NASA TECHNICAL MEMORANDUM

NASA TM X-2151

NASA TM X-2151

FORTRAN IV PROGRAM FOR STUDYING ION - POLAR-MOLECULE COLLISIONS

by R. Bruce Canright, Jr., and John V. Dugan, Jr.

Lewis Research Center

Cleveland, Ohio 44135

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • JANUARY 1971

FORTRAN IV PROGRAM FOR STUDYING ION - POLAR-MOLECULE COLLISIONS

by R. Bruce Canright, Jr., and John V. Dugan, Jr.

Lewis Research Center

SUMMARY

Ion - polar-molecule collisions have been studied by numerically integrating their approximate equations of motion on a digital computer. In this report we present a FORTRAN IV program to do this integration (using conservation of energy as a step-size control) and to give results in the form of collision statistics, microfilm plots, and motion pictures of trajectories. Included are the description of the ion-molecule system, the form of the input and output, the functions of the main program and subroutines, the COMMON structure, a sample case, and the complete FORTRAN IV listings.

INTRODUCTION

Problem Background

The ion-molecule system has been studied extensively (refs. 1 to 3). By the use of a simple model, the Lagrangian can be obtained and the equations of motion derived (see ref. 4). Numerical integration with the aid of a computer gives instantaneous coordinates and velocities. The polar molecule can "look like" either a rod or a symmetric top. When the molecule is represented as a symmetric top, the laboratory coordinate system is as shown in figure 1.

The model potential energy has two terms, a permanent dipole contribution and an induced charge contribution. The Lagrangian for a symmetric-top molecule two scaled units long is written as

$$L = T - V = \frac{m}{2}(\dot{X}^2 + \dot{Y}^2 + \dot{Z}^2) + \frac{I_1}{2}(\dot{\xi}^2 + \dot{\eta}^2 \sin^2 \xi) + \frac{I_2}{2}(\dot{\psi} + \dot{\eta} \cos \xi)^2$$

$$+ \frac{\mu e}{r^3}(-X \sin \xi \sin \eta + Y \sin \xi \cos \eta + Z \cos \xi) + \frac{\alpha e^2}{2r^4}$$

where

T	model kinetic energy
V	model potential energy
m	reduced mass, $M_1M_2/(M_1 + M_2)$
X, Y, Z	Cartesian coordinates of ion (see fig. 1)
$\dot{X}, \dot{Y}, \dot{Z}$	Cartesian velocity components of ion
ξ, η, ψ	angular coordinates of dipole (see fig. 1)
$\dot{\xi}, \dot{\eta}, \dot{\psi}$	angular velocity components of dipole
I_1, I_2	moments of inertia (principal and about symmetry axis, respectively)
μ	dipole moment of molecule
e	electronic charge
r	ion-molecule separation
α	electronic polarizability of molecule

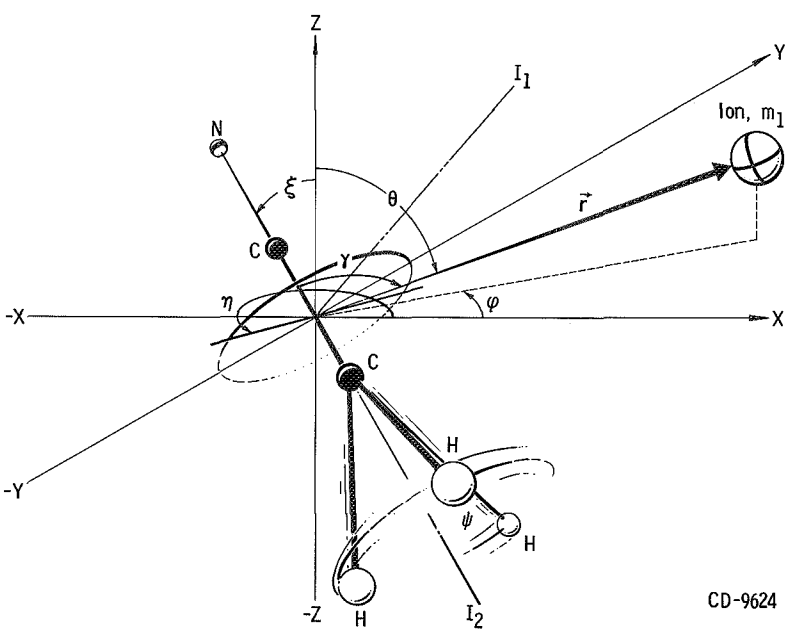


Figure 1. - Coordinate system for ion - polar-molecule pair.

Equations of Motion

From this Lagrangian are derived 12 first-order coupled differential equations which must be integrated. These are the equations of motion.

$$\ddot{X} \equiv \dot{V}_x = \frac{\mu e}{mr^5}\left(-r^2 \sin\xi \sin\eta - 3XF_v\right) - \frac{2\alpha e^2 X}{mr^6} \tag{1}$$

$$\ddot{Y} \equiv \dot{V}_y = \frac{\mu e}{mr^5}\left(r^2 \sin\xi \cos\eta - 3YF_v\right) - \frac{2\alpha e^2 Y}{mr^6} \tag{2}$$

$$\ddot{Z} \equiv \dot{V}_z = \frac{\mu e}{mr^5}\left(r^2 \cos\xi - 3ZF_v\right) - \frac{2\alpha e^2 Z}{mr^6} \tag{3}$$

with $F_v = (-X \sin\xi \sin\eta + Y \sin\xi \cos\eta + Z \cos\xi)$.

$$\dot{X} \equiv V_x \tag{4}$$

$$\dot{Y} \equiv V_y \tag{5}$$

$$\dot{Z} \equiv V_z \tag{6}$$

$$\ddot{\xi} \equiv \dot{\omega}_\xi = \dot{\eta}^2 \sin\xi \cos\xi - \frac{\mu e}{I_1 r^3}(Z \sin\xi - X \cos\xi \sin\eta + Y \cos\xi \cos\eta)$$

$$- \frac{I_2}{I_1}(\dot{\psi}\dot{\eta} \sin\xi + \dot{\eta}^2 \sin\xi \cos\xi) \tag{7}$$

$$\ddot{\eta} \equiv \dot{\omega}_\eta = \frac{\left[-\frac{\mu e}{r^2}(Y \sin\eta + X \cos\eta) - 2I_1\dot{\eta}\dot{\xi} \cos\xi + I_2(\dot{\xi}\dot{\psi} + \dot{\xi}\dot{\eta} \cos\xi)\right]}{I_1 \sin\xi} \tag{8}$$

$$\ddot{\psi} \equiv \dot{\omega}_\psi = \dot{\eta}\dot{\xi} \sin\xi - \ddot{\eta} \cos\xi \tag{9}$$

$$\dot{\xi} \equiv \omega_{\xi} \tag{10}$$

$$\dot{\eta} \equiv \omega_{\eta} \tag{11}$$

$$\dot{\psi} \equiv \omega_{\psi} \tag{12}$$

Numerical Approach

A theoretical check was made on the accuracy of integration: The instantaneous energy was compared at every time step to the initial energy, and the next time step was calculated based on this comparison. For this reason (a predictor-corrector check not required), a variable-order Runge-Kutta method was chosen (refs. 5 to 7), using fourth or fifth order for most problems. Details of choosing the (variable) time step for integration appear in appendix B of reference 8. Essentially, the step size is inversely proportional to the relative difference between current total energy and initial total energy. This has proved to be much more efficient than simple step modification, for example, halving or doubling.

Two important details of the numerical model are

(1) The term $\sin(\xi)$ in the denominator of equation (8) goes to zero as the dipole lines up with the Z-axis, introducing a singularity into the equations of motion. This difficulty is handled by using two spherical (Euler when the dipole is a top, that is, $\psi \neq 0$) coordinate systems for the dipole, an "unprimed" system and a "primed" system (ref. 1). The unprimed system (fig. 1) is used whenever the $\sin(\xi)$ term is greater than some chosen limit (the dipole is outside of some chosen polar cap). The primed system is used and the appropriate equations of motion are integrated whenever the dipole is within this cap.

(2) The program provides an option of two collision potential terms: a hard-sphere potential $V_{r=r_c} = \infty$, or a Lennard-Jones potential $V \sim r^{-12}$. Here r_c is a chosen collision distance (called CD in the program).

COMPUTER STUDIES

Two kinds of studies can be made with this program: collection of statistics and detailed study of individual trajectories. For a collection of statistics, a large number of initial conditions must be generated "randomly." Among other outputs, the program counts "captures;" by definition, the polar molecule captures the ion whenever the ion

comes within a certain prescribed distance from the molecule. An ion may be captured many times during the course of one trajectory; or it may be repulsed, never captured. The capture cross sections for various inputs can then be estimated (ref. 8). The procedure is to integrate many sets of initial conditions for each of a number of values for the impact parameter b. Some fraction $C_R(b)$ of the number of sets for each b will be capture collisions. Then the capture collision cross section σ_c is

$$\sigma_c = \pi \int_0^\infty C_R(b)\, d(b^2)$$

where this integral is approximated numerically by using the calculated (b, $C_R(b)$) points. Here the details of each trajectory are not important.

For a detailed study of individual trajectories, the entire history of variables in an individual case is stored and plotted (refs. 9 and 10). Plots of relative translational motion, rotational energy, and angles against the ion-dipole separation r, show the trajectory at a glance. Motion pictures of the system during its interaction provide a natural view of the collision (in the center-of-mass system). (Plots and motion pictures were made on a Control Data Corporation Model 280 microfilm recorder; the arithmetic was done in double precision on an IBM 7040/7094 DCS.) The program presented herein is applicable for statistics, plots, and motion pictures.

PROGRAM DETAILS

Input

The input to the main program is of two types. First, there is the general input for a group of initial conditions. For convenience, this input is read in by means of NAMELIST (ref. 1), which includes these quantities:

MASS	reduced mass of system, scaled units
MI	moment of inertia, I_1
MUE	μe, scaled units
ALFE2	$\alpha e^2/2$, scaled units
RI	initial separation, scaled to angstroms; placed in Q(1)
ORDER	order of Runge-Kutta integration, usually fourth or fifth
TOTERR	total relative error in energy to be tolerated

MAXHR maximum step-size increase factor

MAXER maximum local error ratio to be tolerated

HI initial step, scaled time

CHANGE $\sin(\xi)$, defines polar "cap," causes coordinate switch

CTDI LOGICAL switch, .TRUE. if collision-time distance is input

CTD collision-time distance - the separation from which measurement of collision lifetime is started. (If not input, the program calculates

$$CTD = \frac{\sigma_L}{\pi b} \sim \frac{1}{b} \sqrt{\frac{\alpha e^2}{\frac{1}{2} mV^2}}$$

where σ_L is the Langevin cross section.) Another criterion sometimes used (but not in the program as listed) is to start measuring collision time when interaction is "felt," that is, when the dipole rotational energy changes by 10 percent from its initial energy. The collision lifetime is the elapsed time the pair spends within this distance.

CD collision distance (Every time the pair gets within this separation they collide, and a capture is counted.)

MAXCAP maximum captures allowed for each trajectory

CD1 boundary for Lennard-Jones polarization potential

HTS minimum step to be tolerated, scaled time

RM I_2/I_1; RM is zero for rod cases

PLOT control array calling for still plots (If not all zero.)

IPLOT program stores and plots every IPLOTth point

CUTPLT LOGICAL switch: if .TRUE., truncates trajectories at 1000 points; if .FALSE., uses disk storage

Second, there is the set of initial coordinates Q and two labels, read in by (OP4F5.3, 1P6E10.3/OPF5.3, 4X, 2I3). Q(1) is not read in here, but is set to RI.

Q(1)	r	Q(5)	ξ	Q(9)	$\dot{\eta}$
Q(2)	θ	Q(6)	$\dot{R}$	Q(10)	$\dot{\xi}$
Q(3)	φ	Q(7)	$\dot{\theta}$	Q(11)	$\dot{\psi}$
Q(4)	η	Q(8)	$\dot{\varphi}$	Q(12)	ψ

Output

For each case, the main program prints out the initial conditions, the final conditions, initial and final energies, impact parameter b, the initial velocity collision time and distance, various computing times, and the statistics of the step-size history (see also the section Sample Case). Units and scaling of the variables are given in table I.

TABLE I. - UNITS AND SCALE FACTORS FOR VARIABLES

IN MAIN PROGRAM

Variable name	Input units	Output units	Units inside program
MASS	$g\times10^{20}$	$g\times10^{20}$	$g\times10^{20}$
MI	$(g\text{-}cm^2)\times10^{36}$	$(g\text{-}cm^2)\times10^{36}$	$(g\text{-}cm^2)\times10^{36}$
MUE	$(esu\text{-}cm)\times10^{24}$	$(esu\text{-}cm)\times10^{24}$	$(esu\text{-}cm)\times10^{24}$
ALFE2	$(esu^2\text{-}cm^3)\times10^{40}$	$(esu^2\text{-}cm^3)\times10^{40}$	$(esu^2\text{-}cm^3)\times10^{40}$
HI	$sec\times10^{14}$	$sec\times10^{14}$	$sec\times10^{14}$
CTD	Å	Å	Å
CD	Å	Å	Å
CD1	Å	Å	Å
HTS	$sec\times10^{14}$	$sec\times10^{14}$	$sec\times10^{14}$
Q(1) or RI	Å	cm	Å
Q(2)	rad	rad	rad
Q(3)	↓	↓	↓
Q(4)			
Q(5)			
Q(6)	$(cm/sec)\times10^{-6}$	cm/sec	$(cm/sec)\times10^{-6}$
Q(7)	$(rad/sec)\times10^{-14}$	rad/sec	$(rad/sec)\times10^{-14}$
Q(8)	↓	↓	↓
Q(9)			
Q(10)			
Q(11)			
Q(12)	rad	rad	rad
V	----------	cm/sec	cm/sec
B	----------	cm	cm
E or IE	----------	$erg\times10^{8}$	$erg\times10^{8}$
RE or IRE	----------	$erg\times10^{8}$	$erg\times10^{8}$
T	----------	sec	$sec\times10^{14}$
REALT	----------	sec	sec
CT	----------	sec	sec
CTDS	----------	cm	cm
TRUN	----------	sec	sec
TPLOT	----------	sec	sec
CAPT	----------	rad	rad

Main Program

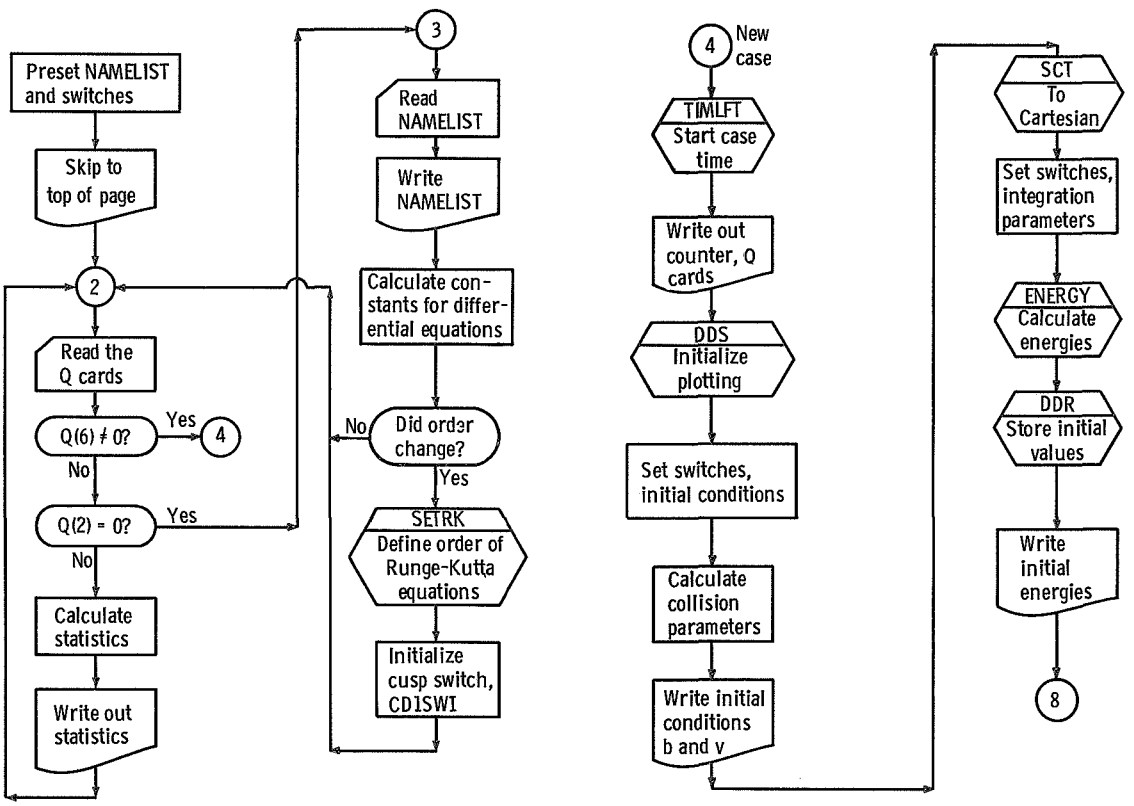
The main program reads the input, sets up the case, monitors the integration, and prints the output. With only slight exceptions, it is the only routine which calls other routines; the call structure is very simple. The arithmetic and logic flow for this program are diagrammed in figure 2.

The important switches local to this program are as follows:

RS — set to 2 if a capture occurs, otherwise set to 1

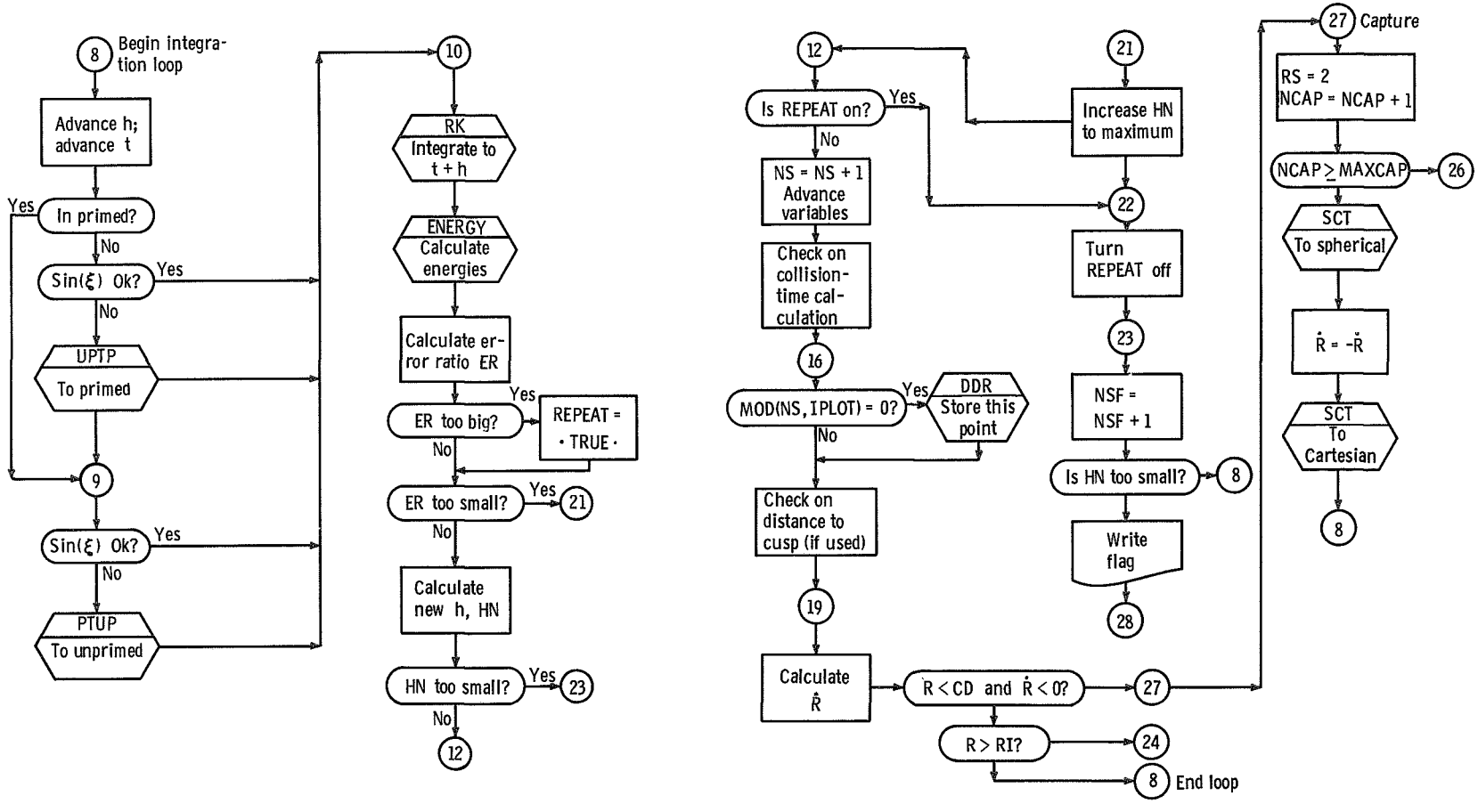
CD1SWI — set to 1 if CD1 > CD, signifying a Lennard-Jones potential (r^{-12}) term is to be included for this case; set to 3 if CD1 ≤ CD

CTSW — keeps track of capture status



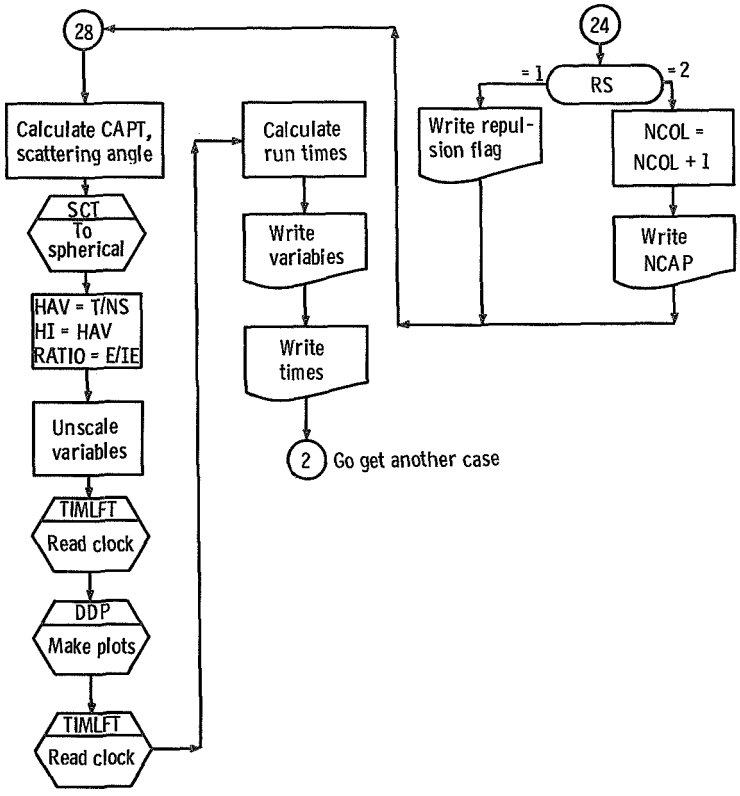
(a) Setting up for a case.

Figure 2. - Main program logic.



(b) Integration loop.

Figure 2. - Continued.



(c) Output for a case.

Figure 2. - Concluded.

CD1SW initially CD1SWI; then keeps track, if necessary, of nearness to Lennard-Jones distance CD1

REPEAT .TRUE. if bad step, .FALSE. if good

Subroutines

In addition to the main program, several subroutines are required. These subroutines are summarized in table II.

Use of COMMON

The following COMMON areas are used in this program:

(1) /DDC/RE, V2, SXI, R, T, KT, ID1, ID2, Q(12), CLRP, RUN(3), PLOT(5), CUTPLT:

TABLE II. - SUBROUTINE SUMMARY

Subroutine name (Entry)	Purpose	Called by-	Calls on-
JACK	Evaluates the equations of motion at time t	RK	
ENERGY	Calculates various energy terms at time t	Main	
RK	Integrates from t to t + h, using variable-order, variable h Runge-Kutta method	Main	JACK
SETRK	Defines the Runge-Kutta coefficients for the current order	Main	
PTUP	Transforms dipole coordinates from primed to unprimed	Main	
UPTP	Transforms dipole coordinates from unprimed to primed	Main	
SCT	Transforms ion coordinates from spherical to Cartesian	Main	
DARC	Calculates a double precision $\cos^{-1}$	Main	
[a]DDS	Makes plots; initializes	Main	HEAD; DOTS; BLANK, TEST
DDR	Stores points; entry in DDS	Main	
DDP	Plots points; entry in DDS	Main	
HEAD or HEADM	Makes headings	DDP	
DOTS	Draws dotted circles (for motion pictures)	DDP	
BLANK, TEST	Makes leaders (for motion pictures)		
PLTDUM	Contains dummy entries for the purpose of this report (Entries are explained in comments cards, see section Program Listings.)	Main; DDP	

[a]Two decks with same name; one used for plots, the other for motion pictures.

This COMMON area appears in the main program, and in DDS and HEAD; the variables are

RE	rotational energy
V2	velocity squared
SXI	$\sin(\xi)$
R	ion-dipole separation
T	time
KT	rotational energy scale factor
ID1, ID2	plot labels
Q(12)	coordinate vector
CLRP	plot label switch
RUN(3)	plot label, BCD
PLOT(5)	plot selector array
CUTPLT	plot truncation switch

(2) /JACKEN/EROC, RM, V, ETRC, EPLC, EPTC, XXC1, XXC2, XC, JCD1, EP, SIG, ETRAN, EPOT, EPOL: This COMMON area appears in the main program and in JACK and ENERGY; the variables are

EROC	$I_1/2$
RM	I_2/I_1
V	velocity
ETRC	$m/2$
EPLC	$\alpha e^2/2$
EPTC	μe
XXC1	$\mu e/m$
XXC2	$2\alpha e^2/m$
XC	$\mu e/I_1$
JCD1	switch for Lennard-Jones potential (If JCD1 = 1, infinite barrier.)
EP	Lennard-Jones interaction potential strength parameter

SIG characteristic Van der Walls diameter

$$\left[V_{L\text{-}J} = EP\left(\frac{SIG}{R}\right)^{12}\right]$$

ETRAN translational energy

EPOT potential energy

EPOL polarization energy

where energy, E = EROT + ETRAN - EPOL - EPOT

(3) /INTSET/C1, C2, C4, RN appears in the main program and in SETRK; the variables are a function of the order n of the Runge-Kutta method:

C1 $(2^n - 1)^{-1}$

C2 $(2^n - 1)^{-1}\, 2^n$

C4 $(MAXHR)^{-n-1}$

RN $1/(n + 1)$

(4) /INTES/TOTERR, MAXER, HI, MAXHR, HMAX, HMIN, HAV, NS, NSF: This COMMON area appears in the main program and in SETRK; it contains the integration parameters

TOTERR total relative error in energy to be tolerated

MAXER maximum local error ratio to be tolerated

HI starting time step for this case

MAXHR maximum step increase factor

HMAX maximum step taken during this integration

HMIN minimum step taken during this integration

HAV average step taken during this integration

NS good step counter during this integration

NSF bad step counter during this integration

(5) /RKC/NV, NVM1, C(11), B(12), A(66): This COMMON area appears in SETRK and RK; these variables are the Runge-Kutta constants, constructed by SETRK and used by RK:

(6) /PCSC/PCS appears in the main program and in DDS, JACK, and ENERGY.

PCS = 1: not in polar cap, unprimed

PCS = 2 in primed system

(7) /LANG/LS appears in the main program and in JACK and ENERGY.

LS = .TRUE.: $\mu = 0$, Langevin case

LS = .FALSE.: not Langevin

(8) /CIRCLE/RADIUS, RVIEW, ON appears in the main program and in DDS and DOTS (used for motion pictures only).

RADIUS — a function of collision distance and of distance from X-Y motion-picture plane; the radius of the dotted molecule or ion (Fig. 3 shows the function of RADIUS and RVIEW.)

RVIEW — the scanning radius for the motion-picture frame, in angstroms (The ion and molecule are filmed when they are within two RVIEW of each other.)

ON — If ON = .TRUE., a dotted circle will be on the frame. If ON = .FALSE, no circle can be drawn.

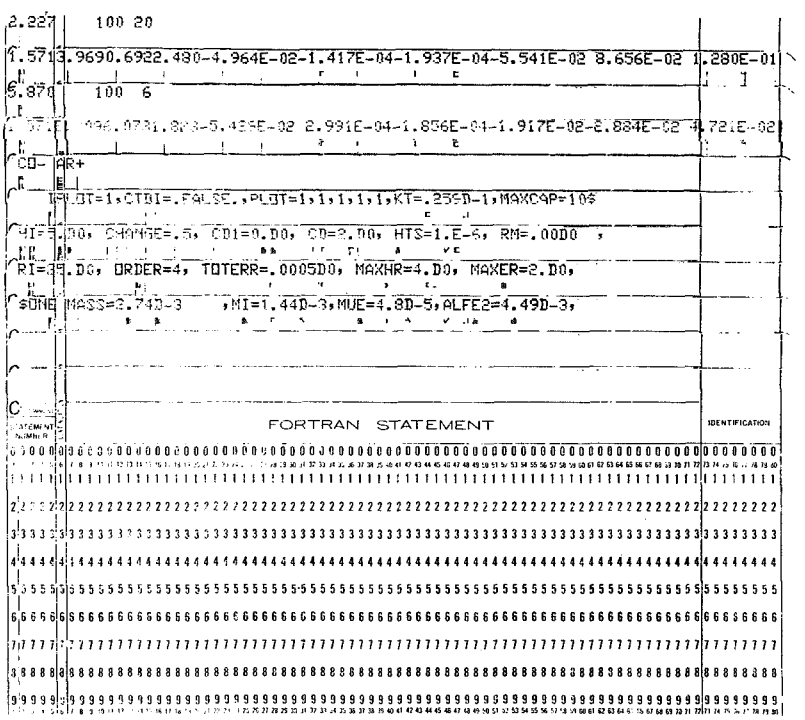


Figure 3. - Input card images.

Sample Case

The input cards for a sample case are shown in figure 3. Two blank cards "trigger" the reading of the NAMELIST and then the RUN array only if plots are required; two Q-cards follow. This case produces the following output on the listing:

```
$ONE
MASS  =       0.274CCCCCCCCCCCCCD-C2, MI    =       0.144000000000000D-02, MUE   =       C.4800CCCCCCCCCCCD-04,
ALFE2 =       0.449CCCCCCCCCCCCCD-02, RI    =       0.250000000000000C 02, ORDER =                4,
TOTERR=       C.50CCCCCCCCCCCCCCD-03, MAXHR =       0.400C000000000000C 01, MAXER =       C.2000CCCCCCCCCCCD 01,
HI    =       0.5CCCCCCCCCCCCCCCD 01, CHANGE=    5.0000000E-01, CTCI  =     F, CTD   =       0.1CCCCCCCCCCCCCCD 02,
CC    =       0.200CCCCCCCCCCCCCD C1, MAXCAP=               10, CEI   =       C.                    , HTS   =    1.0000000E-06,
RM    =       0.                    ,
PLCT  =                1,              1,              1,              1,              1,
KT    =       0.259CCCCCCCCCCCCCD-01, IPLCT =               1, OUTPLT=     T,
$ ENC


CASE    1   INPLT       1.571  2.2C9  6.073  1.823  -5.4290-02   2.991D-04  -1.856C-C4  -1.9170-C2  -2.9840-02   4.7210-02
                                                                                                          5.870     100     6
                     R         THETA        PHI         ETA         XI         PSI
INITIAL       Q      2.50D-C7     1.5710     2.2090      6.0730      1.8230     5.870C    V     5.50D C4    B     4.000-08
CONDITIONS    DQ    -5.43D 04    2.99D 10  -1.86D 10   -1.92D 12   -2.88D 12   4.72C 12   IE     5.03D-06  IRE     8.470-07
FINAL         Q      2.51D-C7     2.C671    -0.7181    -43.5264     1.5306    2E.4459    E     5.C3D-06   RE     3.22D-06  C     1
VALUES        DQ     3.61D 04  -2.78D 10  -1.91D 10   -6.20D 12   -2.51D 12   5.45C 12   RATIO    1.000 TIME     1.01D-11
COMP TIME      8.87     NS      352     NSF      67     HAV       2.865     HMAX      1C.44E    HMIN       C.C52  CAPT =  1.9065
COLLISION TIME    2.12E-12      DISTANCE    8.23E-08      RUN TIME   8.87      PLOT TIME   C.


CASE    2   INPLT       1.571  3.96S  C.692  2.480  -4.964D-02  -1.417C-04  -1.9270-C4  -5.541D-C2   8.656D-02   1.2800-01
                                                                                                          2.227     100    20
                     R         THETA        PHI         ETA         XI         PSI
INITIAL       Q      2.50D-C7     1.5710     3.9690      0.6920      2.4800     2.2270    V     5.00D 04    B     3.000-08
CONDITIONS    DQ    -4.S6D C4   -1.42D 10  -1.94D 10   -5.54D 12   8.66D 12   1.28C 13   IE     9.64D-06  IRE     6.23D-06
FINAL         Q      2.51D-C7     C.9857     2.3194      2.1625      0.2356     1.5187    E     9.64D-C6   RE     7.36D-06  C     1
VALUES        DQ     4.08D 04    1.43D C9  -2.78D 10   -3.74D 13   5.10D 12   5.35D 13   RATIO    1.000 TIME     9.40D-12
COMP TIME     21.63     NS      753     NSF      256     HAV       1.248     HMAX      4.614     HMIN       0.058  CAPT =  2.3051
COLLISION TIME    3.58E-12      DISTANCE    1.21E-07      RUN TIME  21.63      PLOT TIME   0.
```

In addition to this output, motion pictures or still plots can be made by including the desired subroutines. Figure 4 shows a sample motion-picture frame. Figure 5 shows sample still plots.

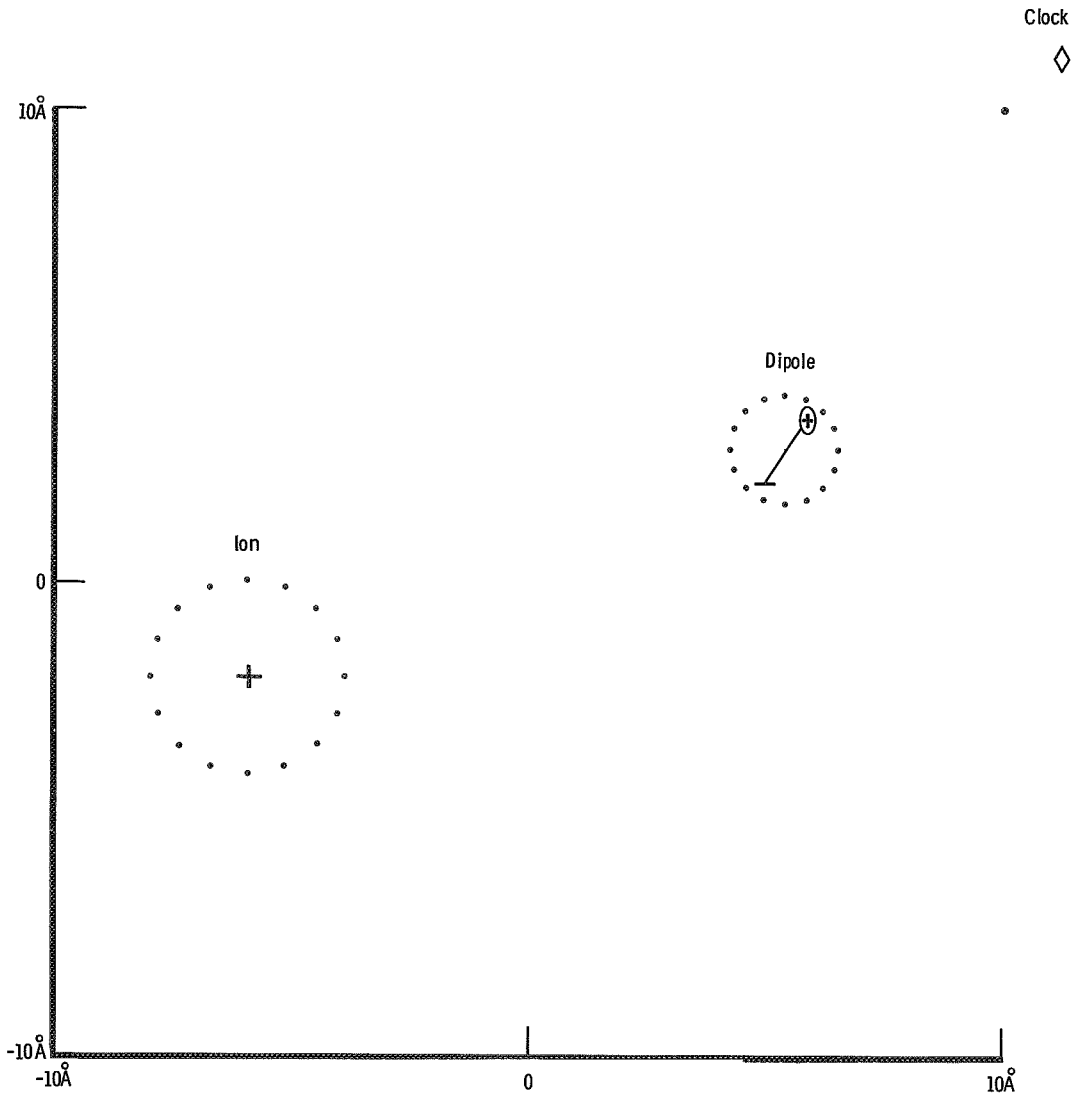
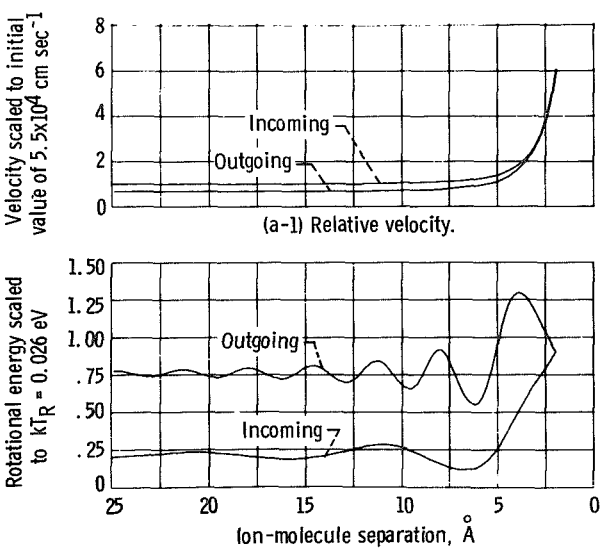


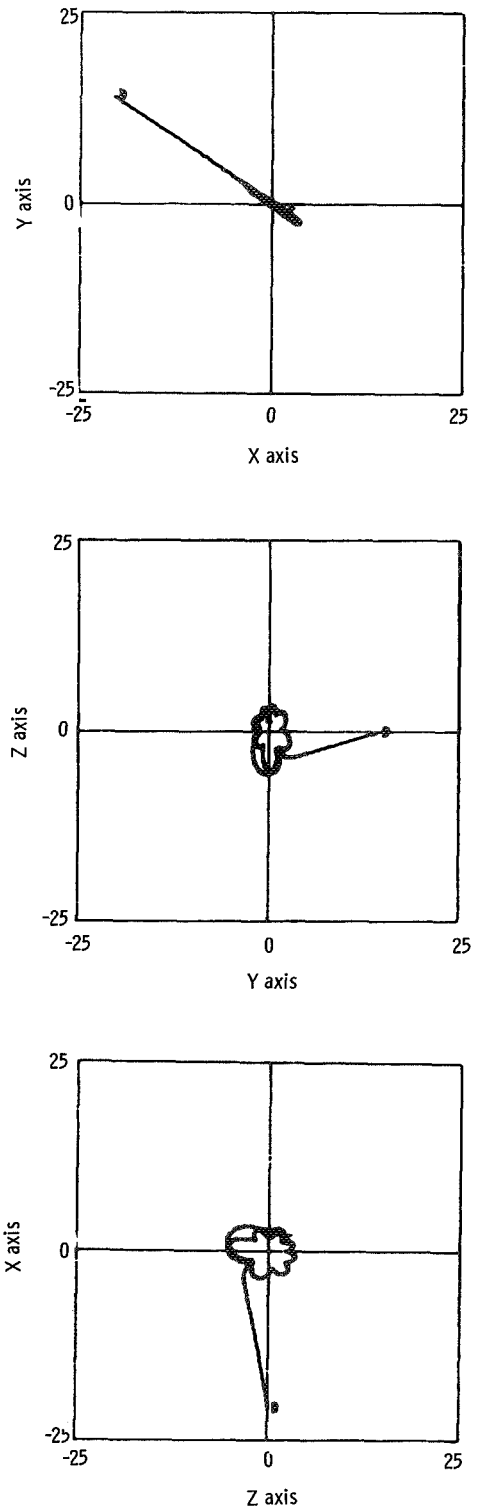
Figure 4. - Sample motion-picture frame. CS-51925



(a-1) Relative velocity.

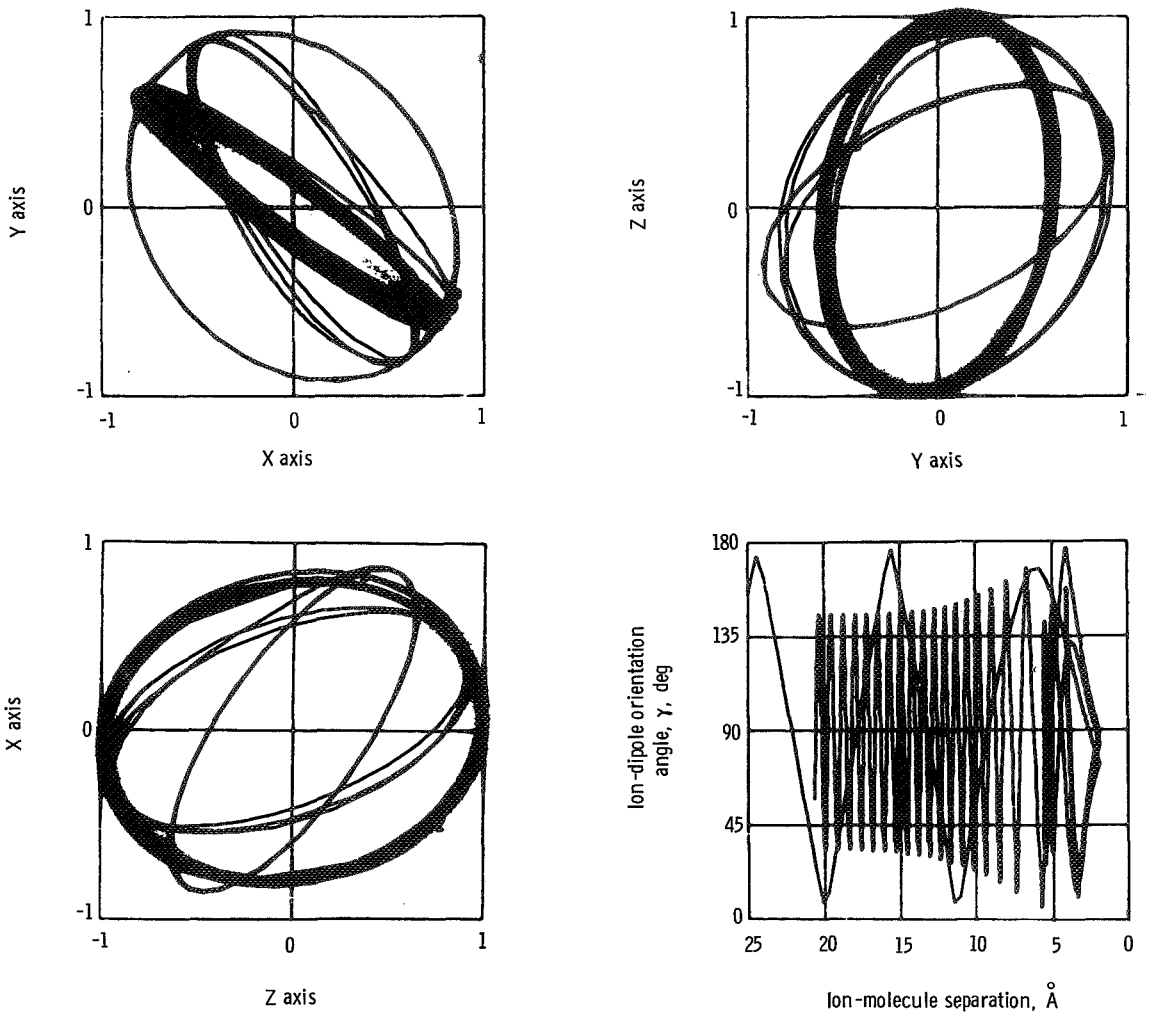
(a-2) Molecular rotational energy.

(a) Variations of ion-molecule relative velocity and polar-molecule rotational energy during Ar^+ + CO single-reflection capture collision.



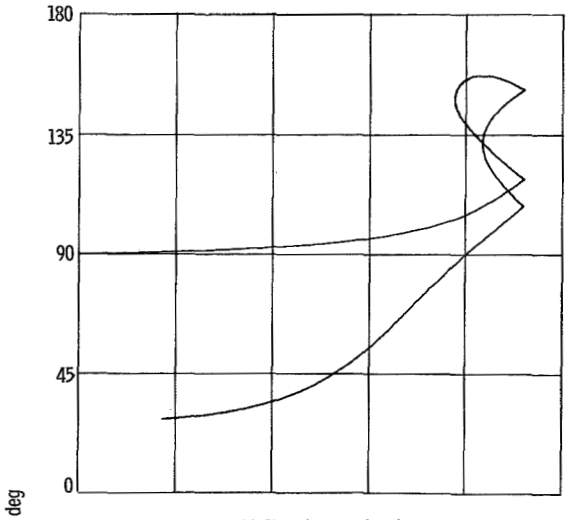
(b) Variation of ion coordinates for NO_2^+ + HCl multiple-reflection capture collision.

Figure 5. - Sample still plots.

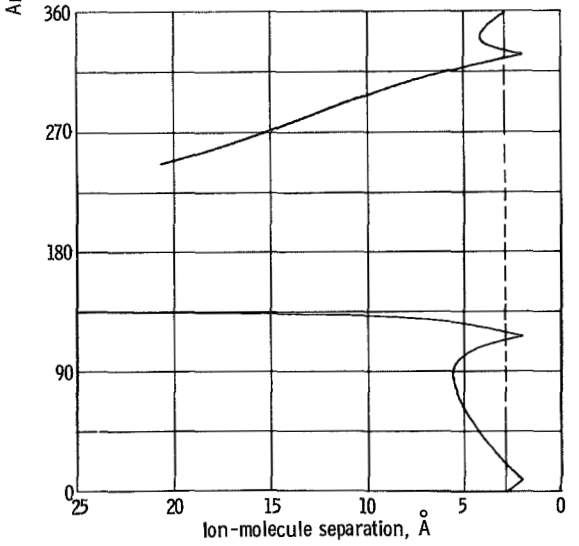


(c) Variations of dipole moment vector and ion-dipole orientation angle (between ion-molecule radius vector and negative end of dipole) during Ar^+ + CO multiple-reflection capture collision.

Figure 5. - Continued.



(d-1) Polar angle, θ.



(d-2) Azimuthal angle, φ; vertical dashed line indicates passing through 0 or 2π.

(d) Variations of coordinate angles for argon ion relative to CO molecule during capture collision with multiple reflections.

Figure 5. - Concluded.

Program Listings

The complete program is listed below in FORTRAN IV (ref. 11). Not included are the routines which actually make the microfilm plots, because they are unique to this Center. A dummy deck, PLTDUM, explains these routines in comments cards so that others can be substituted. Of course, statistics were gathered, and even individual trajectories studied, without plots.

```
$IBFTC MAIN.

C      7090 DOUBLE PRECISION SINGLE SYMMETRIC TOP (WITH ROD AS A SPECIAL
C      CASE) PROGRAM WITH DD80 PLOTTING OPTION
       INTEGER PCS,RS,CTSW,ORDER,CASE,CLRP,PLOT,TNCOL,
      1 TNCASE,CD1SW,CD1SWI
       LOGICAL LS
       LOGICAL REPEAT,CTDI,CUTPLT
       DOUBLE PRECISION TOTERR,MAXER,HI,MAXHR,HMAX,HMIN,HAV,
      1                 C1,C2,C4,RN,
      2                 ERDC,RM,V,ETRC,EPLC,EPTC,XXC1,XXC2,XC,CD1,
      3                 RE,V2,SXI,R,T,KT,Q,
      4                 RI,MASS,MI,MUE,ALFE2,
      5                 B,CAPT,CD,RD,CTD,E,IE,IRE,RATIO,RATIOL,
      6                 H,EA,EL,ER,TMAX,HN,Q1,SZ2,T1,Z,PSID
       DOUBLE PRECISION X,SQRT,ARCOS,EP,SIG,ETRAN,EPOT,EPOL
       LOGICAL ON
       EXTERNAL JACK
       DIMENSION Z(12),Q1(12)
C
C      COMMON FOR MOVIE DECKS (NOT STILL PLOTS)
       COMMON/CIRCLE/ RADIUS,RVIEW,ON
C      COMMON FOR INTEGRATION PARAMETERS AND STATISTICS
       COMMON /INTES/ TOTERR,MAXER,HI,MAXHR,HMAX,HMIN,HAV,NS,NSF
C
C      COMMON FOR COMMUNICATION WITH SETRK. QUANTITIES ARE FUNCTIONS OF T
C      INTEGRATION FORMULA ORDER
       COMMON /INTSET/ C1,C2,C4,RN
C
C      COMMON FOR TRANSMITTING CONSTANTS TO ENERGY AND JACK (JACK = DIFFE
C      ENTIAL EQUATION EVALUATOR) THE TRANSLATIONAL VELOCITY, V, ALSO
C      APPEARS.
       COMMON/JACKEN/ ERDC,RM,V,ETRC,EPLC,EPTC,XXC1,XXC2,XC,JCD1,
      1  EP,SIG,ETRAN,EPOT,EPOL
C      COMMON FOR LANGEVIN SWITCH, LS=.TRUE. IFF MUE=0.
       COMMON/LANG/ LS
C
C      COMMON HOLDS PRIMED-UNPRIMED COORDINATE SYSTEM SWITCH
C                1    = UPRIMED
C                2    = PRIMED
       COMMON /PCSC/ PCS
C
C      COMMON FOR TRANSMITTING CONTROL PARAMETERS, HEADING INFORMATION,
C      AND PLOT QUANTITIES TO PLOTTING SUBROUTINE
       COMMON /DDC/ RE,V2,SXI,R,T,KT,ID1,ID2,Q(12),CLRP,RUN(3),PLOT(5),CJ
      1ITPLT
C
       DATA CASE/0/,LORDER/0/,REPEAT/.FALSE./,CTD/1.D1/,CTDI/.FALSE./
       DATA NCOL,TNCOL,NCASE,TNCASE/0,0,0,0/
C
C      THESE VARIABLES ARE READ IN VIA GENERAL INPUT (NAMELIST) CARDS.
C      MASS, MI, MUE, AND ALFE2 MUST BE SPECIFIED. IF CTDI IS SPECIFIED
C      AS  TRUE, THEN CTD MUST ALSO BE SPECIFIED. IF THE ENTIRE PLOT
C      VECTOR IS NOT SET TO 0, THEN RUN AND KT MUST BE SPECIFIED.
C      MASS    REDUCED MASS OF ION-DIPOLE PAIR
C      MI      MOMENT OF INERTIA IN SCALED UNITS
C      MUE     DIPOLE MOMENT FACTOR IN SCALED UNITS
C      ALFE2   POLARIZATION FACTOR IN SCALED UNITS
C      RI      INITIAL SEPARATION IN ANGSTROMS
```

```
C      ORDER   ORDER OF RUNGE-KUTTA FORMULA TO BE USED
C      TOTERR  TOTAL ERROR TO BE TOLERATED OVER INTEGRATION
C      MAXHR   MAXIMUM STEP SIZE INCREASE FACTOR
C      MAXER   MAXIMUM ERROR RATIO TOLERATED
C      HI      INITIAL STEP SIZE IN SCALED TIME UNITS
C      CHANGE  SIN(XI)=CHANGE INITIATES COORDINATE TRANSFORMATION
C      CTDI    LOGICAL VARIABLE - COLLISION TIME DISTANCE INPUT SWITCH
C      CTD     COLLISION TIME DISTANCE (READ IN ONLY IF CTDI=.TRUE.)
C      CD      REFLECTION DISTANCE (CAPTURE DISTANCE FOR C.S. CALCS)
C      MAXCAP  MAXIMUM NUMBER OF CAPTURES PERMITTED
C      CD1     BOUNDARY OF POLARIZATION POTENTIAL, USED AS L--J SWITCH
C      HTS     MINIMUM STEP SIZE ALLOWED (IN SCALED TIME UNITS)
C      RM      RATIO OF SECOND MOMENT OF INERTIA TO FIRST MOMENT OF INERT
C           FOR ROD CASES RM=0.D0.
C      PLOT    PLOT CONTROL VECTOR. IF(PLOT(I).NE.0) THE I TH PLOT WILL B
C           DRAWN
C      KT      AVERAGE ROTATIONAL ENERGY
C      IPLOT   PLOT EVERY IPLOT TH POINT
C      CUTPLT  LOGICAL VARIABLE - IF TRUE, PLOTS ONLY 1249 POINTS TO AVOI
C           TAPE DELAY
       NAMELIST/ONE/ MASS,MI,MUE,ALFE2,RI,ORDER,TOTERR,MAXHR,MAXER,H
      1I,CHANGE,CTDI,CTD,CD,MAXCAP,CD1,HTS,RM,PLOT,KT,IPLOT,CUTPLT
C
C      REDEFINE FUNCTIONS AS DOUBLE PRECISION
       SQRT(X)=DSQRT(X)
       ARCOS(X)=DARCOS(X)
C
C      PRESET GENERAL INPUT QUANTITIES
       RI = 25.D0
C      EP,SIG   LENNARD-JONES(CUSP) POTENTIAL CONSTANTS
       EP=6.88D-6
       SIG=3.5D0
C
       ORDER=4
       TOTERR = .001D0
       MAXHR = 4.D0
       MAXER = 2.D0
       HI = 2.D0
       CHANGE=.707
       CD = 1.D0
       MAXCAP=10
       HTS=1.E-6
       RM=0.D0
       CTDI=.FALSE.
       IPLOT=1
       CUTPLT=.TRUE.
       DO 1 I1=1,NPLTS
1      PLOT(I1)=0
C
C      SKIP TO TOP OF PAGE
       WRITE (6,31)
C
C      READ A CARD
2      READ (5,32) (Q(I),I=2,12),ID1,ID2
C
C      R DOT NONZERO INDICATES A CASE SET
       IF (Q(6).NE.0.) GO TO 4
```

```
C
C     THETA ZERO INDICATES BLANK SET -  NEXT SET IS GENERAL INPUT
      IF (Q(2).EQ.0.) GO TO 3
C
C     OTHERWISE COMPUTE AND PRINT OUT COLLISION PERCENTAGES.(SKIP IF NO
C     CASES HAVE BEEN RUN YET.)
      IF (NCASE.EQ.0) GO TO 2
      PCCOL=100.*FLOAT(NCOL)/FLOAT(NCASE)
      TNCOL=TNCOL+NCOL
      TNCASE=TNCASE+NCASE
      TPCCOL=100.*FLOAT(TNCOL)/FLOAT(TNCASE)
      WRITE (6,33) NCOL,NCASE,PCCOL,TNCOL,TNCASE,TPCCOL
      NCOL=0
      NCASE=0
      GO TO 2
C
C     READ GENERAL INPUT SET AND WRITE OUT ALL GENERAL INPUT VARIABLES
3     READ (5,ONE)
      WRITE (6,ONE)
C        PASS RADIUS OF CORE TO MOVIE DECKS (IF ANY)
      RADIUS = CD
C
C     READ IN 18 CHARACTER RUN LABEL IF THERE IS TO BE ANY PLOTTING
      IF ((PLOT(1).NE.0).OR.(PLOT(2).NE.0).OR.(PLOT(3).NE.0).OR.(PLOT(4)
     1.NE.0).OR.(PLOT(5).NE.0)) READ (5,34) RUN
C
C     COMPUTE CONSTANTS FOR ENERGY AND DIFFERENTIAL EQUATION EVALUATION
C     SUBROUTINES
      ETRC = .5D0*MASS
      EPLC = .5D0*ALFE2
      XXC2 = 2.D0*ALFE2/MASS
C
C        SPECIAL CASE IF MUE = 0
      IF(MUE.NE.0.D0) GO TO 50
      LS = .TRUE.
      EROC = 0.D0
      EPTC = 0.D0
      XC = 0.D0
      XXC1 = 0.D0
      RM = 0.D0
      GO TO 51
   50 EROC = .5D0*MI
      LS = .FALSE.
      EPTC=MUE
      XC=MUE/MI
      XXC1=MUE/MASS
   51 CONTINUE
C
C
C     IF RUNGE-KUTTA FORMULA ORDER HAS BEEN CHANGED, CALL SETRK TO
C     CALCULATE NEW COEFFICIENTS
      IF (ORDER.EQ.LORDER) GO TO 2
      LORDER=ORDER
      CALL SETRK (ORDER)
C
C     SET CUSP POTENTIAL SWITCH INITIALIZATION VARIABLE
      CD1SWI=3
      IF (CD1.GT.CD) CD1SWI=1
```

```
C
C       RETURN TO READ A NEW SET OF CARDS
        GO TO 2
C
C       NEW CASE - COUNT IT AND CLOCK IN
4       CASE=CASE+1
        CALL TIMLFT (TIN)
        NCASE=NCASE+1
C
C       WRITE OUT THE CASE CARD
        WRITE (6,35) CASE,(Q(I),I=2,12),ID1,ID2
C
C       INITIALIZING CALL TO PLOT ROUTINE
        CALL DDS
C
C       SET CUSP POTENTIAL SWITCHES
        CD1SW=CD1SWI
C       EP,SIG    LENNARD-JONES(CUSP) SWITCH SETTING
        JCD1=1
C
C       RETRIEVE INITIAL CONDITIONS AND UNSCALE THEM FOR OUTPUT
        Q(1)=RI
        Q(6) = -DABS(Q(6))
        DO 5 I=2,6
        Z(I)=Q(I)
      5 Z(I+5) = Q(I+5)*1.D14
        Z(1) = Q(1)*1.D-8
        Z(6) = Q(6)*1.D6
        Z(12)=Q(12)
C
C       COMPUTE VELOCITY, IMPACT PARAMETER, ESTIMATED INTERACTION TIME, AN
C       COLLISION TIME DISTANCE
        SZ2 = DSIN(Z(2))
        V=SQRT(Z(6)*Z(6)+Z(1)*Z(1)*(Z(7)*Z(7)+(Z(8)*SZ2)**2))
        TMAX = 2.D6*RI/V
        B = (Z(1)**2*SQRT(Z(7)**2+(Z(8)*SZ2**2)**2))/V
        IF(.NOT.CTDI)  CTD = SQRT(ALFE2/(ETRC*V*V))*1.D-2/B
C
C       OUTPUT INITIAL CONDITIONS, VELOCITY, AND IMPACT PARAMETER
        WRITE (6,36) (Z(I),I=1,5),Z(12),V,B,(Z(I),I=6,11)
C
C       TRANSFORM TRANSLATIONAL VARIABLES TO RECTANGULAR COORDINATES
        CALL SCT (Q(1),Q(6),Q(1),Q(6),1)
C
C       STORE INITIAL CONDITION VECTOR
        DO 6 I=1,12
6       Z(I)=Q(I)
C
C       INITIALIZE TIME, STEP SIZE MAXIMUM AND MINIMUM STEP SIZE,
C       REPULSION SWITCH, STEP COUNTER, BAD STEP COUNTER, AND CAPTURE
C       COUNTER
        T = 0.D0
        HMIN=HI
        HMAX=HI
        HN=HI
        RS=1
        NS=0
        NSF=0
        NCAP=0
```

```
C     PERMUTE INTERGRATION VARIABLES AND DERIVATIVES TO ORDER R, THETA,P
C     ETA,XI,PSI
      PSID=Q(11)
      DO 7 K=6,10
      I=16-K
7     Q(I+1)=Q(I)
      Q(6)=Q(12)
      Q(12)=PSID
C
C     SET PRIMED-UNPRIMED COORDINATE SWITCH TO UNPRIMED
      PCS=1
C
C     COMPUTE INITIAL ENERGY
      CALL ENERGY(IE,IRE,Q(1),Q(7))
C
C     INITIALIZE ENERGY RATIO AND R
      RATIOL = 1.DO
      R=RI
      RE=IRE
C
C     RECORD INITIAL VALUES FOR PLOTTING
      CALL DDR
C
C     OUTPUT INITIAL ENERGY AND ROTATIONAL ENERGY
      WRITE (6,37) IE,IRE
C
C     INITIALIZE COLLISION TIME SWITCH AND COLLSION TIME
      CTSW=1
      CT=0.
C
C     BEGIN A STEP - SET STEP SIZE AND TRIAL VARIABLE
8     H=HN
      T1=T
C
C     BRANCH IF WE ARE IN PRIMED COORDINATE SYSTEM
      IF (PCS.EQ.2) GO TO 9
C
C     WE ARE IN UNPRIMED SYSTEM. BRANCH IF WE SHOULD STAY IN THIS SYSTEM
      IF(ABS(SIN(SNGL(Q(5)))).GE.CHANGE) GO TO 10
C
C     TRANSFORM TO PRIMED COORDINATE SYSTEM
      PCS=2
      CALL UPTP (Q(1),Q(7),Q(1),Q(7))
C
C     BRANCH TO INTEGRATION CALL
      GO TO 10
C
C     WE ARE IN PRIMED SYSTEM. CHECK TO SEE IF WE SHOULD STAY IN PRIMED
C     SYSTEM. BRANCH TO INTEGRATION CALL IF WE SHOULD
    9 XI = ARCOS(-DSIN(Q(5))*DSIN(Q(4)))
      IF (ABS(SIN(XI)).LT.CHANGE) GO TO 10
C
C     TRANSFORM TO UNPRIMED COORDINATE SYSTEM
      PCS=1
      CALL PTUP (Q(1),Q(7),Q(1),Q(7))
C
C     CALL RUNGE-KUTTA INTEGRATION SUBROUTINE. ORDER OF FORMULA HAS
C     ALREADY BEEN SPECIFIED.
   10 CALL RK(12,T1,H,Q,Q1,JACK)
```

```
C
C      CALL ENERGY ROUTINE AND COMPUTE ENERGY RATIO
   11  CALL ENERGY(E,RE,Q1(1),Q1(7))
       RATIO=E/IE
C
C      COMPUTE ERROR MADE IN THIS STEP
       EL = DABS(RATIO-RATIOL)
C
C      COMPUTE ERROR ALLOWED FOR THIS STEP
       EA = H*(TOTERR - DABS(RATIOL-1.D0))/TMAX
C
C      COMPUTE RATIO OF ACTUAL ERROR TO ALLOWED ERROR
       ER=EL/EA
C
C      TURN STEP REPEAT SWITCH ON IF ERROR RATIO IS TOO LARGE
       IF (ER.GT.MAXER) REPEAT=.TRUE.
C
C      BRANCH TO MAXUMUM STEP SIZE INCREASE SECTION IF ERROR IS VERY SMAL
       IF (ER.LT.C4) GO TO 21
C
C      COMPUTE SIZE OF NEXT INTEGRATION STEP
       HN = H*(1.D0/ER)**RN
C
C      ENTER HERE WITH NEW STEP SIZE
12     CONTINUE
C
C         TRY TO PREVENT HANGUP(ESP. HCL)
       IF(EL.GT.1.D-6 .OR. .NOT.REPEAT) GO TO 60
       REPEAT = .FALSE.
   60  CONTINUE
C
C      BRANCH TO STEP FAILURE SECTION IF REPEAT SWITCH IS ON
       IF (REPEAT) GO TO 22
C
C      SUCCESSFUL STEP - COUNT IT, RECORD LAST RATIO, ADVANCE TIME,
C      UPDATE DEPENDENT VARIABLE VECTOR, UPDATE MAXIMUM AND MINIMUN
C      STEP SIZES, RECORD LAST R, COMPUTE NEW R.
       NS=NS+1
       RATIOL=RATIO
       T=T1
       DO 13 I=1,12
13     Q(I)=Q1(I)
       HMAX=DMAX1(HMAX,H)
       HMIN=DMIN1(HMIN,H)
       R=SQRT(Q(1)*Q(1)+Q(2)*Q(2)+Q(3)*Q(3))
C
C      FORK ON COLLISION TIME SWITCH
       GO TO (14,15,16),CTSW
C
C      CHECK TO SEE IF WE HAVE PASSED COLLISION DISTANCE ON THE WAY IN.
C      IF SO, RECORD TIME AND RESET SWITCH TO SEARCH FOR COLLISION DISTAN
C      ON THE WAY OUT.
14     IF (R.GT.CTD) GO TO 16
       CTSW=2
       CT1=T
       GO TO 16
C
C      CHECK TO SEE IF WE HAVE PASSED COLLISION DISTANCE ON THE WAY OUT.
C      IF SO, RECORD TIME, COMPUTE ELAPSED TIME IN SECONDS, AND RESET SWI
```

```
C       TO BYPASS THIS SECTION.
15      IF (R.LT.CTD) GO TO 16
        CTSW=3
        CT2=T
        CT=(CT2-CT1)*1.E-14
C
C       RECORD EVERY IPLOT TH POINT FOR PLOTTING
   16   IF(MOD(NS,IPLOT).EQ.0)  CALL DDR
C
C       CUSP POTENTIAL SWITCH SETTING SECTION
        GO TO (17,18,19),CD1SW
17      IF (R.GT.CD1) GO TO 20
        CD1SW=2
        JCD1=2
        GO TO 19
18      IF (R.LE.CD1) GO TO 19
        CD1SW=1
        JCD1=1
        GO TO 20
C
C       BRANCH IF CAPTURE CONDITION HAS OCCURRED. (A CAPTURE HAS OCCURRED
C       IF R IS LESS THAN THE COLLISION DISTANCE AND IS DECREASING.)
19      RD=(Q(1)*Q(7)+Q(2)*Q(8)+Q(3)*Q(9))/R
        IF((R.LT.CD) .AND. (RD.LT.0.D0)) GO TO 27
C
C       BRANCH TO REPULSION SECTION IF CASE IS FINISHED. (A CASE IS FINISH
C       IF R IS GREATER THAN THE INITIAL R=RI)
20      IF (R.GT.RI) GO TO 24
C
C       RETURN FOR THE NEXT STEP
        GO TO 8
C
C       MAXIMUM STEP SIZE INCREASE SECTION
21      HN=MAXHR*H
        GO TO 12
C
C       STEP FAILURE SECTION - TURN OFF REPEAT SWITCH, COUNT THE FAILURE,
C       AND RETURN TO REPEAT THE STEP UNLESS NEW STEP SIZE IS TO SMALL.
22      REPEAT=.FALSE.
23      NSF=NSF+1
        IF(ABS(SNGL(HN)).GE.HTS) GO TO 8
C
C       STEP SIZE TOO SMALL - WRITE OUT SIGNAL, REDUCE CASE COUNT, SET PLO
C       CASE TYPE LABEL SWITCH, AND BRANCH TO OUTPUT SECTION
        RATERR = ABS(SNGL(RATIOL-1.D0))
        WRITE (6,38) RATERR
        NCASE=NCASE-1
        CLRP=4
        GO TO 28
C
C       FORK ON REPULSION SWITCH
C                 1  = REPUSION
C                 2  = CAPTURE HAS ALREADY OCCURRED
24      GO TO (25,26),RS
C
C       REPUSION SECTION - WRITE OUT SIGNALS, SET PLOT CASE TYPE LABEL SWI
C       AND BRANCH TO OUTPUT SECTION
```

```
25    WRITE (6,39)
      CLRP=2
      GO TO 28
C
C     COUNT CASE AS CAPTURE, WRITE OUT NUMBER OF CAPTURES THIS CASE.
C     SET PLOT CASE TYPE LABEL SWITCH, AND BRANCH TO OUTPUT SECTION.
26    WRITE (6,40) NCAP
      NCOL=NCOL+1
      CLRP=1
      GO TO 28
C
C     CAPTURE SECTION - COUNT CAPTURE,BRANCH TO CAPTURE CASE END SECTION
C     IF MAXIMUN NUMBER OF CAPTURES PERMITTED HAS OCCURED.
C     OTHERWISE TRANSFORM TRANSLATIONAL VARIABLES TO SPHERICAL
C     COORDINATES, CHANGE SIGN OF R DOT(RADIAL VELOCITY), TRANSFORM TRAN
C     LATIONAL VARIABLES BACK TO RECTANGULAR COORDINATES, SET REPULSION
C     SWITCH TO INDICATE CAPTURE, AND BRANCH BACK TO RESUME INTEGRATION.
27    NCAP=NCAP+1
      IF (NCAP.EQ.MAXCAP) GO TO 26
      CALL SCT (Q(1),Q(7),Q(1),Q(7),2)
      Q(7)=-Q(7)
      CALL SCT (Q(1),Q(7),Q(1),Q(7),1)
      RS=2
      GO TO 8
C
C
C     OUTPUT SECTION - CASE IS FINISHED.
28    CONTINUE
C
C     IF ROTATIONAL VARIABLES ARE IN PRIMED COORDINATE SYSTEM, TRANSFORM
C     TO UNPRIMED SYSTEM.
      IF (PCS.EQ.2) CALL PTUP (Q(1),Q(7),Q(1),Q(7))
C     COMPUTE SCATTERING ANGLE
      CAPT=ARCOS((Q(7)*Z(7)+Q(8)*Z(8)+Q(9)*Z(9))/SQRT((Q(7)*Q(7)+Q(8)*Q(
     18)+Q(9)*Q(9))*(Z(7)*Z(7)+Z(8)*Z(8)+Z(9)*Z(9))))
C
C     TRANSFORM TRANSLATIONAL VARIABLES TO SPHERICAL COORDINATES.
      CALL SCT (Q(1),Q(7),Q(1),Q(7),2)
C
C     UNLESS THERE HAVE BEEN NO SUCCESSFUL STEPS, COMPUTE AVERAGE STEP
C     SIZE AND STORE IT FOR INITIAL STEP SIZE NEXT CASE
      IF (NS.EQ.0) GO TO 29
      HAV = T/DBLE(FLOAT(NS))
      HI=HAV
C
C     COMPUTE RATIO OF FINAL ENERGY TO INITIAL ENERGY
29    RATIO=E/IE
C
C     UNSCALE INDEPENDENT AND DEPENDENT VARIABLES
      T = T*1.D-14
      Q(1) = Q(1)*1.D-8
      Q(7) = Q(7)*1.D6
      DO 30 I=8,12
   30 Q(I) = Q(I)*1.D14
C
C     READ CLOCK AFTER INTEGRATION AND BEFORE PLOTTING
      CALL TIMLFT (TBFP)
```

```
C
C       PLOT THIS CASE
        CALL DDP
C
C       CLOCK OUT AND COMPUTE COMPUTER TIME FOR THIS CASE
        CALL TIMLFT (TOUT)
        REALT=(TIN-TOUT)/60.
C
C       WRITE OUT FINAL CONDITIONS, ENERGY, ROTATIONAL ENERGY, ELAPSED
C       COMPUTER TIME, AND INTEGRATION STATISTICS.
        WRITE (6,41) (Q(I),I=1,6),E,RE,(Q(I),I=7,12),RATIO,T,REALT,NS,NSF,
       1HAV,HMAX,HMIN
C
C       WRITE OUT SCATTERING ANGLE
        WRITE (6,42) CAPT
C
C       SET COLLISION TIME TO CASE TIME IF COLLISON DISTANCE IS NOT LESS
C       THAN INITIAL SEPARATION
        IF (CTD.GE.RI) CT=T
C
C       SCALE COLLISION TIME DISTANCE
        CTDS=CTD*1.E-8
C
C       COMPUTE RUN TIME AND PLOT TIME
        TRUN=(TIN-TBFP)/60.
        TPLOT=(TBFP-TOUT)/60.
C
C       WRITE OUT COLLISION TIME, COLLISION TIME DISTANCE, INTEGRATION TIM
C       AND PLOTTING TIME
        WRITE (6,43) CT,CTDS,TRUN,TPLOT
C
C       RETURN FOR ANOTHER CASE
        GO TO 2
C
C
31      FORMAT (1H1)
32      FORMAT (0P4F5.3,1P6E10.3/0PF5.3,4X,2I3)
33      FORMAT (///6H CASES,I6,11H COLLISIONS,I6,8H PERCENT,F6.1/12H TOTAL
       1 CASES,I6,17H TOTAL COLLISIONS,I6,8H PERCENT,F6.1///)
34      FORMAT (3A6)
35      FORMAT (////5H CASE,I5,3X,5HINPUT,5X,0P4F7.3,1P6D12.3/112X,0PF7.3,
       12X,2I5)
36      FORMAT (24X,1HR,8X,5HTHETA,7X,3HPHI,8X,3HETA,8X,2HXI,7X,3HPSI/1X,7
       1HINITIAL,7X,1HQ,3X,1PD11.2,0P5F11.4,4X,1HV,2X,1PD11.2,4X,1HB,2X,1P
       2D11.2/1X,10HCONDITIONS,3X,2HDQ,3X,1P6D11.2)
37      FORMAT (1H+,87X,2HIE,2X,1PD11.2,2X,3HIRE,2X,1PD11.2)
38      FORMAT (127X,1HS,1PE9.2)
39      FORMAT (123X,9HREPULSION)
40      FORMAT (123X,1HC,I6)
41      FORMAT (1H+,5HFINAL,9X,1HQ,3X,1PD11.2,0P5F11.4,4X,1HE,2X,1PD11.2,3
       1X,2HRE,2X,1PD11.2/1X,6HVALUES,7X,2HDQ,3X,1P6D11.2,4X,5HRATIO,0PF9.
       23,1X,4HTIME,2X,1PD11.2/1X,9HCOMP TIME,3X,0PF7.2,4X,2HNS,3X,I5,4X,3
       3HNSF,3X,I5,4X,3HHAV,3X,F8.3,4X,4HHMAX,3X,F8.3,4X,4HHMIN,3X,F8.3)
42      FORMAT (1H+,106X,6HCAPT =,F8.4)
43      FORMAT (1X,14HCOLLISION TIME,1PE12.2,6X,8HDISTANCE,1PE12.2,6X,8HRU
       1N TIME,0PF7.2,6X,9HPLOT TIME,0PF7.2)
        END
```

```
$IBFTC JACK.

      SUBROUTINE JACK(T,X,DX)
      DIMENSION X(12),DX(12)
      INTEGER PCS
      LOGICAL LS
      DOUBLE PRECISION T,X,DX,EROC,RM,V,ETRC,EPLC,EPTC,XXC1,XXC2,XC,
     1                 VLJ,  R,R2,R3,R5,R6,XI,ETA,SXI,CETA,SETA,
     2                 CXI,PSI,DPSI,DETA,DXI,VPOT,XXC3,EP,SIG
     3 ,ETRAN,EPOT,EPOL
      COMMON/PCSC/ PCS
      COMMON/LANG/ LS
      COMMON /JACKEN/ EROC,RM,V,ETRC,EPLC,EPTC,XXC1,XXC2,XC,JCD1,EP,SIG
     1 ,ETRAN,EPOT,EPOL
      DO 1 I=1,6
    1 DX(I) = X(I+6)
      ETA = X(4)
      XI = X(5)
      PSI = X(6)
      DXI = DX(5)
      DETA = DX(4)
      DPSI = DX(6)
      R = DSQRT(X(1)*X(1) + X(2)*X(2) + X(3)*X(3))
      R2 = R*R
      R3 = R*R2
      R5 = R2*R3
      SETA = DSIN(ETA)
      CETA = DCOS(ETA)
      SXI = DSIN(XI)
      CXI = DCOS(XI)
      GO TO (20,30), PCS
   20 VPOT = 3.D0*(X(1)*SXI*SETA - X(2)*SXI*CETA + X(3)*CXI)
      DX(7) = XXC1*(R2*SXI*SETA - X(1)*VPOT)/R5
      DX(8) = -XXC1*(R2*SXI*CETA + X(2)*VPOT)/R5
      DX(9) = XXC1*(R2*CXI - X(3)*VPOT)/R5
   24 R6 = R3*R3
      XXC3 = XXC2/R6
      DO 25 I=1,3
   25 DX(I+6) = DX(I+6)-X(I)*XXC3
      GO TO (26,27),JCD1
   27 VLJ=EP*(SIG/R)**12
      DO 29 I=1,3
   29 DX(I+6) = DX(I+6) + 12.D0*X(I)*VLJ/R2
   26 IF(LS) GO TO 40
      DX(10) = (XC*(X(1)*CETA + X(2)*SETA)/R3 - 2.D0*DETA*DXI*CXI +
     1 RM*DXI*(DPSI+DETA*CXI))/SXI
      DX(11) = DETA**2*SXI*CXI + XC*(CXI*(X(1)*SETA-X(2)*CETA) - X(3)*
     1 SXI)/R3 - RM*SXI*(DPSI+DETA*CXI)*DETA
      DX(12) = DETA*DXI*SXI - DX(10)*CXI
      GO TO 50
   30 VPOT = 3.D0*(X(1)*CXI - X(2)*SXI*CETA - X(3)*SXI*SETA)
      DX(7) = XXC1*(R2*CXI - X(1)*VPOT)/R5
      DX(8) = -XXC1*(R2*SXI*CETA + X(2)*VPOT)/R5
      DX(9) = -XXC1*(R2*SXI*SETA + X(3)*VPOT)/R5
   34 R6 = R3*R3
      XXC3 = XXC2/R6
      DO 35 I=1,3
   35 DX(I+6) = DX(I+6) - X(I)*XXC3
      GO TO (36,37),JCD1
```

```
   37 VLJ=EP*(SIG/R)**12
      DO 39 I=1,3
   39 DX(I+6) = DX(I+6) + 12.D0*X(I)*VLJ/R2
   36 IF(LS) GO TO 40
      DX(10) = (XC*(X(2)*SETA-X(3)*CETA)/R3 - 2.D0*DETA*DXI*CXI +
     1 RM*DXI*(DPSI+DETA*CXI))/SXI
      DX(11) = DETA**2*SXI*CXI - XC*(X(1)*SXI+CXI*(X(2)*CETA+X(3)*SETA))
     1 /R3 - RM*SXI*(DPSI+DETA*CXI)*DETA
      DX(12) = DETA*DXI*SXI - DX(10)*CXI
      GO TO 50
   40 DX(10) = 0.D0
      DX(11) = 0.D0
      DX(12) = 0.D0
   50 RETURN
      END



$IBFTC ENRGY.

      SUBROUTINE ENERGY(E,EROT,X,DX)
      INTEGER PCS,CLRP,PLOT,RUN
      DOUBLE PRECISION E,EROT,X,DX,
     1                 EROC,RM,V,ETRC,EPLC,EPTC,XXC1,XXC2,XC,
     2                 RE,V2,SXI,R,T,KT,Q,R2,CXI,EP,SIG, ETRAN,
     3                 XI,ETA,PSI,DXI,DETA,DPSI,EPOL,EPOT,CD12
      DOUBLE PRECISION VLJ
      LOGICAL CUTPLT
      DIMENSION X(6),DX(6)
      COMMON/PCSC/ PCS
      COMMON/DDC/ RE,V2,SXI,R,T,KT,ID1,ID2,Q(12),CLRP,RUN(3),
     * PLOT(5),CUTPLT
      COMMON /JACKEN/ EROC,RM,V,ETRC,EPLC,EPTC,XXC1,XXC2,XC,JCD1,EP,SIG
     1 ,ETRAN,EPOT,EPOL
      ETA = X(4)
      XI = X(5)
      PSI = X(6)
      DETA = DX(4)
      DXI = DX(5)
      DPSI = DX(6)
      SXI = DSIN(XI)
      CXI = DCOS(XI)
      R = DSQRT(X(1)*X(1) + X(2)*X(2) + X(3)*X(3))
      R2 = R*R
      EROT = EROC*((DXI*DXI + (DETA*SXI)**2) + RM*(DPSI+DETA*CXI)**2)
      V2 = (DX(1)*DX(1) + DX(2)*DX(2) + DX(3)*DX(3))
      GO TO (1,2), PCS
    1 EPOT = (EPTC/(R*R2))*(X(1)*SXI*DSIN(ETA)-X(2)*SXI*DCOS(ETA)+
     1 X(3)*CXI)
      GO TO (4,5), JCD1
    2 EPOT = (EPTC/(R*R2))*(X(1)*CXI - X(2)*SXI*DCOS(ETA) - X(3)*SXI*
     1 DSIN(ETA))
      GO TO (4,5), JCD1
    4 EPOL = EPLC/(R2*R2)
      GO TO 6
    5 VLJ=EP*(SIG/R)**12
      EPOL=EPLC/(R2*R2)-VLJ
```

```
    6 ETRAN=ETRC*VZ
      E=EROT+ETRAN-EPOL-EPOT
      RETURN
      END
```

```
$IBFTC RK.
      SUBROUTINE RK(NE,T,H,X,XF,F)
      DOUBLE PRECISION T,H,X,XF,A,B,C,G,Y,DTEMP
      DIMENSION X(NE),XF(NE)
      DIMENSION LA(11),G(18,12),Y(18)
      COMMON /RKC/NV,NVM1,C(11),B(12),A(66)
      DATA  LA              /0,1,3,6,10,15,21,28,36,45,55/
      CALL F(T,X,G(1,1))
      DO 30 L=1,NVM1
      DO 20 I=1,NE
      DTEMP = 0.DO
      M=LA(L)
      DO 10 K=1,L
      M=M+1
   10 DTEMP=DTEMP+A(M)*G(I,K)
   20 Y(I)=X(I)+H*DTEMP
      CALL F(T+H*C(L),Y,G(1,L+1))
   30 CONTINUE
      DO 50 I=1,NE
      DTEMP = 0.DO
      DO 40 L=1,NV
   40 DTEMP=DTEMP+B(L)*G(I,L)
   50 XF(I)=X(I)+H*DTEMP
      T=T+H
      RETURN
      END
```

```
$IBFTC SETRK.
      SUBROUTINE SETRK(NM)
      DOUBLE PRECISION A,B,C,C1,C2,C4,RN,
     1                 TOTERR,MAXER,HI,MAXHR,HMAX,HMIN,HAV,
     2                 TS1,CN,CD,AN,AD,BN,BD
      INTEGER NT1( 8),NT2( 8),NT3( 7),NT( 8),NVA( 7)
      INTEGER P
      DIMENSION CN(36),CD(36),AN(148),AD(36),BN(43),BD(7)
      COMMON /RKC/ NV,NVM1,C(11),B(12),A(66)
      COMMON /INTSET/ C1,C2,C4,RN
      COMMON /INTES/ TOTERR,MAXER,HI,MAXHR,HMAX,HMIN,HAV,NS,NSF
      DATA NVA              /2,3,4,6,7,9,12/     ,
     1 NT             /0,1,2,3,4,5,6,7/,
```

```
2           NT1  /1,2,4,7,12,18,26,37/,
3           NT2  /1,3,6,10,16,23,32,44/,
4           NT3  /1,2,5,11,26,47,83/
 DATA CN
1/1.D0,1.D0,1.D0,1.D0,1.D0,1.D0,1.D0,2.D0,1.D0,2.D0,4.D0,1.D0,2.D0,
21.D0,1.D0,1.D0,1.D0,2.D0,1.D0,1.D0,1.D0,8.D0,1.D0,5.D0,1.D0,1.D0,
31.D0,1.D0,1.D0,1.D0,1.D0,2.D0,1.D0,5.D0,5.D0,1.D0/
 DATA CD
1/1.D0,2.D0,1.D0,2.D0,2.D0,1.D0,3.D0,5.D0,1.D0,3.D0,5.D0,3.D0,3.D0,
23.D0,2.D0,2.D0,1.D0,9.D0,3.D0,2.D0,6.D0,9.D0,9.D0,6.D0,1.D0,9.D0,
36.D0,4.D0,10.D0,6.D0,2.D0,3.D0,3.D0,6.D0,6.D0,1.D0/
 DATA BN
1/1.D0,1.D0,1.D0,+.D0,1.D0,1.D0,2.D0,2.D0,1.D0,23.D0,0.D0,125.D0,
A 0.D0,-81.D0,125.D0,
211.D0,C.DC,81.D0,81.D0,-32.D0,-32.D0,11.D0,110201.D0,0.D0,0.D0,
3767936.D0,635040.D0,-59049.D0,-59049.D0,635040.D0,110201.D0,41.D0,
40.D0,0.D0,0.D0,0.D0,216.D0,272.D0,27.D0,27.D0,36.D0,180.D0,41.D0/
 DATA BD
1 /2.D0,6.D0,6.D0,192.D0,120.D0,2140320.D0,840.D0/
 DATA AN / 1.D0,1.D0,-1.D0,2.D0,1.D0,0.D0,1.D0,0.D0,0.D0,1.D0,1.D0,
1 4.D0,6.D0,1.D0,-12.D0,15.D0,6.D0,90.D0,-50.D0,8.D0,6.D0,36.D0,
2 1.D1,6.D0,0.D0,1.D0,0.D0,2.D0,1.D0,4.D0,-1.D0,-1.D0,18.D0,-3.D0,
3 -6.D0,0.D0,9.D0,-3.D0,-6.D0,4.D0,9.D0,-36.D0,63.D0,72.D0,0.D0,
4 -64.D0,2.D0,1.D0,3.D0,1.D0,0.D0,3.D0,23.D0,0.D0,21.D0,-8.D0,
5-4136.D0,0.D0,-13584.D0,5264.D0,13104.D0,105131.D0,0.D0,302016.D0,
6 -107744.D0,-28425e.D0,1701.D0,-775229.D0,0.D0,-2770950.D0,
7 1735136.D0,2547216.D0,81891.D0,328536.D0,23569.D0,0.D0,-122304.D0
8 ,-20384.D0,695520.D0,-99873.D0,-466560.D0,241920.D0,1.D0,1.D0,
8 3.D0,1.D0,0.D0,3.D0,29.D0,0.D0,33.D0,-12.D0,33.D0,0.D0,0.D0,4.D0,
9 125.DC,-21.D0,0.D0,0.D0,76.D0,125.D0,-162.D0,-30.D0,0.D0,0.D0,
A -32.D0,125.D0,0.D0,99.D0,1175.D0,0.D0,0.D0,-3456.D0,-6250.D0,
B 8424.D0,242.D0,-27.D0,293.D0,0.D0,0.D0,-852.D0,-1375.D0,1836.D0,
C -118.D0,162.D0,324.D0,1303.D0,0.D0,0.D0,-4260.D0,-6875.D0,999.D1,
D 1030.D0,C.D0,0.D0,162.D0,-8595.D0,0.D0,0.D0,30720.D0,48750.D0,
E -66096.D0,378.D0,-729.D0,-1944.D0,-1296.D0,3240.D0/
 DATA AD
1 /1.D0,2.D0,1.D0,2.D0,2.D0,1.D0,3.D0,25.D0,4.D0,81.D0,75.D0,3.D0,
2 3.D0,12.D0,16.D0,8.D0,44.D0,9.D0,12.D0,8.D0,216.D0,729.D0,
3151632.D0,1375920.D0,251888.D0,9.D0,24.D0,16.D0,5.D2,972.D0,36.D0,
4 243.D0,324.D0,324.D0,162.D1,4428.D0/
 NN = NM
 IF(NN.EQ.0) RETURN
 IF(MAXHR.LE.0.D0) MAXHR = 4.D0
 P=NN+1
 TS1 = 2.D0**NN
 C1 = 1.D0/(TS1-1.D0)
 C2=TS1*C1
 C4 = 1.D0/(MAXHR**P)
 RN = 1.D0/DBLE(FLOAT(P))
 N=NT(NN)
 NV=NVA(N)
 NVM1=NV-1
 I1=NT2(N)
 I2=NT2(N+1)-1
 K=0
 DO 10 I=I1,I2
 K=K+1
```

```
   10 B(K) = BN(I)/BD(N)
      I1=NTI(N)
      I2=NTI(N+1)-1
      K=0
      L=0
      DO 20  I=I1,I2
      K=K+1
      C(K) = CN(I)/CD(I)
      DO 20 J=1,K
      M=NT3(N)+L
      L=L+1
   20 A(L) = AN(M)/AD(I)
      RETURN
      END
```

```
$IBFTC PTUP.

      SUBROUTINE PTUP(EP,DEP,E,DE)
      DOUBLE PRECISION EP,DEP,E,DE,
     1                 QE,DQE,QEP,DQEP,P,T,S,PD,TD,SD,
     2                 PP,TP,SP,PPD,TPD,SPD,
     3                 SNP,CP,ST,CT,CTP,STP,SN,
     4                 SPP,CPP,SSP,CSP
      DOUBLE PRECISION SIN,COS,ARCCOS,ATAN2,X,Y
      DIMENSION QE(6),DQE(6),QEP(6),DQEP(6),E(6),DE(6),EP(6),DEP(6)
      EQUIVALENCE (QE(4),P),(QE(5),T),(QE(6),S),(DQE(4),PD),(DQE(5),TD),
     1(DQE(6),SD),(QEP(4),PP),(QEP(5),TP),(QEP(6),SP),(DQEP(4),PPD),
     2(DQEP(5),TPD),(DQEP(6),SPD)
      SIN(X) = DSIN(X)
      COS(X) = DCOS(X)
      ARCCOS(X) = DARCOS(X)
      ATAN2(X,Y) = DATAN2(X,Y)
      DO 18 I=4,6
      QEP(I)=EP(I)
   18 DQEP(I)=DEP(I)
      SPP=SIN(PP)
      CPP=COS(PP)
      STP=SIN(TP)
      CTP=COS(TP)
      SSP=SIN(SP)
      CSP=COS(SP)
      CT=-STP*SPP
      T=ARCCOS(CT)
      ST=SIN(T)
      SN = DSIGN(1.D0,ST)
      P=ATAN2(CTP*SN,STP*CPP*SN)
      SNP=SIN(P)
      CP=COS(P)
      S=ATAN2((-CSP*CPP+CTP*SPP*SSP)*SN,(SSP*CPP+CTP*SPP*CSP)*SN)
      SD=((SPD*CTP+PPD)*SNP-(TPD*SPP-SPD*STP*CPP)*CP)/ST
      TD=(TPD*SPP-SPD*STP*CPP+SD*ST*CP)/SNP
      PD=-(TPD*CPP+SPD*STP*SPP+SD*CT)
      DO 22 I=4,6
      E (I)=QE(I)
```

```
   22 DE (I)=DQE(I)
      DO 30 I=1,3
      E(I)=EP(I)
   30 DE(I)=DEP(I)
      RETURN
      END




$IBFTC UPTP.

      SUBROUTINE UPTP(E,DE,EP,DEP)
      DOUBLE PRECISION E,DE,EP,DEP,
     1                 QE,DQE,QEP,DQEP,P,T,S,PD,TD,SD,
     2                 PP,TP,SP,PPD,TPD,SPD,
     3                 SNP,CP,ST,CT,SS,CS,CTP,STP,SN,
     4                 SPP,CPP
      DOUBLE PRECISION SIN,COS,ARCCOS,ATAN2,X,Y
      DIMENSION QE(6),DQE(6),QEP(6),DQEP(6),E(6),DE(6),EP(6),DEP(6)
      EQUIVALENCE (QE(4),P),(QE(5),T),(QE(6),S),(DQE(4),PD),(DQE(5),TD),
     1(DQE(6),SD),(QEP(4),PP),(QEP(5),TP),(QEP(6),SP),(DQEP(4),PPD),
     2(DQEP(5),TPD),(DQEP(6),SPD)
      SIN(X) = DSIN(X)
      COS(X) = DCOS(X)
      ARCCOS(X) = DARCOS(X)
      ATAN2(X,Y) = DATAN2(X,Y)
      DO 18 I=4,6
      QE(I)=E(I)
   18 DQE(I)=DE(I)
      SNP=SIN(P)
      CP=COS(P)
      ST=SIN(T)
      CT=COS(T)
      SS=SIN(S)
      CS=COS(S)
      CTP=ST*SNP
      TP=ARCCOS(CTP)
      STP=SIN(TP)
      SN = DSIGN(1.D0,STP)
      PP=ATAN2(-CT*SN,ST*CP*SN)
      SPP=SIN(PP)
      CPP=COS(PP)
      SP=ATAN2((CS*CP-CT*SNP*SS)*SN,-(SS*CP+CT*SNP*CS)*SN)
      SPD=((SD*ST*CP-TD*SNP)*CPP-(SD*CT+PD)*SPP)/STP
      TPD=(TD*SNP-SD*ST*CP+SPD*STP*CPP)/SPP
      PPD=TD*CP+SD*ST*SNP-SPD*CTP
      DO 22 I=4,6
      EP(I)=QEP(I)
   22 DEP(I)=DQEP(I)
      DO 30 I=1,3
      EP(I)=E(I)
   30 DEP(I)=DE(I)
      RETURN
      END
```

```
$IBFTC SCT.

      SUBROUTINE SCT(S,DS,C,DC,MODE)
      DIMENSION S(1),DS(1),C(1),DC(1)
      DOUBLE PRECISION S,DS,C,DC,R,DR,ST,CT,DTR,SP,DPR,
     1                 X,DX,Y,DY,Z,DZ,R2,CP
      GO TO (10,20), MODE
   10 R = S(1)
      DR = DS(1)
      ST = DSIN(S(2))
      CT = DCOS(S(2))
      DTR = R*DS(2)
      SP = DSIN(S(3))
      CP = DCOS(S(3))
      DPR = R*DS(3)
      C(1) = R*ST*CP
      C(2) = R*ST*SP
      C(3) = R*CT
      DC(1) = DR*ST*CP + DTR*CT*CP - DPR*ST*SP
      DC(2) = DR*ST*SP + DTR*CT*SP + DPR*ST*CP
      DC(3) = DR*CT - DTR*ST
      GO TO 30
   20 X = C(1)
      DX = DC(1)
      Y = C(2)
      DY = DC(2)
      Z = C(3)
      DZ = DC(3)
      R2 = X*X + Y*Y + Z*Z
      S(1) = DSQRT(R2)
      S(2) = DARCOS(Z/S(1))
      S(3) = DATAN2(Y,X)
      DS(1) = (X*DX + Y*DY + Z*DZ)/S(1)
      DS(2) = (Z*DS(1)-S(1)*DZ)/(S(1)*DSQRT(R2-Z*Z))
      DS(3) = (X*DY-Y*DX)/(R2-Z*Z)
   30 RETURN
      END
```

```
$IBFTC DARC.

      DOUBLE PRECISION FUNCTION DARCOS(X)
      DOUBLE PRECISION X,Y,PI,HALFPI ,DATAN,DSQRT
      DATA PI/3.14159265358979320/,HALFPI/1.57079632679489700/
      IF(DABS(X).GT.1.D0) CALL ARERR(36HDARCOS CALLED WITH ARGUMENT .GT.
     1 1.$)
      IF(X.EQ.0.D0) GO TO 10
      Y = DATAN(DSQRT(1.D0-X*X)/X)
      IF(Y.LT.0.D0) Y = Y + PI
      DARCOS = Y
      GO TO 20
   10 DARCOS = HALFPI
   20 RETURN
      END
```

```
$IBFTC DDS..

      SUBROUTINE DDS
      LOGICAL NOPLOT,SKTHET,CUTPLT
      INTEGER PCS,CLRP,PLOT
      DOUBLE PRECISION Q,V2,K,RE,T,KT,SXX
      COMMON/DDC/ RE,V2,SXX,R,T,KT,ID1,ID2,Q(12),CLRP,
     1 RUN(3),PLOT(5),CUTPLT
      COMMON /PCSC/ PCS
      REAL RA(1000),VA(1000),REA(1000),XXA(1000),YYA(1000),ZZA(1000),
     1  THETAA(1000),XA(1000),YA(1000),ZA(1000),GAMMAA(1000),PHIA(1000),
     2  QA(1000,12),IV(2),KTC(2),NEOP
      EQUIVALENCE (QA(1,1),RA(1)), (QA(1,2),VA(1)), (QA(1,3),REA(1)), (Q
     1A(1,4),XXA(1)), (QA(1,5),YYA(1)), (QA(1,6),ZZA(1)), (QA(1,7),THETA
     2A(1)), (QA(1,8),XA(1)), (QA(1,9),YA(1)), (QA(1,10),ZA(1)), (QA(1,1
     31),GAMMAA(1)), (QA(1,12),PHIA(1))
      DATA NPA/12/,LREC/1000/,RTD/57.29578/,EOP/1./,NEOP/0./,BEG/4H$R1B/
     1,END/4H$R1E/,IV/12H            /,KTC/12H            /,LRECM1/999 /
      DATA NPLTS/5/
      NOPLOT=.TRUE.
      DO 1 II=1,NPLTS
1     IF (PLOT(II).NE.0) GO TO 2
      RETURN
2     NOPLOT=.FALSE.
      SKTHET=.TRUE.
      IF (PLOT(2).EQ.0) SKTHET=.FALSE.
      REWIND 1
      NREC=0
      IP=1
      WRITE (6,75)
      RETURN
      ENTRY DDR
      IF (NOPLOT) RETURN
      IF ((CUTPLT).AND.(IP.EQ.LRECM1)) RETURN
      IP=IP+1
      RA(IP) = K
      IF (PLOT(1).EQ.0) GO TO 3
      VA(IP)=SQRT(SNGL(V2))
      REA(IP)=SNGL(RE)
3     IF (PLOT(2).EQ.0) GO TO 4
      XXA(IP)=SNGL(Q(1))
      YYA(IP)=SNGL(Q(2))
      ZZA(IP)=SNGL(Q(3))
      THETAA(IP)=ARCOS(ZZA(IP)/SNGL(R))*RTD
4     IF (PLOT(3).EQ.0) GO TO 8
      GO TO (6,5),PCS
C     RETURN TO ORIGINAL VERSION
5     CXI=-SIN(SNGL(Q(5)))*SIN(SNGL(Q(4)))
      SXI=SQRT(1.-CXI*CXI)
      ETA=ATAN2(COS(SNGL(Q(5))),SIN(SNGL(Q(5)))*COS(SNGL(Q(4))))
      GO TO 7
6     CXI=COS(SNGL(Q(5)))
      SXI=SIN(SNGL(Q(5)))
      ETA=SNGL(Q(4))
7     CETA=COS(ETA)
      SETA=SIN(ETA)
      XA(IP)=SXI*SETA
      YA(IP)=-SXI*CETA
      ZA(IP)=CXI
```

```
      GAMMAA(IP) = ARCOS((SNGL(Q(1))*XA(IP) + SNGL(Q(2))*YA(IP)
     1 + SNGL(Q(3))*ZA(IP))/(SNGL(R))) * RTD
8     IF (PLOT(4).EQ.0) GO TO 9
      IF (SKIHET) GO TO 9
      THETAA(IP)=ARCOS(SNGL(Q(3))/SNGL(R))*RTD
9     IF (PLOT(5).EQ.0) GO TO 10
      PHIA(IP)=ATAN2(SNGL(Q(2)),SNGL(Q(1)))*RTD
      IF (PHIA(IP).LT.0.) PHIA(IP)=PHIA(IP)+360.
10    IF (IP.LT.LREC) RETURN
      IF (NREC.GT.0) GO TO 12
      VI=VA(2)
      RI=ABS(RA(2))
      XXB=XXA(2)
      YYB=YYA(2)
      ZZB=ZZA(2)
      XB=XA(2)
      YB=YA(2)
      ZB=ZA(2)
      DO 11 I1=1,NPA
11    QA(1,I1)=QA(2,I1)
12    NREC=NREC+1
      WRITE (1) QA
      DO 13 I1=1,NPA
13    QA(1,I1)=QA(LREC,I1)
      IP=1
      RETURN
      ENTRY DDP
      IF (NOPLOT) RETURN
      IF (NREC.GT.0) GO TO 15
      RI=ABS(RA(2))
      VI=VA(2)
      XXB=XXA(2)
      YYB=YYA(2)
      ZZB=ZZA(2)
      XB=XA(2)
      YB=YA(2)
      ZB=ZA(2)
      DO 14 I1=1,NPA
14    QA(1,I1)=QA(2,I1)
15    IPL=IP-1
      IF (IPL.GT.1) GO TO 16
      IF (IPL.EQ.1) IPL=LREC
      IF (IPL.EQ.0) IPL=LREC-1
      GO TO 17
16    NREC=NREC+1
17    MT1=1
      IF (NREC.EQ.1) GO TO 18
      MT1=2
      WRITE (1) QA
18    XXE=XXA(IPL)
      YYE=YYA(IPL)
      ZZE=ZZA(IPL)
      XE=XA(IPL)
      YE=YA(IPL)
      ZE=ZA(IPL)
      IF (PLOT(1).EQ.0) GO TO 31
      CALL HEAD
      CALL LRSIZE (0.0,10.,5.0,10.)
      CALL LRMRGN (1.0,0.5,0.5,0.5)
```

```
      CALL LRLEGN (36HION-MOLECULE SEPARATION IN ANGSTROMS,36,0,3.84,5.1
     125,NEQP)
      CALL LRLEGN (35HVELOCITY SCALED TO INITIAL VALUE OF,35,1,0.100,5.3
     15,NEQP)
      CALL LRCNVT (VI*1.E6,3,IV,4,I1,4)
      CALL LRLEGN (IV,I1,1,0.100,8.15,NEQP)
      CALL LRLEGN (7H CM/SEC,7,1,0.100,9.05,NEQP)
      CALL LRANGE(RI,0.0,0.0,0.0)
      CALL LRGRID (2,1,RI/10.,6.)
      GO TO (19,21),MTI
19    DO 20 II=1,IPL
20    VA(II)=VA(II)/VI
      CALL LRCURV (RA,VA,IPL,2,SYMBOL,NEQP)
      GO TO 24
21    REWIND 1
      NP=LREC
      DO 23 IREC=1,NREC
      READ (1) QA
      IF (IREC.EQ.NREC) NP=IPL
      DO 22 II=1,NP
22    VA(II)=VA(II)/VI
23    CALL LRCURV (RA,VA,NP,2,SYMBOL,NEQP)
24    CALL LRSIZE (0.0,10.,0.0,5.0)
      CALL LRLEGN (36HION-MOLECULE SEPARATION IN ANGSTROMS,36,0,3.84,0.1
     125,NEQP)
      CALL LRLEGN (32HROTATIONAL ENERGY SCALED TO KT =,32,1,0.100,0.6,NE
     1QP)
      CALL LRCNVT (KT,3,KTC,4,I1,4)
      CALL LRLEGN (KTC,I1,1,0.100,3.18,NEQP)
      CALL LRLEGN (2HEV,2,1,0.100,4.12,NEQP)
      RESCAL=KT*1.6E-4
      GO TO (25,27),MTI
25    DO 26 II=1,IPL
26    REA(II)=REA(II)/RESCAL
      CALL LRCURV (RA,REA,IPL,2,SYMBOL,NEQP)
      GO TO 30
27    REWIND 1
      NP=LREC
      DO 29 IREC=1,NREC
      READ (1) QA
      IF (IREC.EQ.NREC) NP=IPL
      DO 28 II=1,NP
28    REA(II)=REA(II)/RESCAL
29    CALL LRCURV (RA,REA,NP,2,SYMBOL,NEQP)
30    CALL LRLEGN (1H ,1,0,.5,.5,EOP)
31    IF (PLOT(2).EQ.0) GO TO 48
      CALL HEAD
      CALL LRMRGN (1.0,0.5,0.5,1.0)
      CALL LRSIZE (0.0,5.0,5.0,10.)
      CALL LRANGE (-RI,RI,-RI,RI)
      CALL LRGRID (2,2,RI,RI)
      CALL LRLEGN (6HX AXIS,6,0,2.25,5.125,NEQP)
      CALL LRLEGN (6HY AXIS,6,1,0.100,7.25,NEQP)
      GO TO (32,33),MTI
32    CALL LRCURV (XXA,YYA,IPL,2,SYMBOL,NEQP)
      GO TO 35
33    REWIND 1
      NP=LREC
      DO 34 IREC=1,NREC
```

```
      READ (1) QA
      IF (IREC.EQ.NREC) NP=IPL
34    CALL LRCURV (XXA,YYA,NP,2,SYMBOL,NEOP)
35    CALL LRLABL (BEG,4,0,XXB,YYB,NEOP)
      CALL LRLABL (END,4,0,XXE,YYE,NEOP)
      CALL LRSIZE (5.0,10.,5.0,10.)
      CALL LRLEGN (6HY AXIS,6,0,7.25,5.125,NEOP)
      CALL LRLEGN (6HZ AXIS,6,1,5.100,7.25,NEOP)
      GO TO (36,37),MTL
36    CALL LRCURV (YYA,ZZA,IPL,2,SYMBOL,NEOP)
      GO TO 39
37    REWIND 1
      NP=LREC
      DO 38 IREC=1,NREC
      READ (1) QA
      IF (IREC.EQ.NREC) NP=IPL
38    CALL LRCURV (YYA,ZZA,NP,2,SYMBOL,NEOP)
39    CALL LRLABL (BEG,4,0,YYB,ZZB,NEOP)
      CALL LRLABL (END,4,0,YYE,ZZE,NEOP)
      CALL LRSIZE (0.0,5.0,0.0,5.0)
      CALL LRLEGN (6HZ AXIS,6,0,2.25,0.125,NEOP)
      CALL LRLEGN (6HX AXIS,6,1,0.100,2.25,NEOP)
      GO TO (40,41),MTL
40    CALL LRCURV (ZZA,XXA,IPL,2,SYMBOL,NEOP)
      GO TO 43
41    REWIND 1
      NP=LREC
      DO 42 IREC=1,NREC
      READ (1) QA
      IF (IREC.EQ.NREC) NP=IPL
42    CALL LRCURV (ZZA,XXA,NP,2,SYMBOL,NEOP)
43    CALL LRLABL (BEG,4,0,ZZB,XXB,NEOP)
      CALL LRLABL (END,4,0,ZZE,XXE,NEOP)
      CALL LRSIZE (5.0,10.,0.0,5.0)
      CALL LRLEGN (36HION MOLECULE SEPARATION IN ANGSTROMS,36,0,6.34,0.1
     125,NEOP)
      CALL LRLEGN (26H    PULAR ANGLE IN DEGREES,26,1,5.100,1.50,NEOP)
      CALL LRANGE(RI,0.0,0.0,180.)
      CALL LRGRID (2,2,RI/5.,45.)
      GO TO (44,45),MTL
44    CALL LRCURV (RA,THETAA,IPL,2,SYMBOL,NECP)
      GO TO 47
45    REWIND 1
      NP=LREC
      DO 46 IREC=1,NREC
      READ (1) QA
      IF (IREC.EQ.NREC) NP=IPL
46    CALL LRCURV (RA,THETAA,NP,2,SYMBOL,NEOP)
47    CALL LRLEGN (1H ,1,0,5.5,.5,ECP)
48    IF (PLOT(3).EQ.0) GO TO 65
      CALL HEAD
      CALL LRMRGN (1.0,0.5,0.5,1.0)
      CALL LRSIZE (0.0,5.0,5.0,10.)
      CALL LRANGE (-1.0,1.0,-1.0,1.0)
      CALL LRGRID (2,2,1.0,1.0)
      CALL LRLEGN (6HX AXIS,6,0,2.25,5.125,NEOP)
      CALL LRLEGN (6HY AXIS,6,1,0.100,7.25,NEOP)
      GO TO (49,50),MTL
```

```
49    CALL LRCURV (XA,YA,IPL,2,SYMBOL,NEOP)
      GO TO 52
50    REWIND 1
      NP=LREC
      DO 51 IREC=1,NREC
      READ (1) QA
      IF (IREC.EQ.NREC) NP=IPL
51    CALL LRCURV (XA,YA,NP,2,SYMBOL,NEOP)
52    CALL LRLABL (BEG,4,0,XB,YB,NEOP)
      CALL LRLABL (END,4,0,XE,YE,NEOP)
      CALL LRSIZE (5.0,10.,5.0,10.)
      CALL LRLEGN (6HY AXIS,6,0,7.25,5.125,NEOP)
      CALL LRLEGN (6HZ AXIS,6,1,5.100,7.25,NEOP)
      GO TO (53,54),MTL
53    CALL LRCURV (YA,ZA,IPL,2,SYMBOL,NEOP)
      GO TO 56
54    REWIND 1
      NP=LREC
      DO 55 IREC=1,NREC
      READ (1) QA
      IF (IREC.EQ.NREC) NP=IPL
55    CALL LRCURV (YA,ZA,NP,2,SYMBOL,NEOP)
56    CALL LRLABL (BEG,4,0,YB,ZB,NEOP)
      CALL LRLABL (END,4,0,YE,ZE,NEOP)
      CALL LRSIZE (0.0,5.0,0.0,5.0)
      CALL LRLEGN (6HZ AXIS,6,0,2.25,0.125,NEOP)
      CALL LRLEGN (6HX AXIS,6,1,0.100,2.25,NEOP)
      GO TO (57,58),MTL
57    CALL LRCURV (ZA,XA,IPL,2,SYMBOL,NEOP)
      GO TO 60
58    REWIND 1
      NP=LREC
      DO 59 IREC=1,NREC
      READ (1) QA
      IF (IREC.EQ.NREC) NP=IPL
59    CALL LRCURV (ZA,XA,NP,2,SYMBOL,NEOP)
60    CALL LRLABL (BEG,4,0,ZB,XB,NEOP)
      CALL LRLABL (END,4,0,ZE,XE,NEOP)
      CALL LRSIZE (5.0,10.,0.0,5.0)
      CALL LRLEGN (36HION-MOLECULE SEPARATION IN ANGSTROMS,36,0,6.34,0.1
     125,NEOP)
      CALL LRLEGN (28HORIENTATION ANGLE IN DEGREES,28,1,5.100,1.50,NEOP)
      CALL LRANGE(R1,0.0,0.0,180.)
      CALL LRGRID (2,2,R1/5.,45.)
      GO TO (61,62),MTL
61    CALL LRCURV (RA,GAMMAA,IPL,2,SYMBOL,NEOP)
      GO TO 64
62    REWIND 1
      NP=LREC
      DO 63 IREC=1,NREC
      READ (1) QA
      IF (IREC.EQ.NREC) NP=IPL
63    CALL LRCURV (RA,GAMMAA,NP,2,SYMBOL,NEOP)
64    CALL LRLEGN (1H ,1,0,5.5,.5,EOP)
65    IF (PLOT(4).EQ.0) GO TO 70
      CALL HEAD
      CALL LRSIZE (0.,10.,0.,10.)
      CALL LRMRGN (1.0,0.5,1.0,0.5)
      CALL LRCHSZ (2)
```

```
      CALL LRLEGN (36HION-MOLECULE SEPARATION IN ANGSTROMS,36,0,3.14,0.2
     100,NEOP)
      CALL LRLEGN (22HPOLAR ANGLE IN DEGREES,22,1,0.12,3.80,NEOP)
      CALL LRANGE(R1,0.0,0.0,180.)
      CALL LRGRID (2,2,R1/5.,45.)
      GO TO (66,67),MTL
66    CALL LRCURV (RA,THETAA,IPL,2,SYMBOL,NECP)
      GO TO 69
67    REWIND 1
      NP=LREC
      DO 68 IREC=1,NREC
      READ (1) QA
      IF (IREC.EQ.NREC) NP=IPL
68    CALL LRCURV (RA,THETAA,NP,2,SYMBOL,NEOP)
69    CALL LRLEGN (1H ,1,0,.5,.2,EOP)
   70 IF(PLOT(5).EQ.0) GO TO 100
      CALL HEAD
      CALL LRSIZE (0.,10.,0.,10.)
      CALL LRMRGN (1.0,0.5,1.0,0.5)
      CALL LRCHSZ (2)
      CALL LRLEGN (36HION-MOLECULE SEPARATION IN ANGSTROMS,36,0,3.14,0.2
     100,NEOP)
      CALL LRLEGN (26HAZIMUTHAL ANGLE IN DEGREES,25,1,0.12,3.40,NEOP)
      CALL LRANGE(R1,0.0,0.0,360.)
      CALL LRGRID (2,2,R1/5.,45.)
      GO TO (71,72),MTL
71    CALL LRCURV (RA,PHIA,IPL,2,SYMBOL,NEOP)
      GO TO 74
72    REWIND 1
      NP=LREC
      DO 73 IREC=1,NREC
      READ (1) QA
      IF (IREC.EQ.NREC) NP=IPL
73    CALL LRCURV (RA,PHIA,NP,2,SYMBOL,NEOP)
74    CALL LRLEGN (1H ,1,0,.5,.2,EOP)
  100 CONTINUE
      RETURN
75    FORMAT (24H+THIS CASE TO BE PLOTTED)
      END
```

```
$IBFTC HEAD.

      SUBROUTINE HEAD
      COMMON /DDC/ RE,V2,SXI,R,T,KT,ID1,ID2,Q(12),CLRP,RUN(3),PLOT(5)
     1 ,OUTPLT
      DOUBLE PRECISION RE,V2,SXI,R,T,KT,Q
      REAL CASE(2),NEOP,TYPE(8)
      DATA NEOP / 0. / ,
     1  TYPE/ 48HCAPTURE      REPULSION    OUT OF TIME TOO SMALL     /
      INTEGER CLRP,PLOT
      CALL LRSIZE(0.0,10.,0.0,10.)
      CALL LRCNVT(ID1,1,CASE(1),1,6,0)
      CALL LRCNVT(ID2,1,CASE(2),1,6,0)
      CALL LRCHSZ(2)
```

```
      INDEX = 2*(CLRP-1) + 1
      CALL LRLEGN(3HRUN,3,0,1.50,9.875,NEOP)
      CALL LRLEGN(RUN, 18,0,2.00,9.875,NEOP)
      CALL LRLEGN(4HCASE,4,0,4.80,9.875,NEOP)
      CALL LRLEGN(CASE,12,0,5.425,9.875,NEOP)
      CALL LRLEGN(TYPE(INDEX),12,0,7.125,9.875,NEOP)
      CALL LRCHSZ(1)
      RETURN
      END




$IBFTC PLTCM.

C
C         SPECIAL MOVIE DECK, SIMPLIFIED
      SUBROUTINE DDS
      LOGICAL ON
      LOGICAL CUTPLT
      LOGICAL DIPOLE
      REAL XA(1500),YA(1500),XXA(1500),YYA(1500),TA(1500),NEOP,
     1  DUMX(2),DUMY(2),ZR(1500)
      DOUBLE PRECISION ERUC,RM,V,ETRC,EPLC,MUE,XCA,XCB,XC,CD1
      DOUBLE PRECISION Q,V2,R,RE,T,KT,SXX
      INTEGER PCS,CLRP,PLOT
C         COMMON FOR POTENTIAL SPHERE RADIUS
      COMMON/CIRCLE/ RADIUS,RVIEW,ON
C         COMMON FOR COORD SYSTEM SWITCH
      COMMON/PCSC/ PCS
C         COMMON FOR ALL MAJOR VARIABLES
      COMMON/DDC/ RE,V2,SXX,R,T,KT,ID1,ID2,Q(12),CLRP,RUN(3),PLOT(5)
     1  ,CUTPLT
      COMMON/JACKEN/ ERUC,RM,V,ETRC,EPLC,MUE,XCA,XCB,XC,JCD,CD1
      DIMENSION XAXISX(2),YAXISX(2),YAXISY(2),XAXISY(2)
      DATA XAXISX/-10.,+10./, YAXISX/-10.,-10./
      DATA XAXISY/-10.,-10./, YAXISY/-10.,+10./
      DATA AX,BX,CX,DX,EX /3*2.0,5.0,8.0/
      DATA AY,BY,CY,DY,EY /2.0,5.0,8.0,2.0,2.0/
      DATA LRECM1/1500/, EOP/1./, NEOP/0./
C
C         DUMMY ENTRY TO START PLOTTING
C         MAKES MOVIES IN X--Y PLANE NOW
C
      IP = 0
      ON = .TRUE.
      DIPOLE = .TRUE.
C         SAVE COLLISION RADIUS
      RADMAX=RADIUS
      IF(MUE.EQ.0.D0) DIPOLE = .FALSE.
      WRITE(6,101)
  101 FORMAT(24H+MOVIE MADE IN CM SYSTEM )
      RETURN
C
C         ENTRY TO RECORD VARIABLES
```

```
C
      ENTRY DDR
      IF(IP.GE.LRECM1) RETURN
      IP = IP + 1
C
C        STORE SCALED TIME
C        CLOCK PERIOD IS 10**(-13) SEC
      TA(IP) = AMOD(SNGL(T),10.)*.628319
C
C        STORE ION POSITION
C
C        CENTER OF MASS COORDS
      X = Q(1)
      Y = Q(2)
      PHI = ATAN2(Y,X)
      XXA(IP) = 0.5*SNGL(R)*COS(PHI)
      YYA(IP) = 0.5*SNGL(R)*SIN(PHI)
C
      Z = 0.5*Q(3)
      IF(ABS(Z).GT.10.) GO TO 2
C        SPECIAL RADIUS VARIATION
      RADIUS = RADMAX*(0.5+Z/40.)
      GO TO 3
    2 RADIUS = 0.0
    3 ZR(IP) = RADIUS
C
C        STORE DIPOLE POSITION
      IF(.NOT.DIPOLE) RETURN
      GO TO (6,5), PCS
    5 CXI = -SIN(SNGL(Q(5)))*SIN(SNGL(Q(4)))
      SXI = SQRT(1.0-CXI**2)
      ETA = ATAN2(COS(SNGL(Q(5))),SIN(SNGL(Q(5)))*COS(SNGL(Q(4))))
      GO TO 7
    6 CXI = COS(SNGL(Q(5)))
      SXI = SIN(SNGL(Q(5)))
      ETA = Q(4)
    7 CETA = COS(ETA)
      SETA = SIN(ETA)
C        NOTE SCALING OF DIPOLE LENGTH
C        .57735 IS 1.0/SQRT(3.)
      SCALER=.57735*RADIUS
C        NOTE PHASE SHIFT BECAUSE ETA AHEAD BY PI/2
      XA(IP) = SXI*SETA*SCALER - XXA(IP)
      YA(IP) = -SXI*CETA*SCALER - YYA(IP)
      RETURN
C
C        ENTRY TO MAKE MOVIE
C
      ENTRY DDP
C        CLOSE IN VIEW
      RVIEW = 10.
      CALL LRMON
      CALL LREON
      CALL LRSIZE(0.0,10.,0.0,10.)
      CALL LRMRGN(.75,.75,.75,.75)
      CALL BLANK
      CALL TEST
      CALL BLANK
      CALL TEST
```

```
      CALL LRMRGN(2.350,2.0,2.233,2.0)
      CALL HEADM
      CALL LRCHSZ(2)
      CALL LRANGE(-RVIEW,RVIEW,-RVIEW,RVIEW)
      CALL LRGRID(1,1.0,,0.)
C
C        BEGIN MAIN MOVIE LOOP
C
      DO 80 I=1,IP
      ISAVE = I
C
C        MAKE SURE WE HIT OUTPUT
      CALL TIMLFT(TLEFT)
      IF(TLEFT.LT.900.) RETURN
C
C        LABEL THE PERIMETER (IN ANGSTROMS)
C
      CALL LRLEGN(3H-10.3,0,AX,AY,NEOP)
      CALL LRLEGN(1H0   ,1,0,BX,BY,NEOP)
      CALL LRLEGN(1H0   ,1,0,DX,DY,NEOP)
      CALL LRLEGN(2H10  ,2,0,CX,CY,NEOP)
      CALL LRLEGN(2H10  ,2,0,EX,EY,NEOP)
C
C        DRAW THE AXES
C
      CALL LRCURV(XAXISX,YAXISX,2,2,SYMBOL,NEOP)
      CALL LRCURV(XAXISY,YAXISY,2,2,SYMBOL,NEOP)
C
C
C
C        PLOT ION ONLY IF NEAR ENOUGH
C
      XTEST = XXA(I)
      YTEST = YYA(I)
      IF(ABS(XTEST).GE.RVIEW .OR. ABS(YTEST).GE.RVIEW) GO TO 78
      CALL LRCURV(XTEST,YTEST,1,3,1H+,NEOP)
C        GET RADIUS FOR DOTS
      RADIUS = ZR(I)
      IF(RADIUS.EQ.0.) GO TO 78
      CALL DOTS(XTEST,YTEST)
C
   78 CONTINUE
C
C        PLOT THE CLOCK
      XCLOCK = 0.5*SIN(TA(I)) + 8.
      YCLOCK = 0.5*COS(TA(I)) + 8.
      CALL LRLEGN(1H*,1,0,8.,8.,NEOP)
      CALL LRLEGN(7H$GNG$GF,7,0,XCLOCK,YCLOCK,NEOP)
C
C
C        PLOT THE DIPOLE
      IF(DIPOLE) GO TO 20
C
C        NO DIPOLE--MUE=0
C
C
```

```
C        PLOT CENTER
      CALL LRCURV(-XTEST,-YTEST,1,3,1HO,NEOP)
      CALL DOTS(-XTEST,-YTEST)
      GO TO 75
C
C
C        NEW COMPLICATIONS
C        XTEST,YTEST ARE COORDS OF NEG END
C        X,Y ARE COORDS OF POS END
C
   20 CONTINUE
      XTEST = XA(I)
      YTEST = YA(I)
      X = -XTEST - 2.0*XXA(I)
      Y = -YTEST - 2.0*YYA(I)
      IBOTH = 0
      IF(ABS(XTEST).GE.RVIEW .OR. ABS(YTEST).GE.RVIEW) GO TO 31
      IBOTH = IBOTH + 1
C        NEG END ON
      CALL LRCURV(XTEST,YTEST,1,3,1H-,NEOP)
   31 IF(ABS(X).GE.RVIEW .OR. ABS(Y).GE.RVIEW) GO TO 32
      IBOTH = IBOTH + 1
C        POS END ON
      CALL LRCURV(X,Y,1,3,1H+,NEOP)
      CALL LRLABL(7H$GNW$GF,7,0.,X,Y,NEOP)
   32 IF(IBOTH.LE.1) GO TO 75
      IF(RADIUS.EQ.0.) GO TO 75
      RADIUS = RADMAX - RADIUS
      IF(XTEST.LT.X) GO TO 76
      DUMX(1) = X
      DUMX(2) = XTEST
      DUMY(1) = Y
      DUMY(2) = YTEST
      GO TO 77
   76 DUMX(1) = XTEST
      DUMX(2) = X
      DUMY(1) = YTEST
      DUMY(2) = Y
   77 CONTINUE
      CALL DOTS(-XXA(I),-YYA(I))
      CALL LRCURV(DUMX,DUMY,2,2,SYMBOL,NEOP)
   75 CONTINUE
      CALL LRCURV(0.0,0.0,1,3,1H ,EOP)
   80 CONTINUE
      CALL BLANK
      CALL LREOFF
      CALL LRMOFF
      CALL LRCHSZ(1)
      RETURN
      END
```

```
$IBFTC HEADM.

      SUBROUTINE HEADM
C        HEADER FOR MOVIES
      INTEGER CLRP,PLOT
      REAL CASE(2),TYPE(8)
      DOUBLE PRECISION RE,VV,SXI,R,T,KT,Q
      COMMON/DOC/ RE,VV,SXI,R,T,KT,ID1,ID2,Q(12),CLRP,RUN(3),
     * PLOT(5),CUTPLT
      DATA TYPE/48HCAPTURE      REPULSION    OUT OF TIME TOO SMALL    /
      CALL LRGHSZ(4)
      CALL LRCNVT(ID1,1,CASE(1),1,6,0)
      CALL LRCNVT(ID2,1,CASE(2),1,6,0)
      INDEX = 2*(CLRP-1) + 1
      DO 10 I=1,20
      CALL LRLEGN(3HRUN,3,0,1.0,9.0,0.0)
      CALL LRLEGN(RUN,18,0,2.5,9.0,0.0)
      CALL LRLEGN(4HCASE,4,0,1.0,7.0,0.0)
      CALL LRLEGN(CASE,12,0,2.5,7.0,0.0)
      CALL LRLEGN(TYPE(INDEX),12,0,1.0,5.0,1.0)
   10 CONTINUE
      RETURN
      END




$IBFTC DOTS.

      SUBROUTINE DOTS(XCENTR,YCENTR)
      LOGICAL ON
      REAL PTS(5),PT(4)
      COMMON/CIRCLE/ R,RM,ON
      DATA PT/1.0,.92388,.70711,.38268/, PTS(5)/0.0/
C
      X = XCENTR
      Y = YCENTR
      IF(.NOT.ON) GO TO 20
C        TEST CORNERS
      C = X+R
      IF(C.GE.RM) GO TO 50
      C = X-R
      IF(C.LE.-RM) GO TO 50
      C = Y+R
      IF(C.GE.RM) GO TO 50
      C = Y-R
      IF(C.LE.-RM) GO TO 50
C        CORNERS ON THE PLOT
   20 CONTINUE
C        THIS IS FOR VARIABLE RADIUS
      DO 25 I=1,4
   25 PTS(I) = PT(I)*R
   30 DO 40 I=1,5
      J = 6-I
      CALL LRCURV(+PTS(J)+X,+PTS(I)+Y,1,3,1H*,NEOP)
      CALL LRCURV(+PTS(J)+X,-PTS(I)+Y,1,3,1H*,NEOP)
      CALL LRCURV(-PTS(J)+X,-PTS(I)+Y,1,3,1H*,NEOP)
```

```
      CALL LRCURV(-PTS(J)*X,+PTS(I)*Y,1,3,1H*,NEOP)
   40 CONTINUE
   50 RETURN
      END



$IBFTC BLANK.

      SUBROUTINE BLANK
      REAL X1(2),X2(2),X3(2),Y1(2),Y2(2),Y3(2),Y4(2)
      DATA X1 /-1.0,1.0/, X2 /1.0,1.0/, X3 /-1.0,-1.0/, Y1/-.766,-.766/
      DATA Y2 /-.766,.766/, Y3 /.766,.766/, Y4 /.766,-.766/
C
C        MAKE BLANK FILM
      DO 10 I=1,80
      CALL LRCURV(0.0,0.0,0.1,1HX,1.0)
   10 CONTINUE
      RETURN
C
      ENTRY TEST
C        MAKE TEST PATTERN
      CALL LRGRID(1,1,0.,0.)
      CALL LRANGE(-1.0,1.0,-1.0,1.0)
      CALL LRCURV(X1,Y1,2,2,SYM,0.0)
      CALL LRCURV(X1,Y3,2,2,SYM,0.0)
      CALL LRCURV(X1,Y2,2,2,SYM,0.0)
      CALL LRCURV(X1,Y4,2,2,SYM,0.0)
      CALL LRCURV(X3,Y2,2,2,SYM,0.0)
      CALL LRCURV(X2,Y2,2,2,SYM,1.0)
C
      RETURN
      END



$IBFTC PLTDUM

C
C        DUMMY ENTRIES WITH PURPOSES AND ARGS EXPLAINED
C        MORE INFO IN NASA TM X-1866(1969)
C
C
C*****   MAIN ROUTINE, DRAWS A CURVE
      SUBROUTINE LRCURV(X,Y,NPTS,NTYPE,SYMBOL,EOP)
C        X--        ARRAY OF ABCISSA VALUES
C        Y--        ARRAY OF ORDINATE VALUES
C        NPTS--     NUMBER OF THESE VALUES
C        NTYPE--    TYPE OF PLOT...1=POINTS,2=LINES,
C                                3=SYMBOLS,4=SPECIAL SYMBOLS
C        SYMBOL--   THE PLOTTING SYMBOL
C        EOP--      END OF PLOT SWITCH...0.0=NOT END,1.0=END, ADVANCE
C
```

```
C
C*****   PRINTS OUT LEGENDS
      ENTRY LRLEGN(CHARS,N,NORIEN,X,Y,EOP)
C         CHARS--    ARRAY OF CHARACTERS(BCD) TO BE PRINTED(6 PER WORD)
C         N--        NUMBER OF CHARACTERS...E.G. 12=CHARS DIM. BY 2
C         NORIEN--   ORIENTATION...0=HORIZONTAL,1=VERTICAL PRINTING
C         X--        X COORD OF FIRST CHARACTER, IN ABS. UNITS 0.--10.
C         Y--        Y COORD OF FIRST CHARACTER, IN ABS. UNITS 0.--10.
C         EOP--      AS ABOVE
C
C
C*****   DEFINES USER'S UNITS FOR PLOTTING
      ENTRY LRANGE(XMIN,XMAX,YMIN,YMAX)
C         XMIN--     MIN X VALUE, USER'S UNITS
C         XMAX--     MAX X VALUE, USER'S UNITS
C         YMIN--     MIN Y VALUE, USER'S UNITS
C         YMAX--     MAX Y VALUE, USER'S UNITS
C
C
C*****   LIKE LRLEGN, BUT WORKS IN USER'S UNITS
      ENTRY LRLABL(CHARS,N,NORIEN,X,Y,EOP)
C        ARGS SAME AS FOR LRLEGN, EXCEPT X AND Y NOW IN USER'S UNITS
C
C
C*****   DEFINES MARGINS FOR THIS PLOT
      ENTRY LRMRGN(XLEFT,XRIGHT,YBOTM,YTOP)
C         XLEFT--    LEFT   MARGIN IN ABS. PLOTTING UNITS
C         XRIGHT--   RIGHT  MARGIN IN ABS. PLOTTING UNITS
C         YBOTM--    BOTTOM MARGIN IN ABS. PLOTTING UNITS
C         YTOP--     TOP    MARGIN IN ABS. PLOTTING UNITS
C
C
C*****   DEFINES GRID FOR THIS PLOT
      ENTRY LRGRID(IX,IY,DX,DY)
C         IX--       CODE FOR VERTICAL GRID LINE 0=STANDARD,
C                                  1=DX GIVES NO. OF LINES
C                                  2=DX GIVES INTERVALS BETWEEN LINES
C                                  3=DX GIVES NO. OF TICK MARKS
C                                  4=DX GIVES INTERVAL BETWEEN TICK MARKS
C         IY--       CODE FOR HORIZONTAL GRID LINES, USED AS IX IS USED
C         DX--       NUMBER OR INTERVAL, VERTICAL LINES
C         DY--       NUMBER OR INTERVAL, HORIZONTAL LINES
C
C
C*****   DEFINES PIECE OF FRAME FOR THIS PLOT
      ENTRY LRSIZE(XLEFT,XRIGHT,YBOTM,YTOP)
C         ARGS AS FOR LRMRGN
C
C
C*****   CONVERTS NUMBER TO BCD FOR PRINTING
      ENTRY LRCNVT(VALUE,NTYPE,CHARS,NFORM,N,M)
C         VALUE--    NUMBER TO BE CONVERTED, TYPE IS NTYPE
C         NTYPE--    ...1=VALUE IS INTEGER(FIXED),3=VALUE IS FLOATING
C         CHARS--    ARRAY TO HOLD CHARACTERS INTO WHICH VALUE IS CONV.
C         NFORM--    FORMAT CODE...1=I,3=F,4=E
C         N--        TOTAL NO. OF CHARACTERS
C         M--        CHARACTERS TO RIGHT OF DECIMAL,=0 FOR NFORM=1
C
```

```
C
C*****   TURNS EXTENDED MODE(USES WHOLE FRAME) ON AND OFF
      ENTRY LREON
      ENTRY LREOFF
C*****   TURNS MOVIE MODE(ROTATES 90 DEGREES) ON AND OFF
      ENTRY LRMON
      ENTRY LRMOFF
C
C
C*****   SETS CHARACTER SIZE
      ENTRY LRCHSZ(NSIZE)
C        NSIZE--    DEFINES CHARACTER SIZE, 0=LET PLOTTING ROUT. PICK,
C                                            1=TINY,2=SMALL,3=MED,4=BIG
C
      ENTRY TIMLFT(TIME)
C*****   READS INTERNAL SYSTEM CLOCK,RETURNS TIME LEFT TO END OF RUN
C        TIME--     TIME REMAINING,IN PULSES...ONE PULSE=1/60 SECOND
C
C
C
      ENTRY ARERR(TEXT)
C*****   FORTRAN-CALLED ARITHMETIC ERROR ROUTINE IN OUR SYSTEM,
C*****   TREATED LIKE LIBRARY ARITH ERRORS...CALLED ONLY BY DARCOS
C        TEXT--     MESSAGE TO BE PRINTED WHENEVER CALL IS EXECUTED
C                   TEXT MUST END IN $
C
C
      RETURN
      END
```

CONCLUDING REMARKS

A FORTRAN IV program has been developed to numerically study ion - polar-molecule collisions. Results are obtained in the form of time history plots and motion pictures, as well as collision statistics. This report describes, for potential users, the structure and use of the program.

Lewis Research Center,
National Aeronautics and Space Administration,
Cleveland, Ohio, July 29, 1970,
129-02.

APPENDIX - PROGRAM FOR GENERATING INITIAL CONDITIONS

The main program generates batches of random input conditions in the format expected by the collisions program (on the Q cards). A MAP listing of the random number generator RAND/SAND is also included.

Input

Two kinds of NAMELIST are used, one in the main program, the other in SUBROUTINE GENARY. They contain

Block	Variable	Variable in main program	Meaning
/IN1/	NC		number of cases of random coordinates for this set of values
	PSW		= 0, do not punch cards
			= 1, punch cards
	BETA1		= $1/KT_1$; where KT_1 is the most probable energy, first degree of freedom
	BETA2		= $1/KT_2$; where KT_2 is the most probable energy, second degree of freedom
			BETA2 = 0 for linear molecules
/GENER/	ATE	MIA	first moment of inertia, I_1
		MIB	second moment of inertia, I_2
		RA	initial separation, r_i
		VA	initial velocity
		BA	impact parameter, b

Subroutine GENARY

This subroutine reads the array ATE(42) and scans it looking for the form limit_1, increment_1, limit_2, increment_2, . . . and then generates values of uniform increment;

between limit_i and limit_{i+1}. It can be used to vary any or all of the parameters MIA, MIB, RA, VA, BA, in order to study the effect of these parameters on the trajectories.

Main Program

This program generates, in a series of nested loops, the coordinates and time derivatives, in sets of NC values for each set of parameter values.

Output

The main program prints the parameter values (in NAMELIST/OUT1/) and the random coordinates. Optionally, the coordinates are punched onto cards, suitable for input to the collision program, and the number of cards is printed (in NAMELIST/OUT2/). The coordinates are stored as follows:

Mathematical name	Stored in generator program in-	Stored in collision program in-
R	(not used)	Q(1)
θ	A(n, 1)	Q(2)
φ	A(n, 2)	Q(3)
η	A(n, 3)	Q(4)
ξ	A(n, 4)	Q(5)
ψ	A(n, 9)	Q(11)
$\dot{R}$	DR	Q(6)
$\dot{\theta}$	AA(n, 1)	Q(7)
$\dot{\varphi}$	AA(n, 2)	Q(8)
$\dot{\eta}$	AA(n, 3)	Q(9)
$\dot{\xi}$	AA(n, 4)	Q(10)
$\dot{\psi}$	AA(n, 5)	Q(12)

Listings

The following is a complete listing of the main program, subroutine GENARY, and subroutine RAND/SAND:

```
$IBFTC RANIT

      REAL A(999,10),RA(10),MIA(10),MIB(10),VA(20),BA(100),AA(999,10),
     1  M11,M12
      INTEGER PSW
      DATA TWOPI/6.2831853/
      WRITE(6,1001)
 1001 FORMAT(1H1)
      CALL SAND(STORE)
      NAMELIST/IN1/ NC,PSW,BETA1,BETA2
    5 READ(5,IN1)
      WRITE(6,IN1)
      PSW=2-PSW
      DO 10 IC=1,NC
      CALL RAND(RPHI)
      RTHETA=.5
      CALL RAND(RETA)
      CALL RAND(RXI)
      CALL RAND(RPSI1)
      CALL RAND(RPSI2)
      PSI1=TWOPI*RPSI1
      PSI2=TWOPI*RPSI2
C        THETA
      A(IC,1)=ARCCOS(1.-2.*RTHETA)
C        PHI
      A(IC,2)=TWOPI*RPHI
C        ETA
      A(IC,3)=TWOPI*RETA
C        XI
      A(IC,4)=ARCCOS(1.-2.*RXI)
      A(IC,5)=SIN(PSI1)
      A(IC,6)=COS(PSI1)/(2.*SQRT(RTHETA*(1.-RTHETA)))
      A(IC,7)=COS(PSI2)/(2.*SQRT(RXI*(1.-RXI)))
      A(IC,9)=0.
      IF(BETA2.EQ.0.) GO TO 10
      CALL RAND(RPSI3)
C        PSI
      A(IC,9) = TWOPI*RPSI3
   10 A(IC,8)=SIN(PSI2)
      CALL GENARY(MIA,NM1,10,17HMOMENT OF INERTIA,17)
      CALL GENARY(MIB,NM2,10,24HSECOND MOMENT OF INERTIA ,24)
      CALL GENARY(RA,NR,10,9HINITIAL R,9)
      CALL GENARY(VA,NV,20,16HINITIAL VELOCITY,16)
      CALL GENARY(BA,NB,100,16HIMPACT PARAMETER,16)
      NCARDS=NC*NM1*NR*NV*NB*NM2
      NAMELIST/OUT2/NCARDS
      IF(PSW.EQ.1)WRITE(6,OUT2)
      IG=0
      DO 50 IM1=1,NM1
      M11=MIA(IM1)
```

```
      DO 50 IM2=1,NM2
      MI2=MIB(IM2)
      DO 20 IC=1,NC
      CALL RAND(RRE)
      RE1=-ALOG(RRE)/BETA1
      F2=SQRT(2.*RE1/MIA(IMI))
      F3=0.
      AA(IC,5)=0.
C        ETA DOT
      AA(IC,3)=F2*A(IC,7)
C        XI DOT
      AA(IC,4) = F2*A(IC,8)
      IF(BETA2.EQ.0.) GO TO 20
C        PSI DOT
      AA(IC,5) = SQRT(RE1/MI2) - AA(IC,3)*(1.-2.*RXI)
   20 CONTINUE
      DO 50 IR=1,NR
      R=RA(IR)
      R2=R*R
      DO 50 IV=1,NV
      V=VA(IV)
      DO 50 IB=1,NB
      B=BA(IB)
      F1=VA(IV)*BA(IB)/R2
C        R DOT
      DR=-SQRT(1.-BA(IB)*BA(IB)/R2)*VA(IV)
      DO 30 IC=1,NC
C        THETA DOT
      AA(IC,1)=F1*A(IC,5)
C        PHI DOT
   30 AA(IC,2)=F1*A(IC,6)
C
      NAMELIST/OUT1/MI1,MI2,RE1,RE2,V,B,F1,F2,F3
      WRITE(6,OUT1)
      IG=IG+1
      WRITE(6,1002)((A(IC,I),I=1,4),DR,(AA(IC,I),I=1,4), AA(IC,5),A(IC,9
     1),IG,IC ,IC=1,NC)
 1002 FORMAT(1H0,1X,5HTHETA,4X,3HPHI,4X,3HETA,5X,2HXI,10X,2HDR,5X,
     16HDTHETA,8X,4HDPHI,8X,4HDETA,9X,3HDXI,8X,4HDPSI,4X,3HPSI/
     2(0P4F7.3,1P6E12.3,0PF7.3,4X,2I5))
      GO TO (40,50),PSW
C
   40 PUNCH 1003,((A(IC,I),I=1,4),DR,(AA(IC,I),I=1,4), AA(IC,5), A(IC,9)
     1,IG,IC,IC=1,NC)
 1003 FORMAT(0P4F5.3,1P6E10.3/0PF5.3,4X,2I3)
      PUNCH 1004
 1004 FORMAT(1H1/ )
   50 CONTINUE
      GO TO 5
      END
```

```
$IBFTC GENAR.

      SUBROUTINE GENARY(A,L,N,LABEL,NL)
      REAL A(N),ATE(42),LABEL(1)
      DO 5 I=1,42
    5 ATE(I)=0.
      NAMELIST /GENER/ ATE
      NW=NL/6
      IF(MOD(NL,6).NE.0) NW=NW+1
      WRITE(6,1002) (LABEL(I),I=1,NW)
 1002 FORMAT(1HL,47X,6A6)
      READ(5,GENER)
      DO 10 M=1,21
      IF(ATE(2*M).EQ.0.) GO TO 20
   10 CONTINUE
      CALL ARERR(47HGENER ATE INPUT CARD HAS MORE THAN 41 ENTRIES $)
   20 K=1
      A(1)=ATE(1)
      MM1=M-1
      IF(MM1.EQ.0) GO TO 60
      DO 50 I=1,MM1
      SGN=SIGN(1.,ATE(2*I+1)-ATE(2*I-1))
      D=SIGN(ATE(2*I),SGN)
      WRITE(6,1003) ATE(2*I-1),D,ATE(2*I+1)
 1003 FORMAT(45X,G13.6,1H(,G13.6,1H),G13.6)
      H=D*1.E-4
      C=0.
   30 K=K+1
      IF(K.GT.N)CALL ARERR(43HGENARY IS BEING ASKED TO OVERFILL A VECTOR
     1$)
      C=C+1.
      B=ATE(2*I-1)+C*D
      IF((B+H    )*SGN.GT.ATE(2*I+1)*SGN)GO TO 40
      A(K)=B
      GO TO 30
   40 A(K)=ATE(2*I+1)
   50 CONTINUE
   60 L=K
      RETURN
      END
```

```
$IBMAP RANDOM
       ENTRY   RAND
       ENTRY   SAND
SAND   CLA     3,4
       STA     B          MULTIPLIER IN RANDOM NO. GENERATOR
       CLA     ONE        SET DAM = TO 1 FOR FIRST RANDOM NO.
DAM    STO*    3,4
       TRA     1,4
RAND   SAVE    (4)
       LDQ*    B
       MPY     CONS       BY DAM
```

```
      B STQ     0         STORE THE LOW ORDER PART AT DAM
        CLA     FLC       FLOAT NORMALIZE , AND
        LLS     27        ROUND THE
        FAD     C         RANDOM NO.
R       STO*    3,4
        TRA     1,4
    FLC DCT     000000000200        EXPONENT OF RANDOM NO.
   CONS DEC     30517578125         5 EXP 15
    ONE DEC     1                   ONE
      C DCT     170000000200        NORMALIZING CONSTANT
        END
```

REFERENCES

1. Dugan, John V., Jr.: A Semiclassical Theory of Capture Collisions Between Ions and Polar Molecules. Ph.D. Thesis, Univ. Notre Dame, 1965.
2. Dugan, John V., Jr.; and Magee, John L.: Semiclassical Approach to Capture Collisions Between Ions and Polar Molecules. NASA TN D-3229, 1966.
3. Dugan, John V., Jr.; and Magee, John L.: Capture Collisions Between Ions and Polar Molecules. J. Chem. Phys., vol. 47, no. 9, Nov. 1, 1967, pp. 3103-3113.
4. Goldstein, Herbert: Classical Mechanics. Addison-Wesley Publ. Co., Inc., 1950.
5. Ralston, Anthony: A First Course in Numerical Analysis. McGraw-Hill Book Co., Inc., 1965.
6. Milne, William E.: Numerical Solution of Differential Equations. John Wiley & Sons, Inc., 1953.
7. Butcher, J. C.: On Runge-Kutta Processes of High Order. J. Austral. Math. Soc., vol. 4, pt. 2, 1964, pp. 179-194.
8. Dugan, John V., Jr.; Rice, James H.; and Magee, John L.: Calculation of Capture Cross Sections for Ion-Polar-Molecule Collisions Involving Methyl Cyanide. NASA TM X-1586, 1968.
9. Dugan, John V., Jr.; Canright, R. Bruce, Jr.; and Palmer, Raymond W.: Computer-Made Motion Pictures and Time History Plots of Ion-Polar-Molecule Collisions. NASA TN D-5747, 1970.
10. Dugan, John V., Jr.; and Rice, James H.: A Computer Plotting Description of Ion-Molecule Collisions with Long-Lived Capture Complexes. NASA TN D-5407, 1969.
11. Anon.: IBM 7090/7094 IBSYS Operating System Version 13 FORTRAN IV Language, IBM Form C28-6390-0.

NASA-Langley, 1970 —— 6 E-5832

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA TM X-2151		

4. Title and Subtitle	5. Report Date
FORTRAN IV PROGRAM FOR STUDYING ION - POLAR-MOLECULE COLLISIONS	January 1971
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
R. Bruce Canright, Jr., and John V. Dugan, Jr.	E-5832
	10. Work Unit No. 129-02
9. Performing Organization Name and Address Lewis Research Center National Aeronautics and Space Administration Cleveland, Ohio 44135	11. Contract or Grant No.
	13. Type of Report and Period Covered Technical Memorandum
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D. C. 20546	14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

Ion - polar-molecule collisions have been studied by numerically integrating their approximate equations of motion on a digital computer. In this report we present a FORTRAN IV program to do this integration (using conservation of energy as a step-size control) and to give results in the form of collision statistics, microfilm plots, and motion pictures of trajectories. Included are the description of the ion-molecule system, the form of the input and output, the functions of the main program and subroutines, the COMMON structure, a sample case, and the complete FORTRAN IV listings.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Ion-molecule collisions Equations of motion Differential equations Computer graphics	Unclassified - unlimited

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	57	$3.00

*For sale by the National Technical Information Service, Springfield, Virginia 22151

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
WASHINGTON, D.C. 20546

OFFICIAL BUSINESS

FIRST CLASS MAIL

POSTMASTER: If Undeliverable (Section 15
Postal Manual) Do Not Ret

"The aeronautical and space activities of the United States shall be conducted so as to contribute . . . to the expansion of human knowledge of phenomena in the atmosphere and space. The Administration shall provide for the widest practicable and appropriate dissemination of information concerning its activities and the results thereof."

—NATIONAL AERONAUTICS AND SPACE ACT OF 1958

NASA SCIENTIFIC AND TECHNICAL PUBLICATIONS

TECHNICAL REPORTS: Scientific and technical information considered important, complete, and a lasting contribution to existing knowledge.

TECHNICAL NOTES: Information less broad in scope but nevertheless of importance as a contribution to existing knowledge.

TECHNICAL MEMORANDUMS: Information receiving limited distribution because of preliminary data, security classification, or other reasons.

CONTRACTOR REPORTS: Scientific and technical information generated under a NASA contract or grant and considered an important contribution to existing knowledge.

TECHNICAL TRANSLATIONS: Information published in a foreign language considered to merit NASA distribution in English.

SPECIAL PUBLICATIONS: Information derived from or of value to NASA activities. Publications include conference proceedings, monographs, data compilations, handbooks, sourcebooks, and special bibliographies.

TECHNOLOGY UTILIZATION PUBLICATIONS: Information on technology used by NASA that may be of particular interest in commercial and other non-aerospace applications. Publications include Tech Briefs, Technology Utilization Reports and Technology Surveys.

Details on the availability of these publications may be obtained from:

SCIENTIFIC AND TECHNICAL INFORMATION OFFICE

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Washington, D.C. 20546