 IDA=1,2
JI = JM+1
K1 = KPK1(JM)
IF(K1,LE,0) GO TO 519
K2 = KPK2(JM)
DO 520 K=K1,K2
V = PACK(K,IMO)
A = PACK(K,IMO+1)
SCA(1) = ARS(SIN(A))
SCA(2) = ARS(COS(A))
D = PACK(K,IMO+2)
C ELIMINATE CONSIDERATION OF FRAGMENTS FALLING BELOW A PRIORI T.HOLDS
IF (D,LE,0.) GO TO 520
IF (W,LT,F(1,L)) .OR. V,LT,F(2,L)) GO TO 520
FMV = W*V/CONKF
FKE = W*V*V/CONKF
IF (FMV,LT,F(3,L)) .OR. FKE,LT,F(4,L)) GO TO 520
COMP1210
COMP1220
COMP1230
COMP1240
COMP1250
COMP1260
COMP1270
COMP1280
COMP1290
COMP1300
COMP1310
COMP1320
COMP1330
COMP1340
COMP1350
COMP1360
COMP1370
COMP1380
COMP1390
COMP1400
COMP1410
COMP1420
COMP1430
COMP1440
COMP1450
COMP1460
COMP1470
COMP1480
COMP1490
COMP1500
COMP1510
COMP1520
COMP1530
COMP1540
COMP1550
COMP1560
COMP1570
COMP1580
COMP1590
COMP1600

```

[illegible]

```

GO TO 208
C FUNCTION AX -- TOTAL NO. LETHAL FRAGMENTS (SEE TMR COEF, SET CO, X)
C FUNCTION SX -- PROB. OF KILL (SEE TMR COEF, SET CO, X)
C FUNCTION AX -- (DO NOT ADD AX)
220 ROUN = 1.
IF (F(11,L),GT,0.) ROUN=F(11,L)*C(7,15)
COUN = 1.*C(1,15)*F(10,L)*C(2,15)*X*C(3,15)*C(4,15)
* V*C(6,15)*C(7,15)
DO 221 I=1,2
IF (V,GT,SCA(1)*C(5,15)*C(6,15)) GO TO 222
221 CONTINUE
GO TO 520
222 DUM = "
IF (.NOT.CTWO) GO TO 205
ADUM = EXP(-D*(F(8,L)*SCA(1)+F(9,L)*SCA(2)))
IF (IT,NE,3) GO TO 207
GO TO 205
C CONTRIBUTE TO OUT(K,IPH,LL) AND/OR OUT(K,IPH,IL)
C DUM IS A PARTIAL NUMBER DENSITY.
C ADUM IS A PARTIAL DENSITY*ITV.
209 OUT(K,IPH,LL) = OUT(K,IPH,LL)+DUM
GO TO 520
208 OUT(K,IPH,LL) = OUT(K,IPH,LL)+DUM
207 OUT(K,IPH,IL) = (1-OUT(K,IPH,LL))*ADUM
520 CONTINUE
510 I40 = IM0+3
511 CONTINUE
555 IT = IT+1
L = L+1
553 ITV = ITV+10
521 CONTINUE
C SCALE FUNCTION VALUES, TAKE LOG IF NEEDED.
DO 523 LL=1,NPLOTS
IS = ISP(LL)
IT = ITP(LL)
DO 522 K=1,NRNG
DUM = OUT(K,IPH,LL)
IF (ABS(DUM).LE,1.E-38) GO TO 524
IF (IT,NE,19 .AND. IT,NE,20) GO TO 525

```

Reproduction of this document is prohibited.

```

C NORMALIZE BY DIVIDING BY FRAG. NO. DENSITY, FOR FCNS 08 AND 09 ONLY
ILL = LL+1
IF(IT,FS,19) ILL = LL+2
DUM1 = OUT(K,IPH,ILL)
IF(DUM1.NE.0.) DUM=OUT4/DUM1
525 IF (DUM,LT,OLP(ILL)) DUM=OLP(ILL)
IF (DUM,GT,OLP(ILL)) DUM=OLP(ILL)
IF (MOD(KTIPP(ILL),10),GT,0) DUM=ALOG10(DUM)
GO TO 522
C IF NO VALUE HAS BEEN STORED, SET VALUE UNDEFINED (=1.E+38)
524 DUM = 1.E+38
522 OUT(K,IPH,ILL) = DUM
523 CONTINUE
510 CONTINUE
GO TO 100
363 IF(.NOT.GCCPRT) WRITE(6,FCF) PG,KTCO,KARD
GCCPRT = .TRUE.
WRITE(6,364)
364 FORMAT(42H0***** TOO MANY SETS OF OUTPUT FUNCTIONS)
100 CALL FLUSH
RETURN
END
COMP2410
COMP2420
COMP2430
COMP2440
COMP2450
COMP2460
COMP2470
COMP2480
COMP2490
COMP2500
COMP2510
COMP2520
COMP2530
COMP2540
COMP2550
COMP2560
COMP2570
COMP2580
COMP2590
COMP2600
COMP2610
COMP2620

```

```

SURROUTINE OUTPUT
C
C GENERATE PLOTS USING PRINTER AND/OR WRITE FUNCTIONS TO TAPE
C
C CONTROL CARD TABLES AND GENERAL PROGRAM CONTROL VARIABLES
COMMON /CTRL/ GCRD,GCCRD,GCCPRT,KTCO,FCC(R),KARD(14),PC,PS,PD,PP,
* TXTIME(2),TXDATE(2),WHY(20),KWARD(800)
LOGICAL GCCRD,GCCPRT,WHY
INTEGER TXTIME,TXDATE,FCC,PC,PS,PD,PP
LOGICAL LIST,PLOT,GNKPLT
EQUIVALENCE (KWARD(377),L,ST),(KWARD(369),PLOT),
* (KWARD(353),GNKPLT)
C
C FUNCTION TABLES, PARAMETERS AND VALUE STORAGE
COMMON /PLOTCK/ NR,G,DRNG,NPH,DPH,DPH2,IWD,IVA,IVE,IPOSOP(2),
* NSETS,INORD,
* TTLORD(12),TTLFRG(12),NPLOTS,NX,NY,XMIN,YMIN,DX,DY,NZ,CH(55),
* RECT(R1,41),ITF(20),ISF(20),ITYF(20),IVPF(20),NF(20),F(14,20),
* ITTLF(12,20),KTP(20),CZMIN(20),CZMAX(20),XTR(14,10),YTB(14,10),
* C(7,10),NTB(10),KTB(10),OUT(20,36,6),ITLTP(12,8),KTPP(8),OLP(8),
* OHP(8),ISP(R),ITP(R),PZMI(R),PZMAX(R),ITYP(R),ITYS(R),IVPP(R),
INTEGER CH
C
EXTERNAL PURPLE
DIMENSION TT(41,4),TEMP(6,4),KTEMP(6,4),TEMP3(6,4)
EQUIVALENCE (TT(1,1),TEMP(1,1)),(TT(1,3),KTEMP(1,1)),
* (TT(1,5),TEMP3(1,1))
DATA MXRNG/20/
DATA ELN10/2,30258509/
DATA MXORDR/5/,IHDR/6H*HEAD*/
DATA IPLTAP/2/
C
DO 600 LL=1,NPLOTS
IF(.NOT.PLOT) GO TO 604
I = IFX0(IPOSOP,6,2,0,ITVP(LL))
CALL MOVE(IPOSOP,8,1,IVPP(LL))
WRITE (IPLTAP) IHDR,IPOSOP
WRITE (IPLTAP) TTLORD
WRITE (IPLTAP) TTLFRG
WRITE (IPLTAP) (ITLTP(1,LL),I=1,12)

```

Reproduced from
best available copy.

```

604 PC = PD
IF (QWKPLT) PC=PP
IF (.NOT. (LIST.OR.QWKPLT)) GO TO 605
WRITE (6,602) PC,IORD,IVG,ITLORD,IVA,IVF,ITLFCG,ITYP(LL),
* IVP(LL),CITLPL(I,LL),I=1,12)
602 FORMAT(A1,18X,19HDATA I.D.      =,15,1H-,A1,5X,12A6/
* 19X,19HFRAG I.D.      =,4X,A1,1H-,A1,5X,12A6/
* 19X,19HPLAT I.D.      =,15,1H-,A1,5X,12A6)
IF (PLOT) WRITE(6,603) IORDOP
603 FORMAT(20H PLOT TAPE KEY      =,1X,A6,A2)
WRITE (6,607)
607 FORMAT(1X)
C FCN. VALUES COMPUTED ON POLAR GRID. RE-EXPRESS ON A RECT. GRID.
605 ZMIN = 1.E38
ZMAX = -1.E38
DO 606 I=1,NPNC
DO 606 J=1,NPH
IF (OUT(I,J,LL),GE.(1.E+38)) GO TO 606
ZMIN = AMIN1(ZMIN,OUT(I,J,LL))
ZMAX = AMAX1(ZMAX,OUT(I,J,LL))
606 CONTINUE
Y = YMIN
DO 610 IV=1,NY
X = XMIN
DO 611 IX=1,NX
RECT(IX,IV) =
* POLCAR(OUT(1,1,LL),MXPNG,NRNG,NPH,DRNG,DPH,DRNG,DPH2,X,Y)
611 X = X+DX
610 Y = Y+DY
C OPTIONALLY SMOOTH GRID. (RECT(I,J)=AVG OF IT AND MANY NEIGHBORS)
IORDER = KTOP(LL)/10
IOR = MINO(MXQORD,IORDER)
IF (IOR.LE.0) GO TO 613
IOR1 = IOR+1
IDUM = NX+IOR1
II = 0
IXOR = 1-IOR1
DO 640 IX=1,IDUM
DO 641 IV=1,NY
II = II+1

```

```

OUTP0410
OUTP0420
OUTP0430
OUTP0440
OUTP0450
OUTP0460
OUTP0470
OUTP0480
OUTP0490
OUTP0500
OUTP0510
OUTP0520
OUTP0530
OUTP0540
OUTP0550
OUTP0560
OUTP0570
OUTP0580
OUTP0590
OUTP0600
OUTP0610
OUTP0620
OUTP0630
OUTP0640
OUTP0650
OUTP0660
OUTP0670
OUTP0680
OUTP0690
OUTP0700
OUTP0710
OUTP0720
OUTP0730
OUTP0740
OUTP0750
OUTP0760
OUTP0770
OUTP0780
OUTP0790
OUTP0800

```

```

        IF(IX,GT,IOR1) RECT(IXOR,IY)=TT(IY,II)
        IF(IX,GT,NX) GO TO 641
        IL = MAX0(1,IX-IOR)
        IH = MIN0(NX,IX+IOR)
        JL = MAX0(1,IY-IOR)
        JH = MIN0(NY,IY+IOR)
        DUM = 0.
        DO 642 I=IL,IH
        DO 643 J=JL,JH
        DUM = DUM+RECT(I,J)
        644 CONTINUE
        643 CONTINUE
        642 CONTINUE
        TT(IY,II) = DUM/FLOAT((IH-IL+1)*(JH-JL+1))
        641 CONTINUE
        640 IXOR = IXOR+1
        613 IF (PZMIN(LL).GE.PZMAX(LL)) GO TO 612
        ZMIN = PZMIN(LL)
        ZMAX = PZMAX(LL)
        612 CONTINUE
        C IF REQUESTED, OUTPUT FUNCTION IN POLAR AND RECT. FORM
        IF (.NOT.PLOT) GO TO 645
        WRITE (IPLTAP) NRNG,NPH,DRNG,DPH,ZMIN,ZMAX
        WRITE (IPLTAP) ((OUT(I,J,LL),I=1,NRNG),J=1,NPH)
        WRITE (IPLTAP) NX,NY,XMIN,YMIN,DX,DY,ZMIN,ZMAX
        WRITE (IPLTAP) ((RECT(I,J),I=1,NX),J=1,NY)
        C GEN: PRINTER CONTOUR PLOT OF FEN. (IF GWKPLT ST)
        645 IF (.NOT.GWKPLT) GO TO 600
        CALL SOJAC(PURPLE,YMIN,YMIN+FLOAT(NY-1)*DY,XMIN,XMIN+FLOAT(NX-1)*
        * DX,ZMIN,ZMAX,NX,1,NZ,CH)
        NZ = (ZMAX-ZMIN)/FLOAT(NZ)
        Z = ZMIN
        NL = (NZ+5)/4
        NI = NL-(NL*4-(NZ+2))
        K = 3
        DO 682 J=1,4
        DO 680 I=1,NL
        K = K+1
        T = P(I,J) = Z
        IF (MOD(KTTP(LL),10).GT,0) TEMP(I,J)=EXP(TEMP(I,J)*ELN10)
        KTEMP(I,J) = CH(K)

```

```

OUTP0810
OUTP0820
OUTP0830
OUTP0840
OUTP0850
OUTP0860
OUTP0870
OUTP0880
OUTP0890
OUTP0900
OUTP0910
OUTP0920
OUTP0930
OUTP0940
OUTP0950
OUTP0960
OUTP0970
OUTP0980
OUTP0990
OUTP1000
OUTP1010
OUTP1020
OUTP1030
OUTP1040
OUTP1050
OUTP1060
OUTP1070
OUTP1080
OUTP1090
OUTP1100
OUTP1110
OUTP1120
OUTP1130
OUTP1140
OUTP1150
OUTP1160
OUTP1170
OUTP1180
OUTP1190
OUTP1200

```

```

480 Z = Z+DZ
482 CONTINUE
  NJ = NL-1
  DO 684 J=1,4
    IF(NJ,LE,0) GO TO 687
    DO 686 I=1,NJ
      684 TEMQ(I+1,J) = TEMP(I,J)
      687 IF(J,EG,4) GO TO 684
      TEMQ(1,J+1) = TEMP(NL,J)
      684 CONTINUE
      TEMQ(1,1) = -1.E78
      TEMP(NI,4) = +1.F38
      NJ = 4
      WRITE(6,685)
      685 FORMAT(/57X,19HTHUS SCALES SOJAC.,/)
      DO 683 I=1,NL
        IF(I,GT,NI) NJ=3
        WRITE (6,681) (TEMP(I,J),(K=1,3),TEMP(I,J),J=1,NJ)
        681 FORMAT(4(3X,1PE10,2,4H LF ,3A1.3H LT,E10.2))
      683 CONTINUE
      600 CONTINUE
      100 RETURN
      END

```

C

```

OUTP1210
OUTP1220
OUTP1230
OUTP1240
OUTP1250
OUTP1260
OUTP1270
OUTP1280
OUTP1290
OUTP1300
OUTP1310
OUTP1320
OUTP1330
OUTP1340
OUTP1350
OUTP1360
OUTP1370
OUTP1380
OUTP1390
OUTP1400
OUTP1410
OUTP1420
OUTP1430
OUTP1440

```

```

      FUNCTION PURPLE(X,Y)
      C FUNCTION TABLES, PARAMETERS AND VALUE STORAGE
      COMMON /PLOTCH/ ARNG,DENG,NPH,CPH,CPH2,IVO,IYA,IVF,IPOSOP(?),
      * NSETS,IWORD,
      * TFLORD(12),ITLFFG(12),NPLOTS,NX,NY,XMIN,YMIN,CX,CY,NZ,ICH(55),
      * RECT(R1,41),ITF(20),ISF(20),IVPF(20),NF(20),F(14,20),
      * ITTLF(12,20),KTF(20),OZMIN(20),OZMAX(20),XTB(14,10),YTB(14,10),
      * C(7,10),NTB(10),KTB(10),OUT(20,36,8),ITTLP(12,8),ITPB(8),OLP(P),
      * OHP(8),ISP(8),ITP(8),PZMIN(8),PZMAX(8),ITVP(8),ITVS(8),IVPP(8)
      C
      C COMMON BLOCK PASSED FROM 'SOJAC'
      COMMON /CSOJAC/ CX,GY,IXP,IYP,IX,IY
      C
      C SUPPLIES VALUES OF RECT TO 'SOJAC' (PRINTER CONTOUR PLOTTER)
      C
      XX = (X-YMIN)/GY+1.
      IXX = XX
      XX = 1.0AMOD(XX,1.)
      IXX = MIN0(NY-1,IXX)
      IXX = NY-IXX
      IF(RECT(IY,IXX),1,1.E+3F .AND. RECT(IY,IXX+1),1,1.E+38) GO TO 1
      PURPLE = 1.E+3A
      RETURN
      1 PURPLE = RECT(IY,IXX)*(1,-XX)+RECT(IY,IXX+1)*XY
      RETURN
      END
PURP0010
PURP0020
PURP0030
PURP0040
PURP0050
PURP0060
PURP0070
PURP0080
PURP0090
PURP0100
PURP0110
PURP0120
PURP0130
PURP0140
PURP0150
PURP0160
PURP0170
PURP0180
PURP0190
PURP0200
PURP0210
PURP0220
PURP0230
PURP0240
PURP0250
PURP0260

```

Reproduced from
best available copy.


```

C *****
C * CCARD *      DATA HOUSE      IITRI      SEPT 1970
C *****
C LOGICAL FUNCTION CCARD(S1,C1,L1,IACT,W)
C INTERPRET CONTROL CARD FIELDS IDENTIFIED BY SYNTAX.
C
C S1      TEXT STRING.
C C1      POINTER TO START OF CURRENT FIELD
C L1      LENGTH OF CURRENT FIELD
C IACT    ACTION CODE RETRIEVED BY SYNTAX FROM SYNTAX RULES
C W       GENERAL ARRAY FOR CONTROL CARDS..
C         W(1)--W(10).. FLAGS AND COUNTERS FOR USE BY AND COMMUNI-
C           CATION WITH SYNTAX
C         W(11).. NUMBER OF CONTROL CARD ENTRIES TIMES 7 PLUS 20.
C         W(12)--W(20).. HOLD OUTPUT VALUES FOR USE BY PROGRAM
C           CALLING SYNTAX
C         W(12).. CONTROL CARD TYPE CODE
C         W(13).. NON-SYNTAX ERROR COLUMN (=0 IF NONE)
C         W(14),W(15).. POINT TO START AND END OF FIELD ORDINAL
C           LIST (NOT USED IF W(14).GT.W(15))
C         W(16),W(17).. POINT TO START AND END OF POSITIONAL FIELD
C           LIST (NOT USED IF W(16).GT.W(17))
C
C UPON RETURN, CCARD=. TRUE, IF FIELD IS OK, OR CCARD=.FALSE. IF NOT
C
C CONTROL CARD TYPE 1 IS RESERVED FOR C.CDS. WHICH NEVER REQUIRE
C FURTHER USER ACTION. (I.E., CARD MERELY SETS FLAGS AND VALUES)
C FIELD TYPE CODE 9, IS RESERVED FOR THE 'FIELD DEFAULT VALUE
C REDEFINITION' FLAG. THE REMAINDER OF A C.CD. ON WHICH THIS FLAG
C APPEARS CONSISTS OF FIELD DEF. VAL. REDEFNS. (NOTE.. FLAG DEFAULT
C SETTINGS ARE ENTERED IN THE LOGICAL FORM (NAME=T, NAME=.FALSE.,
C ETC.) DO NOT SIMPLY ENTER 'NAME' OR 'NO'+NAME!)
C
C INTEGER S1(1)
C INTEGER C1,W(1),NULLV(5),INULL(3)
C LOGICAL CRIPE,POSIT,EXIST,INULL,DEFINE
C REAL FNULL
C
C KV AND RV ARE EQUIVALENT NAMES FOR THE 1ST WORD OF KVA.
C
CCARD0010
CCARD0020
CCARD0030
CCARD0040
CCARD0050
CCARD0060
CCARD0070
CCARD0080
CCARD0090
CCARD0100
CCARD0110
CCARD0120
CCARD0130
CCARD0140
CCARD0150
CCARD0160
CCARD0170
CCARD0180
CCARD0190
CCARD0200
CCARD0210
CCARD0220
CCARD0230
CCARD0240
CCARD0250
CCARD0260
CCARD0270
CCARD0280
CCARD0290
CCARD0300
CCARD0310
CCARD0320
CCARD0330
CCARD0340
CCARD0350
CCARD0360
CCARD0370
CCARD0380
CCARD0390
CCARD0400

```

Reproduced from
best available copy.

```

C      KV IS ALWAYS USED FOR LOGICAL AND INTEGER VALUES.
C      RV IS ALWAYS USED FOR REAL VALUES.
C      INTEGER KVA(2)
C      EQUIVALENCE (KVA(1),KV,RV)
C      DATA IRL/14/,INULL,ENULL,FNULL/0,0,1H,.,FALSE,.,0,0/
C      VALUE OF NCHW IS MACHINE DEPENDENT -- MAX, NO CHAR, IN NAME OR VAL
C      DATA NCHW/12/
C      KNL IS MACH. DEP. -- NUMBER OF WORDS ALLOWED FOR A NAME OF VALUE
C      DATA KNL/2/
C      EQUIVALENCE (NULLV(1),INUL(1)),(NULLV(4),INUL(5),FNULL),
C      * (EXIST,EXIST)
C      LMAX = MIN(41,NCHW)
C      GO TO (100,200,300,400,500,600,700,800,900,1000),IACT
C      CONTROL CARD NAME
C      100 N = W(11)
C      W(12) = 0
C      W(13) = 0
C      DO 101 KK=21,.,7
C      I = W( KK+1)
C      102 IF(I.LE.0) GO TO 101
C      II = I+KNL
C      J = KOML(W(1),1,IAPS(W(1)),S1,C1,LMAX)
C      IF (J.FG.0) GO TO 110
C      IF (J.NE.-1) GO TO 104
C      103 IF(W(11).LT.0) GO TO 110
C      104 I = W(II+1)
C      GO TO 102
C      101 CONTINUE
C      GO TO 9
C      CONTROL CARD NAME FOUND
C      110 W(12)= W(KK)
C      KPOS1 = W(KK+2)
C      KPOS = KPOS1
C      W(16)= KPOS1
C      KPOS2 = W(KK+3)-1
C      KEY1 = W(KK+3)
C      KEY2 = W(KK+4)

```

```

KOR1 = W(KK+5)
KOR = KOR1
W(14) = KOR1
KOP2 = W(KK+4)
C SET DEFAULT FIELDS
  IF(KPOS1,GT,KEY2) GO TO 102
  NO 115 K=KPCS1,KEY2
  I = W(K)
  W(I+2) = 0
  IF(W(I),GE,81000) GO TO 115
  W(I+3) = W(I)/1000
  II = I+KML+4
  IF(MOD(W(I),10)-4) 113,114,114
C** LOADING OF DEFAULT VALUE MAY REQUIRE A DO LOOP
  113 W(I+4) = W(II)
  W(I+5) = W(II+1)
  GO TO 115
  114 W(I+4) = W(II)
  GO TO 115
  116 L = MINO(W(I+4),W(II))
  K1 = W(I+5)
  K2 = W(II+1)
  CALL MOVE(W(K1),I,L,W(K2),1)
  J = W(I+4)-L
  IF(J,LE,0) GO TO 115
  CALL MOVE(W(K1),L+1,I+1,W(K1),1)
  CALL MOVE(W(K1),L+2,J-1,W(K1),I+1)
  115 CONTINUE
  120 GRIPE = .FALSE.
  DEFINE = .FALSE.
  GO TO 1
C
C POSITIONAL VALUE OR FIELD NAME
C
  200 KS = C1
  LS = L"MAX
  LS1 = L1
  GO TO 1
C
C PRIOR WORD WAS A FIELD NAME (FOLLOWED BY A VALUE)

```

Reproduced from copy.
best available copy.

```

C
300 POSIT = .FALSE.
C ALL REG. POSIT. FIELDS MUST BE SUPPLIED PRIOR TO 1ST KEY TYPE FIELD
IF(KPOS.GT.KPOS2) GO TO 410
KW = W(KPOS)
IF(W(KV).GE.52000) GO TO 5
GO TO 410

C
C FIELD VALUE. INTERPRET AS INTEGER, ALPHA, LOGICAL, OR REAL.
C
400 KS = C1
LS = LMAX
LS1 = L1
410 K2 = K2+1
IF(K2.LE.C) GO TO 1
KV = C
L = 0
GO TO (440,440,420,430,440,450,460,470),K2
C INTEGER
420 IF (INTI(S1,KS,LS1,0,KV),NE.1) GO TO 495
GO TO 405
C ALPHAMERIC
C** BLANK OUT ALL WORDS OF KVA BEFORE MOVING TEXT
430 KV = IRL
KVA(2) = IRL
CALL MOVE(KV,1,LS,S1,KS)
L = LS
GO TO 405
C LOGICAL
440 IF (ILGI(S1,KS,LS1,0,KV),NE.1) GO TO 405
GO TO 405
C REAL
450 IF (IRFI(S1,KS,LS1,0,0,KV),NE.1) GO TO 495
GO TO 405
C LONG-ALPHAMERIC
460 IF(DEFINE) GO TO 465
LS1 = WIND(MAP,LS1)
CALL MOVE(W(JAP),1,LS1,S1,KS)
W(KW+2) = 1
W(KW+3) = LS1

```

```

CCAR1210
CCAR1220
CCAR1230
CCAR1240
CCAR1250
CCAR1260
CCAR1270
CCAR1280
CCAR1290
CCAR1300
CCAR1310
CCAR1320
CCAR1330
CCAR1340
CCAR1350
CCAR1360
CCAR1370
CCAR1380
CCAR1390
CCAR1400
CCAR1410
CCAR1420
CCAR1430
CCAR1440
CCAR1450
CCAR1460
CCAR1470
CCAR1480
CCAR1490
CCAR1500
CCAR1510
CCAR1520
CCAR1530
CCAR1540
CCAR1550
CCAR1560
CCAR1570
CCAR1580
CCAR1590
CCAR1600

```

CCAR1610
CCAR1620
CCAR1630
CCAR1640
CCAR1650
CCAR1660
CCAR1670
CCAR1680
CCAR1690
CCAR1700
CCAR1710
CCAR1720
CCAR1730
CCAR1740
CCAR1750
CCAR1760
CCAR1770
CCAR1780
CCAR1790
CCAR1800
CCAR1810
CCAR1820
CCAR1830
CCAR1840
CCAR1850
CCAR1860
CCAR1870
CCAR1880
CCAR1890
CCAR1900
CCAR1910
CCAR1920
CCAR1930
CCAR1940
CCAR1950
CCAR1960
CCAR1970
CCAR1980
CCAR1990
CCAR2000



```

GO TO 1
465 IF(W(K),GE,91000) GO TO 5
  S1 = W(K),LS1
  CALL MOVE(W(K),LS1,LS2)
  W(K) = LS1+100+100*(K-1)
  J = 100-LS1
  IF(J,LE,0) GO TO 1
  CALL MOVE(W(K),LS1,LS2)
  CALL MOVE(W(K),LS2,LS3)
  GO TO 1
C SYNOLIC -- SEARCH DICTIONARY FOR MATCH WITH SYMBOL
470 I = W(KW+5)
  L = 0
  KV = 0
471 IF(I,LE,0) GO TO 405
  II = I+KAL
  KV = KV+1
  J = KC(L*(I+1),1,1ARS(K(I)),S1,KC,LS)
  IF (J,EG,0) GO TO 405
  IF (J,NE,-1) GO TO 472
473 IF(W(II),LT,0) GO TO 405
472 I = W(II+1)
  GO TO 471
C CONCLUSION
405 IF(DEFINE) GO TO 477
  W(KW+2) = 1
  W(KW+3) = L
C A DO LOOP MAY BE REQUIRED TO MOVE ALL OF KVA
  W(KW+4) = KV
  IF(K2,GE,7) GO TO 1
  W(KW+5) = KVA(2)
  GO TO 1
C A DO LOOP MAY BE REQUIRED TO MOVE ALL OF KVA
407 IF(W(K),GE,91000) GO TO 5
  II = KW+KWL+4
  W(II) = KV
  IF(K2,LT,7) W(II+1)=KVA(2)
  W(KW) = L*1000+MOD(W(K),1000)
  GO TO 1
C VALUE ERROR

```

```

495 W(KK+2) = -1
    IF(DEFINE) GO TO 5
    W(KK+3) = 0
    W(KK+4) = 0
    IF(K2,GE,7) GO TO 5
    W(KK+5) = 0
    GO TO 5

C PRIOR WORD WAS EITHER A FIELD NAME (W/C A VALUE) OR A POSIT. VALUE
C
C 500 POSIT = .TRUE.
    IF(DEFINE) GO TO 5
C IF ANY REG. POSIT. FIELDS REMAIN UNSATISFIED, DO NOT SRC FOR KEYWORD
    IF(KPOS,GT,KPOS2) GO TO 515
    KW = *(KPOS)
    IF(W(KW),GE,P2000) GO TO 550
C SEARCH FOR POSSIBLE FIELD NAME
C
C 510 IF(.NOT,DEFINE) GO TO 515
    IF (INT1(S1,KS,LS,0,K),NE,1) GO TO 515
    IF(K,LE,0) GO TO 5
    K = KPOS1-1+K
    IF(K,GT,KPOS2) GO TO 5
    KW = K(K)
    K2 = MOD(W(KW),10)
    IF(KOR,GT,KOR2) GO TO 1
    W(KOR) = KW
    KOR = KOR+1
    GO TO 1
C 515 DO 511 KEY=KEY1,KEY2
    KW = W(KEY)
    IF(W(KK+2),NE,0) GO TO 511
    K2 = MOD(W(KW),10)
    K = W(KW+1)
C IF K=V TYPE FIELD ENCOUNTERED, DO NOT SEARCH K OR V TYPE FIELD ENTR.
C IF K OR V TYPE FIELD ENCOUNTERED, DO NOT SRC K=V TYPE ENTRIES
    EXIST = K,LE,0 .OR. K2,LE,1 .OR. K2,GE,9
    IF(POSIT .AND. .NOT,EXIST .OF. .NOT,POSIT .AND. EXIST) GO TO 511
    512 IF(K,LE,0) GO TO 511

```

CCAR2010
 CCAR2020
 CCAR2030
 CCAR2040
 CCAR2050
 CCAR2060
 CCAR2070
 CCAR2080
 CCAR2090
 CCAR2100
 CCAR2110
 CCAR2120
 CCAR2130
 CCAR2140
 CCAR2150
 CCAR2160
 CCAR2170
 CCAR2180
 CCAR2190
 CCAR2200
 CCAR2210
 CCAR2220
 CCAR2230
 CCAR2240
 CCAR2250
 CCAR2260
 CCAR2270
 CCAR2280
 CCAR2290
 CCAR2300
 CCAR2310
 CCAR2320
 CCAR2330
 CCAR2340
 CCAR2350
 CCAR2360
 CCAR2370
 CCAR2380
 CCAR2390
 CCAR2400

CCAR2410
CCAR2420
CCAR2430
CCAR2440
CCAR2450
CCAR2460
CCAR2470
CCAR2480
CCAR2490
CCAR2500
CCAR2510
CCAR2520
CCAR2530
CCAR2540
CCAR2550
CCAR2560
CCAR2570
CCAR2580
CCAR2590
CCAR2600
CCAR2610
CCAR2620
CCAR2630
CCAR2640
CCAR2650
CCAR2660
CCAR2670
CCAR2680
CCAR2690
CCAR2700
CCAR2710
CCAR2720
CCAR2730
CCAR2740
CCAR2750
CCAR2760
CCAR2770
CCAR2780
CCAR2790
CCAR2800

Reproduced from
best available copy.

```

11 = *K*L
41 = IARG(1)
1 = K*L( (K) ) * K1 * G1 * S1 * G1
IF (IARG(1).GT.1) GO TO 41
15 (1) 510,520,519
513 IF(K2,1,1) GO TO 519
IF(LS,1,1) GO TO 519
IF(KOR(1),1,1) GO TO 519
1 = K*L( (K) ) * K1 * G1 * S1 * G1
IF (1,1,1) GO TO 521
IF (1,1,1) GO TO 519
514 IF(W(1),LT,0) GO TO 521
GO TO 519
518 IF(W(1),LT,0) GO TO 521
519 K = W(1)+1
GO TO 512
511 CONTINUE
C NO MATCH FOUND -- OK IF THIS IS ADDITIONAL
IF(KPOS,GT,KPOS2,OK,NOT,POSIT) GO TO 5
550 KW = K(KPOS)
KPOS = KPOS+1
K2 = MOD(W(KV),10)
IF(KOR,GT,KOR2) GO TO 554
W(KOR) = KW
KOR = KOR+1
551 IF(POSIT) GO TO 560
K2 = MOD(W(KV),10)
GO TO 523
C PREPARE FIELD (IF LONG-ALPHA)
560 IF(MOD(W(KV),10).NE,4) GO TO 440
ASSIGN 410 TO KGF
GO TO 546
C FIELD NAME (OR *+NAME) FOUND
520 EXIST = .TRUE.
GO TO 522
521 EXIST = .FALSE.
522 IF(DEFINE) GO TO 523
IF(K2,FG,9) GO TO 540
523 IF(KOR,GT,KOR2) GO TO 530
W(KOR) = KW

```

```

      KOP = KOP+1
C  SET EXISTENCE OR NULL VALUES
530 IF(K2,GT,1) GO TO 535
      W(KK+2) = 1
      W(KK+4) = IEXIST
      GO TO 1
535 IF(K2=6) 536,545,555
536 W(KK+4) = NULLV(K2)
      IF(K2,EG,3) W(KK+5)=NULLV(K2)
      W(KK+3) = 0
      JAP = KW+4
      LAP = NCHW
      MAP = NCHW
      KAP = 1
      GO TO 1
C  FIELD IS LOG-ALPHAFETIC
545 ASSIGN 1 TO KGO
546 II = KW+4
      IF (DEFINE) II=II+KNI
      JAP = W(II+1)
      LAP = W(II)
      MAP = LAP
      KAP = 1
      IF(LAP,LE,0) GO TO 549
      CALL MOVE(W(JAP),1,1,1W,1)
      CALL MOVE(W(JAP),2,LAP-1,0(JAP),1)
      IF (DEFINE) GO TO 547
      W(KK+3) = 0
      GO TO 549
547 W(KK) = MON(W(KK),1000)
549 GO TO KGO
C  SYMBOLIC -- COLLECT ARGUMENT FOR A SEARCH
555 W(KK+4) = 0
      GO TO 1
C  IF A FIELD OF TYPE 9 IS ENCOUNTERED, THEN REST OF CARD SETS DEFAULTS
540 DEFINE = ,TRUE,
      W(12) = 1
      GO TO 1
C
C  END OF CONTROL CARD -- CHECK FOR MISSING REQUIRED FIELDS

```

CCAR2810
 CCAR2820
 CCAR2830
 CCAR2840
 CCAR2850
 CCAR2860
 CCAR2870
 CCAR2880
 CCAR2890
 CCAR2900
 CCAR2910
 CCAR2920
 CCAR2930
 CCAR2940
 CCAR2950
 CCAR2960
 CCAR2970
 CCAR2980
 CCAR2990
 CCAR3000
 CCAR3010
 CCAR3020
 CCAR3030
 CCAR3040
 CCAR3050
 CCAR3060
 CCAR3070
 CCAR3080
 CCAR3090
 CCAR3100
 CCAR3110
 CCAR3120
 CCAR3130
 CCAR3140
 CCAR3150
 CCAR3160
 CCAR3170
 CCAR3180
 CCAR3190
 CCAR3200

```

C
C
400 IF (KPOS1.GT.KPOS2) GO TO 410
C 610 X=KPOS1,KPOS2
C I = V(V)
IF (X(1).LT.522200) GO TO 410
IF (X(1+2).LE.0) GO TO 430
410 CONTINUE
420 IF (GRIP) GO TO 430
CCARD = 'TRUP'
GO TO 440
430 IF (GRIP) GO TO 410
IGRIP = 01
435 W(13) = IGRIP
CCARD = 'FALSE'
440 W(15) = KPOS-1
W(17) = WPOS-1
RETURN
C
C AND TO *-TYPE STRING (PROHIBITED FOR *-ALPHABETIC FIELDS)
C
700 IF (K2.LE.3) X=X+101E.61 OF 1014
CALL CAT(W(CARD),MAP-LAP,SIGMA)
IF (DEFINE) GO TO 705
W(KW+2) = 2
W(KW+3) = MAP-LAP
GO TO 1
705 W(KW) = (MAP-LAP)*1000+WPOS*(K-110000)
GO TO 1
C
C START OF POSITIONAL *-TYPE ALPHA VALUE (RESERVE POSIT. FIELD ENTRY)
C
800 IF (DEFINE) GO TO 4
POSIT = 'FLGS'
IF (KPOS1.GT.KPOS2) GO TO 4
GO TO 550
C
C RETURNS
C
900 CONTINUE
1000 CONTINUE

```



```

9 CCARP = .FALSE.
  RETURN
6 IF(GRIPE) GO TO 4
  IGRIPE = C1+L1
  GO TO 7
5 IF(GRIPE) GO TO 4
  IGRIPE = K+LS
7 GRIPE = .TRUE.
1 CCARP = .TRUE.
  RETURN
END

```

```

CCAR361D
CCAR362D
CCAR363D
CCAR364D
CCAR365D
CCAR366D
CCAR367D
CCAR368D
CCAR369D
CCAR370D
CCAR371D

```

CAT 0010
CAT 0020
CAT 0030
CAT 0040
CAT 0050
CAT 0060
CAT 0070
CAT 0080
CAT 0090
CAT 0100
CAT 0110
CAT 0120
CAT 0130
CAT 0140
CAT 0150
CAT 0160
CAT 0170
CAT 0180
CAT 0190
CAT 0200
CAT 0210
CAT 0220
CAT 0230

```

*****
* CAT *
*****
SUBROUTINE CAT (S1,C1,L1,S2,C2,L2)
  INTEGER S1(1),C1(1),S2(1),C2(1)
  CHARACTER L(1000)
  L = ' '
  IF (L1.LT.0) RETURN
  CALL RCHXTR(S1,C1)
  CALL RCHXTR(S2,C2)
  DO I = 1,L1
    CALL SCHXTR(S1,C1,I,C2,C2+1)
    C1 = C1+1
    L1 = L1-1
  RETURN
END

```

Reproduced from
best available copy.

1 MEO
 IF(,NOT,CS) RETURN
 C = C+L
 L = L-L
 RETURN
 30 IFX0 = -1
 RETURN
 END

IFX00410
 IFX00420
 IFX00430
 IFX00440
 IFX00450
 IFX00460
 IFX00470
 IFX00480

Reproduced from
 best available copy.


```

INT10410
INT10420
INT10430
INT10440
INT10450
INT10460
INT10470
INT10480
INT10490
INT10500
INT10510
INT10520
INT10530
INT10540
INT10550
INT10560
INT10570
INT10580
INT10590
INT10600
INT10610

```

```

IF(1.E0,IP,05,1.E6,IP6) GO TO 16
18 IF(1.LT,100,09,1.E7,109) GO TO 20
N = N+10+1-IP0
IF (NS.EQ,0) GO TO 14
GO TO 10
15 NS = -1
GO TO 10
16 NS = +1
10 CONTINUE
K = C+LL
GO TO 25
20 IF (NS.EQ,0) GO TO 30
INTI = 0
25 IF(VS.LT,0) N=-N
GO TO 35
30 INTI = -1
35 IF(.NOT.QS) RETURN
L = L-K+C
C = K
RETURN
END

```

[illegible]

```

      IF (OS) IL=MIN(L, )
      C SET UP CONVERSION
      IK = C
      IF (IL,LE,C) GO TO 43
      IREI = 1
      IXV = C
      IX = 1
      ID = C
      EV = 5.00
      CX = .FALSE.
      C SEARCH FOR MINUS SIGN (TO ALLOW PROPER HANDLING OF -0.XXX NUMBERS)
      IIC = +1
      CALL RCHXTRK(I,IK)
      DO 11 I=1,IL
      J = ICHXTRK(S,V)
      IF (J,EG,I) GO TO 13
      IF (J,NE,I) GO TO 12
      11 CONTINUE
      I = IL
      GO TO 14
      12 I = I-1
      GO TO 14
      13 IIS = -1
      14 IK = IK+I
      IL = IL-I
      C CONVERT INTEGER PART, LESS SIGN
      15 LL = IL
      II = INT(S,IK,IL,LL,IV)
      IF (IIV,LT,C) GO TO 60
      VV = IIV*IIS
      IF (II,LE,C) GO TO 20
      ID = C
      GO TO 50
      C SEARCH FOR DECIMAL POINT
      20 IF (KOM(S,IK,1,IP,1),NE,C) GO TO 38
      25 IK = IK+1
      IL = IL-1
      IF (IL,LE,C) GO TO 50
      C CONVERT FRACTION (IF PRESENT)
      IF = 0

```

```

IREI0410
IREI0420
IREI0430
IREI0440
IREI0450
IREI0460
IREI0470
IREI0480
IREI0490
IREI0500
IREI0510
IREI0520
IREI0530
IREI0540
IREI0550
IREI0560
IREI0570
IREI0580
IREI0590
IREI0600
IREI0610
IREI0620
IREI0630
IREI0640
IREI0650
IREI0660
IREI0670
IREI0680
IREI0690
IREI0700
IREI0710
IREI0720
IREI0730
IREI0740
IREI0750
IREI0760
IREI0770
IREI0780
IREI0790
IREI0800

```

```

IRE10810
IRE10820
IRE10830
IRE10840
IRE10850
IRE10860
IRE10870
IRE10880
IRE10890
IRE10900
IRE10910
IRE10920
IRE10930
IRE10940
IRE10950
IRE10960
IRE10970
IRE10980
IRE10990
IRE11000
IRE11010
IRE11020
IRE11030
IRE11040
IRE11050
IRE11060
IRE11070
IRE11080
IRE11090
IRE11100
IRE11110
IRE11120
IRE11130
IRE11140
IRE11150
IRE11160
IRE11170
IRE11180
IRE11190
IRE11200

```

Reproduced from
best available copy.

```

IFV = 0
LL = 0
CALL RCHVTX(I1,IK)
DO 31 I=1,IL
  J = ICHXTK(S,U)
  IF(J,EC,IB) GO TO 31
  IF(J,LT,OC,OR,J,GT,IC) GO TO 35
  IF(LL,GF,IXI) GO TO 34
  IFV = IFV+J-U-IFC
  LL = LL+1
31 CONTINUE
  IF = 1
  I = IL+1
  IK = IK+1
  IL = IL-1
  IFV = IFV
  L1 = MOD(LL+50,1)+1
  L2 = (LL+50)/1
  FV = FV/(L1)/TT(L2)
  IF(IIS,LT,OC) FV=FV
34 IF(IF,GT,OC) GO TO 50
  GO TO 40
38 IF(II,LT,OC) GO TO 60
  ID = 0
  C SEARCH FOR 'E'
  40 IF(KOK(S,IK,1,IE,1),NE,OC) GO TO 45
  RX = 'TRUE'
  41 IK = IK+1
  IL = IL-1
  IF(IL,LE,OC) GO TO 40
  C CONVERT EXPONENT
  45 LL = IL
  IX = INT(S,IK,IL,LL,IXV)
  IF (IX,LT,OC,AND. (GX,OR. .NOT,OS)) GO TO 60
  C GENERATE FINAL VALUE AND CHECK SIZE
  50 VV = VV+FV
  IF(DABS(VV),LE,G,OC) GO TO 55
  IXV = IXV-ID

```

```

IREI1210
IREI1220
IREI1230
IREI1240
IREI1250
IREI1260
IREI1270
IREI1280
IREI1290
IREI1300
IREI1310
IREI1320
IREI1330
IREI1340
IREI1350
IREI1360

```

```

IF(IXV,LT,-60,00, IXV,GE,+40) GO TO 60
L1 = 500(IXV+50,0)+1
L2 = (IXV+50)/10
VV = V*IT(L1)*IT(L2)
IF(DABS(VV),LT,SPALL,OR,CA'S(VV),GT,P13) GO TO 60
55 V = VV
IF(IX,LE,0) IREI=0
GO TO 70
C ERROR DETECTED
60 IREI = -1
61 V = 0.
70 IF(.NOT,OS) RETURN
C = IK
L = IL
RETURN

```

```

C *****
C * ISCAN *
C *****
C
C      JUNK HOUSE      JITRI      SEPT 1970
C
C      FUNCTION ISCAN(S1,C1,L1),TRT,C2,C3)
C
C      SCANS UP TO L1 CH., STARTING WITH C1 OF STRING S1. PICKS
C      UP TOKENS, DEFINED BY USER, VIA TRT TABLE.
C      A TOKEN IS 1) 1 OR MORE CH. VIA ZERO ENTRY IN TRT TABLE, OR
C      2) 1 CH. VIA NONZERO TRT ENTRY
C      TOKEN TYPE IS INDICATED BY RETURNED VALUE OF ISCAN
C
C      ISCAN = 0 TEXT (1 OF MORE ZERO-TRT-ENTRY CH.)
C      ISCAN = 1 END OF STRING (L1=0 ON INPUT, C1 OUT=C1 IN)
C      ISCAN = GT.1 DELIMITER (NON-ZERO-TRT-ENTRY CH.)
C      C1 1ST CH. SCANNED, ON OUTPUT, IS NEXT CH. TO BE SCANNED.
C      L1 NO. CH. TO BE SCANNED, ON OUTPUT, IS NO. CH. LEFT,
C      TRT 64 WORD INTEGER TABLE, 1 ENTRY FOR EACH POSSIBLE CH.
C      C2 = INPUT VALUE OF C1
C      L2 = LEFT OF TOKEN. (SET IF END OF STRING)
C      STRING S IS PACKED (6 CH/WORD)
C
C      INTEGER S1(1),TRT(64)
C      INTEGER C1,C2,CN
C
C      L=0 SIGNALS END-OF-STRING
C      IF(L1,GT,0) GO TO 1
C      L2 = C
C      CN = C1
C      ISCAN = 1
C      GO TO 6
C
C      TRANSLATE AND TEST S1, USING PROVIDED TRT TABLE
C      1 I = ITRT(S1,C1,L1,TRT,C1)
C      IF(I) 4,5,6
C      4 IF(CN,GT,C1+1) GO TO 7
C      DELIMITER FOUND
C      L2 = 1
C      ISCAN = IARS(I)
C      GO TO 6
C      TEXT FOUND

```

Reproduced from
best available copy.

ISCAN410
 ISCAN420
 ISCAN430
 ISCAN440
 ISCAN450
 ISCAN460
 ISCAN470
 ISCAN480
 ISCAN490

7 CV = CV-1
 5 L2 = CV-C1
 ISCAN = 0
 C SET UP FOR NEXT SCAN, AND SET TCKEN LOCATION
 6 L1 = L1-L2
 C2 = C1
 C1 = CV
 RETURN
 END

```

ITRT0010
ITRT0020
ITRT0030
ITRT0040
ITRT0050
ITRT0060
ITRT0070
ITRT0080
ITRT0090
ITRT0100
ITRT0110
ITRT0120
ITRT0130
ITRT0140
ITRT0150
ITRT0160
ITRT0170
ITRT0180
ITRT0190
ITRT0200
ITRT0210
ITRT0220
ITRT0230
ITRT0240
ITRT0250
ITRT0260
ITRT0270
ITRT0280
ITRT0290
ITRT0300
ITRT0310

```

```

*****
* IRT *
*****
      FUNCTION IRT(C1, C2, IRT)
      EXAMINE TO 11 CHAR. STARTING W/CH. C1 OF STRING S1.
      STOP AT 1ST CH. WHERE THE NEXT CH. IS NOT '0'.
      RETURN WITH C1 = LEFT CH. (C1), AND
      IRT = 1. IF NO '0' FOUND AT END OF STRING
      = 0. IF NO '0' FOUND. IF COUNTERED
      = N. INDEX OF TYPE '0' FOUND. MORE WILL FOLLOW
      STRING S1 IS PACKET (6 CHAR/WORD)
      INTEGER S1(1), IRT(44),
      INTEGER C1, C2
      CALL RCHXTR(X, C1)
      IC = C1-1+11
      DO 1 I=C1, IC
      J = ICHXTR(S1, J)+1
      IRT = IRT(J)
      IF(IRT.NE.0) GO TO 2
1 CONTINUE
      CN = IC+1
      IRT = 0
      RETURN
2 CN = I+1
      IF(I.EQ.IC) IRT=IRT
      RETURN
      END

```

Reproduced from
best available copy.

KOML0010
KOML0020
KOML0030
KOML0040
KOML0050
KOML0060
KOML0070
KOML0080
KOML0090
KOML0100
KOML0110
KOML0120
KOML0130
KOML0140
KOML0150
KOML0160
KOML0170
KOML0180
KOML0190
KOML0200
KOML0210
KOML0220
KOML0230
KOML0240
KOML0250
KOML0260
KOML0270
KOML0280
KOML0290
KOML0300
KOML0310
KOML0320
KOML0330
KOML0340
KOML0350
KOML0360
KOML0370

```

*****
* KOML *
*****
FUNCTION KOML(S1,C1,L1,S2,C2,L2)
INTEGER S1(1),C1(1),S2(1),C2(1)
COMPARE TWO STRINGS OVER THE LENGTH OF THE SHORTER STRING,
THE TWO STRINGS ARE..
IF L1 CHAR. OF STRING S1, STARTING WITH CHAR. C1, AND
THE L2 CHAR. OF STRING S2, STARTING WITH CHAR. C2,
UPON RETURN, KOML IS SET TO..
  =2 IF S1 L1 S2
  =1 IF S1 L1 S2
  =0 IF S1 L1 S2
  =1 IF S1 L1 S2
  =2 IF S1 L1 S2
L = L1
KOML = 0
IF (L1-L2) 4,5,3
3 L = L2
KOML = +1
GO TO 5
4 KOML = -1
5 IF (L1-L2) RETURN
CALL RCHXTR(K1,C1)
CALL RCHXTR(K2,C2)
DO 6 I=1,L
IF (ICHXTR(S1,K1)-ICHXTR(S2,K2)) 2,6,1
6 CONTINUE
RETURN
2 KOML = -2
RETURN
1 KOML = +2
RETURN
END

```

Reproduced from
best available copy.


```

      IF(SC,GT,0.) SC=ALOG10(SC)
      IF(SC,LT,0.) SC=EC-3.
      SC = AINT(SC/3.)*3.
      SC = 10.**(-AINT(SC))
      WRITE(6,202) YS,SC,YE
202  FORMAT(11X,6HYM1 =,E12.4,31X,10HX MULT. BY,E12.4,32X,6HYMAX =,
      * E12.4/)
      DSC = DX*SC
      XSC = XS*SC
      X=XS
      DO 102 J=1,N
      OGRID=.FALSE.
      IF(KGI,GT,NGI) GO TO 110
      IF(1,LT,IG(KGI)) GO TO 110
      OGRID = ,TRUE.
      KGI = KGI+1
110  KGJ = 1
      DO 101 J=1,MY
      IF(OGRID) GO TO 111
      IF(KGJ,GT,NGJ) GO TO 112
      IF(1,LT,IG(KGJ)) GO TO 112
      KGJ = KGJ+1
111  POT(J) = DOT
112  GO TO 101
113  POT(J) = BL
101  CONTINUE
      CALL FCN(X,N,VAL)
      DO 103 J=1,N
      IV = (VAL(J)-YS)/DY+1,
      IF(IV,GE,1 .AND. IV,LE,MY) POT(IV)=CHAR(J)
103  CONTINUE
      WRITE(6,151) XSC,(POT(J),J=1,MY)
151  FORMAT(1X,F8,2,2X,121A1)
      XSC = XSC+DSC
102  X=X+DX
      RETURN
      END
NOTA041D
NOTA0420
NOTA0430
NOTA0440
NOTA0450
NOTA0460
NOTA0470
NOTA0480
NOTA0490
NOTA0500
NOTA0510
NOTA0520
NOTA0530
NOTA0540
NOTA0550
NOTA0560
NOTA0570
NOTA0580
NOTA0590
NOTA0600
NOTA0610
NOTA0620
NOTA0630
NOTA0640
NOTA0650
NOTA0660
NOTA0670
NOTA0680
NOTA0690
NOTA0700
NOTA0710
NOTA0720
NOTA0730
NOTA0740
NOTA0750
NOTA0760
NOTA0770

```



```

IF (U, I, LT, 2, ) GO TO 52
IF (U, I, LT, 2, ) GO TO 52
51 J = (U, I, - 0, I, ) / 2, + 1,
IF (U, I, J, 0, ) GO TO 52
IF (U, I, J, 0, ) GO TO 52
J = J + 1
IF (U, I, J, 0, ) GO TO 52
IF (U, I, J, 0, ) GO TO 52
52 J = J + 1
GO TO 51
53 NG = J + 1
54 IF (U, I, J, 0, ) GO TO 52
55 RETURN
END

```

Reproduced from
best available copy.

PPGR0410
PPGR0420
PPGR0430
PPGR0440
PPGR0450
PPGR0460
PPGR0470
PPGR0480
PPGR0490
PPGR0500
PPGR0510
PPGR0520
PPGR0530
PPGR0540

CHARACTER STORE AND EXTRACT ROUTINES

RETURN

*(P),

SET POINTER 'A' TO CHAR. NO. OF STRING.

FORTRAN CALL IS..
CALL SCHXTR(S,P)

SCHXTR*

LA,C16 AC,0 , CLEAR TOP OF DIVIDEND
LA A1,P1,R11 , LOAD BOTTOM 1ST. CHAR. COUNT)
AA,C16 A1,P , PLUS 5
CI,C16 AC,0 , DIVIDE BY 6 TO GET..
SA,C1 AC,C,R11 , ST. WORD 'C.' (MOVEMENT), AND
SA,C2 A1,P,R11 , ST. CHAR. NO. WITHIN WORD (REM.)
J 3,P11

SET NEXT CHAR. IN STRING 'A' TO BIT PATTERN IN THE LOWER 6
BITS OF 'C'. 'A' IS STRING PTR.

FORTRAN CALL IS..
CALL SCHXTR(S,P,P)

SCHXTR*

LA,C1 A3,P1,P11 , GET CURR. WORD NO.
LA,C2 A2,P1,P11 , GET CURR. CHAR. NO.
LA A1,C,B11 , GET ADDR. OF START OF STRING
AA A1,A3 , COMPUTE ADDR. OF CURRENT WORD..
AA,A,C14 A1,1 , IS START-1+WORD NO.
LA AC,P2,P11 , GET CHARACTER TO BE STORED.
EX STORCH,A2 , EXECUTE PROPER CHAR. STORE INSTR.
LNU A1,NEXTC , JUMP TO PTR UPDATE ROUTINE
J 4,P11 , AND EXIT

RETRIEVE NEXT CHAR. FROM STRING 'SI'. 'SI' IS STRING PTR.

FORTRAN CALL IS..
* = ICHXTR(S,P)

RCHX0010
RCHX0020
RCHX0030
RCHX0040
RCHX0050
RCHX0060
RCHX0070
RCHX0080
RCHX0090
RCHX0100
RCHX0110
RCHX0120
RCHX0130
RCHX0140
RCHX0150
RCHX0160
RCHX0170
RCHX0180
RCHX0190
RCHX0200
RCHX0210
RCHX0220
RCHX0230
RCHX0240
RCHX0250
RCHX0260
RCHX0270
RCHX0280
RCHX0290
RCHX0300
RCHX0310
RCHX0320
RCHX0330
RCHX0340
RCHX0350
RCHX0360
RCHX0370
RCHX0380
RCHX0390
RCHX0400

ICHXTK*	LA,01	A3,*1,R11	. GET CURR. WORD NO.	RCHX0410
	LA,02	A2,*1,R11	. AND CURR. CHAR. NO.	RCHX0420
	LA	A1,0,R11	. GET ADDR OF START OF STRING	RCHX0430
	AA	A1,A3	. COMPUTE ADDR OF CURRENT WORD..	RCHX0440
	ANA,014	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,F11	. 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, AND	RCHX0530
	SA	A3,*1,R11	. UP WORD NO BY 1	RCHX0540
	J	0,11		RCHX0550
	SA,02	A2,*1,R11	. SAME WORD. STORE NEW CHAR NO	RCHX0560
	J	,A1		RCHX0570
				RCHX0580
				RCHX0590
				RCHX0600
				RCHX0610
				RCHX0620
				RCHX0630
				RCHX0640
				RCHX0650
				RCHX0660
				RCHX0670
				RCHX0680
				RCHX0690
				RCHX0700
				RCHX0710
				RCHX0720
				RCHX0730
				RCHX0740
				RCHX0750
				RCHX0760
				RCHX0770
				RCHX0780
				RCHX0790
				RCHX0800

SIMILAR TO SCHXTK, BUT CHARACTERS ARE STORED FROM RIGHT TO LEFT.

FORTRAN CALL IS..
CALL PCHXTK(S,V,K)

Reproduced from
best available copy.

PCHXTK*	LA,01	A3,*1,R11	. GET CURR. WORD NO.
	LA,02	A2,*1,R11	. GET CURR. CHAR. NO.
	LA	A1,0,R11	. GET ADDR OF START OF STRING
	AA	A1,A3	. COMPUTE ADDR OF CURRENT WORD..
	ANA,014	A1,1	. IS START-1+WORD NO.
	LA	A0,*2,R11	. GET CHARACTER TO BE STORED.
	EX	STORECH,A2	. EXECUTE PROPER CHAR. STORE INSTR.
	LMJ	A1,PREVCH	. JUMP TO PTR UPDATE ROUTINE
	J	4,R11	. AND EXIT

SIMILAR TO ICXHTK, BUT CHARACTERS ARE RETRIEVED FROM RIGHT TO LEFT.

FORTRAN CALL IS..
K = JCHXTK(S,V)

SUBROUTINE SOJAC(FCN,X1,X2,YMAX,XMIN,YMAX,FINT,IMAX,IYR,INT,NE,L)

 * SOJAC *

 *** X-VALUES ARE SCALED BY APPROPRIATE POWER OF TEN.
 PLOTS CONTROLLED BY FCN(X,Y) ON PRINTER.
 NOTE: IN CALLING ROUTINE FCN, MUST APPEAR IN AN 'EXTERNAL'
 STATEMENT.
 X RANGES BETWEEN XMIN AND YMAX. X INCREASES DOWN THE PAGE.
 Y RANGES BETWEEN YMIN AND YMAX. Y INCREASES ACROSS THE PAGE.
 FCN(X,Y) IS EXPECTED TO RANGE BETWEEN FMIN AND FMAX.
 L(3) IS PLOTTED IF A FUNCTION VALUE IS .GE. 1.E+14 (I.E. UNRE-
 FINED.) L(4) IS PLOTTED FOR VALUES BELOW FMIN. L(NF+2) FOR ABOVE
 FMAX. THE REGION BETWEEN FMIN AND FMAX IS DIVIDED INTO NE EQUAL
 INTERVALS. INTERVAL K IS PLOTTED WITH SYMBOL L(K+1).
 TO HELP SMOOTH THE CONTOURS, THIS PROCEDURE IS FOLLOWED. IF THE
 FUNCTION VALUE AT A POINT BE WITHIN 0.05 INTERVAL WIDTH FROM THE
 EDGE OF AN INTERVAL, IT WILL BE PLOTTED WITH A BORDER CHARACTER.
 IF THE PLOT POSITION ABOVE THE CURRENT POSITION (I.E., SAME Y,
 LOWER X) BE BLANK, THE BORDER CHARACTER IS L(2). OTHERWISE IT IS
 L(1). THIS PROCEDURE IS NOT USED IF L(1) AND L(2) ARE BLANK.
 THE PLOT IS IYR PRINT POSITIONS WIDE. THE LENGTH OF THE PLOT IS
 WHATEVER IT HAS TO BE TO KEEP THE UNIT OF LENGTH THE SAME IN THE
 X AND Y DIRECTIONS.
 IF INT = 0, SOJAC RETURNS AFTER COMPUTING THESE QUANTITIES, FOUND
 IN COMMON /CSOJAC/. X, Y, THE AMOUNT X AND Y ARE STEPPED
 BETWEEN EVALUATION POINTS, AND IYR, THE NUMBER OF POINTS IN THE
 X DIRECTION. THERE ARE IYR*IXR POINTS PLOTTED IN ALL.
 IF INT.GT.0, A PLOT IS GENERATED.
 THE Y RANGE AND THE WIDTH OF AN INTERVAL
 OF X IS PRINTED ALONG THE LEFT MARGIN,
 ARE PLOTTED, THE VALUE OF X IS PRINTED ALONG THE LEFT MARGIN,
 SOJAC0010
 SOJAC0020
 SOJAC0030
 SOJAC0040
 SOJAC0050
 SOJAC0060
 SOJAC0070
 SOJAC0080
 SOJAC0090
 SOJAC0100
 SOJAC0110
 SOJAC0120
 SOJAC0130
 SOJAC0140
 SOJAC0150
 SOJAC0160
 SOJAC0170
 SOJAC0180
 SOJAC0190
 SOJAC0200
 SOJAC0210
 SOJAC0220
 SOJAC0230
 SOJAC0240
 SOJAC0250
 SOJAC0260
 SOJAC0270
 SOJAC0280
 SOJAC0290
 SOJAC0300
 SOJAC0310
 SOJAC0320
 SOJAC0330
 SOJAC0340
 SOJAC0350
 SOJAC0360
 SOJAC0370
 SOJAC0380
 SOJAC0390
 SOJAC0400

```

SOJA0410
SOJA0420
SOJA0430
SOJA0440
SOJA0450
SOJA0460
SOJA0470
SOJA0480
SOJA0490
SOJA0500
SOJA0510
SOJA0520
SOJA0530
SOJA0540
SOJA0550
SOJA0560
SOJA0570
SOJA0580
SOJA0590
SOJA0600
SOJA0610
SOJA0620
SOJA0630
SOJA0640
SOJA0650
SOJA0660
SOJA0670
SOJA0680
SOJA0690
SOJA0700
SOJA0710
SOJA0720
SOJA0730
SOJA0740
SOJA0750
SOJA0760
SOJA0770
SOJA0780
SOJA0790
SOJA0800

COMMON /CSQAC/D,IP,IX,IV,1,0

INTEGER POT,POT,PL,IG(30)
LOGICAL CFILL,CFILL,CFILL
DIMENSION POT(12,2),L(1)
DATA POT,PL,MG/1,1,14,13/

IVV = IVR
IF(IV,LE,0,CF,IV,GT,121) IVV = 121
CFILL = L(1),NE,PL,CF, L(2),NE,PL
NGJ = MAX(2,MING(IVV/20+1,MG-2))
MG1 = MG-NGJ
CALL PGRIG(VMIN,VMAX,VV,IG,NGJ,IG,20)
SET IF X AND Y SCALERS, GRID LINE INTERVALS
YR = YMAX-VMIN
XR = XMAX-VMIN
PY = YR/FLOAT(IVV-1)
PX=DY*1.666667
PX2 = PX/2.
IXR = XR/DX+1.5
CALL PGRIG(VMIN,XVY,VMIN,IG,IG(NGJ+1),12)
NG1 = NGI+NGJ
MG1 = NGJ+1

IF (INT,LE,0) RETURN
IF IY,LE,0, ASSUME THAT ONLY SCALE DATA WAS WANTED
SET UP FUNCTION RANGE, AND CONTOUR INTERVAL
PR = FMAX-EMIN
ENF = IF
CF = PR/ENF
XSC = MAX1(ABS(VMIN),ABS(VMAX))
IF(PR,GT,0,AND, XSC,GT,0) GO TO 130
WRITE(6,231) ENF,FMAX
FORMAT(19HENDLL PLOT, ENF =,F15.7,9H, FMAX =,E15.7)
131 RETURN
130 XSC = ALOG10(XSC)
IF(XSC,LT,0.) XSC=XSC-3.
XSC = AINT(XSC/3.)*3.
XSC = -AINT(XSC)
XSC = 10.**XSC

```

```

C
DXS = DX*XS
      PRINT PLT, I, J, K, L
      WRITE(6,152) X, Y, Z, XSC, YSC, ZSC
152  FORMAT(11X,6HYPER) =E12.4,12X,10HOUTOUR INTERVAL =,E12.4,12X,
      * 10HX MULT. BY,EC,2,12X,4HMAX =,E12.4/)
      X = X*IN
      XS = X*XS
      II = 2
      DO 102 I=1,IV
      KK = II
      II = 3-II
      GGRID = .FALSE.
      IF(KG1.GT.NG1) GO TO 103
      IF(J.LT.IG(KG1)) GO TO 104
      GGRID = .TRUE.
      KG1 = KG1+1
      FOR KGJ = 1
      Y = Y*IN
      DO 103 J=1,IVY
      POT(J,II) = POT
      IF (GGRID) GO TO 103
      IF(KGJ.GT.NGJ) GO TO 104
      IF(J.LT.IG(KGJ)) GO TO 104
      KGJ = KGJ+1
      GO TO 103
101 VAL = FCN(X,Y)
      CONVERT FUNCTION VALUE INTO PLOT SYMBOL
      IF(VAL.GE.1.E+38) GO TO 114
      TV = (VAL-FMIN)/DF+5.
      ATV = ATVT(TV+.05)
      IX = TV
      IF(CFILL .AND. ID.GE.5 .AND. IV.(E.NF+4 .AND. ABS(TV-ATV).(E.0.05)
      * GO TO 110
      IX = MIND(NF+5,MAX(4,IFIX(TV)))
      POT(J,II) = L(IX)
      GO TO 111
110 POT(J,II) = L(IX)
      IF(1.EC.1) GO TO 111
      IF(POT(J,KK).NE.PL) POT(J,II)=I(2)
      GO TO 111

```

Reproduced from
best available copy.

```

114 POT(J,II) = L(3)
111 CONTINUE
103 V=V+DY
C
161 WRITE(6,151) X5,(POT(J,II),J=1,IVY)
151 FORMAT(1X,F8.2,2X,121A1)
      GO TO 112
112 X5 = X5+DX
102 X=X+DX
      RETURN
      END
SOJA1210
SOJA1220
SOJA1230
SOJA1240
SOJA1250
SOJA1260
SOJA1270
SOJA1280
SOJA1290
SOJA1300
SOJA1310

```

```

LOGICAL U
LOGICAL OPF5(3,6)
EQUIVALENCE (Y(1),C),(Y(2),00),(Y(3),A1),(Y(4),A2),(Y(5),A),
* (Y(6),K),(Y(7),T),(Y(8),F)
DATA OPF5/.TRUE.,.3*,.FALSE.,.TRUE.,.TRUE.,.FALSE.,.2*,.TRUE.,.3*,.FALSE.,
* 2*,.TRUE.,.FALSE.,.TRUE.,.TRUE.,.FALSE.,.2*,.TRUE./
C** NWDR IS NO. 'WORDS' PER SYNTAX RULE
DATA NWDR/2/

C
C EMPTY STACK, SET UP SCAN, POINT TO INITIAL RULE
C
S = 0
LL = L1+i
R = (IX=1)*NWDR+i
I = -1

C
C SCAN -- PICK UP ONE TOKEN (GROUP) FROM STRING. EITHER TEXT OR DELIM.
C
I ISAVE = I
I = ISCAN(S1,C1,L1,TRY,C2,I2)

C
C APPLY A RULE TO LATEST TOKEN
C
2 CALL SYNTAXF(9,Y,X(P))
GO TO (100,200,300,400,500,505,600,601,700,800,810,850,860),0
9 SYNTAX = .FALSE.
8 IF(O.EQ.2) RETURN
C1 = C1+L2
L1 = L1+L2
RETURN

C
C 0=1 -- PROCEDURE CALL
C
100 S = S+2
IF(S.GT.NB) GO TO 9
C STACK STRING INFO AND RETURN ADDR.
SYK(S+1) = C1-L2
SYK(S) = B
C RULE=PROCEDURE ADDRESS IS KEPT AS ARGUMENT OF CALL
B = A

```

```

SYNT0410
SYNT0420
SYNT0430
SYNT0440
SYNT0450
SYNT0460
SYNT0470
SYNT0480
SYNT0490
SYNT0500
SYNT0510
SYNT0520
SYNT0530
SYNT0540
SYNT0550
SYNT0560
SYNT0570
SYNT0580
SYNT0590
SYNT0600
SYNT0610
SYNT0620
SYNT0630
SYNT0640
SYNT0650
SYNT0660
SYNT0670
SYNT0680
SYNT0690
SYNT0700
SYNT0710
SYNT0720
SYNT0730
SYNT0740
SYNT0750
SYNT0760
SYNT0770
SYNT0780
SYNT0790
SYNT0800

```

```

      GO TO 2
C
C 0=2 == EXAMINE A TOKEN FOR (NO) MEMBERSHIP IN A SET OF TOKEN TYPES
C
200 GO TO (300,210,220,230,240),00
210 SYNTAX = I.FQ, A1
    GO TO 209
220 SYNTAX = I.NE, A1
    GO TO 209
230 SYNTAX = A1.LE, I, AND, I.LE, A2
    GO TO 209
240 SYNTAX = I.LT, A1, OR, I.GT, A2
C
C CHECK VALUE OF 'SYNTAX' -- DETERMINES SUCCESS/FAILURE
C
209 IF(SYNTAX) GO TO 290
C
C SYNTAX EXAMINATION (OR OTHER TEST) FAILS
C
280 CALL SYNTAXF(R,Y,X(R))
    B = F
    IF (B.GT,0) GO TO 2
    IF NO F-ADDR., POP STACK (RETURN FROM RULE-PROC.) IF STACK NON-EMPTY
282 IF(S.LE,0) GO TO 9
    B = STK(S)
    C1 = STK(S=1)
    L1 = LL=C1
283 S = S+2
    CALL SYNTAXF(R,Y,X(R))
    B = F
    IF(B.LE,0) GO TO 282
    GO TO 1
C
C SYNTAX EXAMINATION (OR OTHER TEST) SUCCEEDS
C
290 CALL SYNTAXF(I0,Y,X(B1))
    IF (K.LE,0) GO TO 291
    IF (,NOT, U(S1,C2,L2,K,W)) GO TO 280
291 B = 1
C
C B IS A SUCCESSFUL BRANCH ADDRESS

```

SYNT0810
 SYNT0820
 SYNT0830
 SYNT0840
 SYNT0850
 SYNT0860
 SYNT0870
 SYNT0880
 SYNT0890
 SYNT0900
 SYNT0910
 SYNT0920
 SYNT0930
 SYNT0940
 SYNT0950
 SYNT0960
 SYNT0970
 SYNT0980
 SYNT0990
 SYNT1000
 SYNT1010
 SYNT1020
 SYNT1030
 SYNT1040
 SYNT1050
 SYNT1060
 SYNT1070
 SYNT1080
 SYNT1090
 SYNT1100
 SYNT1110
 SYNT1120
 SYNT1130
 SYNT1140
 SYNT1150
 SYNT1160
 SYNT1170
 SYNT1180
 SYNT1190
 SYNT1200

```

295 IF(B.LE.C) GO TO 292
    IF(C.LE.2) GO TO 1
    GO TO 2
C IF NO T-ADDR., POP STACK IF NON-EMPTY (I.E. RETURN FROM RULE-PROC.)
292 IF(S.LE.0) GO TO 4
    R = STK(S)
    S = S-2
    GO TO 290
C
C O=3 -- FORCE GOOD-SYNTAX RESPONSE (UNLESS USER ROUTINE RETURNS FALSE)
300 SYNTAX = .TRUE.
    GO TO 290
C
C O=4 -- FORCE BAD-SYNTAX RESPONSE
400 SYNTAX = .FALSE.
    GO TO 280
C
C O=5 -- COMPARE W(A1) WITH A2
505 A2 = W(A2)
500 IF(W(A1)=A2) 501,502,503
501 J = 1
    GO TO 510
502 J = 2
    GO TO 510
503 J = 3
510 IF (OPF5(J,00)) GO TO 300
    GO TO 400
C
C O=7 -- SET W(A1) TO SOME FCN OF W(A1) AND A2
C O=8 -- SET W(A1) TO SOME FCN OF W(A1) AND W(A2)
601 A2 = W(A2)
600 GO TO (611,613,615,617,619,621,623,625,627,629,631,633,635,637,
    * 639,641,643,645),00
611 W(A1) = A2
    GO TO 300

```

Reproduced from
best available copy.

SYNT1610
SYNT1620
SYNT1630
SYNT1640
SYNT1650
SYNT1660
SYNT1670
SYNT1680
SYNT1690
SYNT1700
SYNT1710
SYNT1720
SYNT1730
SYNT1740
SYNT1750
SYNT1760
SYNT1770
SYNT1780
SYNT1790
SYNT1800
SYNT1810
SYNT1820
SYNT1830
SYNT1840
SYNT1850
SYNT1860
SYNT1870
SYNT1880
SYNT1890
SYNT1900
SYNT1910
SYNT1920
SYNT1930
SYNT1940
SYNT1950
SYNT1960
SYNT1970
SYNT1980
SYNT1990
SYNT2000

Reproduced from
best available copy.

413 W(A1) = -A2
GO TO 300
415 W(A1) = 1/25(A2)
GO TO 300
417 W(A1) = -1/5(A2)
GO TO 300
419 W(A1) = W(A1)+A2
GO TO 300
421 W(A1) = W(A1)-A2
GO TO 300
423 W(A1) = W(A1)*A2
GO TO 300
425 IF(A2.EQ.0) GO TO 400
W(A1) = W(A1)/A2
GO TO 300
427 IF (A2.EQ.0) GO TO 400
W(A1) = (W(A1)-1-A2)/A2
GO TO 300
429 IF (A2.LE.0) GO TO 400
W(A1) = MOD(W(A1),A2)
GO TO 300
431 IF (A2.LT.0) GO TO 400
W(A1) = W(A1)**A2
GO TO 300
433 W(A1) = MIN0(W(A1),A2)
GO TO 300
435 W(A1) = MAX0(W(A1),A2)
GO TO 300
437 W(A1) = ISIGN(W(1),A2)
GO TO 300
439 SYNTAX = A2.LE.0
GO TO 200
441 SYNTAX = A(A1).GT.0 .OR. A2.GT.0
GO TO 200
443 SYNTAX = W(A1).GT.0 .AND. A2.GT.0
GO TO 200
445 SYNTAX = W(A1).GT.0 .AND. A2.LE.0 .OR. W(A1).LF.0 .AND. A2.GT.0
GO TO 200
C 0=9 -- SET W(A1) TO SOME 'SPECIAL VALUE'
C

```

C
700 GO TO (705,710,720,730,740,750,760,770),C0
C SET W(A1) TO VALUE OF NEXT CHAR. (FALSE RETURN IF I1.LE.C)
705 IF(I1.LE.C) GO TO 706
CALL RCHXTR(J,C1)
W(A1) = ICHXTR(S1,J)
C1 = C1+1
L1 = L1+1
GO TO 300
706 W(A1) = C
GO TO 400
C SET W(A1) TO TYPE-CODE OF PRIOR TOKEN
C (TOKEN PRESUMABLY FROM 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) = R+NWOR
GO TO 300
C SET W(A1) TO F-ADDR OF CURRENT RULE
C** (ANY USER EXIT USED WITH THIS RULE MUST RETURN WITH I1.TRUE.)
770 CALL SYNTRF(R,Y,>(P))
W(A1) = F
GO TO 300
C
C 0=10 -- GO TO RULE AT (T)+W(A1)
C 0=11 -- GO TO RULE AT *+1+W(A1)
C 0=12 -- GO TO RULE WHOSE NUMBER IS AT (T)+W(A1)
C 0=13 -- GO TO RULE WHOSE NUMBER IS AT *+1+W(A1)

```

```

SYNT2010
SYNT2020
SYNT2030
SYNT2040
SYNT2050
SYNT2060
SYNT2070
SYNT2080
SYNT2090
SYNT2100
SYNT2110
SYNT2120
SYNT2130
SYNT2140
SYNT2150
SYNT2160
SYNT2170
SYNT2180
SYNT2190
SYNT2200
SYNT2210
SYNT2220
SYNT2230
SYNT2240
SYNT2250
SYNT2260
SYNT2270
SYNT2280
SYNT2290
SYNT2300
SYNT2310
SYNT2320
SYNT2330
SYNT2340
SYNT2350
SYNT2360
SYNT2370
SYNT2380
SYNT2390
SYNT2400

```

C*** NOTE.. FOR ALL OF THE ABOVE, W(A1) MUST RANGE BETWEEN 0 AND 42

```

C      800 CALL SYNTAXF(7,Y,X(4))
        R = T
        GO TO 840
      810 R = R+NWDR
      840 IF (W(A1),LT,0 .OR. W(1),GT,42) GO TO 400
        R = R+W(A1)*NWDR
        GO TO 295
      850 CALL SYNTAXF(7,Y,X(4))
        R = T
        GO TO 890
      860 R = R+NWDR
      890 IF (W(A1),LT,0 .OR. W(A1),GT,42) GO TO 400
        R = R+W(A1)
        R = (X(B)-1)*NWDR+1
        GO TO 295
      END
SYNT2410
SYNT2420
SYNT2430
SYNT2440
SYNT2450
SYNT2460
SYNT2470
SYNT2480
SYNT2490
SYNT2500
SYNT2510
SYNT2520
SYNT2530
SYNT2540
SYNT2550
SYNT2560
SYNT2570
SYNT2580

```

Reproduced from
best available copy.

REGNAM

SPECIAL FIELD UNPACKING ROUTINE DESIGNED SOLELY FOR USE BY
FUNCTION "SYNTAX" TO RECODE SYNTAX RULES.

THE FIELDS ARE REFINED AS FOLLOWS..

FLD	WORD	BIT	LEN	CODE	DESCRIPTION
1	1	04	04	C	OPERATION CODE
2	1	12	04	CO	OPERATION MODIFIER
3	1	16	04	A1	ARGUMENT OR REGISTER NO. 1
4	1	20	04	A2	ARGUMENT OR REGISTER NO. 2
5	1	24	12	A	PROCEDURE ADDRESS (0=1 ONLY) (NOTE THAT A OVERLAPS A1 AND A2)
6	2	00	12	K	ACTION CODE (USED IF TRUE TEST)
7	2	12	12	T	TRUE ADDRESS
8	2	24	12	F	FALSE ADDRESS

FIELDS OF RULE 'X' ARE STORED IN LIST 'T' AS DIRECTED BY
CONTROL CODE 'I'.

IF I=1-R,

SET T(1) TO CONTENTS OF FIELD 'I' IN 'X'.

IF I=9 SET T(1) THRU T(5)

IF I=10 SET T(6) AND T(7)

FORTHAN CALL IS..

CALL SYNTAX(1,T,X)

*(0)

SYNTAX* LA

LA A2,0,B11

LA A3,2,B11

LA A0,1,B11

ANA,14 A0,1

SA,01 A0,ST1

EX RL-1,A2

SA A0,0,A2

J 4,B11

LA,12 A0,0,A3

LA,11 A0,0,A3

STF

RL

*(A2) = 1

*(A3 POINTS TO X

*(A0 POINTS TO T-1

*(SAVE ADDR OF T-1

*(EXECUTE UNPACK INSTR (DEP ON I)

*(STORE 1 FIELD (FOR I=1-5 ONLY)

*(IF I=1-5, UNPACK JUST 1 FIELD

SYNT0010
SYNT0020
SYNT0030
SYNT0040
SYNT0050
SYNT0060
SYNT0070
SYNT0080
SYNT0090
SYNT0100
SYNT0110
SYNT0120
SYNT0130
SYNT0140
SYNT0150
SYNT0160
SYNT0170
SYNT0180
SYNT0190
SYNT0200
SYNT0210
SYNT0220
SYNT0230
SYNT0240
SYNT0250
SYNT0260
SYNT0270
SYNT0280
SYNT0290
SYNT0300
SYNT0310
SYNT0320
SYNT0330
SYNT0340
SYNT0350
SYNT0360
SYNT0370
SYNT0380
SYNT0390
SYNT0400

T6	LA,9	A0,0,A3	SYNT0410
	LA,6	A0,0,A3	SYNT0420
	LA,1	A0,0,A3	SYNT0430
	J	T6	SYNT0440
	J	T7	SYNT0450
	J	T8	SYNT0460
	J	T15	SYNT0470
	J	T67	SYNT0480
	LMJ	A1,A5	SYNT0490
	J	A,B,1	SYNT0500
T7	LMJ	A1,A7	SYNT0510
	J	A,B,1	SYNT0520
T8	LA	A5,1,A3	SYNT0530
	LSSL	A5,24	SYNT0540
	SZ	A4	SYNT0550
	LDL	A4,12	SYNT0560
	LSSL	A4,1	SYNT0570
	AMA,14	A4,1	SYNT0580
	SA	A4,R,A0	SYNT0590
	J	A,B,1	SYNT0600
F6	LA	A5,1,A3	SYNT0610
	SZ	A4	SYNT0620
	LDL	A4,12	SYNT0630
	SA	A4,A,A0	SYNT0640
	J	O,A1	SYNT0650
F7	LA	A5,1,A3	SYNT0660
	LSSL	A5,12	SYNT0670
	SZ	A4	SYNT0680
	LDL	A4,12	SYNT0690
	LSSL	A4,1	SYNT0700
	AMA,14	A4,1	SYNT0710
	SA	A4,7,A0	SYNT0720
	J	O,A1	SYNT0730
T15	LA,12	A4,7,A3	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

	GET FIELD 6 UNDESIGNED	SYNT0410
	GET FIELD 7 UNDESIGNED	SYNT0420
	GET FIELD 8 UNDESIGNED	SYNT0430
	GET FIELDS 1-5 AT OFF	SYNT0440
	GET FIELDS 6 AND 7 AT OFF	SYNT0450
		SYNT0460
		SYNT0470
		SYNT0480
		SYNT0490
		SYNT0500
		SYNT0510
		SYNT0520
		SYNT0530
		SYNT0540
		SYNT0550
		SYNT0560
		SYNT0570
		SYNT0580
		SYNT0590
		SYNT0600
		SYNT0610
		SYNT0620
		SYNT0630
		SYNT0640
		SYNT0650
		SYNT0660
		SYNT0670
		SYNT0680
		SYNT0690
		SYNT0700
		SYNT0710
		SYNT0720
		SYNT0730
		SYNT0740
		SYNT0750
		SYNT0760
		SYNT0770
		SYNT0780
		SYNT0790
		SYNT0800

SYN77810
 SYN77820
 SYN77830
 SYN77840
 SYN77850
 SYN77860
 SYN77870
 SYN77880

44,6,10
 44,6,13
 44,6,16
 44,6,19
 44,6,22
 44,6,25
 44,6,28
 44,6,31

SA
 LA
 SA
 LA
 LA
 LA
 LA
 LA

T67