228p.

NASA Contractor Report 172464

INVESTIGATION OF THE EXTERNAL FLOW ANALYSIS
FOR DENSITY MEASUREMENTS AT HIGH ALTITUDE

(NASA-CR-172464) INVESTIGATION OF THE EXTERNAL FLOW ANALYSIS FOR DENSITY MEASUREMENTS AT HIGH ALTITUDE Final Report, 1 Jul. 1979 - 31 Aug. 1982 (Princeton Univ., N. J.) 228 p CSCL 01A G3/02 N87-11693 Unclas 44703

George K. Bienkowski

PRINCETON UNIVERSITY
Princeton, New Jersey

Grant NSG-1630
November 1984

[redacted]

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665

NASA Contractor Report 172464

INVESTIGATION OF THE EXTERNAL FLOW ANALYSIS FOR DENSITY MEASUREMENTS AT HIGH ALTITUDE

George K. Bienkowski

PRINCETON UNIVERSITY
Princeton, New Jersey

Grant NSG-1630
November 1984

[redacted]

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665

I. INTRODUCTION

Accurate experimental determination or verification of aerodynamic force coefficients (C_D and/or C_L) requires accurate simultaneous measurements of the forces (or accelerations) and the dynamic pressure ($q = 1/2\rho\ U^2$). Comparison with theoretical predictions requires independent knowledge of the density (ρ) and velocity (U) to establish the proper values of the non-dimensional parameters such as Reynolds number (Re) and Mach number (M). These parameters in principle require independent measurement of temperature and measurement or inference of viscosity. Under hypersonic conditions during the early phases of re-entry the Mach number becomes a secondary parameter while the relevant Reynolds number is based on viscosity within the gas layer near the vehicle and is only weakly dependent on free stream temperature. During the earliest part of re-entry, independent knowledge of density is necessary to establish the degree of rarefaction generally measured by the Knudsen number $Kn = \lambda/L$ where λ is a relevant mean free path and L the characteristic physical dimension (either vehicle size for the overall flowfield or entrance dimensions for the local behavior at the instrument).

The Shuttle Upper Atmosphere Mass Spectrometer (SUMS) Experiment(1) is designed to provide independent measurement of $q = 1/2\ \rho U^2$ within the high altitude range. When combined with information of vehicle velocity, it will provide independent determination of upper atmosphere density and coupled to accelerometer data will give the aerodynamic force coefficients within a regime difficult to simulate on the ground. The experiment is primarily intended to provide information between about 80 Km and 140 Km where rarefaction effects on the force coefficients are most important for a vehicle of the size of the Space Shuttle. It is also a regime where information on the atmosphere is relatively sparse as it lies below the altitude traversed by satellites and above that regularly assessed by ground launched meterological vehicles. The interpretation of the measurements, however, requires an adequate understanding of the flowfield around the Space Shuttle within the vicinity of the SUMS experiment in order to provide the proper data reduction procedure and an assessment of the accuracy of the results.

At sufficiently low altitudes (below about 80 Km for the Space Shuttle), conventional pitot probe measurements can provide the dynamic pressure with straightforward data reduction and relatively minor corrections. At sufficiently high altitude (above about 150 Km), free molecular theory can be used to infer free stream conditions from surface measurements. The forces, however, are small and of little interest while the measurements require instruments of high sensitivity and are therefore difficult. In the intermediate range of altitudes where SUMS is designed to provide data, the typical molecular mean free path is of the same order as the characteristic vehicle dimensions. Figure 1 shows the variation with altitude of the free stream Knudsen number ($Kn_\infty = \lambda_\infty/D$) based on the free stream mean free path λ_∞ and the Shuttle diameter D at the location of the SUMS orifice just ahead of the wheel well. Note that $Kn_\infty = .01$ at about 87 Km and $Kn_\infty = 10.0$ at about 136 Km. Intermolecular collisions can, therefore, neither be neglected (free molecular theory) nor represented by the resultant transport properties (continuum theory) over the major portion of the SUMS measurement regime. The gas properties at the entrance to the instrument are, therefore, dependent on a flowfield that can only be determined on the basis of a "molecular" theory.

In addition to the above "external flow problem", needed to establish properties near the surface of the vehicle, the entrance region of the instrument is typically either smaller or comparable to a local mean free path. In such circumstances, the connection between the gas properties at some distance into the internal plumbing and those at the vehicle surface, can be very sensitive to the velocity and angular distribution of the incoming molecules. This requires both a high degree of detail from the "external flow" and a local analysis that must assess the molecular behavior at the instrument entrance. We shall refer to this as the "entrance problem". Figure 1 also shows an approximate band of Knudsen numbers for the entrance region of the SUMS experiment. Kn_s is based on the mean free path λ_s at the vehicle surface and the orifice diameter ($d_o = .235$ cm) with surface properties fitted between free molecular results at high altitude and continuum Newtonian values at low altitude. Kn_c is a similar Knudsen number based on conditions behind the "entrance" tube with λ_c estimated on free molecular results using Hughes and deLeeuw theory[2] at high altitude, with continuum constant pressure results applied at low altitude. Note that conditions within the entrance tube range from clearly free molecular behavior above about 110 Km to transitional behavior near 80 Km with fully continuum results only approached at the lowest altitude of interest.

The subsequent connection between the properties immediately behind the entrance tube and the measurements at the mass spectrometer shall be referred to as the "internal problem". The analytical procedures for calculating pressure profiles through the internal plumbing are well established and will be further verified by instrument calibration[1].

During the preliminary phase of NASA Grant NSG 1630 (July 1979 to November 1979), the feasibility of examining the "external flow problem" for the Space Shuttle nose region within the relevant altitude range was established. A previously developed Direct Simulation Monte Carlo Computer Code[3,4,5] was found to be suitable as the starting point for this geometry and altitude range. Preliminary results were obtained at 87, 95, 105, 115 Km altitudes.

During the subsequent grant periods (November 1979 to September 1982) improvements in the modelling of the geometry and the molecular interactions have been incorporated in the external flow computer code. A number of runs at altitudes of 87, 95, 105, and 115 Km have been made to obtain a range of the relevant parameters and to provide input information at the SUMS entrance location. The "entrance problem" has been examined both by using published information (2,6,7), and a previously developed Monte Carlo code for internal geometries (6. Because of the combination of entrance geometry (very long tube) and the range of local Knudsen number over the altitudes considered, a totally new "entrance" computer code had to developed. This code provides the connection between the flux information at the orifice entrance obtained from the external code and the local gas properties behind the entrance tube where the gas is in equilibrium with the "cold" walls of the internal plumbing. This new code has only been exercised to a limited extent, but preliminary results relating the pressure within the tube behind the tile to the free stream dynamic pressure have been obtained. This information coupled with an appropriate calibration of the mass spectrometer provides the basis for a viable data reducion procedure of the SUMS experiment.

Section IIA contains a brief description of the operation of the EXTERNAL computer code (the detailed code is attached in Appendix A). Section IIB describes the issues associated with geometric modelling of the shuttle nose region and the modelling of intermolecular collisions including rotational energy exchange and a preliminary analysis of the vibrational excitation and dissociation effects. Section IIC discusses the selection of the trial runs and presents the major results.

Section IIIA contains a brief describtion of both the original version and the modified present code (INTERNAL) for the entrance problem (Appendix C contains the code listing). Section IIIB contains a disucssion justifying the selection of geometric, collisional and surface modelling parameters used for the trial runs. Section IIIC presents the preliminary results and discusses the major effects.

Section IV presents the conclusions that can be drawn from the present study, provides a preliminary estimate of the data reduction procedure and suggests future work.

II. EXTERNAL FLOWFIELD

The physical properties of the gas monitored by the SUMS instrument are not those of the free stream but are altered both by the intermolecular interactions in the external flowfield and by the combination of intermolecular and surface interactions within the entrance orifice and tubing leading to the instrument. External flowfield effects can be summarized in terms of the relation between local "stagnation" pressure and the free stream dynamic pressure ($q = 1/2\rho U^2$) at sufficiently low altitudes. Within the "transition" regime of interest (80–140 Km) the gas entering the orifice is neither in equilibrium with the surface nor simply related to the free stream. The only currently available technique for describing the flowfield within this regime and providing sufficiently detailed data on the physical state of the gas at the surface is the Direct Simulation Monte Carlo Code.

A. DIRECT SIMULATION MONTE CARLO COMPUTER CODE

"Monte Carlo" is the technique of using a simulated situation and random numbers to generate solutions from which information for the real case is then deduced statistically. The Monte Carlo approach ranges from being a strictly mathematical technique for evaluating the complicated multi-dimensional Boltzmann collision integral to a complete simulation of a number of molecules, with randomness introduced only in the initial conditions. A modification of this latter approach is the one used in the present development. It consists of simutaneously following a large number of particles which yields, to some degree, a "direct simulation" of the processes taking place. Because there are finite limits on computer storage space, a modification to the direct simulation technique was developed by G.A. Bird (Ref. 5) wherein the real gas is simulated by several thousand "sample" particles populated into cells of the sample space considered. For collision calculations, all the particles in one cell are used as a representation of the local distribution function from which collision pairs are chosen at random, but in proportion to their collision probability based on the real gas. This preceding discussion applies to a general program incorporating the direct simulation procedure. A specific computer program for the generalized three-dimensional program for axisymmetric bodies in a hypersonic multi-fluid flow is described below.

The program (EXTERNAL) conducts numerical experiments with a model multi-component gas. The real gas is simulated by several thousand molecules which may be thought of as a representative sample of the many billions of molecules in the corresponding real gas. The positions and velocity components of the simulated molecules are stored in the computer and typical collisions are computed among them as a time parameter is advanced. Since the flow is at an angle of attack to the body, three position coordinates, three velocity components and appropriate internal energy levels must be stored for each simulated molecule.

The computation of collisions starts at zero time with the molecules moving along the flow axis at the required freestream Mach number. The body is inserted into this flow at the zero time and the desired steady flow is obtained as the large-time solution of the resulting unsteady flow.

The free-stream flow vector lies in the x-y plane. The simulated region is bounded by the x-y plane as a plane of symmetry, an outer cylindrical boundary (the axis of the cylinder is the x axis), and two planes parallel to the y-z plane. These boundaries must be set sufficiently far from the body to eliminate interference. The simulated region is divided into a number of cells which are sufficiently small for the expected change in flow properties across the cells to be small.

The first step is to generate the initial, or zero time, configuration of molecules. The molecules are distributed over the simulated region and the velocity components assigned are appropriate to a gas in Maxwellian equilibrium and moving at the required Mach number. The body is then inserted into the flow and the molecules are allowed to move and collide among themselves. The move and collide processes are uncoupled by computing a number of collisions appropriate to a time interval Δt_m equivalent to a small fraction of the mean time between collisions, and then moving the molecules through distances appropriate to Δt_m and their instantaneous velocities. The distortion produced in the molecular paths by this approximation is small as long as Δt_m is small compared with the mean time between collisions, and smaller than the typical transit time of a molecule through a cell.

Since the change in flow properties over the width of one cell is assumed small, the molecules in a cell at any instant may be regarded as a sample of the molecules at the location of the cell. The relative location of the various molecules within the cell can then be disregarded. A pair of molecules is chosen at random from those within the cell under consideration and is retained or rejected in such a way that the probability of retention is proportional to the relative collision probability for the appropriate interaction law. When a pair is retained, a typical collision is computed between the two molecules and the new velocity components and internal energies are stored in place of the old ones.

In general, the relative number ratio of the species of molecules in the multi-component gas will differ from unity, requiring the computation of different types of collisions. There is, therefore, one time counter for each type of collision in each cell. For each collision, the correct time counter is advanced for the cell by an amount appropriate to the collision parameters. The probability of collision, and therefore the time advancement per collision, is made proportional to the number of molecules in the cell, and the relative velocity and cross-sections of the colliding molecules. Collisions are computed in each cell until all the time counters have advanced through at least a time Δt_m. When this procedure has been carried out for all cells, the overall experiment time is advanced by Δt_m and the molecules are moved through appropriate distances.

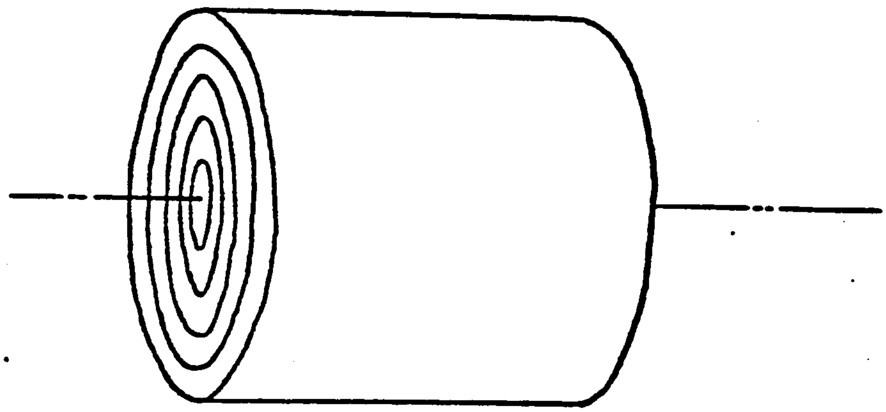
The set of molecules in each cell changes as the molecules are moved and appropriate conditions must be applied at the boundaries of the region begin simulated. Every boundary is treated as a source of molecules with velocity components representative of molecules moving in thermal equilibrium at the appropriate fraction of the free-stream Mach number. (The fractional Mach number is determined by the cosine of the angle between the local boundary normal and the flow direction.) Any molecule which moves outward across any boundary is regarded as being lost and is removed from the sample. The plane of symmetry (the x-y plane) is regarded as a specularly reflecting surface. Interactions with the body are also computed. The body consists of a number of conic sections rotated about the axis of symmetry. Each section must be separately specified according to the coefficients of the defining equation, a procedure to be described later in the report. For the purpose of computing the momentum and energy transfers to the surface, each region of the body can be subdivided into smaller sections. Within these smaller sections, the following three parameters must be specified: wall temperature/gas temperature, energy accommodation coefficient for each species, and tangential (momentum) accommodation coefficient for each species.

After the flow has settled down to a steady state, the number flux, momentum and energy transfers to the surface are accumulated and used to compute the aerodynamic data. The time required to establish steady flow is usually assumed to be close to the time required for the free-stream flow to traverse several body lengths. The overall number flux, drag, and heat transfer coefficients are determined, along with their distribution along the surface.

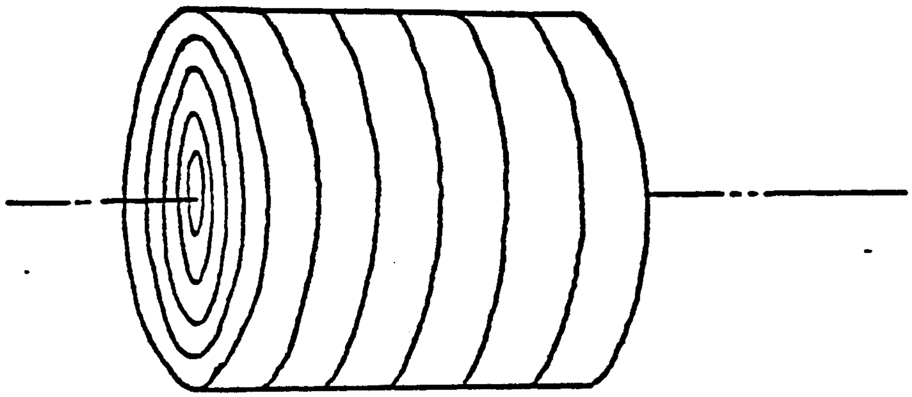
In addition, it is possible to generate data on the body surface which can be used as input to the companion program, INTERNAL, described separately. (INTERNAL) computes the flow field regime inside an axially symmetric cavity, such as might be used for a spacecraft-borne sensor. The input data needed for this computation includes the molecular distribution functions present at the orifice to the cavity, the orifice being on the surface of the spacecraft.) This data consists of velocity and internal energy samples in three coordinate directions for all species of the mixture.

Flow field properties are also computed. Instantaneous values are sampled at appropriate time intervals and these are time-averaged for greater accuracy. Number densities, velocities and temperatures are printed for each cell.

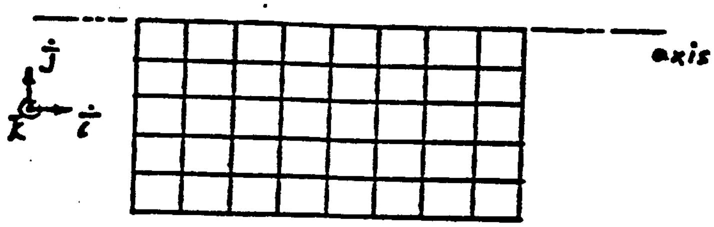
The numerical experiment takes place in a cylindrical block of space whose axis is coincident with the axis of revolution of the conic surfaces comprising the test body. This space is subdivided into cells in which the flow field properties of the experiment can be monitored. Cylindrical surfaces concentric with the axis partition the space into nested clindrical volumes, as shown:



Planes parallel to the end-faces of the cylinders divide the cylinders into a stack of nested rings.



Finally, planes perpendicular to the preceding planes and passing through the axis, called radial planes, divide up the rings in the azimuthal direction, producing cells which are wedge-shaped pieces of rings. The geometry is more easily visualized if one considers a trace of the cell configuration in a radial plane. A typical planar trace is shown:



The axis shown is the axis of symmetry of the sample space (and of the test body). This axis is considered to be the x axis. When this direction must be specified in vector algebra computations, a unit vector $\bar{i}$ is assumed in the x direction.

Assume that the planar trace shown above is bounded by the x-axis as described and by the -y axis. This plane is at 0° azimuth angle and is called the zero plane. It is the plane normally depicted when describing the sample space.

A unit vector $\bar{j}$ points in the +y direction. The z direction points out of the paper, and in this direction is the unit vector $\bar{k}$, which is given by $\bar{i} \times \bar{j}$.

The flow velocity is in the direction $\bar{i} \cos\alpha + \bar{j} \sin\alpha$, where α is the defined angle of attack between 0° and 90°. Since the flow is thus in the xy plane, there is symmetry in the z direction. That is, any condition in effect at +z is also in effect at -z. Thus azimuth angles need be specified only from 0° to 180° where 0° is in the -y direction, 90° is in the +z direction, and 180° is in the +y direction.

Now the fact is that the gas density in the vicinity of the stagnation point of the body can become many tens of times higher than the density far from the body. It is thus desirable to use small cells in this region while the cells are larger in the regions of relatively low density. This partitioning of cell sizes is accomplished in two ways.

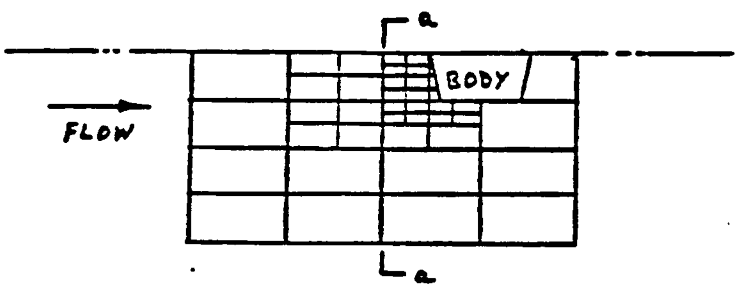
First, the rings can be divided azimuthally into two different sizes. This is done by specifying an angle, called THETAZ, and the number of wedge divisions both below and above this theta plane, called $NWEDGE_1$ and $NWEDGE_2$. ("Below the theta plane" means azimuth angles between 0° and THETAZ, and "above the theta plane" means azimuth angles between THETAZ and 180°.)

Second, the cells below the theta plane can be subdivided in the axial and radial directions down to second and then third level cells. In the 0° radial plane representation of the sample space, this would appear as large rectangles being sub-divided into small rectangles.

In this way, the sample space geometry can be tailored to the configuration of the test body angle of attack to the flow. The following examples are presented to clarify the above statement. Assume for all cases that the test body in question is a short cylinder. As explained in the section TEST BODY, the cylinder cannot have flat end faces, so the ends are cones with apex angles of about 175°.

a) For 0° angle of attack, the flow impinges directly on the left face of the cylinder.

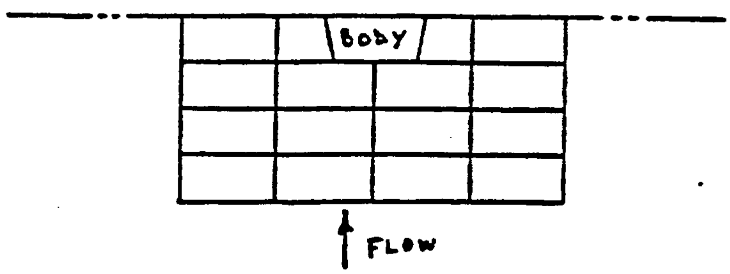
It is thus desirable to have, if possible, constant azimuthal angles since there is no angles since there is no azimuthal assymetry. One possible configuration is therefore: THETAZ = 180° NWEDGE $_1$ = 6, NWEDGE $_2$ = 0 (producing 30° wedges), and the axial and radial directions can be subdivided any convenient way, producing a zero radial plane that looks like:



In this type of geometry, all radial planes are the same as the zero plane.

b) For 90° angle of attack, the flow impinges on the curved cylindrical surface of the cylinder at the bottom. It is thus necessary to have small azimuthal wedges on the lower portion of the cylinder at and near the stagnation point, while larger wedges will suffice on the upper portions of the body (in the wake of the flow).

For instance, an acceptable set of parameters is: THETAZ = 60°, $NWEDGE_1$ = 3, $NWEDGE_2$ = 3 (producing 20° and 40° wedges), and again the axial and radial directions can be subdivided in any convenient way. Any radial plane up to 60° looks like the radial plane in example (a), with the body and smaller cells centered about a-a, while any radial plane between 60° and 180° looks like:



c) For an angle of attack between 0° and 90°, the configuration looks generally like that of example (b)). In this case, however, the theta plane generally should be at an azimuth angle which is relatively low (near 60°) for a high angle of attack (45° - 90°) and relatively high (near 120°) for a low angle of attack.

Because the test body is located on the axis of the cylindrical sample space, for each particle that interacts directly with the body, many more do not. In the interest of minimizing the program running time necessary to permit a statistically sufficient number of particles to strike the body, the computation makes use of zonal weighting factors. That is to say, each particle in the sample-space in reality represents one or more particles, the actual number depending upon the weighting factor of the zone in which the particle currently exists.

Up to five cylindrical boundaries are selected across which the zonal weighting factors change. These boundaries are specified in terms of the number of first level cells between the axis and the boundary. The change in the zonal weighting factor across each boundary can be given by:

$$LF_n = \frac{LD_{n+1} + LD_n}{\prod_{j=0}^{n-1} LF_j} \qquad n=1,2,\ldots 5$$

where the LD values are the number of first level cells between the axis and the cylindrical boundaries; and $LF_0=1$, $LD_6=LD_5=LD_K=NH$ were $K \geq K_{last}$.

This equation is the result of having the zonal weighting factors defined in such a way that they are equal to the ratio of sample space volume in the zone above the weighting-factor boundary to the sample space volume in the zone below the weighting factor boundaries. The importance of the region near the axis can be emphasized by choosing LF values larger than those given above.

The sample space is populated with a distribution of gas molecules. Each molecule is assigned a velocity, a rotational energy and a position, such that the sample space is uniformly (albeit in a random manner) filled with a gas in thermal equilibrium and flowing at the required Mach number. Each molecule is assigned a number corresponding to its zonal weighting factor. The molecule thus represents in actuality a number of molecules (including itself) equal to the zonal weighting factor. While the molecule moves in such a way as to stay within the given zone, its weighting factor does not change. If it crosses a weighting factor boundary while moving in toward the axis, a number of molecules is added to the distribution. The number of molecules added is equal to:

$$\frac{\text{old weighting factor}}{\text{new weighting factor}} - 1$$

For instance, assume that above a boundary, the zonal weighting factor is 6, and below the boundary it is 3. Hence 6/3 - 1, or 1, molecule must be added to the distribution when the molecule crosses the boundary. This is clearer if one considers that above the boundary, the molecule represents a total of 6 molecules. When the molecule drops below the boundary, it can only represent 3 molecules, so another molecule must be added to the distribution to represent the other 3 molecules.

The added molecule(s) is (are) assigned the same position and velocity components as the original molecule. While this does not approximate true kinetic theory at first, in practice the positions and velocities are soon randomized by collision processes.

On the other hand, if a molecule crosses a weighting factor boundary while moving away from the axis of the sample space, there is a probability that it must be dropped from the molecular distribution. The probability is given by

$$1 - \frac{\text{old weighting factor}}{\text{new weighting factor}}$$

The random number generator is used to generate a histogram of disappearing molecules to match the actual probability of disappearance. The whole idea behind using weighting factors is to increase test body - flow field interaction in a given running time. Thus, when body surface quantities are accumulated (like flux, energy, momentum, heat), they are accumulated in terms of the weighted number of molecules striking the body. This is particularly important if the body exists in two or more weighting zones, so that surface quantities in different zones can be correctly compared.

The program is set up to handle a test body consisting of a sequence of connected conic sections rotated about a common axis of symmetry. Some typical surfaces which can be considered without modification are sections of spheres, cones, cylinders, ellipsoids, hyperboloids, and paraboloids. A disc perpendicular to the symmetry axis cannot be handled without modification, since it represents a multi-valued function in r. Cones with very large apex angles (~180°) are used in place of discs. The procedure used to specify body surfaces is to generate a form of the standard equation of the surface in question, substitute into this equation the actual values of the constants, and reduce the standard form to the following type of equation:

$$Ax^2 + Br^2 + Cx + D = 0$$

where A, B, C, and D are numerical values

These coefficients are used as input data to the program. The program requires additional parameters for each conic surface. Each surface can be axially divided into several segments for the purpose of accumulating the body surface parameters like flux, heat transfer, etc. The x-coordinate on the right side of the segment is required as data. (The segments are divided azimuthally by the same radial planes that partition the sample-space.) In addition, the temperature, energy accommodation coefficients and tangential accommodation coefficients for each species are required for each segment. The temperature of a surface is normalized by the free-stream temperature.

The tangential accommodation coefficient specifies the fractional part of incoming tangential momentum that is lost to the surface. For the reflection model used in this program, this also represents the fractional part of the molecules colliding with the body surface whose collisions are diffuse. The remainder collide specularly.

The energy accommodation coefficient specifies the ratio of the net molecular energy flux absorbed by the body to that energy which would be absorbed if all re-emission were appropriate to equilibrium at the surface temperature of the body. This coefficient together with the tangential accommodation coefficient and the surface temperature determines the effective temperature of the diffuse component of the re-emitted molecules in the reflection model used in this program.

It is also possible to collect velocity samples of the colliding molecules on a restricted number of surface segments to generate distributions for the internal flow program. For this prupose, molecules that collide with the body surface are considered in two classes. The class called "UNCOLLIDED" consists of "free-stream" molecules. That is, these molecules have not previously collided with the body surface or with any other molecules other than "free-stream" molecules. The other class, called "COLLIDED", consists of molecules which have previously collided with either the body surface or with other "COLLIDED" molecules.

Appendix A contains the full listing of the computer code in Fortran IV. The main program consists of dimensioning statements coupled to a fiarly detailed description of the input cards (using the NAMELIST input). The main operating program is called RUN which in turn calls the appropriate subroutines. Figure 2 shows a schematic of the flow chart for the operation of the Program EXTERNAL.

B. Modelling of the SUMS Problem

1. Geometric Modelling

The computer code described above requires an axisymetric geometry of the body, although the flow vector can be at an arbitrary direction. The Space Shuttle nose geometry in the vicinity of the SUMS orifice has to be modelled by an equivalent body of revolution. A paraboloid of revolution around as axis inclined at about 8° with respect to the shuttle axis models the lower surface cross sections of the body both in the symmetry plane and in the transverse direction reasonably well. Figure 3 shows a sketch of the actual and modelled Space Shuttle nose geometry. This model will be called the parabola model, and the flow direction of 32° with respect to the axis will be used to represent the 40° angle of attack of the Space Shuttle.

Both for the purpose of benchmarking the test runs against continuum results and to achieve better resolution an alternative axisymmetric flow model was examined. This model consists of a hyperbola rotated about the flow velocity vector passing through the stagnation point. Figure 4 shows a typical cell geometry for this model. This is also the model used by Professor Clark Lewis for his continuum calculations. As will be seen from the results in the next section the latter (axisymmetric) model is questionable as a representation of the flow in the stagnation region of the

Space Shuttle, at least at higher altitudes.

2. *Modelling of Cross Sections*

In order to avoid uncertainties associated with the choice of hard sphere cross-section to best model the "typical" collision, an energy dependent inverse power law cross-section collision code has been incorporated in the program. The power law exponent and reference cross-section is chosen to provide the best fit to the viscosity temperature dependence over the range between the wall and stagnation conditions which represents the energy range of the important collisions. Figure 5 shows that viscosity can be matched to within a few percent over the relevant range with a single choice of exponent and reference conditions

Using the exponent of N=.552 and a reference temperature of 1000°K four axisymmetric cases of a hyperbola at 0° have been run to simulate 87KM, 95KM, 105KM and 115KM altitudes. Since in these cases rotational energy was not included these represent a ficticious monatomic gas of $\gamma=5/3$. Figure 6 shows "smoothed" temperature profiles normalized to the free stream temperature. Note that at the two lower altitudes a relatively "flat" Rankine Hugoniot (R.H.) region exists, while at 105KM R.H. conditions are barely reached and at 115KM the temperature peaks at about 70% of R.H. temperature. The shock layer in all cases is, however, very much thicker than the inviscid result based on nose radius R_N. Note, however, that the hyperbola is very blunt and it is not clear that the nose radius is the appropriate dimension or that this models properly the flow about the Space Shuttle Nose.

3. *Rotational Energy Exchange*

On the basis of previous experience in modelling rotational energy exchange in Reference 4 the model of Larsen and Borgnakke [9] was chosen as appropriate for the blunt geometry under investigation. The External Flow Program was therefore re-coded to include an arbitrary number of internal energy modes but with a single relaxation time related to the parameter ϕ which ranges from ϕ=0 representing no exchange to =1 simulating maximum available exchange at each collision. An indication of the effect at 115KM is shown in Figure 7 and 8 for the most rapid energy exchange (ϕ=1). Note that, as expected, the peak temperature is lower for the γ=1.4 case because some of the energy goes into rotation. Also note that significant non-euqilibrium exists between translation and rotation even at this maximum energy exchange rate corresponding to ϕ=1. The effect on surface properties shown in Figure 8 is virtually non-existent for the shear, but noticeable on the heat flux and pressure.

4. *Preliminary Attempts at Comparison with Continuum Results*

Some preliminary comparisons of both the monatomic (γ =1.667) and diatomic (γ=1.4) runs for the axisymmetric hyperbola were made with continuum results provided by Prof. C.H. Lewis based on continuum theory [10,11]. Among the surface properties only the pressure distribution agreed reasonably well. Since this is the property least sensitive to detailed flow field behavior the agreement is not a very sensitive test. A typical comparison is shown on Figure 8. A comparison of heat flux is not shown on this figure as it is not the same scale at this high altitude. The continuum result of Prof. Lewis gives the stagnation point heat transfer coefficient $C_H = Q/1/2\rho U^3 = 2.8$ which is physically unrealistic. At 105KM the continuum heat transfer coefficient C_H is below one but still appreciably above the Monte Carlo results. At 95KM the Monte Carlo results give heat transfer that is almost twice as large as Prof. Lewis's result for γ=1.4.

The most significant discrepancy between the continuum and Monte Carlo results arises in the shock layer thickness. While the definition is somewhat arbitrary, a comparison of the subsonic region in the vicinity of the stagnation point does give a good indication of the extent of influence of the "downstream" portions of the body. Figure 9 shows the Monte Carlo results for the M=1 line at 115KM together with the shock line from C.H. Lewis. This discrepancy led to a whole re-examination of the modelling of the nose geometry.

5. *Re-examination of Geometric Modelling*

Since the subsonic region appears to extend to the "shoulder" wherever the hyperbola is terminated as shown in Figure 9, the applicability of the hyperbola model becomes suspect, at least for the Monte Carlo calculation where upstream influence cannot be eliminated. In order to further assess the effect of the geometric model on the results a comparison of the hyperbola at 0° to the parbola at 32° are presented in Figures 10 and 11. It can be seen from Figure 10 that the subsonic layer in the vicinity of the SUMS orifice is significantly different in the two cases. Figure 11 shows that while the pressure is not dramatically affected by the model the heat flux and shear are significantly altered. On the basis of these results it was concluded that while the runs for the hyperbola may indicate trends they are neither representative of the SUMS region on the Space Shuttle nor good candidates for benchmarking with the continuum results. All subsequent runs were therefore made using the geometric model of a parabola at 32° angle of attack.

6. *Inelastic vibrational Excitation and Dissociations*

Because of the high energies of collision between free-stream and reflected molecules, inelastic collisions (certainly vibration and dissociation) are in principle possible. At the highest altitude the number of such collisions is expected to be insignificant because the cross-sections are low enough so that a typical molecule will reach the body with a negligible probability of a previous vibrationally exciting or dissociating collision. At the lower altitudes 87 to 95Km the number of collisions suffered by a typical molecule before reaching the body surface is measured in the tens or hundreds and therfore inelastic cross-sections that are only a few hundredths of the elastic and rotational ones may produce significant effects. The detail needed to properly model these collisions in the Monte Carlo Programs far exceeds the available experimental information, which primarily gives overall rate constants. An investigation of the best combination of analytical and empirical information was initiated early within the grant period. A theoretical attempt to couple low energy vibrational excitation experimental information to the highly non-equilibrium high energy collisions through theoretical work was initiated. Appendix B contains a Master's thesis presenting the formalism and giving the initial results of a theoretical formalism. Based on those results, coupled to some limited experimental data, a method for determining the probabilities of specific outcomes in individual collisions in the Monte Carlo Programs can be developed. The increase in computing times, however, may make the feasability of using such a code, for anything but benchmarking, prohititive.

C. MAJOR RESULTS FOR EXTERNAL FLOW

The primary effort during the grant period up to February 1981 was spent in developing the code for external flow and establishing the appropriate geometry to model the region of the Space Shuttle in the vicinity of the SUMS experiment. The major results to that date are presented in the renewal proposal for the period February 1, 1981 to July 30, 1982 which also served as a progress report on the previous grant period.[12] We will only summarize those results here and update them on the basis of additional external flow computations.

A representation of the shuttle nose geometry as a paraboloid of revolution around an axis 8° from the actual shuttle axis (Figure 3) was found to adequately model the windward side of the shuttle in the vicinity of the SUMS orifice. Computatons for both this model at an effective angle of attack of 32° and an alternative axisymmetric flow about hyperboloid model centered on an axis through the nominal stagnation point, demonstrated that the paraboloid model at angle of attack is necessary to adequately model conditions near the SUMS orifice. The typical body and computation cell geometry is shown in Figure 12.

A resonable indication of the flowfield can be obtained by examining density contours or Mach number contours about the body. Figure 13 shows the sonic lines and the M = 5 lines indicating approximately the outer extent of the "shock" layer at 95 Km and 115 Km, within the plane formed by the velocity vector and the axis of the modelled paraboloid. Note the greater "shock" layer thickness at the higher altitude. Also note the fact that even at 95 Km, the shock layer is a large fraction of the local body dimension such as nose radius. The "shock" thickness is a major portion of the entire "shock layer" casting doubt on any calculation incorporating a thin "shock" assumption. Figure 14 shows some density contours on the "windward" side of the body at 95 Km altitude as well as the sonic line. Note the rather constant density rise towards the body with no discernible separation between "shock" and "boundary layer." Also shown are estimates of the stagnation streamline and another streamline along which the velocity only goes slightly subsonic. Note the rather gradual turning along the latter streamline and the rather diffuse nature of the shock layer even at this altitude where the nominal free stream Knudsen number is 0.04. In order to give a better picture of the three-dimensional aspect of the flowfield, a sketch of one-half of the paraboloid and some contour plots of M = 1. and M = 5. lines are shown on Figure 15.

Information in Figures 13 through 15 gives some indication of the nature of the flowfield in the vicinity of the SUMS orifice. The ultimate objective, however, is to establish properties at the vehicle surface. Figure 16 shows the variation of two surface fluxes (normal pressure and heat flux) at 95 Km along the four cross-planes shown in Figure 15. The normal pressure p_s normalized by $1/2\rho U^2$ ($C_p = p_s/1/2\rho U^2$) is shown on the right side of the figure while a heat transfer coefficient ($C_H = Q/1/2\rho U^3$) is shown on the left. Figure 17 shows the variation versus angle in the cross-plane nearest the one containing the SUMS orifice, and compares the results to some theoretical and semi-empirical predictions. The pressure coefficient, as expected, lies approximately between the free molecular (C_p(F.M.)) value and the modified Newtonian (C_p(NEWT) value, up to about 90°. (Note that at 90° around the axis the local angle (β) between the velocity vector and the surface normal is approximately 64° at this cross-plane.) The heat transfer coefficient lies substantially below the free molecular value. It is also compared to a semi-empirical extrapolation of experimental results of stagnation heat transfer on hemispheres presented in reference 13. Direct comparison is clearly dubious due to the substantially different body geoemtry and the implicit assumption of totally local behavior contained in the cosβ variation with local angle of the surface to the velocity vector. Some additional degree of uncertainty is contained in the choice of "body size" that is used to evaluate the correlation parameter K_R^2.

The ultimate objective of the external flow calculations is to provide information on the properties of the gas entering the SUMS orifice. It is the potential non-equilibrium nature of the entering distributin of molecules that is responsible for the difference between the surface "pressure" and the gas pressure within the internal plumbing around the mass spectrometer. The non-equilibrium aspect is most commonly represented by a temperature jump and a velocity slip within the continuum formalism. Under the highly rarefied conditions at the upper end of the altitude range of interest, even that description may be inadequate because of the highly non-Boltzmann distribution of the molecules arriving at the surface. Figure 18 shows the distribution function of the flux of molecules arriving at the SUMS orifice at 95 Km and 115 Km obtained from the external program. The flux distribution is plotted both versus molecular velocities normal to the local surface and tangential within the plane formed by the free stream velocity vector and the local normal. Note that at 95 Km one could fit the distribtion by a Maxwedllian with some temperature different than the body temperature and a slip velocity which is comparable to the free stream mean molecular speed. At 115 Km the distribtion is clearly composed of two components, the free stream molecules arriving unperturbed at the surface, and the collided molecules having a broader distribution possibly representable by another Maxwellian. Clearly this potential bimodal character of the incoming molecules must be recognized in the evaluation of the "entrance" problem at the SUMS entrance.

III. Entrance Problem

As a companion to the initial version of the external code (EXT) an internal code (INT) was developed [8]. The objective of this code was to determine local properties and surface fluxes in a cavity connected by a tube to the exterior surface. The input molecular distribution is obtained from the surface flux information provided by the external code. While the code is in principle general to allow a wide variety of geometric configurations it is optimized for a short tube-large cavity geometry. Early attempts to apply this code directly to the SUMS inlet geometry resulted in very long running times with little assurance that a steady state solution had been achieved. A total recoding of the entrance problem was therefore implemented resulting in a code (INTERNAL) that allows greater flexibility in handling the long tube geometry of SUMS as well as incorporates all of the changes in intermolecular collisions, numbers of species and rotational energy exchange developed for the code EXTERNAL.

A. Entrance Computer Code INTERNAL

The purpose of the program described in this section is to determine the fluid field inside a cavity which consists of a connected sequence of conic sections rotated about an axis of symmetry. The cavity is considered in two parts: a main cavity on whose interior surface the sensor will be located, and an inlet tube whose orifice is presumed to be at the exterior surface of a spacecraft. A detailed code listing is given in Appendix C.

The input data for this program includes a molecular distribution function which is obtained from an external-flow run. Some of the other data refers to input parameters of the external-flow run. Thus it is seen that the pair of programs can be run as a set, computing the conditions inside a cavity which exist for a given set of conditions in the undisturbed free-stream flow. The programs have not yet been directly coupled, although they are written with this intent.

The program was constructed by turning an external-flow program inside out. In doing this, the basic molecule/body collision mechanism is preserved with only minor changes, while the molecule/molecule collision mechanism are not changed in any way.

A general description of the way in which the internal-flow program conducts the experiment is unnecessary since the description in the preceding section generally applies here. Any important differences will be described as they occur.

The numerical experiment takes place in a cylindrical block of space quite similar to that used in external flow. However, the inlet region must be cylindrical in shape while the cavity region can be defined by conic sections.

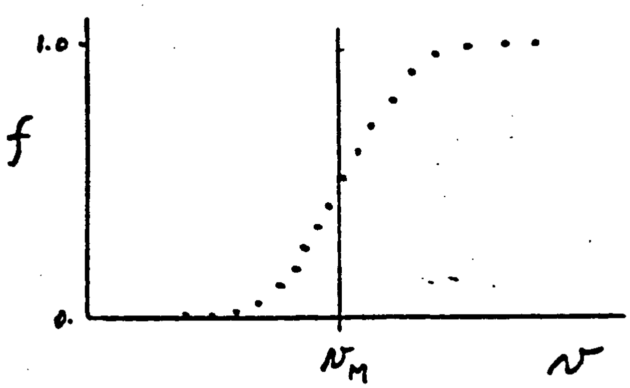
There is no subdivision of 1st level cells into smaller sized cells. Also, the number of azimuthal wedges is specified for a full 360° (since there is no plane of symmetry determined by an input flow direction) and is the same for both regions. No weighting factors are used as the desired surface fluxes on the walls generally occur near the outer edges of the sample space.

The orientation of the sample space is determined by the geometry of the particular body surface segment in external flow which is used as the orifice area of internal flow. The body surface normal of external flow becomes the x-axis (axis of symmetry) of internal flow; the direction given by the cross product of the x-axis of external flow and the body surface normal becomes the z-axis of internal flow; and the direction given by the cross product of the above cross product and the body surface normal becomes the y-axis.

The zero radial plane is the positive x-y plane, and azimuth angles range from 0° to 360° , with the psoitive z-axis at 90° .

The sample space is initially populated with gas molecules in a manner similar to that used in external flow. In this program, however, the gas molecules are initially in thermal with the walls locally. The selection of the density profile initially in the tube indeed poses a problem and is very critical in minimizing running time. The code therefore allows an arbitrary initial distribution specified by the user. The choices and procedures for selecting them are described in the next section. The key objective is to provide a distribution that is consistent with the input flux and will not have to be severly altered in magnitude to arrive at the steady state.

The molecular input distributions are generated from data produced by an external-flow run. This data consists of velocity samples of molecules impinging on the external body surface segment which is considered to be the orifice area of the inlet region. The form of the velocity samples is a series of horizontal S-shaped curves, one for each molecule type, in each of three directions, for both UNCOLLIDED (free-stream) and COLLIDED molecules. Each s-curve gives the fraction of molecules (impinging on the body surface segment) with a velocity $\leq$ the given velocity. A typical curve for UNCOLLIDED molecules is given [v_M is the velocity corresponding to the center velocity of the distribution as computed in the external flow program]:



The information from these curves is used directly to generate the molecular distribution of the molecules entering the inlet.

The population (pressure) inside the cavity region is selected initially as an input. The computer simulation proceeds until a steady profile inside the tube entrance region is generated. Since this does not necessarily require zero net flux, a series of runs with different cavity pressures is generated and a cross plot of flux versus pressure is the output for a single external flow input condition. The equilibrium solution if desired can be obtained from the zero flux point.

B. Scope of "Entrance Problem" for SUMS experiment

Within the free molecular regime the "entrance problem" has been examined by Hughes and deLeeuw[2]. In order to indicate the potential magnitude of the problem, Figure 20 shows the variation of both the surface pressure (p_s) and the chamber pressure (p_c) versus angle of attack under free molecular conditions at infinite speed ratio and a very long tube (conditions approached at high altitude during Space Shuttle re-entry). For the SUMS location, the local angle of attack β is approximately 28° giving a possible difference of a factor of 10 between the surface pressure (p_s) and the internal chamber (p_c) that directly affects the mass spectrometer reading. While this theoretical result is expected to be accurate at 140 Km and above, at lower altitudes the local flux distribution will have both a directed and a rather diffuse component (see Figure 18), and also the effect of internal collisions within the tube will begin to play a role (see Figure 1).

Because the tube length to diameter ratio is very large (37) the time constant and therefore the computing time to reach equilibrium is very long. For this reason no attempt is made to simulate the problem all the way to the condition where the chamber pressure behind the tube is at the correct value to nearly balance the net flux. Instead a series of runs with different assumed "cavity" pressures are performed and the zero net flux (or a given small value for the dynamic condition) can be selected to interpolate the correct "cavity" pressure. This procedure if it proves generally successful can of course be automated within the code.

C. Preliminary Results of INTERNAL Code

Preliminary results for the SUMS entrance tube geometry are presented as a couple points on Figure 21 giving to the ratio p_c/p_s versus altitude.

Also shown are some theoretical curves for full continuum (Continuum), for continuum external flow but free molecular flow through the tube (No slip, $Kn_s \rightarrow \infty$), and for fully free molecular flow (Free Molecule). In addition, some points using a best fit Maxwellian to the external flow flux and free molecular internal flow from reference 2, are shown at 95 Km and 115 Km. These results are preliminary, but they do indicate the magnitude of the correction and the fact that no simple use of the currently available results can cover the entire range of altitudes of interest.

IV. DISCUSSION

If we couple information from INTERNAL (Figure 21) to the results of the external flow computations (summarized as a plot of $p_s/1/2\rho U^2$ versus altitude in Figure 22), we can produce a preliminary estimate of the data reduction curve that could be coupled to the calibration of the mass spectrometer to deduce the $q = 1/2\rho U^2$ during Space Shuttle re-entry. Figure 23 is a cross-plot of the q/p_c versus p_c obtained from Figures 21 and 22 with the 87 and 105 Km results only estimated on the basis of interpolation of Figure 21. The establishment of such a data reduction curve for the nominal re-entry conditions, together with associated error bars as well sensitivities to wall temperature and angle of attack is necessary for the proper interpretation of data to be obtained by the SUMS instrument.

The data measured by the mass spectrometer in the SUMS experiment essentially provides collector currents of charged species of different masses. Calibration of the instrument can relate these to the overall pressure and composition at the entrance to the instrument being calibrated. Since the actual environment of the flux of molecules to be encountered under flight conditions cannot be simulated the calibration is performed with the incident flux essentially in equilibrium with the instrument outer walls (room temperature and no flow). The data reduction procedure must therefore relate the effective environment in the ground test simulation to the desired dynamic pressure q under the flowing non-equilibrium condition and through the calibration to the instrument measurements.

As discussed above, capability now exists for calculating the surface flux distribution of molecules entering an opening at the SUMS location. The computational procedures for connecting that information to the pressure immediately behind the entrance tube has also been developed, although not fully exercised over the entire range of parameters applicable to the SUMS experiment. That pressure, in turn, can be directly related to the ground test environment used to produce the calibration curves for the mass spectrometer.

The primary objective of future work must be to provide a data reduction procedure that relates the spectrometer reading to the free stream dynamic pressure ($q = 1/2\ \rho U^2$). With currently available procedures a relation between the pressure p_c and q such as the preliminary one shown on Figure 23 can be obtained using the best available information on the flight parameters such as velocity, angle of attack, tile temperature, surface conditions, molecular collision parameters, etc. This relation must then be combined with the calibration curve where the instrument readings are related either to p_c directly, or to a calibration pressure which can be related to p_c by conventional means. A single plot of q versus total measured collector current can thus be obtained from the combination of these results.

The relation between the dynamic pressure q and the calibration pressure p_c depends on many parameters of the problem. Some of these, such as flight velocity, angle of attack, and tile surface temperature are expected to vary only slightly from their nominal values. Since the actual measured values of these quantities will be available on each individual flight, corrections to the data reduction relation should be evaluated in the form of sensitivity coefficients for small changes from the nominal. Studies to determine the effects of these parameters are necessary to establish the significant sensitivity coefficients that must be incorporated into the data reduction scheme.

The parameters such as surface accommodation, surface recombination, and free stream composition can also affect the results. In addition, modeling simiplifications of both the geometry of the problem and the molecular collision phenomena can alter the quantitative value of the relation between q and p_c. Because of the unavailability of any in-flight measurements that could lead to an evaluation of these parameters, bounds on the uncertainties they produce should be studied. Sensitivity of the data reduction relation to the most significant of these can then produce bounds on uncertainties on the dynamic pressure q, due to reasonable variations. The final goal of future work is an algorithm for the evaluation of the dynamic pressure q, from the calibration pressure p_c, together with error bounds, due to the uncertainties associated with the external flow and the entrance problems.

REFERENCES

1. Blanchard, R.C., Ducket, R.J. and Hinson, E.W. - A Shuttle Upper Atmosphere Mass Spectrometer (SUMS) Experiment - AIAA paper No. 82-1334 (August, 1982).

2. Hughes, P.C. and deLeeuw, J.H. - Theory for the Free Molecule Impact Probe at an Angle of Attack - in Rarefied Gas Dynamics, Fourth Symposium (Academic, New York, 1965) Vol. I, pp. 653-676.

3. Bienkowski, G.K., McKinley, R.R., and Siskind, S.M. - Monte Carlo Solutions for Hypersonic Flow of Binary Mixtures over Axially Symmetric Bodies - in Rarefied Gas Dynamics, Seventh Symposium (1970) Vol. II., pp. 861-874.

4. Pullin, D.I., Harvey, J.K. and Bienkowski, G.K. - Hypersonic Leading Edge Flow of a Diatomic Gas by the Direct Simulation Method - in Rarefied Gas Dynamics, Ninth Symposium (1974), Vol. II, p. D-5.

5. Bird, G.A., Molecular Gas Dynamics, Oxford University Press, London, (1976).

6. Harbour, P.J. and Bienkowski, G.K. - Method for the Interpretation of Surface Pressure Measurements under Rarefied Hypersonic Conditions - in Physics of Fluids, Vol. 16, (1973), pp. 600-611.

7. Potter, J.L., Kinslow, M., and Boylan, D.E. - Influence of Orifice on Measured Pressures - in Rarefied Gas Dynamics, Fourth Synposium (Academic,. New York, 1966) Vol. II, pp. 175-194.

8. Final Report for Low-altitude Satellite Interaction Study (Neutral Gases), Contract No. NAS5-11241, RCA AED R-3635-F, October, 1970.

9. Larsen, P.S. and Borgnakke, C., "Statistical Collision Model for Simulating Polyatomic Gas with Restricted Energy Exchange" in Rarefied Gas Dynamics, Ninth Symposium, paper A7 - July 1974.

10. Lewis, C.H., Gogineni, P.R. and Murray, A.L., "Comparison of Viscous Shock-Layer Predictions with Experiments for Blunted Cones in Rarefied Transition Regime." in Rarefied Gas Dynamics 11th Symposium, Cannes, France July, 1978.

11. Gogineni, P.R. and Lewis, C.H., "VSLPNS: Three-Dimensional Viscous Hypersonic Flow Predictions Using Combined Shock-Layer and Parabolized Navier-Stokes Equations" Report No. VPI-AERO-094, Virginia Polytechnic Institute and State University, April, 1979.

12. Bienkowski, G.K. - Renewal Proposal on NASA Grant NSG 1630 for February 1, 1981 to July 30, 1982.

13. Coleman, G.T., Metcalf, S.C., Berry, C.J. - Heat Transfer to Hemisphere Cylinders and Bluff Cylinders Between Continuum and Free Molecular Flow Limits - in Rarefied Gas Dynamics, Tenth Symposium (1976) - Paper 202.

FIGURE 1. RELEVANT KNUDSEN NUMBERS VERSUS ALTITUDE

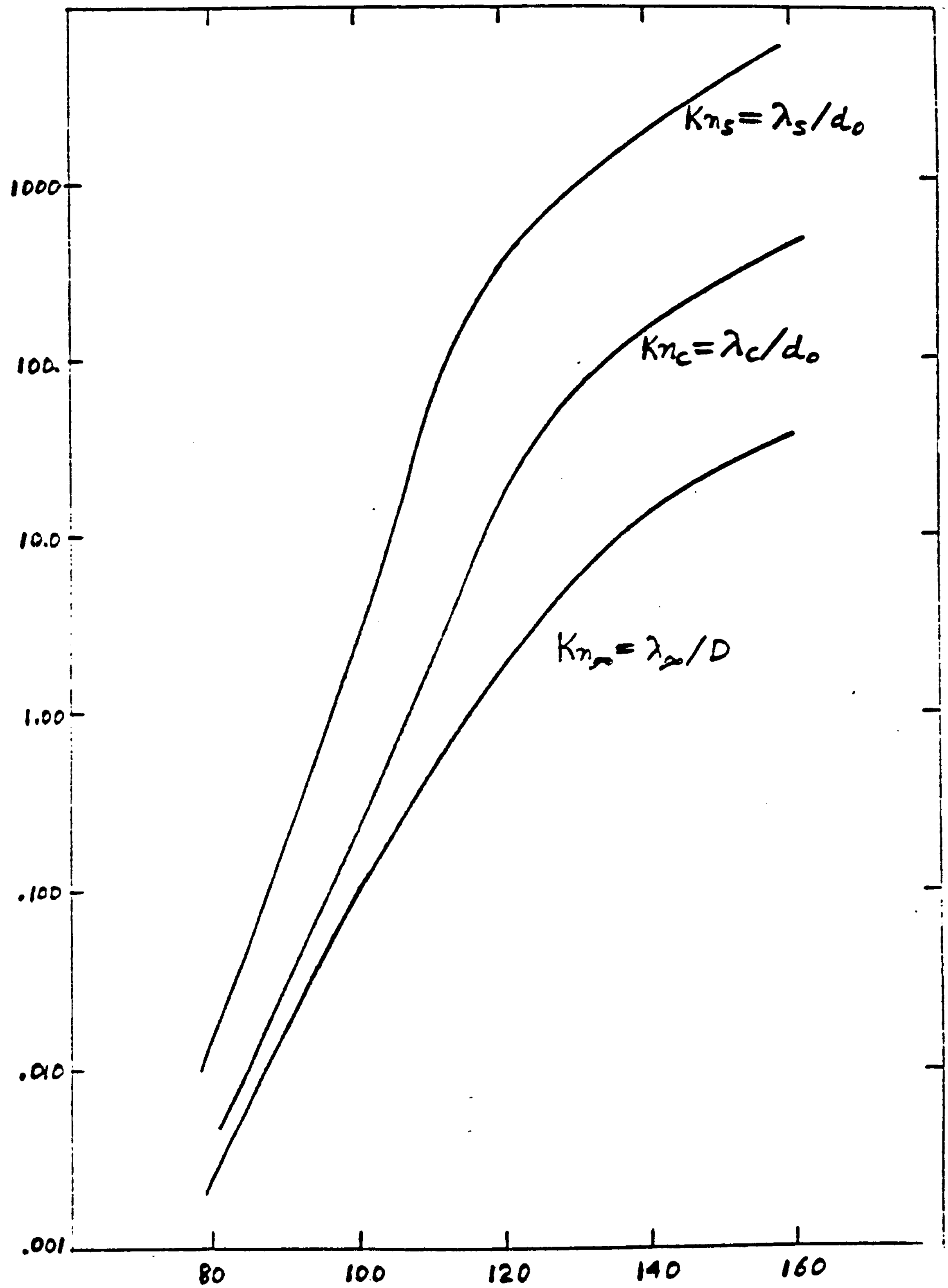


FIGURE 2. SYSTEM FLOW-CHART

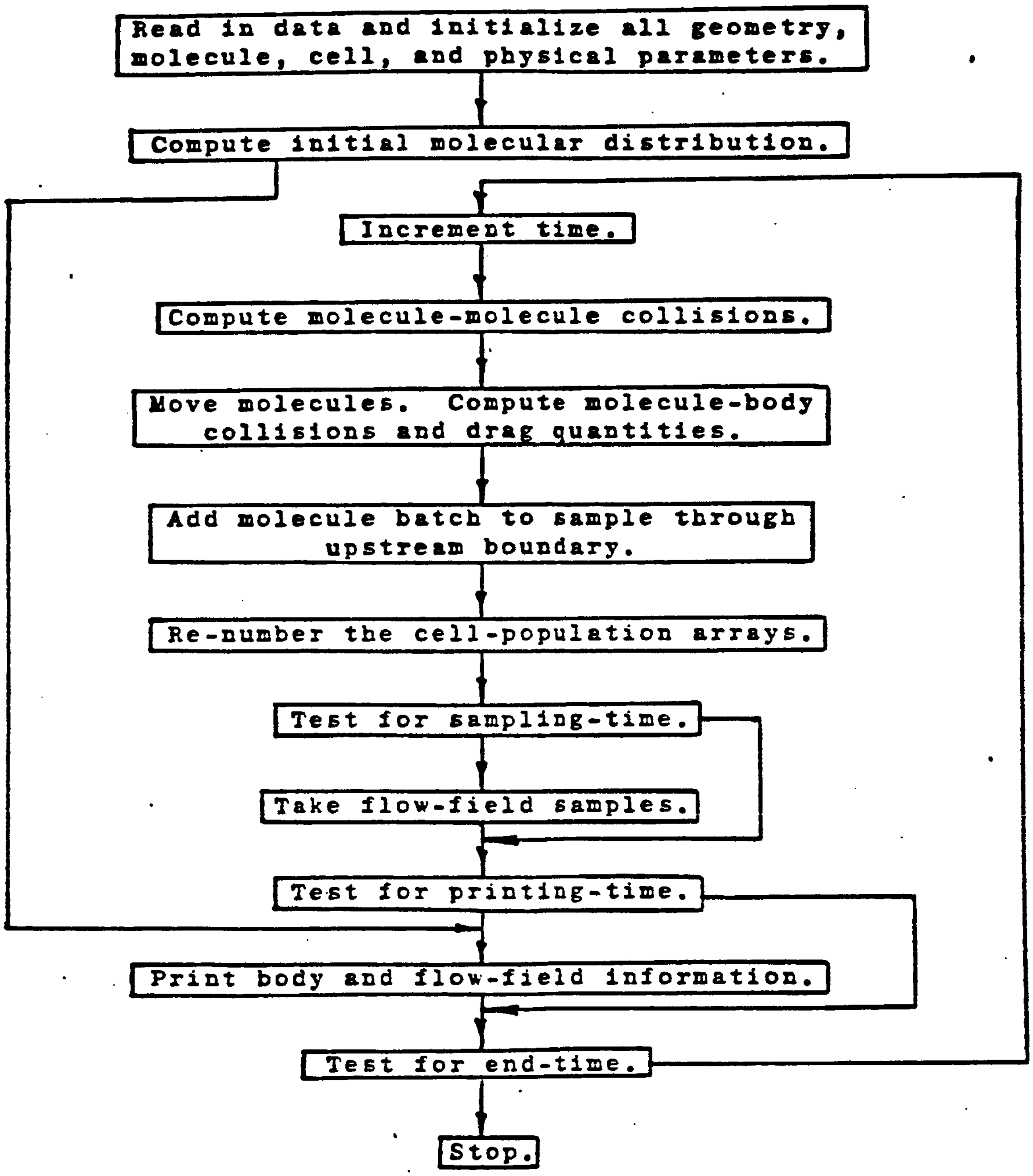


FIGURE 3. MODELING OF SPACE SHUTTLE NOSE (PARABOLA)

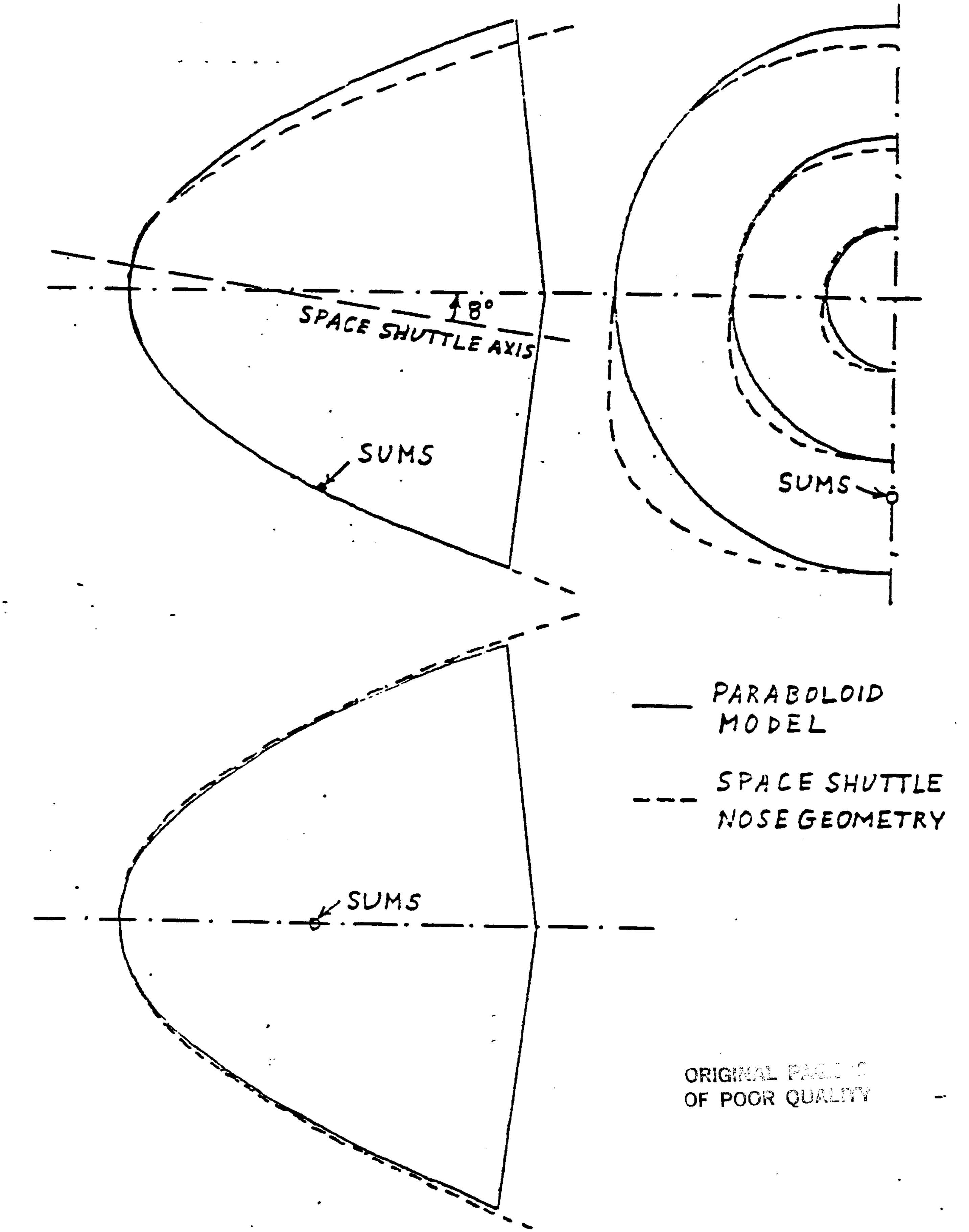


FIGURE 4. TYPICAL CELL CONFIGURATION FOR HYPERBOLA

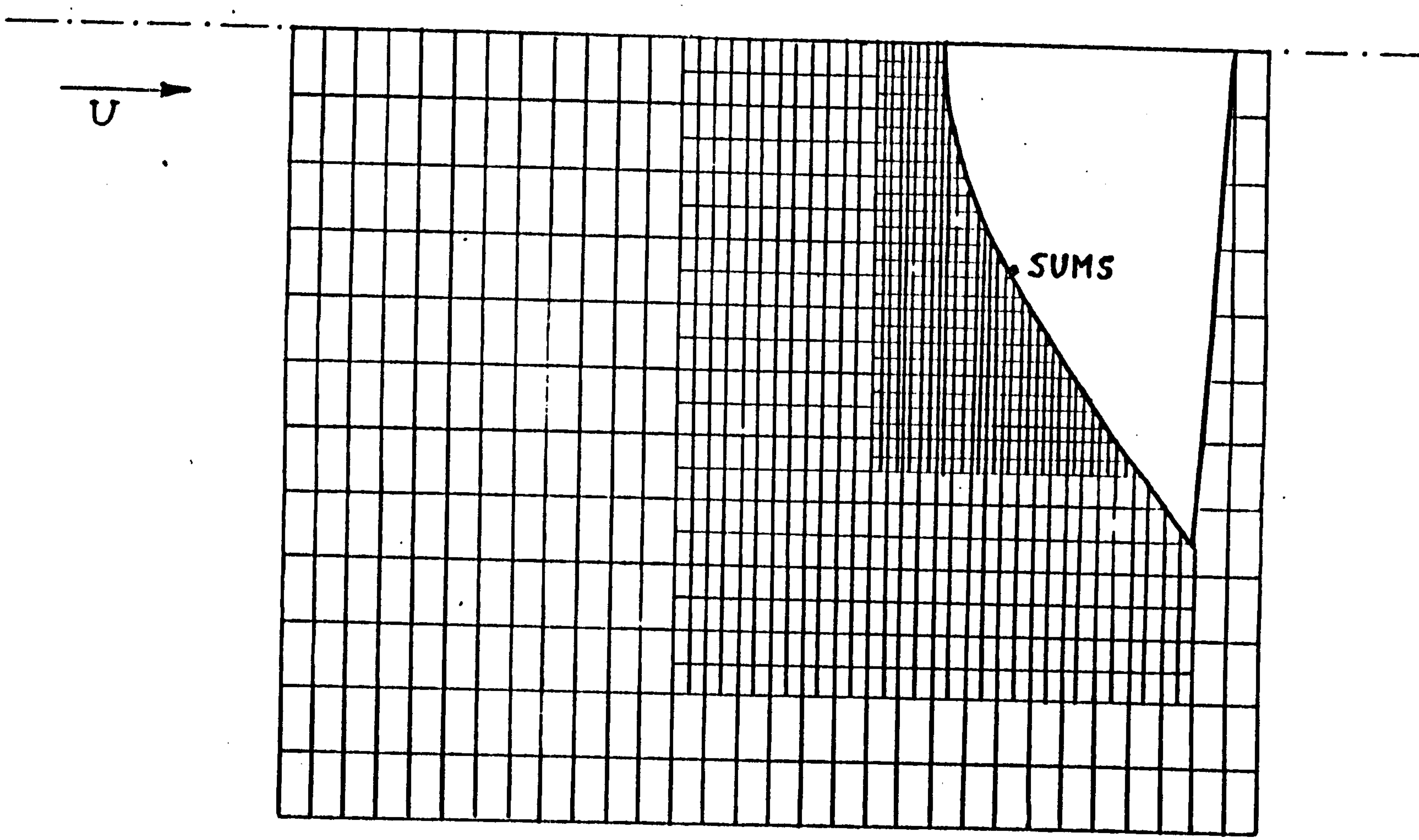


FIGURE 5. MODELLING OF VISCOSITY-TEMPERATURE DEPENDENCE

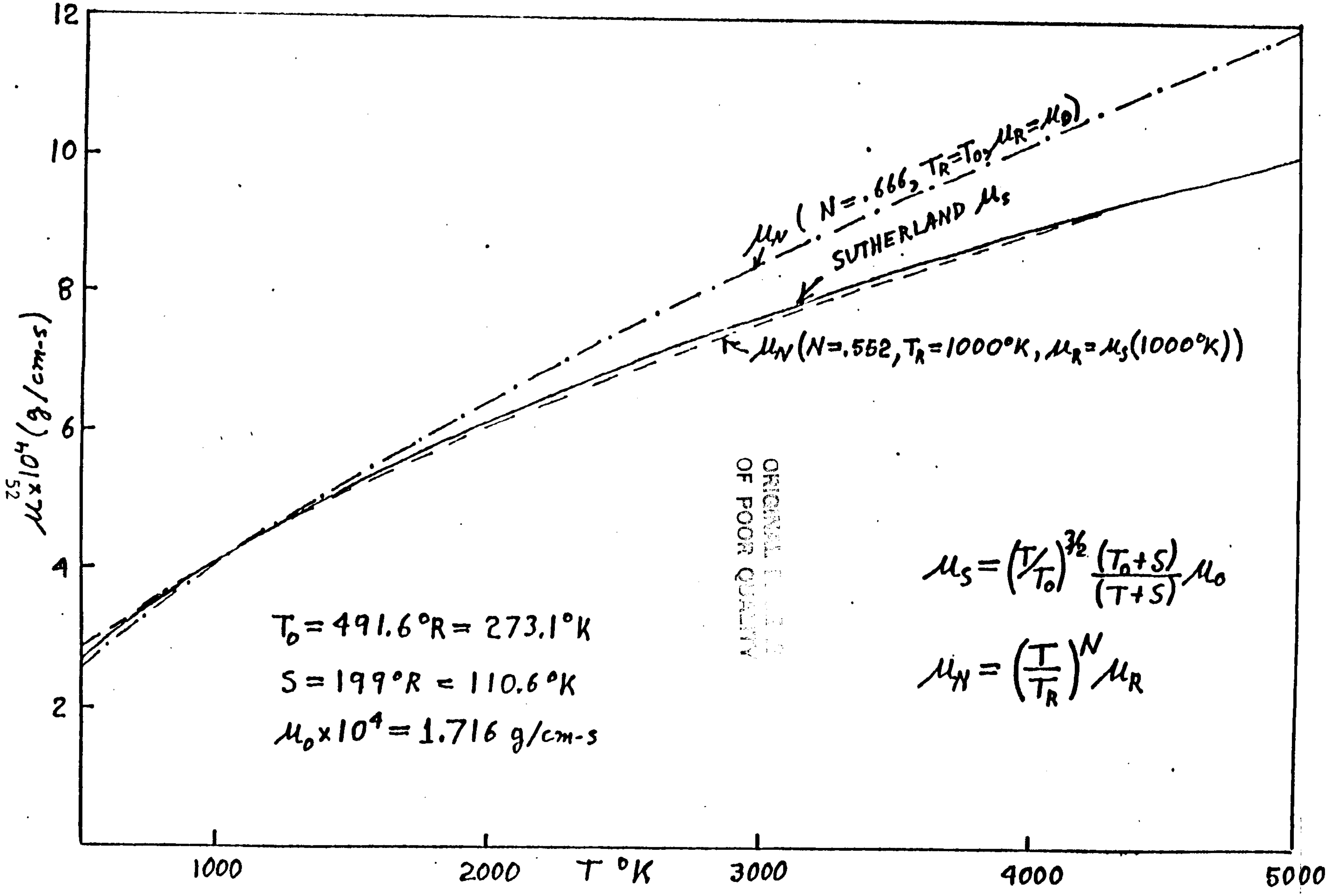
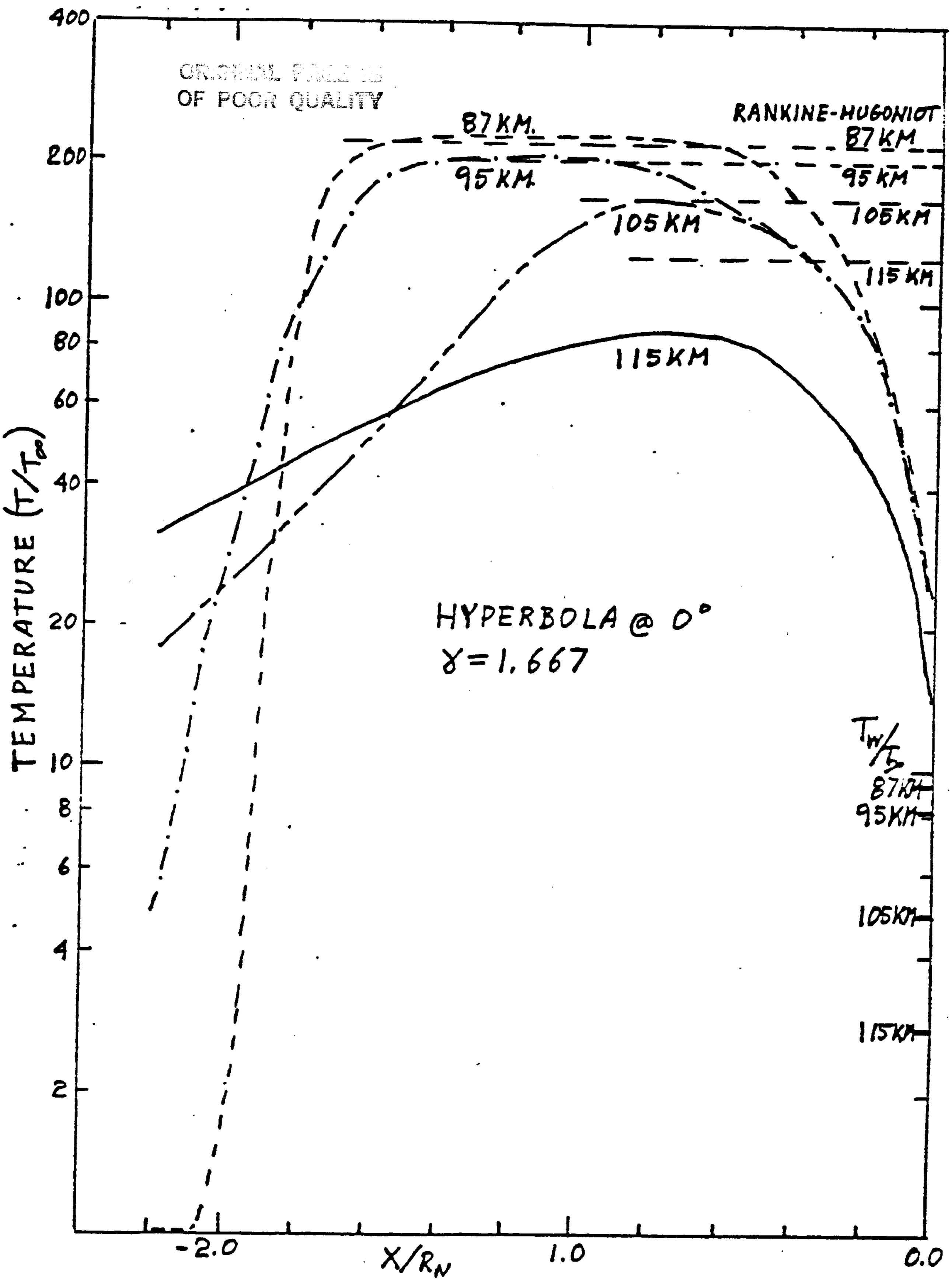


FIGURE 6. TEMPERATURE DISTRIBUTIONS (EFFECT OF ALTITUDE)



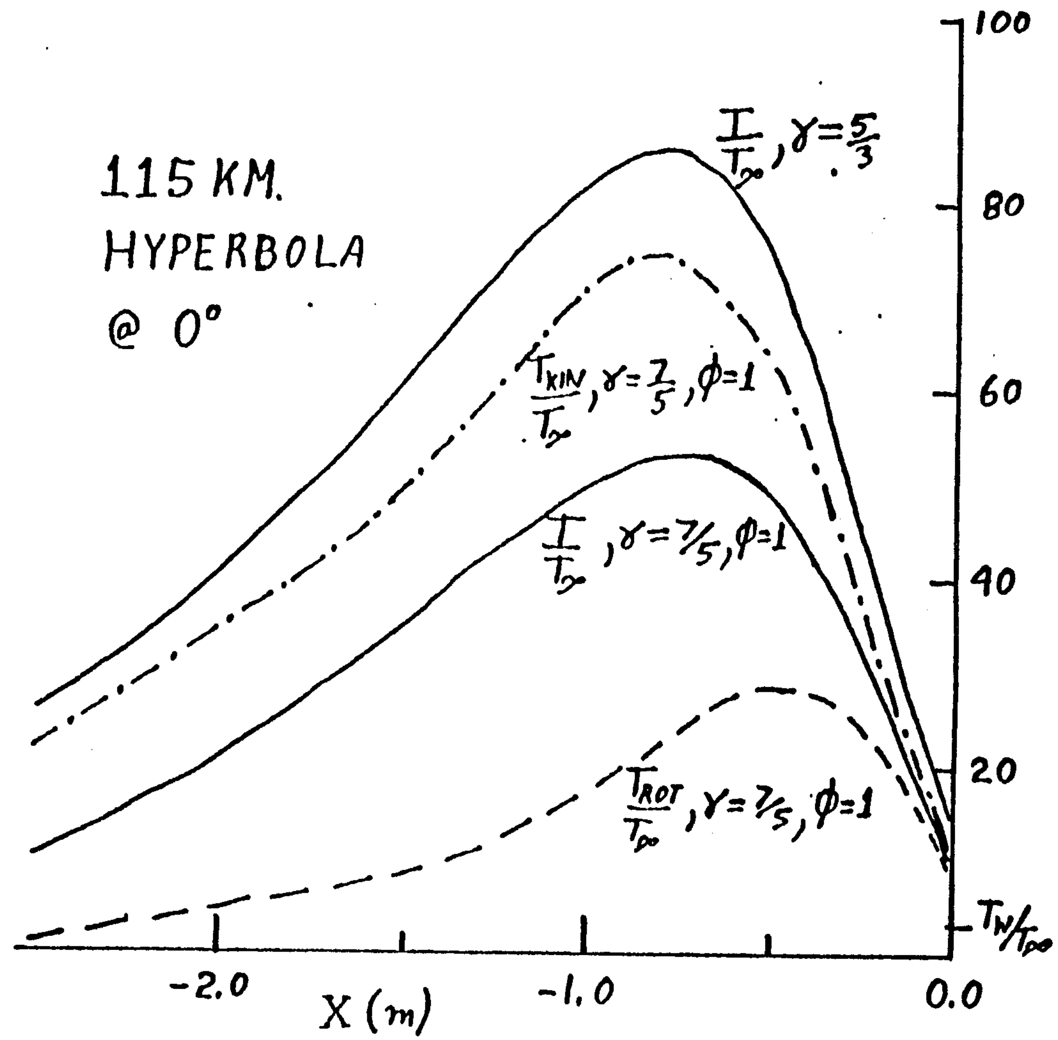
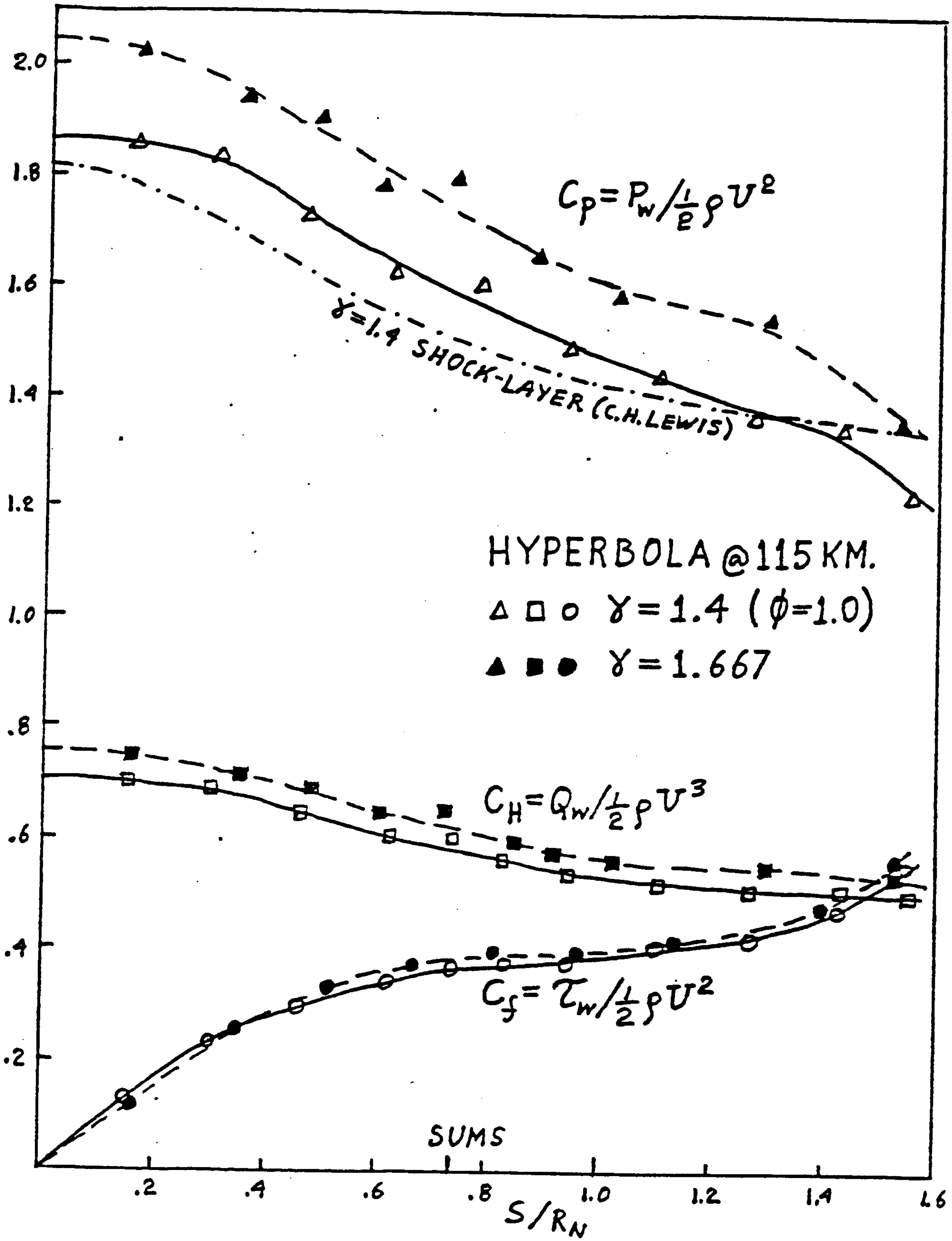


FIGURE 7. TEMPERATURE DISTRIBUTION ALONG STAGNATION STREAMLINE (EFFECT OF γ)

FIGURE 8. SURFACE FLUXES (EFFECT OF γ)



Normalized Distance from Stagnation Point

FIGURE 9. SONIC LINES (EFFECT OF BODY SIZE)

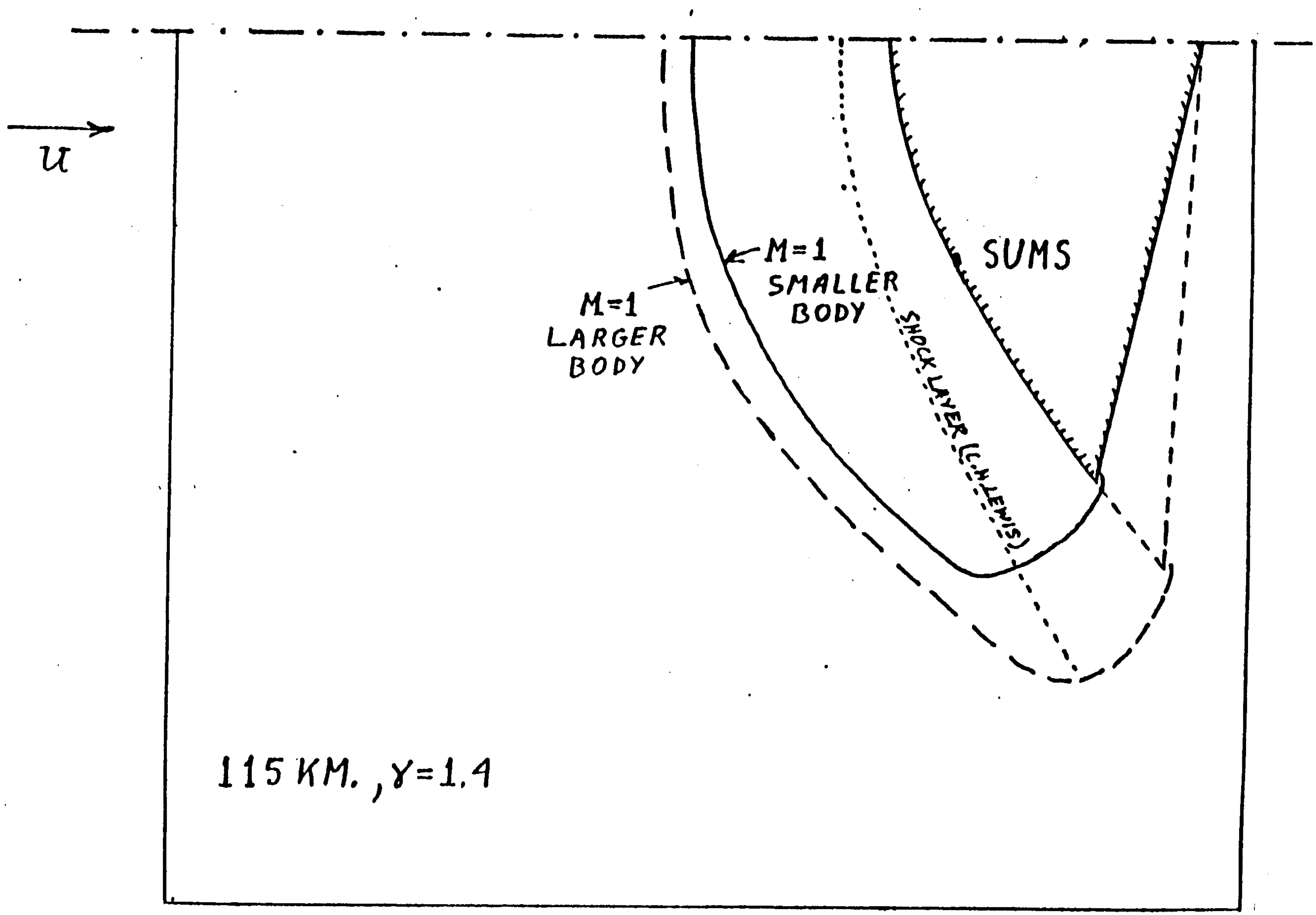


FIGURE 10. SONIC LINES (EFFECT OF GEOMETRIC MODEL)

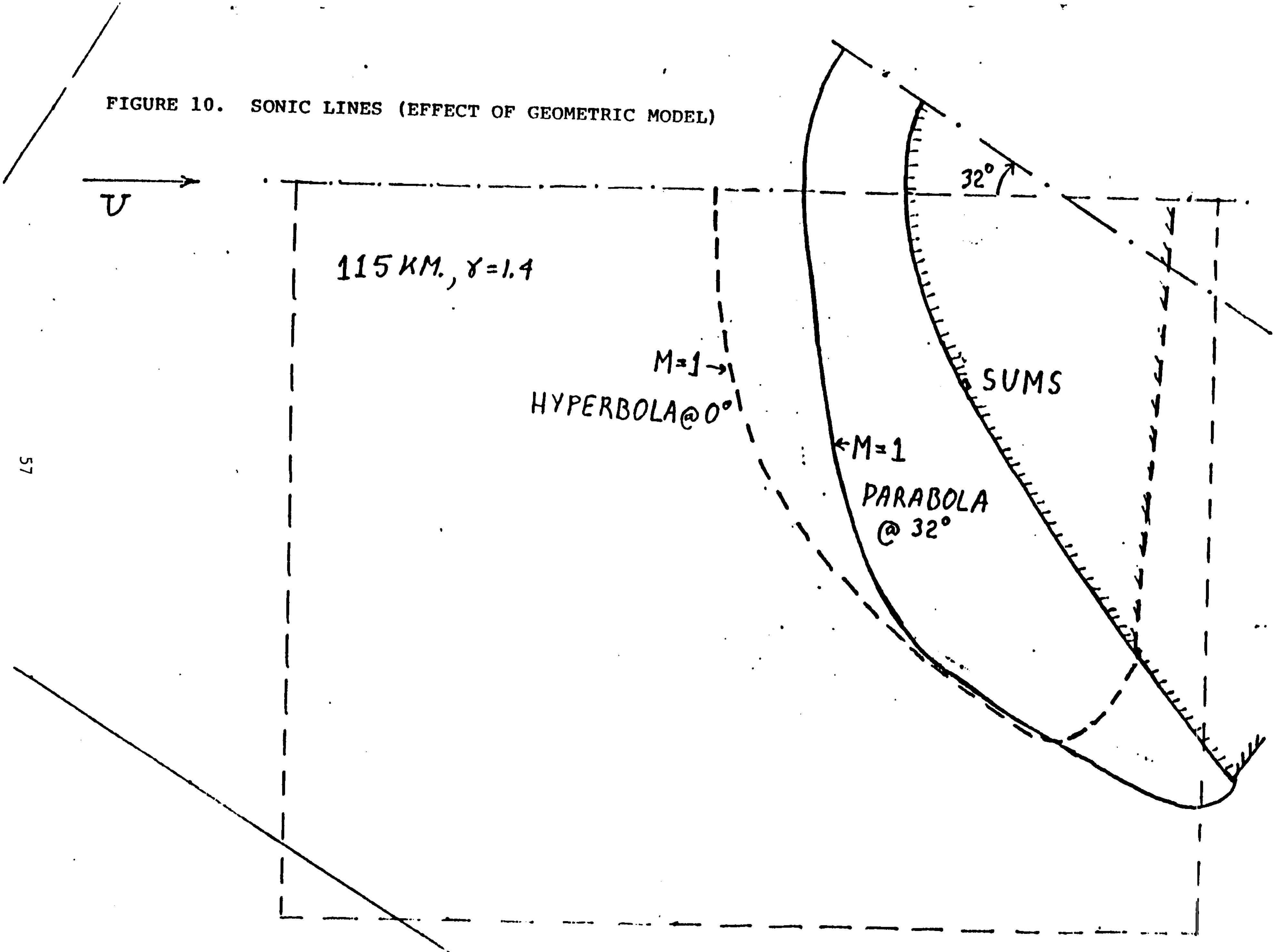
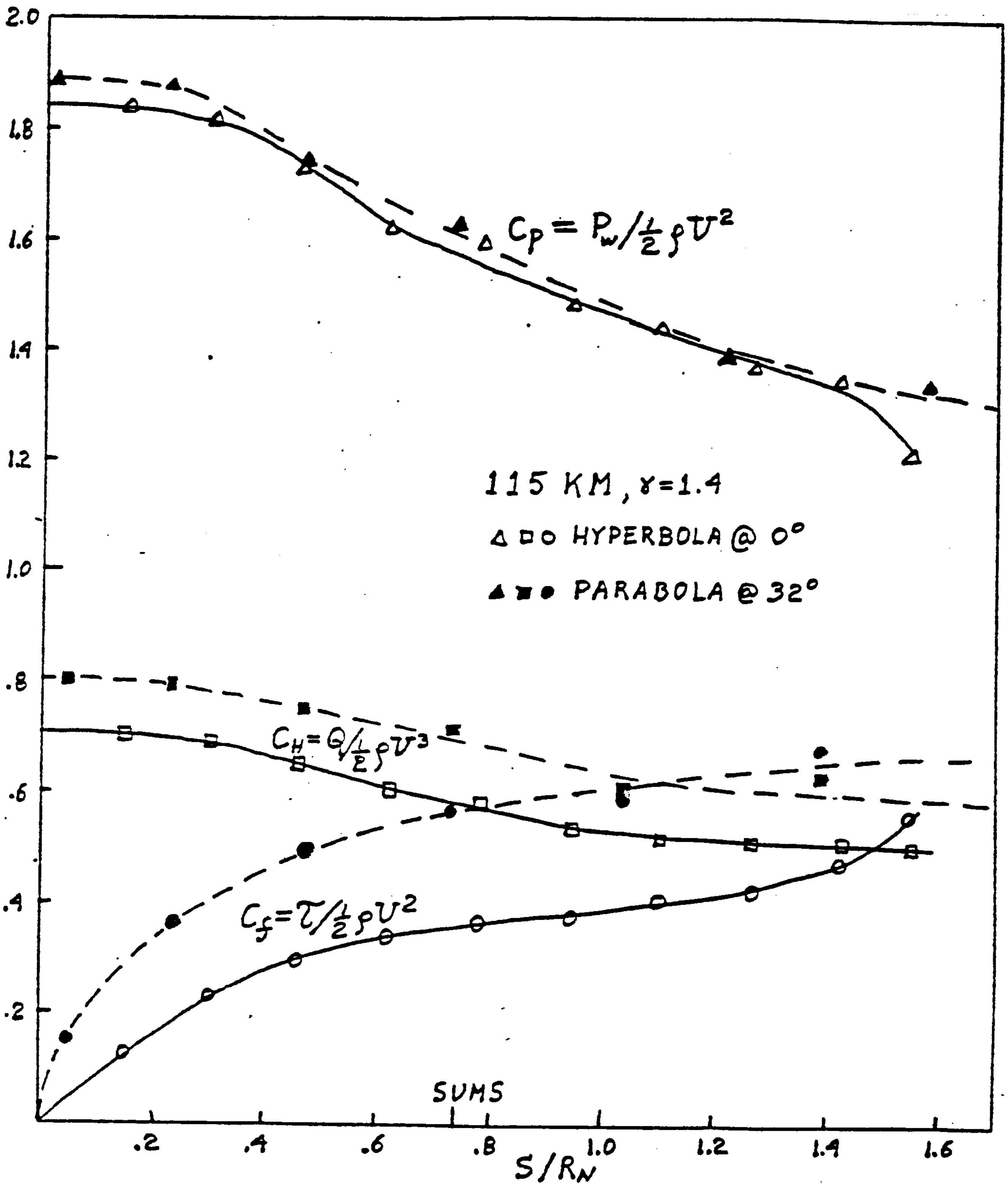


FIGURE 11. SURFACE FLUXES (EFFECT OF GEOMETRIC MODEL)



Normalized Distance from Stagnation Point

ORIGINAL PAGE IS
OF POOR QUALITY

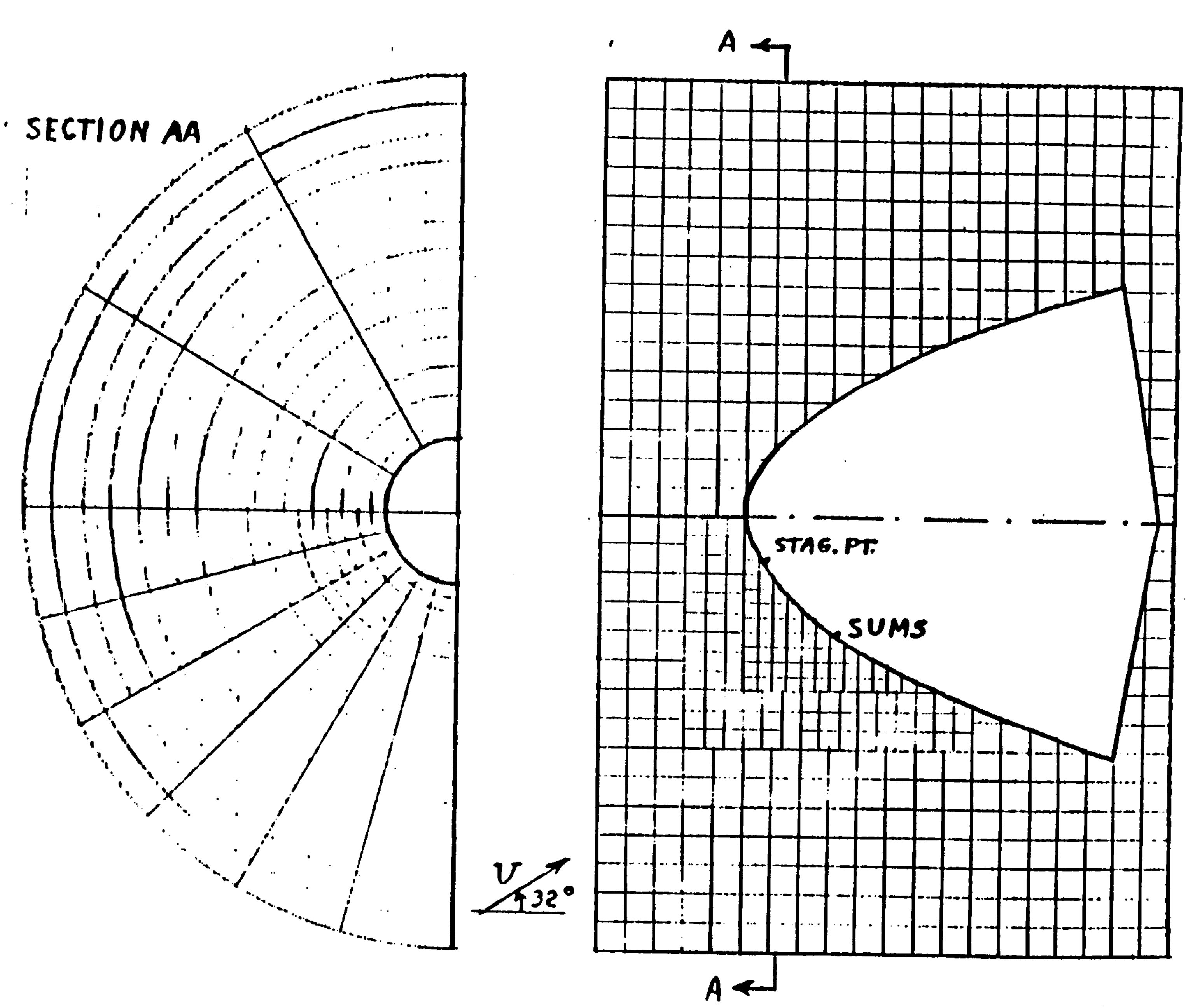


FIGURE 12. TYPICAL CELL CONFIGURATION (95 Km RUN)

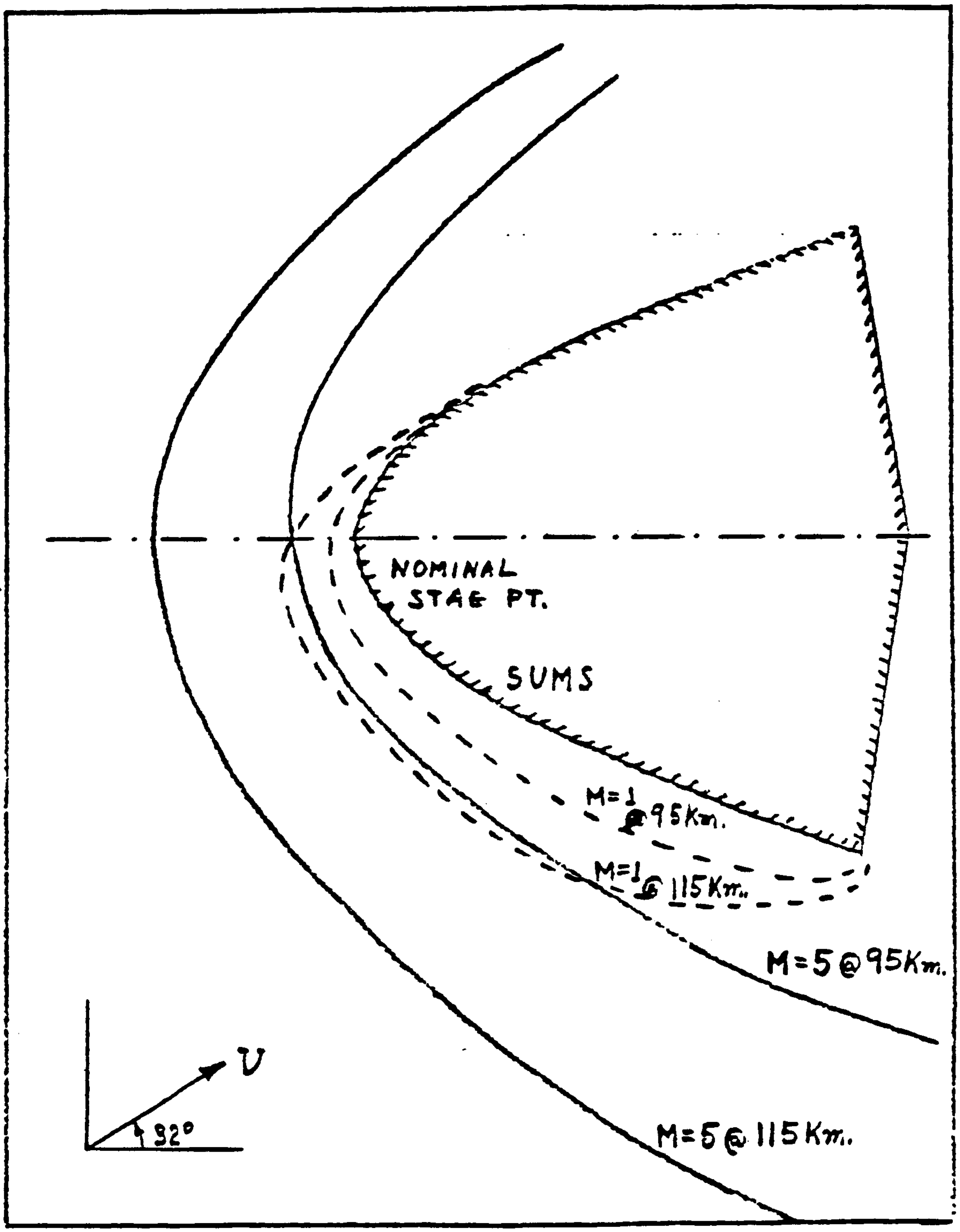


FIGURE 13. MACH NUMBER CONTOURS IN SYMMETRY PLANE AT 95 Km and 115 Km

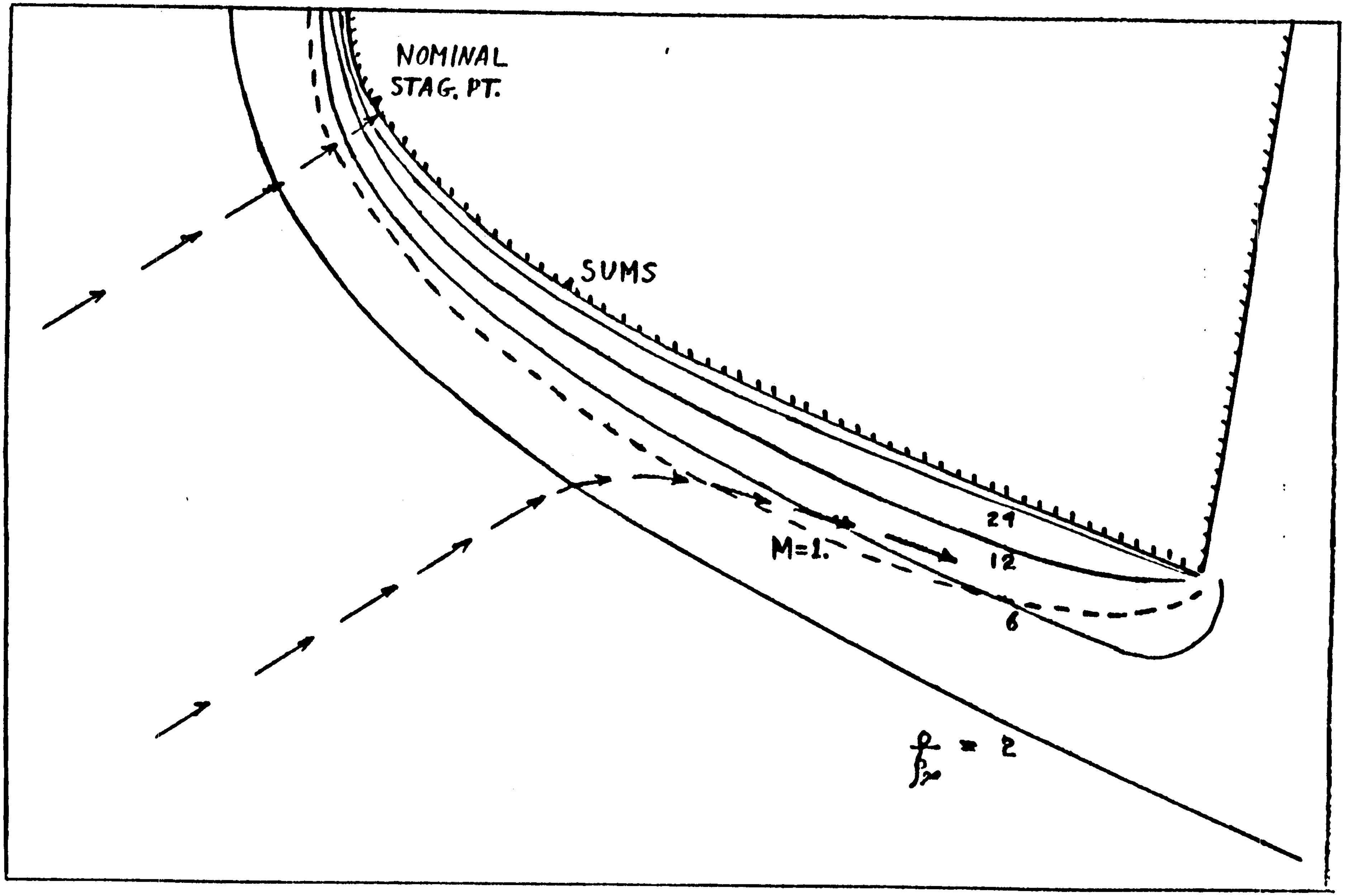
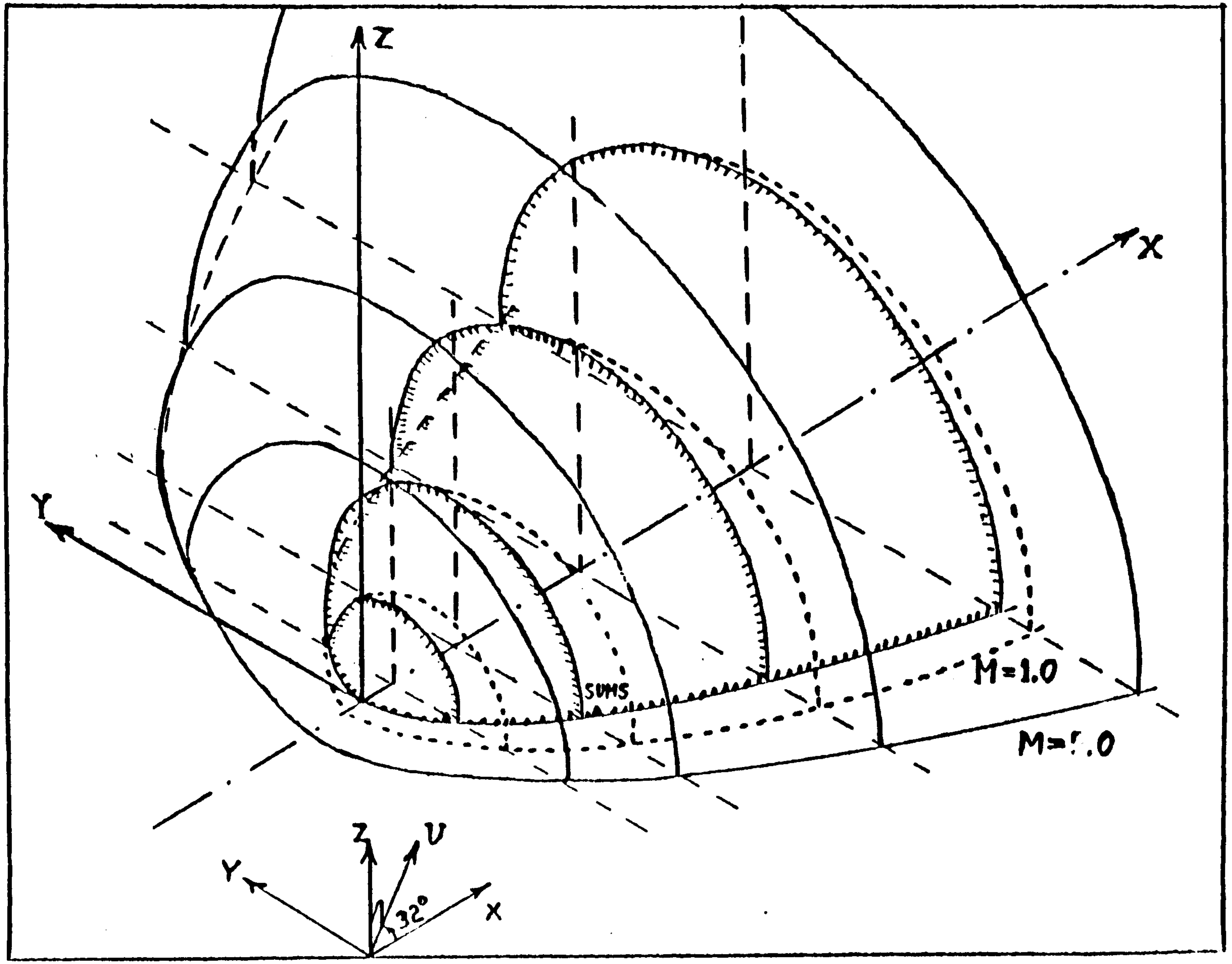


FIGURE 14. DENSITY CONTOURS ON WINDWARD SIDE AT 95 Km ALTITUDE

FIGURE 1[illegible]. THREE DIMENSIONAL SKETCH OF MACH NUMBER CONTOURS
AT 95 Km ALTITUDE



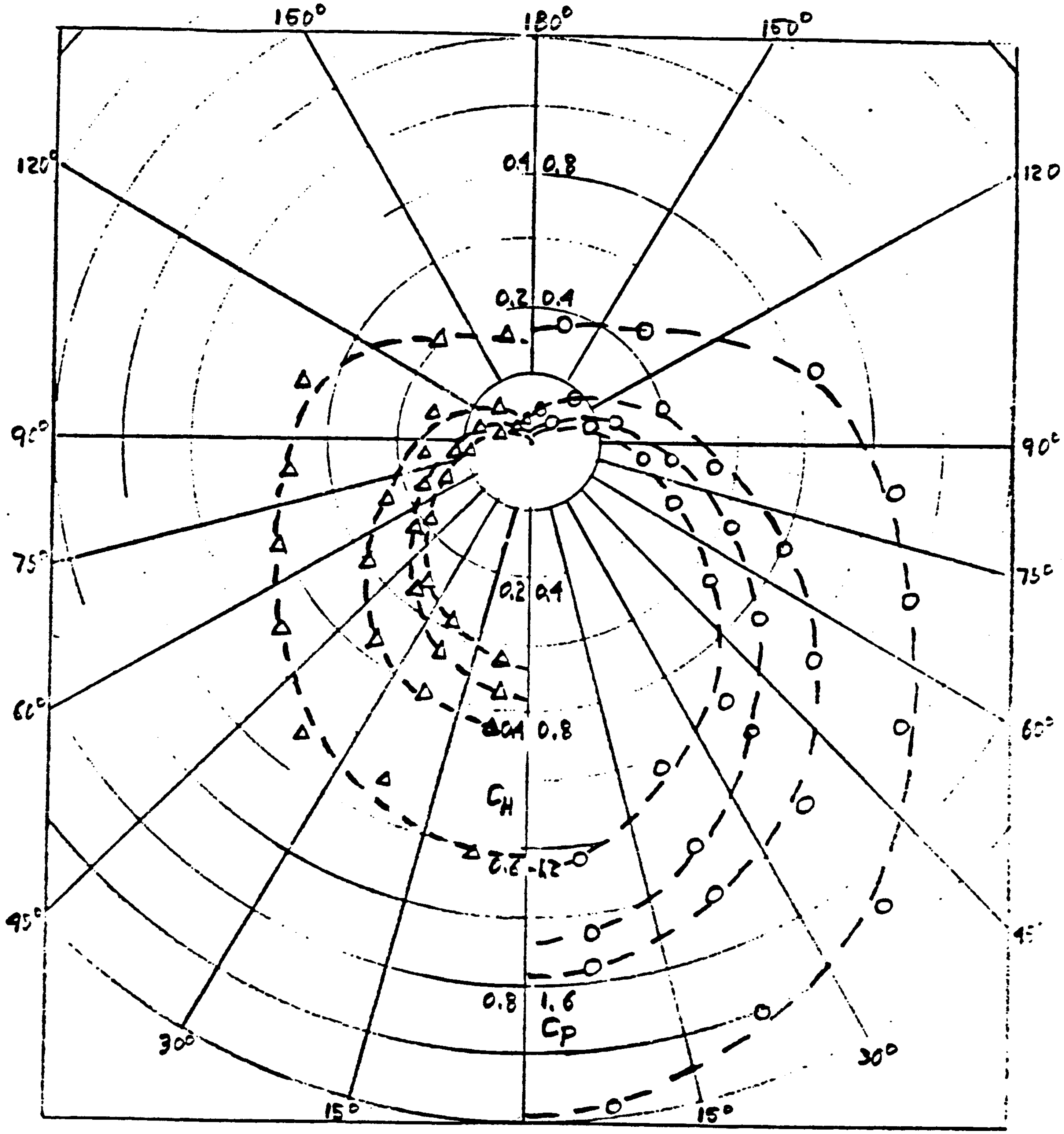


FIGURE 16. SURFACE PROPERTIES; PRESSURE COEFFICIENT $C_p = p_s/1/2\rho U^2$ AND HEAT TRANSFER COEFFICIENT $C_H = Q/1/2\rho U^3$ IN FOUR CROSS-PLANES AT 95 Km ALTITUDE

FIGURE 17. SURFACE PROPERTIES; PRESSURE C_p AND HEAT TRANSFER COEFFICIENT C_H VERSUS ANGLE IN CROSS-PLANE x = 75 cm (NEAR THE SUMS ENTRANCE) AT 95 Km ALTITUDE

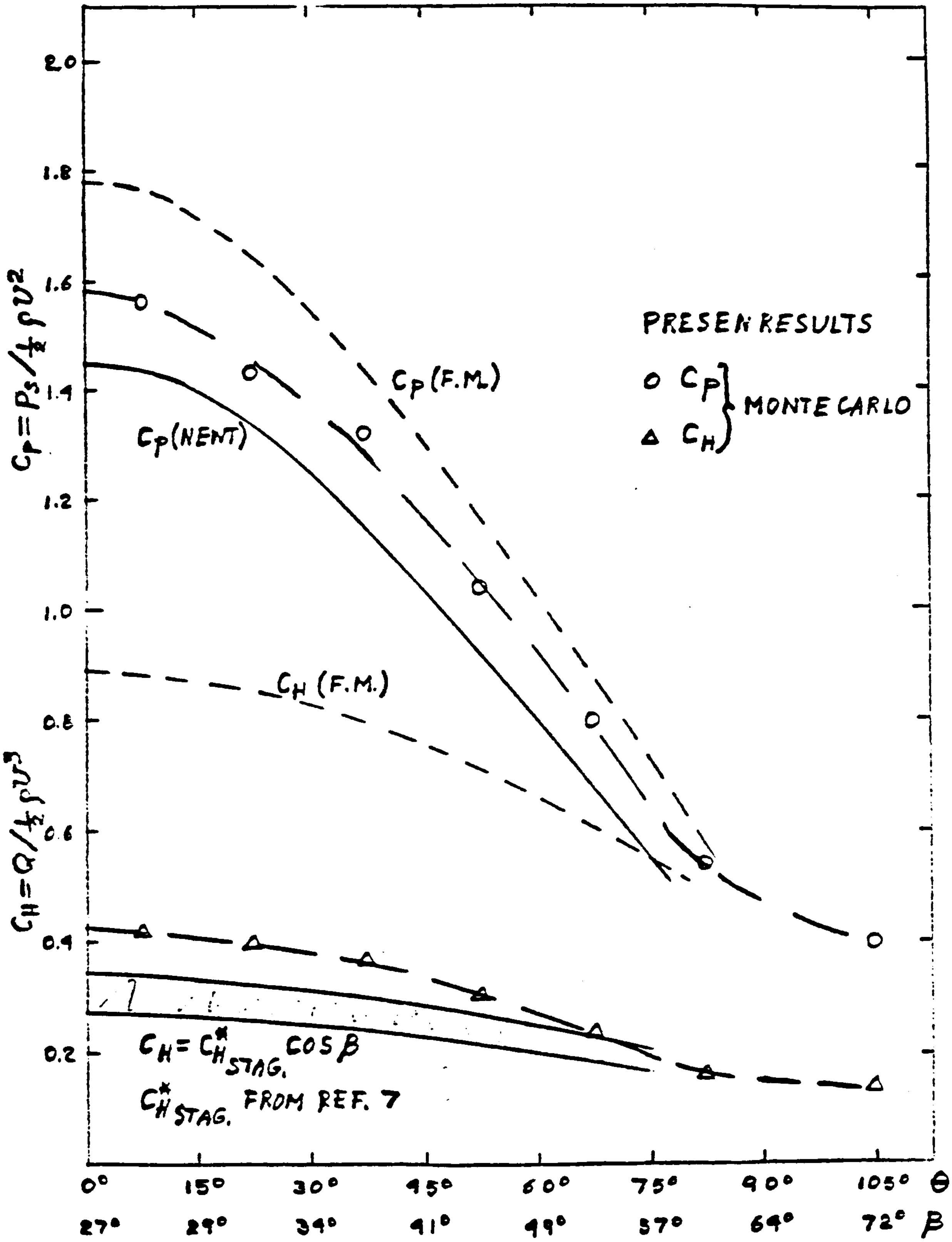


FIGURE 18. FLUX DISTRIBUTION FUNCTION AT SUMS INLET

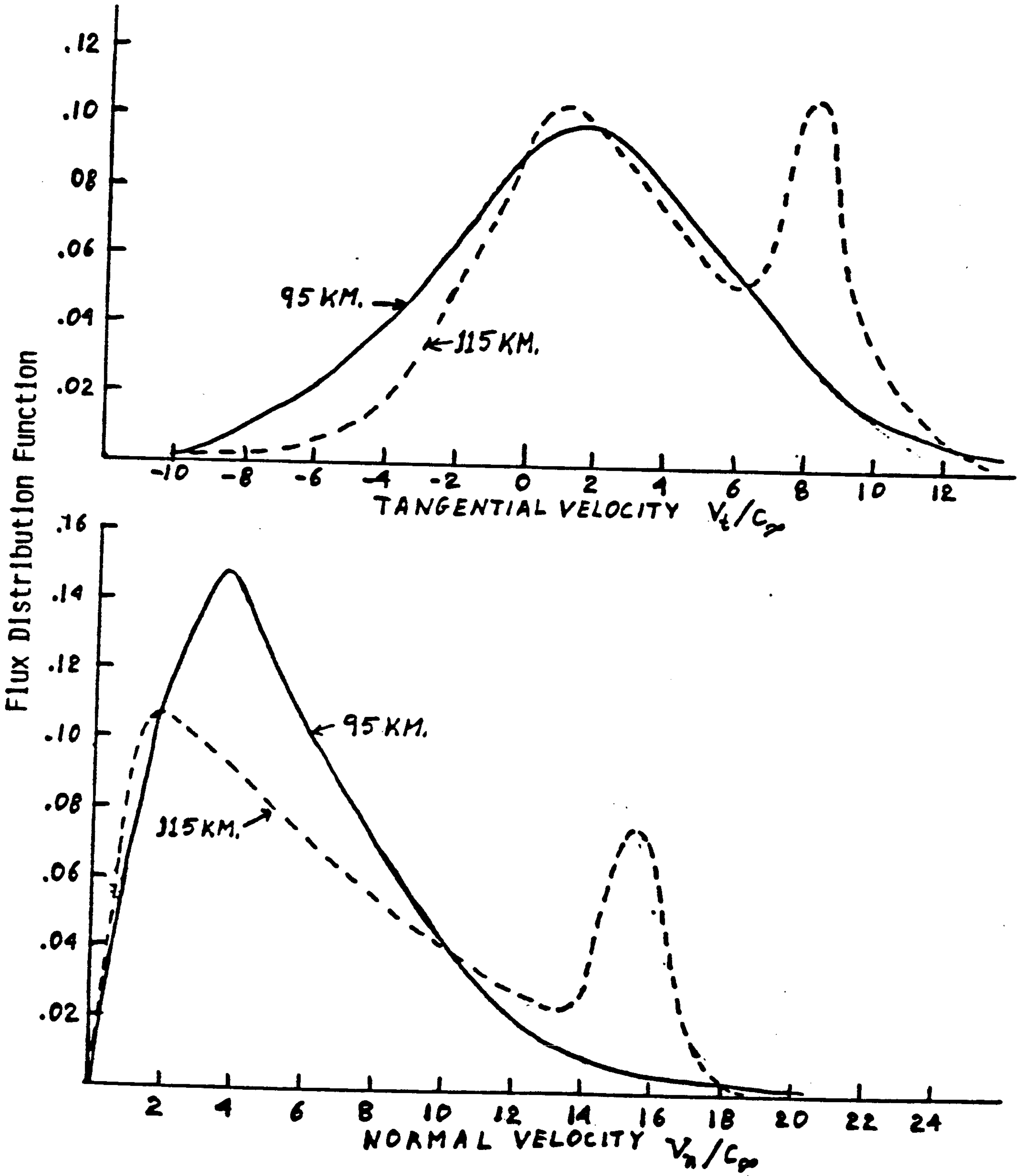


FIGURE 19. SUMS INLET GEOMETRY

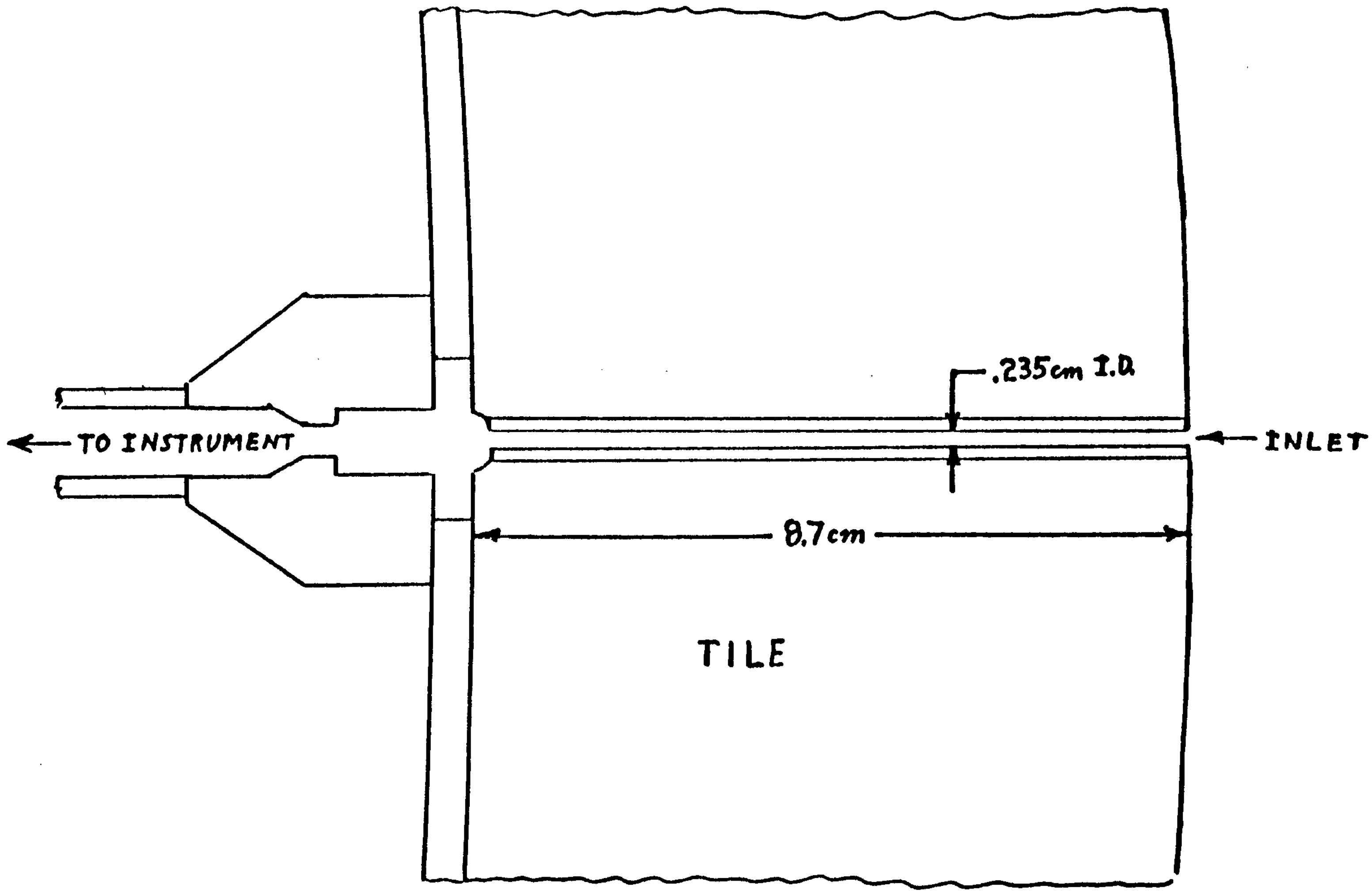


FIGURE 20. FREE MOLECULAR PROBE RESPONSE ($S \to \infty$, $d/L \to 0$)

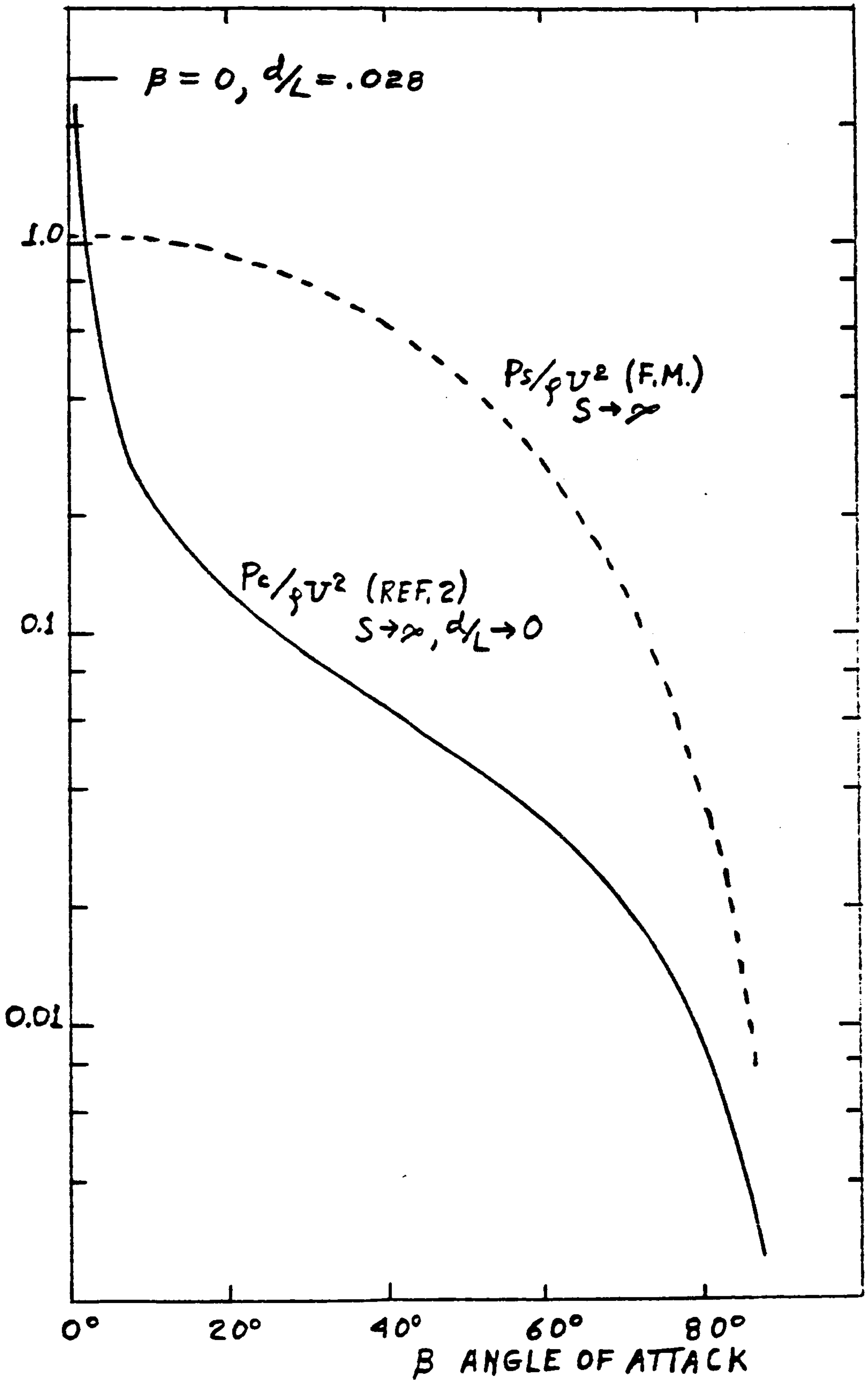


FIGURE 21. PRESSURE CORRECTION FACTOR (p_c/p_s)

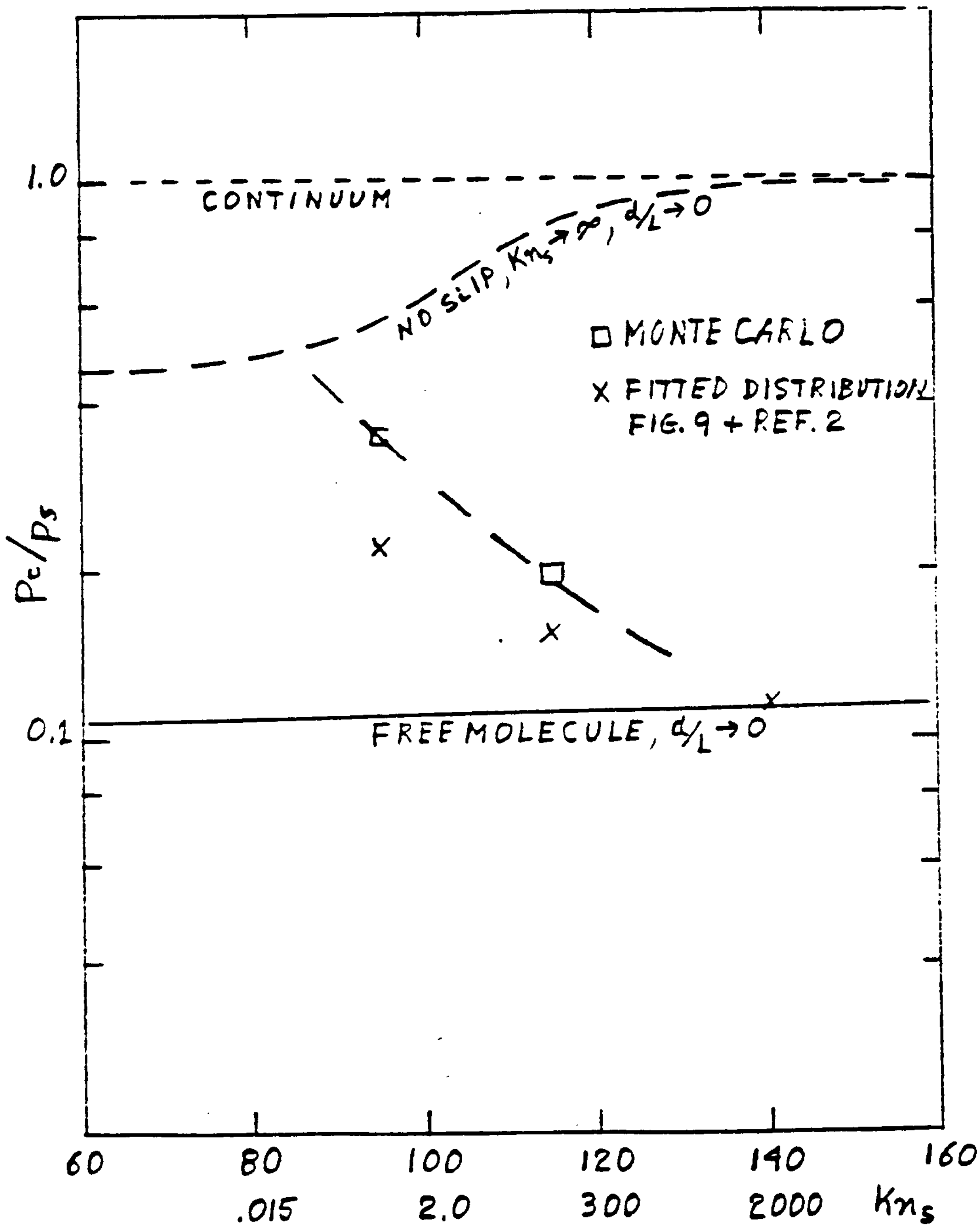


FIGURE 22. SURFACE PRESSURE AT SUMS INLET VERSUS ALTITUDE

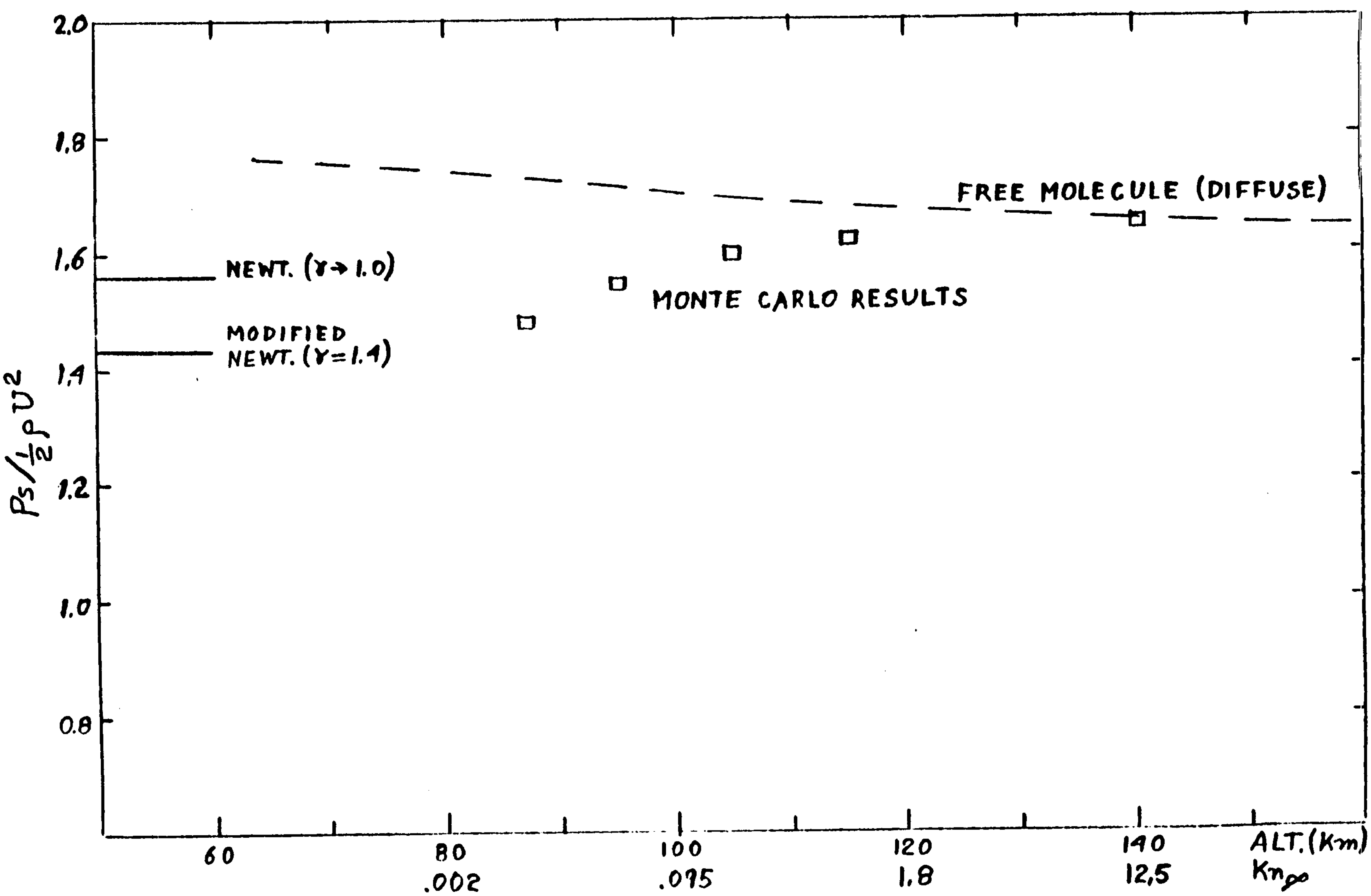
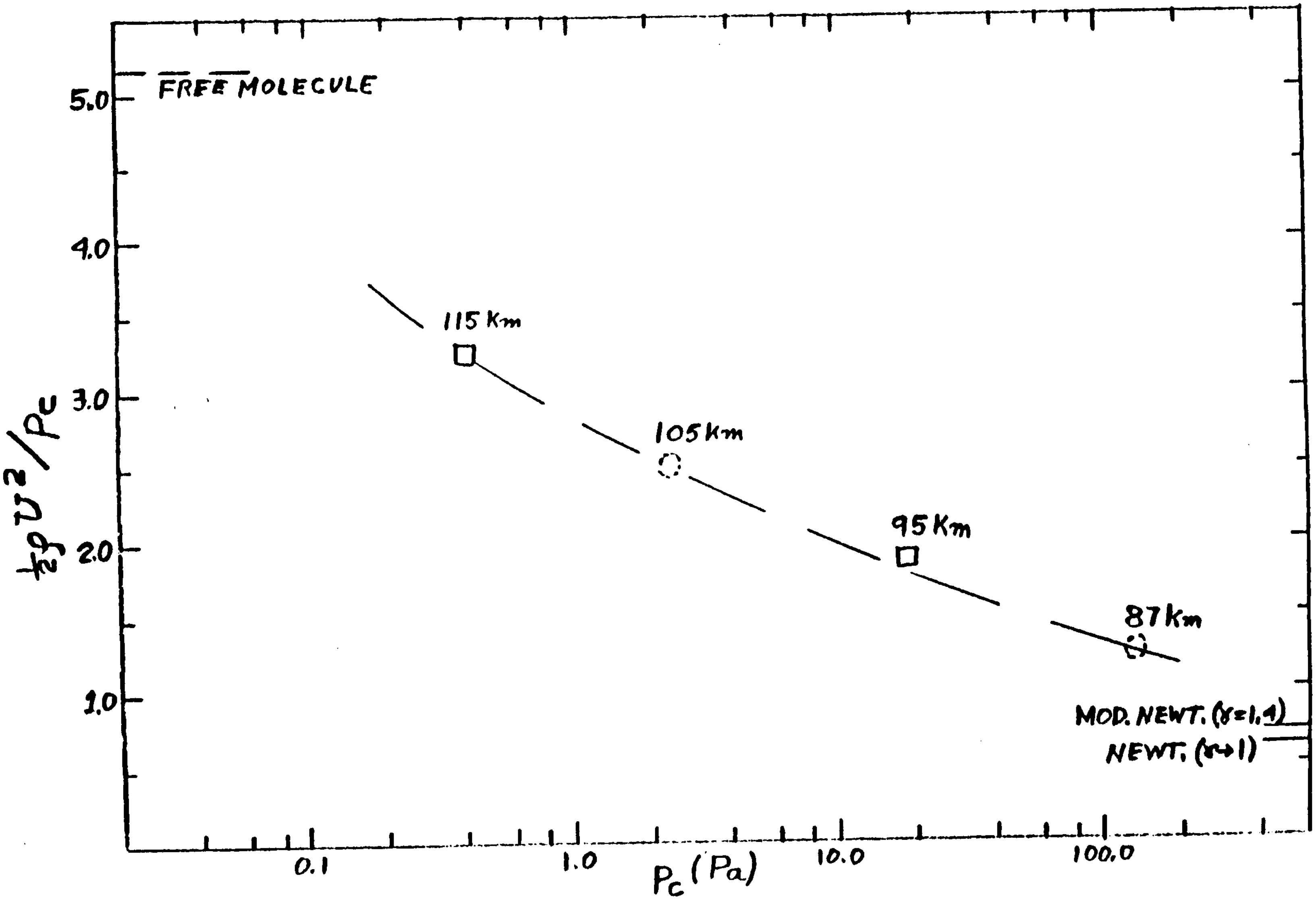


FIGURE 23. PRELIMINARY ESTIMATE OF DATA REDUCTION RELATION q/p_c VERSUS p_c



```
// JOB GKB 0367425.GKBSPACE N=EXTCOMP REG=500 T=.8 P=150 C=0
/*FORMAT PR,DDNAME=SYSMSG,DEST=VM370
// EXEC FORTXCL,PARM.FORT='XREF'
//FORT.SYSIN DD *
C      MAIN PROGRAM FOR MONTE CARLO 3-D EXTERNAL FLOW CALCULATIONS
C     OBJECTIVE OF THIS MAIN PROGRAM IS TO SET THE DIMENSIONS
C     MAIN RUNNING PROGRAM IS  *** RUN ***
C
C     FOLLOWING TWO CARDS HAVE TO BE ELIMINATED FOR NON IBM MACHINES
C*********************************************************************
      INTEGER*2 LB,NBM,NBN,NB,NBP,NBT,NBS,NUMCEL
      INTEGER*2 LM,LPF,LCOL,LKW
C*********************************************************************
C
C     THE NEXT CARD IS ASSOCIATED WITH PRINCETON RANDOM NUMBER GENERATOR
C*********************************************************************
      COMMON/RANCOM/NRAN(4)
C*********************************************************************
C
C     THE FOLLOWING DIMENSION STATEMENTS SET THE MAJOR ARRAY DIMENSIONS
C     AND MUST BE CONSISTENT WITH THE FOLLOWING DATA CARD -----------
C
C     NSP=NUMBER OF SPECIES  - EXAMPLE BELOW NSP=1
C     NMB=NUMBER OF BOXES WITHOUT COUNTING SUBDIVISIONS OR THOSE
C         OCCUPIED BY THE BODY   -  EXAMPLE BELOW NMB=2600
C     NMC=NUMBER OF FINAL CELLS   -  EXAMPLE BELOW NMC=1500
C     NMP=MAX NUMBER OF MOLECULES OF EACH SPECIES ALLOWED IN PROGRAM.
C         IF EXCEEDED, PROGRAM EITHER FAILS OR RESTARTS AT BEGINNING
C         WITH NUMBER REDUCED BY 10%  - EXAMPLE BELOW NMP=20000
C     NPB=MAXIMUM NUMBER IN EACH CELL - EXAMPLE NPB=100
C
      DIMENSION DBA(1,1500),NB(1,1500),NBP(1,1500),NBT(1,1500),LKW(1500)
      DIMENSION TMP(1,1500),TMPA(1,1500),XV(1,1500),XVA(1,1500)
      DIMENSION YV(1,1500),YVA(1,1500),ZV(1,1500),ZVA(1,1500),DR(1,1500)
      DIMENSION TRP(1,1500),TRPA(1,1500),NBS(1,1500),NBM(1,1500)
      DIMENSION NBN(1500),T(1,1,1500)
      DIMENSION LB(20000),LM(1,20000),ER(1,20000)
      DIMENSION LPF(1,20000),PAU(1,20000),PAV(1,20000),PAW(1,20000)
      DIMENSION PAX(1,20000),PAY(1,20000),PAZ(1,20000),LCOL(1,20000)
      DIMENSION PNB(2600),XC(2600),YC(2600),ZC(2600),NUMCEL(2600)
      DATA NSP/1/,NMB/2600/,NMC/1500/,NMP/20000/,NPB/100/
C
C
    2 FORMAT(/17X,'NORMAL TERMINATION OF THE PROGRAM')
      NAMELIST/DIM/NSP,NMB,NMC,NMP,NPB,NRAN
C
C     INITIALIZATION OF RANDOM NUMBER GENERATOR - PRINCETON ROUTINE
C
      NRAN(1)=0
      NRAN(2)=0
      NRAN(3)=0
      NRAN(4)=0
C
C     PRINTOUT OF MAJOR ARRAY DIMENSIONS USED ABOVE
```

ORIGINAL PAGE IS
OF POOR QUALITY


```
      WRITE(6,DIM)
C
C     CALL OF MAIN OPERATING PROGRAM WHICH REQUIRES INPUTS:
C     &CONTRL,&TIMES,&FLOREF,&MOLEC,&SHAPES,&GEOM,&COUPLE
C     THESE INPUTS ARE ALL CURRENTLY IN THE NAMELIST FORMAT
C     AND MAY HAVE TO BE CHANGED IF THAT CONVENTION IS NOT AVAILABLE
C     BRIEF DESCRIPTION OF THE PARAMETERS FOLLOWS
C &CONTRL - ONE OCCURRENCE (NEW OR RESTART)
C PARAMETER  DEFAULT   DEFINITION OR EXPLANATION
C NAME       8 BLANKS  ANY ALPHANUMERIC NAME UP TO 8 CHARACTERS
C TITLE      24 BLNKS  ANY ALPHANUMERIC TITLE UP TO 24 CHARACTERS
C PERCNT     .001      ACCURACY IN INTEGRATION PROCEDURES
C ICOPY      1         NUMBER OF ADDITIONAL COPIES OF OUTPUT
C DUMP       .TRUE.    IF TRUE WILL CAUSE SYSTEM DUMP FOR ANY OF 12
C                      PROGRAMMER DESIGNED ERROR HALTS.
C DEBUG(1)   .FALSE.   IF TRUE WILL PRINT MESSAGE WHEN CELL POP. EXCEEDS MNB
C DEBUG(2)   .FALSE.   IF TRUE WILL PRINT CPU TIME AROUND EACH PART OF LOOP
C DEBUG(3)   .TRUE.    IF TRUE WILL PRINT CPU TIME REMAINING AT END OF LOOP
C NEW        .TRUE.    IF TRUE - NEW RUN - IF FALSE - RESTART OF RUN
C SAVE       .FALSE.   IF TRUE - SNAPSHOT SAVED ON TAPE(9) FOR RESTART
C REDO       .FALSE.   IF TRUE PROGRAM WILL AUTOMATICALLY RESTART WITH 90%
C                      OF TOTAL IF TOTAL CELL POPULATION EXCEEDS MNM
C
C &TIMES  -  ONE OCCURRENCE (NEW OR RESTART)
C PARAMETER  DEFAULT   DEFINITION OR EXPLANATION
C DTM        - - -     REAL NUMBER - FRACTION OF MEAN FREE TIME PER CYCLE
C ITS        - - -     INTEGER - NUMBER OF CYCLES PER SAMPLE
C ITP        - - -     INTEGER - NUMBER OF CYCLES BETWEEN PRINTOUTS
C TST        - - -     INTEGER - ESTIMATE OF NUMBER OF CYCLES TO STEADY STATE
C TLIM       - - -     INTEGER - TOTAL NUMBER OF CYCLES TO END OF RUN -
C                      WILL TERMINATE SOONER IF CPU TIME IS TO BE EXCCEDED
C
C &FLOREF  -  ONE OCCURRENCE (NEW RUN ONLY)
C PARAMETER  DEFAULT   DEFINITION
C INM        - - -     INITIAL NUMBER OF MOLECULES  INM<MNM< OR = NMP
C MNM        - - -     MAXIMUM NUMBER OF MOLECULES PER SPECIES
C MNB        - - -     MAXIMUM NUMBER PER CELL - DIAGNOSTIC ONLY
C MSP        - - -     NUMBER OF MOLECULAR SPECIES  (MAX. IS 3)
C MET        0         IF 0 - DATA IS IN SI (METRIC) UNITS
C                      IF>0 - DATA IS IN ENGLISH UNITS
C U          - - -     FLOW VELOCITY (M/SEC) OR (FT/SEC)
C ANGLE      - - -     ANGLE OF ATTACK (DEGREES)
C RNU        0.0       ARRAY GIVING MOLE FRACTIONS OF SPECIES IN FREE STREAM
C RMA        0.0       ARRAY GIVING MOLECULAR WEIGHTS OF SPECIES ABOVE
C TF         - - -     FREE STREAM TEMPERATURE (K OR R)
C DENF       - - -     FREE STREAM NUMBER DENSITY (NUM/M**3 OR NUM/FT**3)
C
C &MOLEC  -  ONE OCCURRENCE (NEW RUN ONLY)
C PARAMETER  DEFAULT   DEFINITION
C TRE        - - -     REFERENCE TEMPERATURE FOR MOLECULAR DATA
C DIR        0.0       CROSS-SECTIONS AT REFERENCE TEMP. (MSPXMSP)
C ETA        0.0       PARAMETERS IN DIFFUSION AND VISCOSITY LAW (MSPXMSP)
C PHI        0.0       PARAMETERS FOR ROTATIONAL RELAXATION (MSPXMSP)
C CHI        0.0       ROTATIONAL DEGREE OF FREEDOM PARAMETER (NROT/2 - 1)
C ACR        .001      ACCURACY IN MOLECULAR COLLISION CALCULATIONS
```

```
:
: &SHAPES  -  ND+1 OCCURRENCES WHERE ND=NUMBER OF BODY SEGMENTS (NEW RUN)
: PARAMETER  DEFAULT   DEFINITION
:              FIRST OCCURRENCE
: BODY(1)     0.0       STARTING POINT OF BODY FROM FRONT OF CELLS (M OR FT)
: BODY(I)  I>1 - -      NEED NOT BE SPECIFIED
:              SUBSEQUENT OCCURRENCES (ND)
: BODY(1)     - - -     X COORDINATE FROM FRONT OF BODY OF THE DOWNSTREAM
:                       EDGE OF THE CURRENT BODY SEGMENT
: BODY(2)     - - -     TEMPERATURE OF THIS BODY SEGMENT
: BODY(3)     - - -     SURFACE AREA/TOTAL CROSS-SECTIONAL AREA FOR SEGMENT.
:                       IF 0.0 PROGRAM WILL COMPUTE THIS QUANTITY
: BODY(4)     - - -     SWITCH - IF 0.0 THIS SEGMENT'S EQ. WILL APPEAR LATER
:                       IF >0.0 THE EQ. OF THIS AND PRECEEDING SEGMENTS IS
:                       GIVEN BY BODY(6+2*MSP) TO BODY(9+2*MSP)
: BODY(5)     - - -     SWITCH - IF NOT 0.0 THIS IS THE LAST SHAPES CARD
: BODY(I)  I EVEN       ALPHA - ENERGY ACCOMODATION COEFFICIENT FOR SPECIES
: BODY(J)  J ODD        SIGMA - TANGENTIAL ACCOMODATION COEFF. FOR SPECIES
:          I AND J < (6+2*MSP)
: BODY(6+2*MSP)         ORIGIN OF COORDINATES WITH RESPECT TO BODY START
:                       FOR THE EQUATION OF THIS BODY SECTION
: BODY(7+2*MSP) -       COEFFICIENTS A,B,C IN THE EQUATION
:  BODY(9+2*MSP)        R**2+A*X**2+B*X+C=0.0 FOR THIS BODY SECTION
:
: &GEOM  - ONE OCCURRENCE  (NEW RUN ONLY)
: PARAMETER  DEFAULT   DEFINITION
: NWEDGE      - - -     TWO INTEGERS GIVING THE NUMBER OF WEDGES BELOW
:                       AND ABOVE THE ANGLE THETAZ
: THETAZ      - - -     ANGLE FROM NEG. Y AXIS DIVIDING DIFF. WEDGE SIZES
: RMB         - - -     MAX. BODY RADIUS - IF 0. WILL BE COMPUTED BY PROGRAM
: BW          - - -     WIDTH (DEL X) OF FIRST LEVEL CELLS (M OR FT)
: BH          - - -     HEIGHT (DEL R) OF FIRST LEVEL CELLS (M OR FT)
: NW          - - -     NUMBER OF FIRST LEVEL CELLS IN X DIRECTION
: NH          - - -     NUMBER OF FIRST LEVEL CELLS IN RADIAL DIRECTION
: NL            1       NUMBER OF LEVELS OF CELLS
: NFA           0       NUMBER OF FIRST LEVEL CELLS AHEAD OF LEVEL 2
: NCA           0       NUMBER OF FIRST LEVEL CELLS SUBDIVIDED INTO SECOND
:                       LEVEL CELLS ALONG THE X DIRECTION
: NHA           0       AS ABOVE BUT IN RADIAL DIRECTION
: MW            0       NUMBER OF SECOND LEVEL CELLS IN THE X DIPECTION
: MH            0       NUMBER OF SECOND LEVEL CELLS IN THE RADIAL DIRECTION
: NFB           0       NUMBER OF SECOND LEVEL CELLS AHEAD OF LEVEL 3
: NCB           0       NUMBER OF SECOND LEVEL CELLS SUBDIVIDED INTO THIRD
:                       LEVEL CELLS ALONG THE X DIRECTION
: NHB           0       AS ABOVE BUT IN RADIAL DIRECTION
: LW            0       NUMBER OF THIRD LEVEL CELLS IN X DIRECTION
: LH            0       NUMBER OF THIRD LEVEL CELLS IN RADIAL DIRECTION
: LD          - - -     NUMBER OF FIRST LEVEL CELLS FROM AXIS IN R DIRECTION
:                       FOR WEIGHTING FACTOR BOUNDARIES (5 INTEGERS)
: LF          - - -     WEIGHTING FACTOR RATIOS AT BOUNDARIES ABOVE (5 INTEGERS)
:
: &COUPLE  - ONE OCCURRENCE  (NEW RUN ONLY) - DISTRIBUTION FUNCTION
: PARAMETER  DEFAULT   DEFINITION
: NS          - - -     THE NUMBER OF BODY SEGMENTS FOR ACCUMULATING VELOCITY
:                       DISTRIBUTION FUNCTION INFORMATION
```

```
C MS           - - -      ARRAY(NS) OF AXIAL SEGMENT NUMBERS
C IWS          - - -      ARRAY(NS) OF AZIMUTHAL WEDGE NUMBERS
C VEL         3.,3.,4.    THERMAL VELOCITY SPREAD FOR THE UNCOLLIDED MOLECULES
C MJ             20       NUMBER OF DEL V REGIONS FOR SAMPLING VELOCITY SPACE
C SL           - - -      ARRAY GIVING LOWER BOUND ON THE VELOCITY SAMPLE OF
C                         COLLIDED MOLECULES (NSP X NS X 3)
C DELS         - - -      THE RANGE (SL<V<SL+DELS) FOR SAMPLE OF COLLIDED
C                         MOLECULES  (NSP X NS X 3)
C     IF DISTRIBUTION FUNCTION INFORMATION IS NOT DESIRED USE:
C &COUPLE NS=0 &END
C
C
C
C     A SAMPLE INPUT DECK IS GIVEN BELOW:
C
C&CONTRL NAME='SHUT','TLE ',TITLE='HYPE','RBOL','A AT',' 95K','M  M','ON. ',
C  DEBUG=.F.,.T.,.T.,NEW=.T.,SAVE=.T.,ICOPY=0,REDO=.T. &END
C&TIMES DTM=.025,ITS=5,ITP=1000,TST=400,TLIM=1000 &END
C&FLOREF INM=4500,MNM=20000,MNB=100,MSP=1,MET=0,U=7485.9,ANGLE=0.0,RNU=1.,2*0.,
C  RMA=28.94,0.,0.,TP=195.51,DENP=2.52E+19 &END
C&MOLEC TRF=1000,DIF=3.5E-19,ETA=.104,PHI=0.0,CHI=-1.,ACR=.001 &END
C&SHAPES BODY=1.00 &END
C&SHAPES BODY=.0173,1590.,3*0.0,2*1.0 &END
C&SHAPES BODY=.0672,1590.,3*0.0,2*1.0 &END
C&SHAPES BODY=.1444,1590.,3*0.0,2*1.0 &END
C&SHAPES BODY=.2432,1590.,3*0.0,2*1.0 &END
C&SHAPES BODY=.3579,1590.,3*0.0,2*1.0 &END
C&SHAPES BODY=.4842,1590.,3*0.0,2*1.0 &END
C&SHAPES BODY=.6192,1590.,3*0.0,2*1.0 &END
C&SHAPES BODY=.7405,1590.,3*0.0,2*1.0 &END
C&SHAPES BODY=.7500,1590.,0.,1.,0.,2*1.,0.,-1.423278,-2.286,0. &END
C&SHAPES BODY=.9000,1590.,0.0,4*1.0,0.9000,-111.82,2*0.0 &END
C&GEOM NWEDGE=1,0,THETAZ=180.,RMB=0.0,BW=.05,BH=.1,NW=40,NH=24,NL=2,
C  NPA=15,NCA=20,NHA=16,MW=40,MH=32,LD=1,2,4,8,12,LP=3,2,2,2,1 &END
C&COUPLE NS=3,MS=1,5,8,IWS=1,1,1,VEL=2.5,2.5,2.0,MJ=20,SL=9*1.,18*-9.,
C  DELS=27*20. &END
C
C
      CALL RUN(NSP,NMB,NMC,NMP,NPB,DBA,NB,NBF,NBT,TMP,TMPA,XV,XVA,YV,
     1YVA,ZV,ZVA,T,DB,FNB,XC,YC,ZC,NUMCEL,LM,LPF,PAU,PAV,PAW,PAX,PAY,
     2PAZ,LCOL,TRP,TRPA,ER,LKW,NBS,LB,NBM,NBN)
      WRITE(6,2)
      STOP
      END

      SUBROUTINE RUN(NSP,NMB,NMC,NMP,NPB,DBA,NB,NBF,NBT,TMP,TMPA,XV,XVA,
     1 YV,YVA,ZV,ZVA,T,DB,FNB,XC,YC,ZC,NUMCEL,LM,LPF,PAU,PAV,PAW,PAX,
     2PAY,PAZ,LCOL,TRP,TRPA,ER,LKW,NBS,LB,NBM,NBN)
C     MAIN RUNNING PROGRAM ** RUN ***  CALLS ALL OTHER SUROUTINES
C**********************************************************************
C
      INTEGER*2 LM,LPF,LCOL,LKW
      INTEGER*2 LB,NBM,NBN,NB,NBF,NBT,NBS,NUMCEL
      INTEGER PRT,SAMP,TST,TLIM,TIME,Q
      LOGICAL DUMP,DEBUG(3),SAVE,NEW,REDO
```

```
      REAL INTGRL,LAM,MU,NU,JAY,KAY
      DIMENSION BTA(3),C1(3),C2(3),C3(3),C7(3),C8(3),DPA(3),PL(3)
      DIMENSION DELANG(2),PDN(3),HTI(3),HTR(3),JNT(3),KNM(3),NM(3),SR(3)
      DIMENSION NAME(2),TITLE(6),NWEDGE(2),LD(5),LP(5),LWP(6),RLD(6)
      DIMENSION RNU(3),RMA(3),WTM(3), CHI(3), DIR(3,3),DAM(3,3),PHI(3,3)
      DIMENSION ETA(3,3), CN8(3,3), CNG(3),CMG(3), CN(3,3,3),CM(3,3,3)
      DIMENSION CTI(3,3),CTR(3,3),CNI(3,3),CNR(3,3),LEV(3),SN(3),ST(3)
      DIMENSION D1(3),D2(3),D3(3),D4(3),BODY(15),DBG1(3,3),LIMIT(10)
      DIMENSION VL(3,3),DELV(3,3),SSA(2,3),SSB(2,3),VEL(3),COEFF(4,9)
      DIMENSION XLIM(12),NTCF(3,3),MS(3),IWS(3),TANGN(3),NCOL(3,3)
      DIMENSION FV(3,3,2,20,3),NTCV(3,3,2,20,3),SL(3,3,3),DELS(3,3,3)
      DIMENSION XCB(18),XS(18),YCB(18),TB(18),ALPHA(3,18),SIGMA(3,18)
      DIMENSION NTS(3,18,12),NTSF(3,18,12),UTL(3,18,12),UTT(3,18,12)
      DIMENSION VTS(3,18,12),HTSI(3,18,12),HTS(3,18,12)
      DIMENSION UTLI(3,18,12),UTTI(3,18,12),VTSI(3,18,12)
      DIMENSION ENT(2,3,6,12),REM(2,3,6,12),ENTS(3,12),REMS(3,12)
      DIMENSION FTH(3,12),THETA(12),DTH(12)
      DIMENSION LB(NMP),NBN(NMC),NBM(NSP,NMC),LM(NSP,NMP)
      DIMENSION ER(NSP,NMP),TRP(NSP,NMC),TRPA(NSP,NMC)
      DIMENSION DBA(NSP,NMC),NB(NSP,NMC),NBF(NSP,NMC),NBT(NSP,NMC)
      DIMENSION TMP(NSP,NMC),TMPA(NSP,NMC),XV(NSP,NMC),XVA(NSP,NMC)
      DIMENSION YV(NSP,NMC),YVA(NSP,NMC),ZV(NSP,NMC),ZVA(NSP,NMC)
      DIMENSION T(NSP,NSP,NMC),DB(NSP,NMC),LKW(NMC),NBS(NSP,NMC)
      DIMENSION FNB(NMB),XC(NMB),YC(NMB),ZC(NMB),NUMCEL(NMB)
                                                                        RUN0360
      DIMENSION LPF(NSP,NMP),PAU(NSP,NMP),PAV(NSP,NMP),PAW(NSP,NMP)
      DIMENSION PAX(NSP,NMP),PAY(NSP,NMP),PAZ(NSP,NMP),LCOL(NSP,NMP)
                                                                        RUN0440
      EXTERNAL FNCTN,FNCTB                                              RUN0450
                                                                        RUN0460
      COMMON /RANCOM/NRAN(4),KAWLS
      COMMON /FIRST/NL,NW,NH,MW,MH,LW,LH,NXA,NXB,NCA,NCB,NFA,NFB,NHA,NHBRUN0480
      COMMON /SECND/BW,BH,BWB,BHB,BWC,BHC,XLB,XLC                       RUN0490
      COMMON /THIRD/PI,NREG,S,SINANG,COSANG,AKN                         RUN0500
      COMMON /FORTH/NBX,RM,XR,DUMP,C9,RWFM,LL                           RUN0510
      COMMON /FIFTH/ND,TIME,DTM,TI,ITS,ITP,TST,TLIM,RMA,RNU,DIR         RUN0520
      COMMON /SIXTH/RMB,XSTART,JNM,MNM,MNB,NEW,SAVE,PERCNT,NSR,TR       RUN0530
      COMMON /SVNTH/LAM,MU,NU,MT,N,J,X,Y,Z,TUSE                         RUN0540
      COMMON/EIGTH/DENF,U,TF,ANGLE,TRF,CHI,PHI,ETA,WTM,DAM,VELR,XREF
      NAMELIST/CONTRL/NAME,TITLE,PERCNT,ICOPY,DUMP,DEBUG,NEW,SAVE,REDO
      NAMELIST/TIMES/DTM,ITS,ITP,TST,TLIM                               RUN0580
      NAMELIST/FLOREF/INM,MNM,MNB,MSP,MET,U,ANGLE,RNU,RMA,TF,DENF
      NAMELIST/MOLEC/TRF,DIR,ETA,PHI,CHI,ACR
      NAMELIST/SHAPES/BODY                                              RUN0570
      NAMELIST/GEOM/NWEDGE,THETAZ,RMB,BW,BH,NW,NH,NL,NFA,NCA,NHA,MW,MH,N
     1HB,LW,LH,LD,LP,NFB,NCB
      NAMELIST/COUPLE/NS,MS,IWS,VEL,MJ,SL,DELS                          RUN0590
      DATA  IC/0/,ICOPY/1/
      DATA DBG1/' GAS',' AT ','110 ','FLOW',' AT ','130 ',' RUN',' AT ',RUN0630
     1'303 '/                                                           RUN0640
      DATA LIMIT/12,9,18,500,3600,70,900,3,20,3/
      DATA TITLE/'    ','    ','    ','    ','    ','    '/           RUN0660
      DATA NAME/'    ','    '/                                          RUN0670
      DATA CPC/0.0/,CPM/0.0/,CPB/0.0/,CPJ/15.0/
                                                                        RUN0680
```

```
C*********************************************************************RUN0690
C                                                                     RUN0700
C                                                                     RUN0710
C                  FORMATS                                            RUN0720
C                                                                     RUN0730
C                                                                     RUN0740
    1 FORMAT(1H1)                                                     RUN0750
    2 FORMAT(1H1/17X,'RARIFIED SUPERSONIC FLOW OF BINARY GAS',T74,'I') RUN0760
    3 FORMAT('+',103X,'COPY ',I2)                                     RUN0770
    4 FORMAT(/17X,'FLOW THROUGH ALL THE BOUNDARIES'//)
    5 FORMAT(3X,3I4,6E18.6/)
    6 FORMAT(7X,2I4,    E18.6,72X,E18.6)
   30 FORMAT('1TIME =',F6.3,60X,'RANDOM NUMBER GENERATOR HAS BEEN CALLED
     1 ',I10,' TIMES')                                                RUN0800
   31 FORMAT(' CPU TIME LEFT- ',F8.3)                                 RUN0810
   32 FORMAT(7X,'-MOLECULES-'/3X,3I6)
   33 FORMAT(' TIME = ',F8.3,5X,'COLLISION LOOP=',F8.3,5X,'MOVE LOOP = '
     1,F8.3,5X,'TOTAL TIME = ',F8.3/21X,'2ND MOVE LOOP =',F8.3,5X,
     2'CLEANUP LOOP=',F8.3,4X,'PARTICLE NUMBERS = ',4I6)
   34 FORMAT(9X,'-MOLECULAR COLLISIONS-'/3(3I14/))
   35 FORMAT(2X,'-COLLISIONS WITH SURFACE-'/3X,3I8)
   36 FORMAT(' MAXIMUM NUMBER OF MOLECULES SO FAR- ',I6//)            RUN0880
   38 FORMAT(' EXCESS MOLECULES OCCURRED IN ',3A4)                    RUN0890
   40 FORMAT(/' SOMETHING IS WRONG WITH BOX NUMBERING IN RUN '/5I5,2E17.RUN0900
     17,3I5,2E17.7,5I5/2E17.7)                                        RUN0910
   42 FORMAT(/' SOMETHING WRONG IN COMPUTING XSTART'/1X,8E16.8)       RUN0920
   44 FORMAT(' NB(',I2,',',I4,') POPULATION EXCEEDED ',I3,' IN MAIN AT TRUN0930
     1IME = ',F7.3)                                                   RUN0940
   50 FORMAT(///'    SNAP SAVED ON TAPE')                             RUN0950
C                                                                     RUN0960
C*********************************************************************RUN0970
C                                                                     RUN0980
      CPA=ELTIME(0)
      CALL NOUNDF
      LIMIT(4)=NMC
      LIMIT(5)=NMP
      LIMIT(6)=NPB
      LIMIT(7)=NMB
      LIMIT(10)=NSP
      KAWLS=0
      PI=3.141593                                                     RUN1030
      PIROOT=SQRT(PI)                                                 RUN1040
      MET=0
      LARGE=0                                                         RUN1060
      NL=1                                                            RUN1080
      NFA=0                                                           RUN1090
      NCA=0                                                           RUN1100
      NHA=0                                                           RUN1110
      MW=0                                                            RUN1120
      MH=0                                                            RUN1130
      NFB=0                                                           RUN1140
      NCB=0                                                           RUN1150
      NHB=0                                                           RUN1160
      LW=0                                                            RUN1170
      LH=0                                                            RUN1180
```

```
      MJ=20
      DUMP=.TRUE.
      DEBUG(1)=.FALSE.
      DEBUG(2)=.FALSE.
      DEBUG(3)=.TRUE.
      SAVE=.FALSE.
      NEW=.TRUE.
      REDO=.FALSE.
      PERCNT=.001
      ACR=.001
      DO 58 I=1,15
   58 BODY(I)=0.0
      DO 60 I=1,3
      RNU(I)=0.0
      RMA(I)=0.0
      CHI(I)=0.0
      DO 59 J=1,18
      ALPHA(I,J)=1.0
   59 SIGMA(I,J)=1.0
      DO 60 K=1,3
      ETA(I,K)=0.0
      PHI(I,K)=0.0
   60 DIR(I,K)=0.0
      VEL(1)=3.
      VEL(2)=3.
      VEL(3)=4.
      WRITE(6,1)
      READ(5,CONTRL)
      WRITE(6,CONTRL)
      IF(NEW) GO TO 103
      REWIND 9
      READ(9) DENF,U,XREF,TRF,  KAWLS,NL,NW,NH,MW,MH,LW,LH,NXA,NXB,NCA
     1        ,NCB,NFA,NFB,NHA,NHB,BW,BH,BWB,BHB,BWC,BHC,XLB,XLC,PI,NREG
     2        ,S,SINANG,COSANG,AKN,NBX,RM,XR,ND,TIME,DTM,TI,ITS,ITP,TST
     3        ,TLIM,RMA,RNU,DIR,XSTART,JNM,MNM,MNB,TR,BZC,CN7,DRP,FCF
     4        ,PHA,HTF,INM,ITYPE,JTYPE,MJ,NAV,NMAX,NS,NWEDG,PRT,SAMP
     5        ,BTA,C1,C2,C3,C7,C8,DAM,DPA,PL,DELANG,FDN,HTI,HTR,JNT,KNM
     6        ,NM,WTM,C4,VRM,NCOL,LD,LP,LWF,RLD,CTI,CTR,CNI,CNR,LEV,SN
     7        ,ST,D1,D2,D3,D4,SSA,SSB,MS,NSP,NMB,NMC,NMP,NPB,NRAN,VELR
     8        ,IWS,TANGN,XLIM,COEFF,XCB,XS,YCB,TB,ALPHA,SIGMA,NTS,NTSP
     9        ,UTL,UTT,VTS,HTS,HTSI,ENT,REM,ENTS,REMS,FTH,THETA,DTH,TMPA
     A        ,DBA,NB,NBF,NBT,TMP,XV,XVA,YV,YVA,ZV,ZVA,T,DE,FNB,XC,YC,ZC
     B        ,NUMCEL,PAU,PAV,PAW,PAX,PAY,PAZ,FV,NTCV,NTCF,LPF,LCOL,LM
     C,ETA,PHI,CHI,CN,CM,CNG,CMG,CN8,TRP,TRPA,THETAZ,NWEDGE,MSP,ANGLE,TF
     D,UTLI,UTTI,VTSI,ER,RMB,LKW,NBS,LB,NBM,NBN
      REWIND 9
      DTMO=DTM
      READ(5,TIMES)
      WRITE(6,TIMES)
      IF(DTM.EQ.DTMO) GO TO 100
      AIME=TIME*DTMO
      TIME=AIME/DTM+0.1
      DO 99 J=1,NSP
      DO 99 L=1,NWEDG
      ENTS(J,L)=ENTS(J,L)*DTM/DTMO
```

```
      DO 98 K=1,6
      DO 98 I=1,2
      ENT(I,J,K,L)=ENT(I,J,K,L)*DTM/DTMO
   98 CONTINUE
   99 CONTINUE
  100 IF(TI.GT.0.0)  TST=TI/DTM
      WRITE(6,2)
      WRITE(6,4)
      WRITE(6,5) (((I,J,L,(ENT(I,J,K,L),K=1,6),L=1,NWEDG),J=1,MSP),I=1,2)
      WRITE(6,6)  ((J,L,ENTS(J,L),FTH(J,L),L=1,NWEDG),J=1,MSP)
      WRITE(6,2)                                                        RUN1540
      CALL PRINTA(THETAZ,NWEDGE,TITLE,NAME,XCB,YCB,TB,ALPHA,SIGMA,LD,LF,RUN1550
     1XLIM,COEFF,LIMIT,MSP)
      CALL PRINTB(FNA,MSP,FNB,LEV,LWF,NM,RLD,XLIM,XC,YC,ZC,NB,NUMCEL,LKW
     1,MSP)
      GO TO 280                                                         RUN1590
  103 READ(5,TIMES)
      WRITE(6,TIMES)
      READ(5,FLOREF)
      WRITE(6,FLOREF)
      READ(5,MOLEC)
      WRITE(6,MOLEC)
      IF(MSP.GT.LIMIT(10)) CALL DIAG(10,LIMIT(10),MSP)
      CHIM=0.0
      RMR=0.0
      DMR=0.0
      DO 115 M=1,MSP
      RMR=RMR+RMA(M)*RNU(M)
      CHIM=CHIM+CHI(M)*RNU(M)
      DO 115 K=1,MSP
  115 DMR=DMR+RNU(M)*RNU(K)*DIR(M,K)*(TRF/TF)**(ETA(M,K)/2.)
      XREF=1./(DENF*DMR*1.414214)
      VELR=SQRT(16628.64*TF/RMR)
      IF(MET.NE.0)  VELR=SQRT(99437.92*TF/RMR)
      TMR=XREF/VELR
      S=U/VELR
      XLIM(1)=0.                                                        RUN1610
      TR=0.                                                             RUN1620
      YR=0.                                                             RUN1630
      NREG=0                                                            RUN1640
      ND=0                                                              RUN1650
      READ(5,SHAPES)
      WRITE(6,SHAPES)
      XO=BODY(1)/XREF
  104 READ(5,SHAPES)                                                    RUN1670
      WRITE(6,SHAPES)
      ND=ND+1                                                           RUN1680
      IF(ND.GT.LIMIT(3))  CALL DIAG(3,LIMIT(3),ND)                      RUN1690
      XCB(ND)=BODY(1)/XREF+XO
      TB(ND)=BODY(2)/TF
      YCB(ND)=BODY(3)
      DO 1104 M=1,MSP
      ALPHA(M,ND)=BODY(4+2*M)
 1104 SIGMA(M,ND)=BODY(5+2*M)
      IF(TB(ND).GT.TR)  TR=TB(ND)                                       RUN1770
```

```
      IF(YCB(ND).GT.YR) YR=YCB(ND)                                      RUN1780
      IF(BODY(4).EQ.0.0) GO TO 104
      NREG=NREG+1                                                       RUN1800
      XLIM(NREG+2)=XCB(ND)
      XT=BODY(6+2*MSP)/XREF+XO
      A=BODY(7+2*MSP)
      B=BODY(8+2*MSP)/XREF
      C=BODY(9+2*MSP)/XREF**2
      COEFF(1,NREG)=A
      COEFF(2,NREG)=1.0
      COEFF(3,NREG)=B-2.*A*XT
      COEFF(4,NREG)=A*XT**2-B*XT+C
      IF(BODY(5).EQ.0.0) GO TO 104
      IF(NREG.GT.LIMIT(2)) CALL DIAG(2,LIMIT(2),NREG)                   RUN1860
      NSTEP=NREG+2                                                      RUN1870
      A=COEFF(1,1)                                                      RUN1880
      B=COEFF(3,1)                                                      RUN1890
      C=COEFF(4,1)                                                      RUN1900
      DISC=B*B-4.*A*C                                                   RUN1910
      IF(DISC.LT.0.) DISC=0.                                            RUN1920
      DISC=SQRT(DISC)                                                   RUN1930
      IF(A.NE.0.) GO TO 108                                             RUN1940
      XSTART=-C/B                                                       RUN1950
      GO TO 109                                                         RUN1960
  108 X1=.5/A*(-B+DISC)                                                 RUN1970
      X2=.5/A*(-B-DISC)                                                 RUN1980
      XSTART=AMIN1(X1,X2)                                               RUN1990
      XMAX=AMAX1(X1,X2)
      IF(XMAX.LT.XLIM(3)) XSTART=XMAX
      IF(XSTART.GT.0.) GO TO 109                                        RUN2000
      WRITE(6,42)XSTART,A,COEFF(2,1),B,C,DISC,X1,X2                     RUN2010
      IF(DUMP) CALL ABEND(1)                                            RUN2020
      STOP                                                              RUN2030
  109 XLIM(2)=XSTART                                                    RUN2040
      A=COEFF(1,NREG)                                                   RUN2050
      B=COEFF(3,NREG)                                                   RUN2060
      C=COEFF(4,NREG)                                                   RUN2070
      DISC=B*B-4.*A*C                                                   RUN2080
      IF(DISC.LT.0.) DISC=0.                                            RUN2090
      DISC=SQRT(DISC)                                                   RUN2100
      IF(A.NE.0.) GO TO 111                                             RUN2110
      XLIM(NSTEP)=-C/B                                                  RUN2120
      GO TO 112                                                         RUN2130
  111 X1=.5/A*(-B+DISC)                                                 RUN2140
      X2=.5/A*(-B-DISC)                                                 RUN2150
      XLIM(NSTEP)=AMAX1(X1,X2)                                          RUN2160
      XMIN=AMIN1(X1,X2)
      IF(XMIN.GT.XLIM(NSTEP-1)) XLIM(NSTEP)=XMIN
      IF(XLIM(NSTEP).GT.XLIM(NSTEP-1)) GO TO 112
      WRITE(6,42) XLIM(NSTEP),A,COEFF(2,NREG),B,C,DISC,X1,X2
      IF(DUMP) CALL ABEND(1)
      STOP
  112 AKN=1./(XLIM(NSTEP)-XSTART)                                       RUN2170
      XCB(ND)=XLIM(NSTEP)                                               RUN2180
      DO 260 N=1,3
```

```
      MS(N)=0
      IWS(N)=0
      DO 259 M=1,3
      NCOL(N,M)=0
      DO 258 K=1,3
      SL(N,M,K)=0.0
  258 DELS(N,M,K)=0.0
  259 CONTINUE
  260 CONTINUE
      READ(5,GEOM)
      READ(5,COUPLE)                                                    RUN2200
      WRITE(6,GEOM)
      WRITE(6,COUPLE)
      BW=BW/XREF
      BH=BH/XREF
      RMB=RMB/XREF
      IF(RMB.GT.0.) GO TO 264
      DO 262 K=1,NREG
      XBEG=XLIM(K+1)
      XEND=XLIM(K+2)
      A=COEFF(1,K)
      B=COEFF(2,K)
      C=COEFF(3,K)
      D=COEFF(4,K)
      REND=SQRT(ABS((A*XEND**2+C*XEND+D)/B))
      XPEAK=0.
      IF(A.NE.0.) XPEAK=-.5*C/A
      IF((XPEAK.LE.XBEG).OR.(XPEAK.GE.XEND)) GO TO 261
      RTEMP=SQRT(ABS((A*XPEAK**2+C*XPEAK+D)/B))
      IF(RTEMP.GT.REND) REND=RTEMP
  261 IF(REND.GT.RMB) RMB=REND
  262 CONTINUE
  264 CONTINUE
      NWEDG=NWEDGE(1)+NWEDGE(2)
      IF(NWEDGE(2).EQ.0) THETAZ=180.0
      IF(SAVE) REWIND 9                                                 RUN2220
      IF(NWEDG.GT.LIMIT(1)) CALL DIAG(1,LIMIT(1),NWEDG)
      IF(MNM.GT.LIMIT(5)) CALL DIAG(5,LIMIT(5),MNM)                     RUN2230
      IF(MNB.GT.LIMIT(6)) CALL DIAG(6,LIMIT(6),MNB)                     RUN2240
      IF(NS.GT.LIMIT(8)) CALL DIAG(8,LIMIT(8),NS)                       RUN2250
      IF((NS.NE.0).AND.(MJ.GT.LIMIT(9)))CALL DIAG(9,LIMIT(9),MJ)
      JNM=INM                                                           RUN2290
      DELANG(1)=THETAZ/NWEDGE(1)                                        RUN2300
      DELANG(2)=0.0
      IF(NWEDGE(2).NE.0) DELANG(2)=(180.-THETAZ)/NWEDGE(2)
      SINANG=SIN(ANGLE/180.*PI)                                         RUN2320
      COSANG=COS(ANGLE/180.*PI)                                         RUN2330
      XR=BW*NW                                                          RUN2380
      XLIM(NSTEP+1)=XR                                                  RUN2390
      RM=BH*NH                                                          RUN2400
      VOL=PI*RM*RM*XR
      NXA=NW*NH*NWEDG                                                   RUN2420
      NXB=MW*MH*NWEDGE(1)                                               RUN2430
      NXC=LW*LH*NWEDGE(1)                                               RUN2440
      IXA=NW*NH*NWEDGE(1)                                               RUN2450
```

```
      NBX=NXA+NXB+NXC                                                   RUN2460
      IF(NBX.GT.LIMIT(7)) CALL DIAG(7,LIMIT(7),NBX)                     RUN2470
      BR=SQRT(TR)                                                       RUN2480
      DO 113 N=1,5                                                      RUN2490
  113 RLD(N)=BR*LD(N)                                                   RUN2500
      RLD(6)=RM                                                         RUN2510
      LWF(1)=1                                                          RUN2520
      RWFM=RLD(1)                                                       RUN2530
      B=RWFM*RWFM                                                       RUN2540
      C=B                                                               RUN2550
      DO 114 N=2,6                                                      RUN2560
      A=RLD(N)*RLD(N)                                                   RUN2570
      LWF(N)=LWF(N-1)*LF(N-1)                                           RUN2580
      B=B+(A-C)/LWF(N)                                                  RUN2590
      D=RLD(N)/LWF(N)                                                   RUN2600
      IF(D.GT.RWFM) RWFM=D                                              RUN2610
  114 C=A                                                               RUN2620
      INM=INM*RM*RM/B                                                   RUN2630
      DDN=INM/VOL
      DO 140 MT=1,MSP
      WTM(MT)=RMA(MT)/RMR
      FDN(MT)=RNU(MT)*DDN
      DFA(MT)=RNU(MT)
      BTA(MT)=SQRT(WTM(MT))
      SR(MT)=S*BTA(MT)                                                  RUN2730
      SN(MT)=SR(MT)*COSANG                                              RUN2740
      ST(MT)=SR(MT)*SINANG                                              RUN2750
      DO 117 K=1,MSP
      DAM(K,MT)=DIR(K,MT)*(TRF/TF)**(ETA(K,MT)/2.)/DMR
      CN8(K,MT)=DDN/DAM(K,MT)*1.414214
      BT=AMIN1(BTA(K),BTA(MT))
      VR1=S+3.*(1.+SQRT(TR))/BT
      VR2=3.*SQRT((1.+2.*S**2/(5.+CHIM))*(1./WTM(K)+1./WTM(MT)))
      CM(K,MT,1)=AMAX1(VR1,VR2)
      CN(K,MT,1)=RAND(0)*CM(K,MT,1)
      DP=PHI(K,MT)*(CHI(K)+CHI(MT)+2.)-1
      DS=PHI(K,MT)*(2.-.5*ETA(K,MT))-1.0
      DO 917 N=2,3
      XPM=ACR**AMIN1(DP,DS)
      IF((DP.GT.0.).AND.(DS.GT.0.))XPM=(DP/(DP+DS))**DP*(DS/(DP+DS))**DS
      XPN=ACR**AMAX1(DP,DS)
      IF((DP.LT.0.).AND.(DS.LT.0.))XPN=(DP/(DP+DS))**DP*(DS/(DP+DS))**DS
      CM(K,MT,N)=XPM-XPN
      CN(K,MT,N)=RAND(0)*CM(K,MT,N)
      DP=CHI(K)
      DS=CHI(MT)
  917 CONTINUE
  117 CONTINUE
      ARG=SN(MT)                                                        RUN2760
      DO 119 NT=1,2                                                     RUN2770
      D=ERRF(ARG)                                                       RUN2780
      TEMPA=EXP(-SN(MT)*SN(MT))/3.544908+0.5*ARG*D                      RUN2790
      TEMPA=TEMPA/BTA(MT)                                               RUN2800
      TEMPC=0                                                           RUN2810
      DO 118 N=1,6                                                      RUN2820
```

```
      TEMPB=RLD(N)*RLD(N)                                              RUN2830
      TEMPD=INM*TEMPA*DTM/(XR*RM*RM)*(TEMPB-TEMPC)/LWF(N)              RUN2840
      DO 116 K=1,NWEDG                                                 RUN2850
      DELTH=DELANG(1)                                                  RUN2860
      IF(K.GT.NWEDGE(1)) DELTH=DELANG(2)                               RUN2870
      ENT(NT,MT,N,K)=TEMPD*DELTH/180.                                  RUN2880
  116 REM(NT,MT,N,K)=0.                                                RUN2890
      TEMPC=TEMPB                                                      RUN2900
  118 CONTINUE                                                         RUN2910
      SSR=ARG*ARG                                                      RUN2920
      SSA(NT,MT)=ARG+SQRT(SSR+2.)                                      RUN2930
      SSB(NT,MT)=SSA(NT,MT)*(.25*SSA(NT,MT)-ARG)                       RUN2940
  119 ARG=-ARG                                                         RUN2950
      CHT=CHI(MT)
      IF(CHT.GT.0.) CMG(MT)=CHT**CHT*EXP(-CHT)
      IF(CHT.EQ.0.) CMG(MT)=1.0
      IF(CHT.LT.0.) CMG(MT)=ACR**CHT*EXP(-ACR)
      CMG(MT)=RAND(0)*CMG(MT)
      ANG=0.                                                           RUN2970
      N=0                                                              RUN2960
      ARG=ST(MT)                                                       RUN2980
      TEMPD=INM/PI*DTM/RM*2./(BTA(MT)*LWF(6))                          RUN2990
      DO 135 M=1,2
      I=NWEDGE(M)                                                      RUN3010
      IF(I.EQ.0) GO TO 135
      DO 134 K=1,I                                                     RUN3020
      N=N+1                                                            RUN3030
      AA=ANG                                                           RUN3040
      BB=DELANG(M)                                                     RUN3050
      CC=AA+BB                                                         RUN3060
      ANG=ANG+BB                                                       RUN3070
      THETA(N)=AA                                                      RUN3080
      DTH(N)=BB                                                        RUN3090
      AA=AA*PI/180.                                                    RUN3100
      CC=CC*PI/180.                                                    RUN3110
      ARGA=ARG*COS(AA)                                                 RUN3120
      ARGC=ARG*COS(CC)                                                 RUN3130
      ENTS(MT,N)=0.                                                    RUN3140
      REMS(MT,N)=0.                                                    RUN3150
      FTH(MT,N)=0.                                                     RUN3160
      TEMPA=0.                                                         RUN3170
      IF(ARGA.LE.-10.) GO TO 120                                       RUN3180
      TEMPA=FNCTM(ARGA,PIROOT,L,COEFF)                                 RUN3190
      FTH(MT,N)=2.*PIROOT*TEMPA                                        RUN3200
  120 TEMPB=0.                                                         RUN3210
      IF(ARGC.LE.-10.) GO TO 125                                       RUN3220
      TEMPB=FNCTM(ARGC,PIROOT,L,COEFF)                                 RUN3230
  125 SUM1=TEMPA+TEMPB                                                 RUN3240
      TEMPC=.5*SUM1*(CC-AA)*TEMPD                                      RUN3250
      IF(TEMPC.LT.1.E-06) GO TO 134                                    RUN3260
      CALL SIMPSN(ARGA,ARGC,0,INTGRL,PERCNT,COEFF,PIROOT,SUM1,FNCTM)   RUN3270
      ENTS(MT,N)=INTGRL*TEMPD*(CC-AA)                                  RUN3280
  134 CONTINUE                                                         RUN3290
  135 CONTINUE
  140 CONTINUE                                                         RUN3330
```

```
      XS(1)=.5*(XCB(1)-XSTART)*AKN
      DO 155 N=2,ND
  155 XS(N)=(.5*(XCB(N)+XCB(N-1))-XSTART)*AKN
      IF(NS.EQ.0) GO TO 160
      DO 159 I=1,NS
      N=MS(I)
      X=XS(N)/AKN+XSTART
      J=0
  157 J=J+1
      IF(X.GT.XLIM(J+2)) GO TO 157
      CALL HEIGHT(X,Y,J,COEFF,3)
      TANGN(I)=(IWS(I)-.5)*DELANG(1)
      IF(IWS(I)*DELANG(1).GT.THETAZ) TANGN(I)=THETAZ+(IWS(I)-NWEDGE(1)-.
     15)*DELANG(2)
      Z=Y*SIN(TANGN(I)*PI/180.)
      Y=-Y*COS(TANGN(I)*PI/180.)
      CALL NORMAL(EYE,JAY,KAY,ONE,COEFF)
      SNN=-S*(COSANG*EYE+SINANG*JAY)
      ST1=S*(COSANG*ONE-EYE*SINANG*JAY/ONE)
      ST2=-S*SINANG*KAY/ONE
      DO 159 MT=1,MSP
      VL(MT,1)=AMAX1(0.,SNN-VEL(1)/BTA(MT))
      VL(MT,2)=ST1-VEL(2)/BTA(MT)
      VL(MT,3)=ST2-VEL(3)/BTA(MT)
      DELV(MT,1)=SNN+VEL(1)/BTA(MT)-VL(MT,1)
      DELV(MT,2)=2.*VEL(2)/BTA(MT)
      DELV(MT,3)=2.*VEL(3)/BTA(MT)
      AMJ=MJ-1
      DO 159 K=1,3
      DO 159 J=1,MJ
      PV(MT,I,1,J,K)=VL(MT,K)+(J-1)/AMJ*DELV(MT,K)
      PV(MT,I,2,J,K)=SL(MT,I,K)+(J-1)/AMJ*DELS(MT,I,K)
  159 CONTINUE
  160 CONTINUE
      IF(YR.GT.0.) GO TO 169
      A=XSTART
      B=XCB(1)
      L=1
      SUM1=FNCTN(A,PIROOT,L,COEFF)+FNCTN(B,PIROOT,L,COEFF)
      CALL SIMPSN(A,B,L,INTGRL,PERCNT,COEFF,PIROOT,SUM1,FNCTN)
      INTGRL=INTGRL/(ABS(COEFF(2,L))*RMB*RMB)*(B-A)
      YCB(1)=INTGRL
      DO 168 N=2,ND
      A=XCB(N-1)
      B=XCB(N)
      IF(XLIM(L+2).GE.B) GO TO 167
      L=L+1
  167 SUM1=FNCTN(A,PIROOT,L,COEFF)+FNCTN(B,PIROOT,L,COEFF)
      CALL SIMPSN(A,B,L,INTGRL,PERCNT,COEFF,PIROOT,SUM1,FNCTN)
      INTGRL=INTGRL/(ABS(COEFF(2,L))*RMB*RMB)*(B-A)
  168 YCB(N)=INTGRL
  169 LV=2
      IF(NWEDGE(2).EQ.0) LV=1
      CALL CELL(THETAZ,LV,BW,BH,NW,NH,0.,0,0,DELANG,NWEDGE,XC,YC,ZC,FNB)
      IF(NL.LT.2) GO TO 170
```

```
      CALL ZERO(NW,NH,NFA,NCA,NHA,0,NWEDGE(1),FNB)                      RUN3900
      XLB=BW*NFA                                                        RUN3910
      BWB=BW*NCA/NW                                                     RUN3920
      BHB=BH*NHA/NH                                                     RUN3930
      CALL CELL(THETAZ,1,BWB,BHB,NW,NH,XLB,NXA,0,DELANG,NWEDGE,XC,YC,ZC,RUN3940
     1FNB)                                                              RUN3950
      IF(NL.LT.3) GO TO 170                                             RUN3960
      CALL ZERO(NW,NH,NFB,NCB,NHB,NXA,NWEDGE(1),FNB)                    RUN3970
      XLC=XLB+BWB*NFB                                                   RUN3980
      BWC=BWB*NCB/LW                                                    RUN3990
      BHC=BHB*NHB/LH                                                    RUN4000
      CALL CELL(THETAZ,1,BWC,BHC,LW,LH,XLC,NXA,NXB,DELANG,NWEDGE,XC,YC,ZRUN4010
     1C,FNB)                                                            RUN4020
  170 CALL SBTRCT(1,NXA,NXA,BW,BH,DELANG,XC,YC,FNB,XLIM,COEFF)          RUN4030
      LEV(1)=NBX+1
      LEV(2)=NBX+1
      IF(NL.LT.2) GO TO 190                                             RUN4040
      NI=NXA+1                                                          RUN4050
      NP=NXA+NXB                                                        RUN4060
      LEV(1)=NI                                                         RUN4070
      CALL SBTRCT(NI,NP,NP,BWB,BHB,DELANG,XC,YC,FNB,XLIM,COEFF)         RUN4080
      IF(NL.LT.3) GO TO 190                                             RUN4090
      NI=NP+1                                                           RUN4100
      NP=NBX                                                            RUN4110
      LEV(2)=NI                                                         RUN4120
      CALL SBTRCT(NI,NP,NP,BWC,BHC,DELANG,XC,YC,FNB,XLIM,COEFF)         RUN4130
  190 FNA=0.0                                                           RUN4140
      M=0                                                               RUN4150
      DO 210 N=1,NBX                                                    RUN4160
      NUMCEL(N)=0                                                       RUN4170
      IF(FNB(N).LE.0.) GO TO 210                                        RUN4180
      M=M+1                                                             RUN4190
      NUMCEL(N)=M                                                       RUN4200
      FNA=FNA+FNB(N)                                                    RUN4210
      DYC=BH/2.
      IF(N.GT.NXA) DYC=BHB/2.
      IF(N.GT.NXA+NXB) DYC=BHC/2.
      YTC=YC(N)+DYC
      DO 200 LA=1,6
      IF(YTC.LE.RLD(LA)) GO TO 201
  200 CONTINUE
  201 LKW(M)=LWF(LA)
  210 CONTINUE                                                          RUN4220
      NPX=M                                                             RUN4230
      IF(NPX.GT.LIMIT(4)) CALL DIAG(4,LIMIT(4),NPX)                     RUN4240
  220 TIME=0                                                            RUN4250
      LARGE=0                                                           RUN4260
      SAMP=0                                                            RUN4270
      PRT=0
      NAV=0                                                             RUN4290
      AIME=0.                                                           RUN4300
      TI=-1.                                                            RUN4310
      NMAX=0                                                            RUN4380
      DO 250 MT=1,3
      C1(MT)=RAND(0)                                                    RUN4410
```

```
      C2(MT)=RAND(0)                                              RUN4420
      C3(MT)=RAND(0)                                              RUN4430
      C7(MT)=RAND(0)                                              RUN4440
      C8(MT)=RAND(0)                                              RUN4450
      D1(MT)=RAND(0)                                              RUN4460
      D2(MT)=RAND(0)                                              RUN4470
      D3(MT)=RAND(0)                                              RUN4480
      D4(MT)=RAND(0)                                              RUN4490
      FL(MT)=0.                                                   RUN4500
      HTI(MT)=0.                                                  RUN4510
      HTR(MT)=0.                                                  RUN4520
      JNT(MT)=0                                                   RUN4530
      NM(MT)=0                                                    RUN4540
      DO 230 N=1,3                                                RUN4550
      CTI(MT,N)=0.                                                RUN4560
      CTR(MT,N)=0.                                                RUN4570
      CNI(MT,N)=0.                                                RUN4580
  230 CNR(MT,N)=0.                                                RUN4590
      DO 240 N=1,KD                                               RUN4600
      DO 240 K=1,NWEDG                                            RUN4610
      NTS(MT,N,K)=0                                               RUN4620
      NTSF(MT,N,K)=0                                              RUN4630
      HTSI(MT,N,K)=0.                                             RUN4640
      UTLI(MT,N,K)=0.
      UTTI(MT,N,K)=0.
      VTSI(MT,N,K)=0.
      UTL(MT,N,K)=0.                                              RUN4650
      UTT(MT,N,K)=0.                                              RUN4660
      VTS(MT,N,K)=0.                                              RUN4670
  240 HTS(MT,N,K)=0.                                              RUN4680
      DO 250 I=1,3
      NTCF(MT,I)=0                                                RUN4820
      DO 250 L=1,2                                                RUN4830
      DO 250 K=1,3                                                RUN4840
      DO 250 J=1,MJ                                               RUN4850
      NTCV(MT,I,L,J,K)=0                                          RUN4860
  250 CONTINUE                                                    RUN4870
      DO 245 N=1,NPX                                              RUN4690
      DO 245 MT=1,NSP
      NB(MT,N)=0                                                  RUN4700
      NBF(MT,N)=0                                                 RUN4710
      NBS(MT,N)=0
      NBT(MT,N)=0                                                 RUN4720
      DBA(MT,N)=0.                                                RUN4730
      XVA(MT,N)=0.                                                RUN4740
      YVA(MT,N)=0.                                                RUN4750
      ZVA(MT,N)=0.                                                RUN4760
      TMPA(MT,N)=0.                                               RUN4790
      TRPA(MT,N)=0.0
      DO 245 NN=1,NSP
      T(MT,NN,N)=0.0
      FND=DDN
  245 CONTINUE                                                    RUN4800
      DRF=2./(FND*S*S*RMB*RMB*PI)                                 RUN4930
      FCF=1./(FND*S*RMB*RMB*PI)                                   RUN4940
```

```
      HTF=.5*DRF/S
      C9=RAND(0)*RWFM                                                   RUN4960
      LL=FNA/VOL*INB                                                    RUN4970
      WRITE(6,2)
      WRITE(6,4)
      WRITE(6,5)(((I,J,L,(ENT(I,J,K,L),K=1,6),L=1,NWEDG),J=1,MSP),I=1,2)
      WRITE(6,6) ((J,L,ENTS(J,L),FTH(J,L),L=1,NWEDG),J=1,MSP)
      WRITE(6,2)                                                        RUN5000
      CALL PRINTA(THETAZ,NWEDGE,TITLE,NAME,XCB,YCB,TB,ALPHA,SIGMA,LD,LF,RUN5010
     1XLIM,COEFF,LIMIT,MSP)
      CALL GAS(NWEDG,THETAZ,DELANG,NWEDGE,BTA,C1,DFA,NM,RLD,LWF,FNB,DB,NRUN5030
     1B,NBF,LPF,PAU,PAV,PAW,PAX,PAY,PAZ,XLIM,COEFF,LM,LIMIT(4),LIMIT(6),RUN5040
     2LARGE,MNM,MNB,DEBUG(1),LCOL,NUMCEL,MSP,ER,CHI,CNG,CMG,NSP,LB,NBM,
     3NBN)
      CPUTYM=TFIND(0)
      IF(LARGE.NE.0) GO TO 345                                          RUN5060
      DO 265 I=1,MSP
  265 IF(NM(I).GT.NMAX) NMAX=NM(I)
      CALL PRINTB(FNA,MSP,FNB,LEV,LWF,NM,RLD,XLIM,XC,YC,ZC,NB,NUMCEL,LKW
     1,NSP)
      IF(DEBUG(2)) WRITE(6,1)
      CALL ACCUM(NMC,NPB,FNB,NB,PAU,PAV,PAW,ER,TMP,TBP,XV,YV,ZV,LM,MSP,
     1NSP,LPF,NBF,NBM)
      CPA=ELTIME(0)
      CPI=CPA
      GO TO 340                                                         RUN5130
  280 TIME=TIME+1                                                       RUN5140
  285 LARGE=0
      CPI=ELTIME(0)
      AIME=TIME*DTM                                                     RUN5170
      IF(DEBUG(1))WRITE(6,33)AIME,CPC,CPM,CPI,CPB,CPA,(NM(I),I=1,3),NMAX
      PRT=PRT+1                                                         RUN5180
      SAMP=SAMP+1                                                       RUN5190
      CALL COLIDE(CN,CM,WTM,DB,DBA,NB,NCOL,LCOL,PAU,PAV,PAW,ER,T,LM,MSP,
     1LIMIT(4),LIMIT(6),NUMCEL,ETA,PHI,CHI,CN8,NSP,LPF,LKW,NBF,NBM)
      KNM(1)=0                                                          RUN5220
      KNM(2)=0                                                          RUN5230
      KNM(3)=0
      CPC=ELTIME(0)
      IF(DEBUG(1))WRITE(6,33)AIME,CPC,CPM,CPI,CPB,CPA,(NM(I),I=1,3),NMAX
      CALL MOVE(0,AKN,MJ,NS,NWEDG,THETAZ,XSTART,LIMIT(3),LIMIT(1),LIMIT RUN5240
     1(8),LIMIT(9),DELANG,NWEDGE,BTA,C2,C3,DFA,FL,HTI,HTR,JNT,KNM,NM,XCBRUN5250
     2,XLIM,MS,IWS,NTCF,NTCV,FV,CTI,CTR,CNI,CNR,ALPHA,SIGMA,COEFF,HTS,HTRUN5260
     3SI,NTS,NTSF,UTL,UTT,VTS,PAU,PAV,PAW,PAX,PAY,PAZ,LPF,LCOL,TB,MSP,ER
     4,CHI,CNG,CMG,NSP,UTLI,UTTI,VTSI)
      KNM(1)=NM(1)                                                      RUN5280
      KNM(2)=NM(2)                                                      RUN5290
      KNM(3)=NM(3)
      CPM=ELTIME(0)
      IF(DEBUG(1))WRITE(6,33)AIME,CPC,CPM,CPI,CPB,CPA,(NM(I),I=1,3),NMAX
      CALL FLOW(NWEDG,MNM,LARGE,BTA,C1,C7,C8,D1,D2,D3,D4,DTH,NM,SN,ST,THRUN5300
     1ETA,LWF,RLD,FTH,ENTS,REMS,SSA,SSB,PAU,PAV,PAW,PAX,PAY,PAZ,LPF,ENT,RUN5310
     2REM,LCOL,MSP,ER,CHI,CNG,CMG,NSP)
      IF(LARGE.NE.0) GO TO 345                                          RUN5330
      CPB=ELTIME(0)
```

```
      IF(DEBUG(1))WRITE(6,33)AIME,CPC,CPM,CPI,CPB,CPA,(NM(I),I=1,3),NMAX
      CALL MOVE(1,AKN,MJ,NS,NWEDG,THETAZ,XSTART,LIMIT(3),LIMIT(1),LIMIT
     1(8),LIMIT(9),DELANG,NWEDGE,BTA,C2,C3,DFA,FL,HTI,HTR,JNT,KNM,NM,XCBRUN5350
     2,XLIM,MS,IWS,NTCF,NTCV,FV,CTI,CTR,CNI,CNR,ALPHA,SIGMA,COEFF,HTS,HTRUN5360
     3SI,NTS,NTSF,UTL,UTT,VTS,PAU,PAV,PAW,PAX,PAY,PAZ,LPF,LCOL,TB,MSP,ER
     4,CHI,CNG,CMG,NSP,UTLI,UTTI,VTSI)
      DO 330 MT=1,MSP
      CPB=CPB+ELTIME(0)
      IF(DEBUG(1))WRITE(6,33)AIME,CPC,CPM,CPI,CPB,CPA,(NM(I),I=1,3),NMAX
      M=0                                                               RUN5410
      DO 290 N=1,NBX                                                    RUN5420
      IF(NUMCEL(N).EQ.0) GO TO 290
      M=M+1                                                             RUN5440
      NB(MT,M)=0                                                        RUN5450
      NBF(MT,M)=0                                                       RUN5460
  290 CONTINUE                                                          RUN5470
      NG=NM(MT)                                                         RUN5480
      N=0                                                               RUN5490
  295 N=N+1                                                             RUN5500
      IF(N.GT.NG) GO TO 310                                             RUN5510
      X=PAX(MT,N)                                                       RUN5520
      Y=PAY(MT,N)                                                       RUN5530
      Z=PAZ(MT,N)                                                       RUN5540
      R=SQRT(Y*Y+Z*Z)                                                   RUN5550
      ARG=Y/R                                                           RUN5560
      TANG=180.*(1.-ARCCOS(ARG)/PI)                                     RUN5570
      IWDGE=TANG/DELANG(1)                                              RUN5580
      IF((IWDGE.GE.NWEDGE(1)).AND.(DELANG(2).NE.0.)) IWDGE=(TANG-THETAZ)
     1/DELANG(2)+NWEDGE(1)
      IF(IWDGE.LT.0) IWDGE=0
      IF(IWDGE.GE.NWEDG) IWDGE=NWEDG-1                                  RUN5600
      L=X/BW+1.                                                         RUN5610
      IF(L.GT.NW) L=NW                                                  RUN5620
      M=R/BH                                                            RUN5630
      IF(M.GE.NH) M=NH-1                                                RUN5640
      K=(IWDGE*NH+M)*NW+L                                               RUN5650
      IF(K.LE.NXA) GO TO 296                                            RUN5660
      WRITE(6,40)L,M,K,MT,N,X,Y,NWEDGE,NWEDG,TANG,Z,IWDGE,NH,NW         RUN5670
      IF(DUMP) CALL ABEND(4)                                            RUN5680
      STOP                                                              RUN5690
  296 KW=0                                                              RUN5700
      IF(NL.EQ.1) GO TO 300                                             RUN5710
      IF(IWDGE.GE.NWEDGE(1)) GO TO 300
      IF(FNB(K).GT.0.) GO TO 300                                        RUN5730
      L=(X-XLB)/BWB+1.                                                  RUN5740
      IF(L.GT.MW) L=MW                                                  RUN5750
      M=R/BHB                                                           RUN5760
      IF(M.GE.MH) M=MH-1                                                RUN5770
      K=(IWDGE*MH+M)*MW+L+NXA                                           RUN5780
      IF(K.LE.NXA+NXB) GO TO 297                                        RUN5790
      WRITE(6,40)L,M,K,MT,N,X,Y,NWEDGE,NWEDG,TANG,Z,IWDGE,MH,MW,NXA     RUN5800
      IF(DUMP) CALL ABEND(5)                                            RUN5810
      STOP                                                              RUN5820
  297 IF(NL.EQ.2) GO TO 300                                             RUN5830
      IF(FNB(K).GT.0.) GO TO 300                                        RUN5840
```

```
      L=(X-XLC)/BWC+1.                                                  RUN5850
      IF(L.GT.LW) L=LW                                                  RUN5860
      M=R/BHC                                                           RUN5870
      IF(M.GE.LH) M=LH-1                                                RUN5880
      K=(IWDGE*LH+M)*LW+L+NXA+NXB                                       RUN5890
      IF(K.LE.NBX) GO TO 300                                            RUN5900
      WRITE(6,40)L,M,K,MT,N,X,Y,NWEDGE,NWEDG,TANG,Z,IWDGE,LH,LW,NXA,NXB RUN5910
      IF(DUMP) CALL ABEND(6)                                            RUN5920
      STOP                                                              RUN5930
300   LA=0                                                              RUN5940
      IF(NUMCEL(K).EQ.0) GO TO 306
301   LA=LA+1                                                           RUN5960
      IF(R.GT.RLD(LA)) GO TO 301                                        RUN5970
      KW=LWP(LA)                                                        RUN5980
      KKW=LPP(MT,N)                                                     RUN5990
      IF(KW.EQ.KKW) GO TO 305                                           RUN6000
      IF(KW.LT.KKW) GO TO 302                                           RUN6010
      M=KW/KKW                                                          RUN6020
      A=RAND(0)                                                         RUN6030
      B=M                                                               RUN6040
      B=1./B                                                            RUN6050
      IF(A.LT.B) GO TO 305                                              RUN6060
306   PAX(MT,N)=PAX(MT,NG)                                              RUN6070
      PAY(MT,N)=PAY(MT,NG)                                              RUN6080
      PAZ(MT,N)=PAZ(MT,NG)                                              RUN6090
      PAU(MT,N)=PAU(MT,NG)                                              RUN6100
      PAV(MT,N)=PAV(MT,NG)                                              RUN6110
      PAW(MT,N)=PAW(MT,NG)                                              RUN6120
      ER(MT,N)=ER(MT,NG)
      LPP(MT,N)=LPP(MT,NG)                                              RUN6130
      LCOL(MT,N)=LCOL(MT,NG)                                            RUN6140
      N=N-1                                                             RUN6150
      NM(MT)=NM(MT)-1                                                   RUN6160
      NG=NM(MT)                                                         RUN6170
      GO TO 295                                                         RUN6180
302   M=KKW/KW-1                                                        RUN6190
      IF((NM(MT)+M).LE.MNM) GO TO 307
      LARGE=3                                                           RUN6230
      GO TO 345                                                         RUN6240
307   CONTINUE
      DO 304 L=1,M                                                      RUN6200
      NM(MT)=NM(MT)+1                                                   RUN6210
      NG=NM(MT)
      PAX(MT,NG)=PAX(MT,N)                                              RUN6260
      PAY(MT,NG)=PAY(MT,N)                                              RUN6270
      PAZ(MT,NG)=PAZ(MT,N)                                              RUN6280
      PAU(MT,NG)=PAU(MT,N)                                              RUN6290
      PAV(MT,NG)=PAV(MT,N)                                              RUN6300
      PAW(MT,NG)=PAW(MT,N)                                              RUN6310
      ER(MT,NG)=ER(MT,N)
      LCOL(MT,NG)=LCOL(MT,N)                                            RUN6320
304   LPP(MT,NG)=KW                                                     RUN6330
305   LPP(MT,N)=KW                                                      RUN6340
      Q=NUMCEL(K)                                                       RUN6350
      J=NB(MT,Q)+1                                                      RUN6360
```

```
      IF(J.LE.MNB) GO TO 308                                          RUN6370
      IF(DEBUG(1)) WRITE(6,44) MT,K,MNB,AIME
  308 NB(MT,Q)=J                                                      RUN6400
      NBF(MT,Q)=NBF(MT,Q)+KW                                          RUN6410
      LB(N)=Q
      GO TO 295                                                       RUN6430
  310 CONTINUE
      NBM(MT,1)=0
      DO 320 N=1,NBX                                                  RUN6450
      M=NUMCEL(N)
      IF(M.EQ.0) GO TO 320
      A=NBF(MT,M)                                                     RUN6480
      DB(MT,M)=A*DFA(MT)/FNB(N)                                       RUN6490
      NBM(MT,M+1)=NBM(MT,M)+NB(MT,M)
      NBN(M)=NBM(MT,M)
  320 CONTINUE                                                        RUN6500
      IF(NM(MT).GT.NMAX) NMAX=NM(MT)                                  RUN6510
      DO 325 N=1,NG
      Q=LB(N)
      NBN(Q)=NBN(Q)+1
      NA=NBN(Q)
  325 LM(MT,NA)=N
  330 CONTINUE                                                        RUN6520
      IF(SAMP.LT.ITS) GO TO 335
      CALL ACCUM(NMC,NPB,FNB,NB,PAU,PAV,PAW,ER,TMP,TRP,XV,YV,ZV,LM,MSP,
     1NSP,LPF,NBF,NBM)
      SAMP=0                                                          RUN6570
      IF(TIME.LE.TST) GO TO 335
      CALL AVRGE(FNB,DB,DBA,NB,NBT,XV,YV,ZV,XVA,YVA,ZVA,TMP,TMPA,TRP,TRP
     1A,MSP,NSP,NBF,NBS)
      NAV=NAV+1                                                       RUN6600
  335 CPA=ELTIME(0)
      CPI=CPC+CPM+CPB+CPA
      CPJ=2.*CPI+5.
  340 CPUTYM=TFIND(0)
      IF(DEBUG(2))WRITE(6,33)AIME,CPC,CPM,CPI,CPB,CPA,(NM(I),I=1,3),NMAX
      IF((TIME.GE.TLIM).OR.(CPUTYM.LE.CPJ)) GO TO 345
      IF(PRT.LT.ITP) GO TO 280                                        RUN6650
      PRT=0                                                           RUN6660
  345 WRITE(6,30)AIME,KAWLS                                            RUN6670
      IF(DEBUG(3))    WRITE(6,31)CPUTYM
      WRITE(6,32)  (NM(I),I=1,3)
      WRITE(6,34)  ((NCOL(I,J),J=1,3),I=1,3)
      WRITE(6,35)  (JNT(I),I=1,3)
      IF(LARGE.NE.0) GO TO 360                                        RUN6740
      WRITE(6,36)NMAX                                                 RUN6750
      IF(.NOT.SAVE) GO TO 355                                         RUN6900
      IF(PRT.NE.0.AND.CPUTYM.GT.CPJ.AND.TIME.LT.TLIM) GO TO 355
      WRITE(9)DENF,U,XREF,TRP,   KAWLS,NL,NW,NH,MW,MH,LW,LH,NXA,NXB,NCA
     1        ,NCB,NPA,NPB,NHA,NHB,BW,BH,BWB,BHB,BWC,BHC,XLB,XLC,PI,NREGRUN6930
     2        ,S,SINANG,COSANG,AKN,NBX,RM,XR,ND,TIME,DTM,TI,ITS,ITP,TST RUN6940
     3        ,TLIM,RMA,BNU,DIR,XSTART,JNM,MNM,MNB,TB,BZC,CN7,DRF,FCF    RUN6950
     4        ,FNA,HTF,INM,ITYPE,JTYPE,MJ,NAV,NMAX,NS,NWEDG,PRT,SAMP     RUN6960
     5        ,BTA,C1,C2,C3,C7,C8,DAM,DPA,FL,DELANG,FDN,HTI,HTR,JNT,KNM  RUN6970
     6        ,NM,WTM,C4,VRM,NCOL,LD,LP,LWP,RLD,CTI,CTR,CNI,CNR,LEV,SN   RUN6980
```

```
     7        ,ST,D1,D2,D3,D4,SSA,SSB,MS,NSP,NMB,NMC,NMP,NPB,NRAN,VELR
     8        ,IWS,TANGN,XLIM,COEFF,XCB,XS,YCB,TB,ALPHA,SIGMA,NTS,NTSP  RUN7000
     9        ,UTL,UTT,VTS,HTS,HTSI,ENT,REM,ENTS,REMS,FTH,THETA,DTH,TMPARUN7010
     A        ,DBA,NB,NBF,NBT,TMP,XV,XVA,YV,YVA,ZV,ZVA,T,DB,FNB,XC,YC,ZCRUN7020
     B        ,NUMCEL,PAU,PAV,PAW,PAX,PAY,PAZ,PV,NTCV,NTCF,LPF,LCOL,LM  RUN7030
     C,ETA,PHI,CHI,CN,CM,CNG,CMG,CN8,TRP,TRPA,THETAZ,NWEDGE,MSP,ANGLE,TF
     D,UTLI,UTTI,VTSI,ER,RMB,LKW,NBS,LB,NBM,NBN
      REWIND 9                                                          RUN7040
      WRITE(6,50)                                                       RUN7050
  355 CONTINUE
      IF(TIME.LE.TST) GO TO 350
      DT=AIME-TI                                                        RUN6770
      CALL PRINT1(DT,COSANG,SINANG,RMA,RNU,DRF,FCP,HTP,FL,HTI,HTR,CTI,
     1CTR,CNI,CNR)
      CALL PRINT2(AKN,XSTART,DT,RNU,RMA,DRF,FCP,HTP,UTLI,UTTI,VTSI,HTSI,
     1DELANG,NWEDGE,XS,XCB,YCB,HTS,NTS,NTSP,UTL,UTT,VTS,LIMIT(3),
     2LIMIT(1),MSP)
      IF(NS.NE.0) CALL PRINT3(        MSP,MJ,NS,NWEDG,LIMIT(3),LIMIT(1),
     1LIMIT(8),LIMIT(9), RMA,XS,IWS,MS,TANGN,NTSP,NTCF,NTCV,PV)
      CALL PRINT4(MSP,CHI,RNU,NSP,TRPA,NUMCEL,FDN,WTM,DBA,NBS,TMPA,XVA,
     1YVA,ZVA,1,NBT,XC,YC,ZC,LEV,LKW)
      GO TO 353                                                         RUN6860
  350 CONTINUE
      CALL PRINT4(MSP,CHI,RNU,NSP,TRP,NUMCEL,FDN,WTM,DB,NB,TMP,XV,YV,ZV,
     10,NBF,XC,YC,ZC,LEV,LKW)
  353 IF(DEBUG(2)) WRITE(6,1)
      IF((TIME.LT.TLIM).AND.(CPUTYM.GT.CPJ)) GO TO 280
      IF(IC.EQ.ICOPY) RETURN
      IC=IC+1                                                           RUN7080
      WRITE(6,2)
      WRITE(6,4)
      WRITE(6,5) (((I,J,L,(ENT(I,J,K,L),K=1,6),L=1,NWEDG),J=1,MSP),I=1,2)
      WRITE(6,6)  ((J,L,ENTS(J,L),FTH(J,L),L=1,NWEDG),J=1,MSP)
      WRITE(6,2)                                                        RUN7120
      WRITE(6,3) IC                                                     RUN7130
      CALL PRINTA(THETAZ,NWEDGE,TITLE,NAME,XCB,YCB,TB,ALPHA,SIGMA,LD,LP,RUN7140
     1XLIM,COEFF,LIMIT,MSP)
      CALL PRINTB(FNA,MSP,FNB,LEV,LWP,NS,RLD,XLIM,XC,YC,ZC,NB,NUMCEL,LKW
     1,NSP)
      SAVE=.FALSE.                                                      RUN7180
      GO TO 345                                                         RUN7190
  360 WRITE(6,38) (DBG1(I,LARGE),I=1,3)                                 RUN7200
      IF(REDO) GO TO 364
      IF(DUMP) CALL ABEND(9)                                            RUN7220
      STOP                                                              RUN7230
  364 CONTINUE
      IF(NEW) GO TO 365
      READ(9) DENF,U,XREF,TEF,  KAWLS,NL,NW,NH,MW,MH,LW,LH,NXA,NXB,NCA
     1        ,NCB,NFA,NFB,NHA,NHB,BW,BH,BWB,BHB,BWC,BHC,XLB,XLC,PI,NREG
     2        ,S,SINANG,COSANG,AKN,NBI,RM,XR,ND,TIME,DTM,TI,ITS,ITP,TST
     3        ,TLIM,RMA,RNU,DIR,XSTART,JNM,MNM,MNB,TR,BZC,CN7,DRF,FCP
     4        ,FNA,HTF,INM,ITYPE,JTYPE,MJ,NAV,NMAX,NS,NWEDG,PRT,SAMP
     5        ,BTA,C1,C2,C3,C7,C8,DAM,DFA,FL,DELANG,FDN,HTI,HTR,JNT,KNM
     6        ,NM,WTM,C4,VRM,NCOL,LD,LP,LWP,RLD,CTI,CTR,CNI,CNR,LEV,SN
     7        ,ST,D1,D2,D3,D4,SSA,SSB,MS,NSP,NMB,NMC,NMP,NPB,NRAN,VELR
```

```
    8          ,IWS,TANGN,XLIM,COEFF,XCB,XS,YCB,TB,ALPHA,SIGMA,NTS,NTSF
    9          ,UTL,UTT,VTS,HTS,HTSI,ENT,REM,ENTS,REMS,FTH,THETA,DTH,TMPA
    A          ,DBA,NB,NBF,NBT,TMP,XV,XVA,YV,YVA,ZV,ZVA,T,DB,PNB,XC,YC,ZC
    B          ,NUMCEL,PAU,PAV,PAW,PAX,PAY,PAZ,FV,NTCV,NTCF,LPF,LCOL,LM
    C,ETA,PHI,CHI,CN,CM,CNG,CMG,CN8,TRP,TRPA,THETAZ,NWEDGE,MSP,ANGLE,TF
    D,UTLI,UTTI,VTSI,BR,RMB,LKW,NBS,LB,NBM,NBN
      REWIND 9
 365  JNM=9*JNM/10                                                    RUN7260
      ANM=INM
      INM=9*ANM/10
      DDN=.9*DDN
      DRF=DRF/.9
      FCF=FCF/.9
      HTF=HTF/.9
      DO 370 MM=1,MSP
      FDN(MM)=FDN(MM)*INM/ANM
      DO 366 KK=1,MSP
 366  CN8(KK,MM)=CN8(KK,MM)*.9
      DO 370 LT=1,NWEDG
      ENTS(MM,LT)=ENTS(MM,LT)*INM/ANM
      REMS(MM,LT)=0.0
      DO 370 NK=1,2
      DO 370 NJ=1,6
      ENT(NK,MM,NJ,LT)=ENT(NK,MM,NJ,LT)*INM/ANM
 370  REM(NK,MM,NJ,LT)=0.0
      IF(NEW) GO TO 220
      TST=TIME+TST
      TI=-1.
      PRT=ITP
      WRITE(6,2)
      WRITE(6,4)
      WRITE(6,5) (((I,J,L,(ENT(I,J,K,L),K=1,6),L=1,NWEDG),J=1,MSP),I=1,2)
      WRITE(6,6)  ((J,L,ENTS(J,L),FTH(J,L),L=1,NWEDG),J=1,MSP)
      WRITE(6,2)
      IF((LARGE.EQ.2).OR.(LARGE.EQ.3)) GO TO 280
      REDO=.FALSE.
      GO TO 360
      END                                                             RUN7410

      SUBROUTINE DIAG(N,ITEST,NUM)                                    DIAG010
      REAL*8 PARAM(10)/'  NWEDGE','    NREG','      ND','     NPX',
     1 MNM','     MNB','     NBX','      NS','      MJ','     MSP'/
                                                                      DIAG040
----------------------------------------------------------------------DIAG050
                                                                      DIAG060
                     FORMATS                                          DIAG070
                                                                      DIAG080
   32 FORMAT(9X,'ENT,REM,ENTS,REMS,FTH,THETA,DTH')                    DIAG090
   42 FORMAT(///5X,43H ARRAY DIMENSIONS ARE ABOUT TO BE VIOLATED./)   DIAG100
   44 FORMAT(5X,18H MAXIMUM VALUE IS I5,20H, WHEREAS YOU INPUT I5,3H  (,DIAG110
     1A8,1H))                                                         DIAG120
   56 FORMAT(/5X,78H IF YOU DESIRE TO USE THIS VALUE, THE FOLLOWING ARRADIAG130
     1YS MUST BE RE-DIMENSIONED./)                                    DIAG140
   62 FORMAT(9X,'HTS,HTSI,NTS,NTSF,UTL,UTT,VTS')                      DIAG150
   64 FORMAT(9X,'XLIM,COEFF'//11X,'NOTE THAT THE XLIM ARRAY MUST BE DIMEDIAG160
```

```
     1NSIONED TO 3 MORE THAN THE COEFF ARRAY.')                       DIAG170
  66 FORMAT(9X,'XCB,XS,YCB,TB,ALPHA,SIGMA')                           DIAG180
  68 FORMAT(9X,'DBA,NB,NBF,NBT,TMP,TMPA,XV,XVA,YV,YVA,ZV,ZVA,T,DB')   DIAG190
  70 FORMAT(9X,'LPF,PAU,PAV,PAW,PAX,PAY,PAZ,LCOL')                    DIAG200
  72 FORMAT(8X,3H LM)                                                 DIAG210
  74 FORMAT(//5X,76H IF YOU CHANGE THE ARRAY DIMENSIONS, ALSO CHANGE THDIAG220
     1E 'LIMIT' DATA STATEMENT.)                                      DIAG230
  75 FORMAT(9X,'ALL ARRAYS ASSOCIATED WITH SPECIES')
  76 FORMAT(9X,'FNB,XC,YC,ZC,NUMCEL')                                 DIAG240
  78 FORMAT(9X,'FV,NTCV,NTCF,MS,IWS,SL,DELS,TANGN')                   DIAG250
  80 FORMAT(9X,'FV,NTCV')                                             DIAG260
C                                                                     DIAG270
C---------------------------------------------------------------------DIAG280
C                                                                     DIAG290
      WRITE(6,42)                                                     DIAG300
      WRITE(6,44)ITEST,NUM,PARAM(N)                                   DIAG310
      WRITE(6,56)                                                     DIAG320
      GO TO (1,2,3,4,5,6,7,8,9,10),N
   1  WRITE(6,62)                                                     DIAG340
      WRITE(6,32)                                                     DIAG350
      GO TO 11
   2  WRITE(6,64)                                                     DIAG370
      GO TO 11
   3  WRITE(6,66)                                                     DIAG390
      WRITE(6,62)                                                     DIAG400
      GO TO 11
   4  WRITE(6,68)                                                     DIAG420
      GO TO 11
   5  WRITE(6,70)                                                     DIAG440
      GO TO 11
   6  WRITE(6,72)                                                     DIAG460
      GO TO 11
   7  WRITE(6,76)                                                     DIAG480
      GO TO 11
   8  WRITE(6,78)                                                     DIAG500
      GO TO 11
   9  WRITE(6,80)                                                     DIAG520
      GO TO 11
  10  WRITE(6,75)
  11  WRITE(6,74)
      STOP                                                            DIAG540
      END                                                             DIAG550

      SUBROUTINE PRINTA(THETAZ,NWEDGE,TITLE,NAME,XCB,YCB,TB,ALPHA,SIGMA,PRA0010
     1LD,LF,XLIM,COEFF,LIMIT,MSP)
      INTEGER TST,TLIM,TIME                                           PRA0030
      LOGICAL SAVE,NEW                                                PRA0040
      DIMENSION NWEDGE(2),LIMIT(1),TITLE(6),NAME(2),XCB(1),YCB(1),TB(1)
      DIMENSION ALPHA(3,1),SIGMA(3,1),LD(1),LF(1),XLIM(1),COEFF(4,1)   PRA0060
      DIMENSION RNU(3),RMA(3),CHI(3),DIR(3,3),PHI(3,3),ETA(3,3)
      DIMENSION WTM(3),DAM(3,3),VELS(3),XSP(3)
      COMMON /FIRST/NL,NW,NH,MW,MH,LW,LH,NXA,NXB,NCA,NCB,NFA,NFB,NHA,NHBPRA0070
      COMMON /SECND/BW,BH                                             PRA0080
      COMMON /THIRD/PI,NREG,S,SINANG,COSANG,AKN                       PRA0090
      COMMON /FIFTH/ND,TIME,DTM,TI,ITS,ITP,TST,TLIM,RMA,RNU,DIR       PRA0100
```

```
      COMMON /SIXTH/RMB,XSTART,JNM,MNM,MNB,NEW,SAVE,PERCNT,NSR,TR        PRA0110
      COMMON/EIGTH/DENF,U,TF,ANGLE,TRF,CHI,PHI,ETA,WTM,DAM,VELR,XREF
      DATA NOT/'NOT '/                                                   PRA0120
C                                                                        PRA0130
C------------------------------------------------------------------------PRA0140
C                                                                        PRA0150
C                    FORMATS                                             PRA0160
C                                                                        PRA0170
    1 FORMAT(16X,40('-'),T74,'I'//9X,'3-D',I2,'-FLUID PROGRAM - ')
    2 FORMAT('+',31X,A4)                                                 PRA0190
    3 FORMAT('+',35X,       'A RESTART OF A PREVIOUS RUN',T74,'I'/12X,2PRA0200
     1A4,' - ',6A4,'  - ',I2,' REGIONS',T74,'I',16(/T74,'I'))            PRA0210
    4 FORMAT(7X,'FRONT OF BODY =',E12.4,' XSTART  MAX HEIGHT =',E12.4,'
     1RMB',T74,'I'/7X,'X-LIMIT',T37,'BODY COEFFICIENTS',T74,'I')
    6 FORMAT(5F14.6,3X,'I')                                              PRA0240
   10 FORMAT(1X,72('-'))                                                 PRA0250
   12 FORMAT(//14X,'PARAMETERS OF SEGMENTS FOR BODY COLLISIONS',T96,'I'/
     18X,'X-COORD.   TEMP.    ALPHA1   ALPHA2   ALPHA3   SIGMA1   SIGMA2
     2   SIGMA3    AREAS',T96,'I')
   14 FORMAT(4X,E12.4,7F9.4,E12.4,T96,'I')
   16 FORMAT(23X,'WEIGHTING FACTORS'/1X,10I6,T96,'I')
   17 FORMAT(///25X,'ARRAY STORAGE USED'/5X,I6,'       *',10I6,T96,'I')
   18 FORMAT(1H1/17X,'LENGTH OF CELL IN MEAN-FREE-PATHS = ',F12.4,'  BW'
     A,T76,'I'
     1/17X,'HEIGHT OF CELL IN MEAN-FREE-PATHS = ',F12.4,'  BH',T76,'I'
     2/16X,'NUMBER OF L1 CELLS ALONG FLOW AXIS =',I13,'  NW',T76,'I'
     3/17X,'NUMBER OF L1 CELLS IN RADIAL DIR. =',I13,'  NH',T76,'I'
     4/21X,'NUMBER OF LEVELS OF CELL SIZE =',I13,'  NL',T76,'I')
   20 FORMAT(11X,'NUMBER OF L1 CELLS IN FRONT OF L2 CELLS =',I13,'  NFA'
     1,T76,'I'/15X,'NUMBER OF AXIAL SUBDIVIDED L1 CELLS =',I13,'  NCA',T
     276,'I'/14X,'NUMBER OF RADIAL SUBDIVIDED L1 CELLS =',I13,'  NHA',T7
     36,'I'/16X,'NUMBER OF L2 CELLS ALONG FLOW AXIS =',I13,'  MW',T76,'I
     4'/17X,'NUMBER OF L2 CELLS IN RADIAL DIR. =',I13,'  MH',T76,'I')
   22 FORMAT(11X,'NUMBER OF L2 CELLS IN FRONT OF L3 CELLS =',I13,'  NFB'
     1,T76,'I'/15X,'NUMBER OF AXIAL SUBDIVIDED L2 CELLS =',I13,'  NCB',T
     276,'I'/14X,'NUMBER OF RADIAL SUBDIVIDED L2 CELLS =',I13,'  NHB',T7
     36,'I'/16X,'NUMBER OF L3 CELLS ALONG FLOW AXIS =',I13,'  LW',T76,'I
     4'/17X,'NUMBER OF L3 CELLS IN RADIAL DIR. =',I13,'  LH',T76,'I')
   23 FORMAT(3X,'NUMBER OF AZIMUTHAL WEDGES IN LOWER ',I3,' DEGREES =',IPRA0470
     113,'  NWEDGE1 I'/3X,'NUMBER OF AZIMUTHAL WEDGES IN UPPER ',I3,' DE
     2GREES =',I13,'  NWEDGE2 I'/)
   24 FORMAT(16X,'BASIC TIME INTERVAL FOR COLLISIONS =',E13.4,'  DTM
     1 I'/8X,'TIME INTERVAL FOR SAMPLING FLOW FIELD INFO =',E13.4,'  DTS
     2      I'/24X,'TIME INTERVAL FOR PRINTING =',E13.4,'  DTP     I'/9X,
     3'TIME TO STEADY-STATE CONDITIONS (ASSUMED) =',E13.4,'  TST     I'/
     419X,'TIME AT WHICH RUN IS TERMINATED =',E13.4,'  TLIM    I'/)
   26 FORMAT(9X,'INITIAL NUMBER OF MOLECULES - EITHER TYPE =',I13,'  INM
     1      I'/9X,'MAXIMUM NUMBER OF MOLECULES - EITHER TYPE =',I13,'  MN
     2M      I'/1X,'MAX NUMBER OF MOLECULES IN ANY CELL - EITHER TYPE =',PRA0570
     3I13,'  MNB      I')
   27 FORMAT(//22X,'VELOCITY OF FREE STREAM FLOW =',E13.4,' U',T76,'I'/1
     19X,'SPEED RATIO OF FREE STREAM FLOW =',E13.4,' S',T76,'I'/19X,'MAC
     AH NUMBER OF FREE STREAM FLOW =',E13.4,' M',T76,'I'/19X,'SPECIFIC H
     BEAT RATIO (CALCULATED)=',E13.4,' GAMMA',T76,'I'/           35X,'ANG
     2LE OF ATTACK =',F13.4,' ANGLE    I'/16X,'NUMBER DENSITY OF FREE ST
```

```
   3REAM FLOW =',E13.4,' N',T76,'I'/19X,'TEMPERATURE OF FREE STREAM FL
   4OW =',F13.4,' TF',T76,'I'/16X,'MOLE FRACTIONS OF FREE STREAM FLOW
   5=',3E13.4,' RNU I'/16X,'MOLECULAR WEIGHTS OF SPECIES ABOVE =',3F13
   6.4,' RMA I')
 28 FORMAT(//10X,'REFERENCE TEMPERATURE FOR MOLECULAR DATA =',F12.4,'
   1TRF',T90,'I'/14X,'CROSS-SECTION',26X,'TEMP EXPONENT',T90,'I'/3(3X,
   23E12.4,3X,3F12.6,T90,'I'/)/ 5X,'CHI/2-1',11X,'ROTATIONAL PARAMETER
   3 PHI',T90,'I'/3(F12.4,5X,3F12.6,T90,'I'/))                          PRA0670
 29 FORMAT(9X,'DATA SAVED ON TAPE 9')
 30 FORMAT(//31X,'REF MOLECULAR SPEED =',E13.4,' VELR',T76,'I'/20X,'SP
   1ECIES FREE STREAM MOLECULAR SPEEDS',T76,'I'/14X,3E16.6,T76,'I'/26X
   2,'REFERENCE MEAN FREE PATH =',E13.4,' XREF',T76,'I'/26X,'SPECIES M
   3EAN FREE PATHS',T76,'I'/14X,3E16.6,T76,'I'/11X,'LONGITUDINAL KNUDS
   4EN NUMBER (CALCULATED)=',E13.4,' AKN',T76,'I'/13X,'TRANSVERSE KNUD
   5SEN NUMBER (CALCULATED)=',E13.4,' AKT',T76,'I')
                                                                        PRA0680
C-----------------------------------------------------------------------PRA0690
                                                                        PRA0700
      IARRAY=708+LIMIT(3)*(32+56*LIMIT(1))+20*LIMIT(2)+LIMIT(4)*(120+4*LPRA0710
     1IMIT(6))+56*LIMIT(5)+LIMIT(8)*(68+96*LIMIT(9))+20*LIMIT(7)+224*LIMPRA0720
     2IT(1)                                                             PRA0730
      WRITE(6,1) MSP                                                    PRA0750
      IF(NEW) WRITE(6,2) NOT                                            PRA0760
      WRITE(6,3) NAME,TITLE,NREG                                        PRA0770
      WRITE(6,4)XSTART,BMB                                              PRA0780
      DO 100 I=1,NREG                                                   PRA0790
  100 WRITE(6,6)XLIM(I+2),(COEFF(J,I),J=1,4)                            PRA0800
      WRITE(6,10)                                                       PRA0810
      WRITE(6,12)                                                       PRA0820
      DO 110 I=1,ND
  110 WRITE(6,14) XCB(I),TB(I),(ALPHA(J,I),J=1,3),(SIGMA(J,I),J=1,3),YCB
     1(I)                                                               PRA0850
      WRITE(6,10)                                                       PRA0860
      WRITE(6,16)(LD(N),LF(N),N=1,5)
      WRITE(6,17)IARRAY,(LIMIT(I),I=1,10)                               PRA0880
      WRITE(6,18)BW,BH,NW,NH,NL                                         PRA0890
      IF(NL.GT.1) WRITE(6,20)NPA,NCA,NHA,MW,MH                          PRA0900
      IF(NL.GT.2) WRITE(6,22)NPB,NCB,NHB,LW,LH                          PRA0910
      IETAZ=THETAZ                                                      PRA0920
      JETAZ=180-IETAZ                                                   PRA0930
      WRITE(6,23)IETAZ,NWEDGE(1),JETAZ,NWEDGE(2)                        PRA0940
      DTS=DTM*ITS                                                       PRA0950
      DTP=DTM*ITP                                                       PRA0960
      AST=DTM*TST                                                       PRA0970
      ALIM=DTM*TLIM
      CHT=0.0
      DO 120 J=1,MSP
  120 CHT=CHT+CHI(J)*RNU(J)
      GAMMA=(7.+2.*CHT)/(5.+2.*CHT)
      AM=S*SQRT(2./GAMMA)                                               PRA0980
      WRITE(6,24)DTM,DTS,DTP,AST,ALIM
      WRITE(6,26) JNM,MNM,MNB
      WRITE(6,27)U,S,AM,GAMMA,ANGLE,DENP,TF,(RNU(I),I=1,3),(RMA(I),I=1,3
     1)
      WRITE(6,28) TRF,((DIR(I,K),K=1,3),(ETA(I,K),K=1,3),I=1,3),(CHI(I),
```

```
     1(PHI(I,K),K=1,3),I=1,3)
      DO 210 I=1,3
      VELS(I)=0.0
  210 XSP(I)=0.0
      DO 220 J=1,MSP
      VELS(J)=VELR/SQRT(WTM(J))
      XT=0.0
      DO 215 M=1,MSP
  215 XT=XT+RNU(M)*DAM(J,M)*SQRT(1.+WTM(J)/WTM(M))
  220 XSP(J)=1.414214*XREF/XT
      AKT=1./RMB
      WRITE(6,30) VELR,(VELS(I),I=1,3),XREF,(XSP(I),I=1,3),AKN,AKT
      IF(SAVE) WRITE(6,29)                                            PRA1040
      RETURN                                                          PRA1050
      END                                                             PRA1060
:
      SUBROUTINE PRINTB(FNA,MSP,FNB,LEV,LWP,NM,RLD,XLIM,XC,YC,ZC,NB,
     1NUMCEL,LKW,N)
      INTEGER*2 LKW,NB,NUMCEL
      DIMENSION FNB(1),LEV(1),LWP(1),NM(1),RLD(1),XLIM(1),XC(1)
      DIMENSION YC(1),ZC(1),NB(N,1),NUMCEL(1),LKW(1)
      COMMON /FIRST/NL                                                PRB0050
      COMMON /THIRD/PI,NREG,S,SINANG,COSANG,AKN                       PRB0060
      COMMON /FORTH/NBX                                               PRB0070
    1 FORMAT(1H1)                                                     PRB0080
    2 FORMAT(2X,'-----------------------------CELL GEOMETRY------------PRB0090
     1-----------'/2X, 'BOX  LEVEL   POSITION OF CENTER    VOLUME  WPRB0100
     2EIGHTING  POPULATION'/2X,'NUM.',12X,'X',7X,'Y    THETA',12X,'FACTOPRB0110
     3R  ',15X,'  CELL#')
    3 FORMAT(1X,I4,I5,3X,2F8.3,F7.1,E12.3,2X,I2,4X,3I5,3X,I4)
    4 FORMAT(2X,'---------------TOTALS-------------',E12.4,8X,3I5)
      WRITE(6,1)                                                      PRB0150
      WRITE(6,2)
      DO 200 I=1,NBX                                                  PRB0170
      IF(NUMCEL(I).EQ.0) GO TO 200
      X=(XC(I)-XLIM(2))*AKN                                           PRB0180
      Y=YC(I)*AKN                                                     PRB0190
      LEVEL=1                                                         PRB0200
      IF(NL.LT.2) GO TO 120                                           PRB0210
      IF(I.LT.LEV(1)) GO TO 120                                       PRB0220
      LEVEL=2                                                         PRB0230
      IF(NL.LT.3) GO TO 120                                           PRB0240
      IF(I.LT.LEV(2)) GO TO 120                                       PRB0250
      LEVEL=3                                                         PRB0260
  120 CONTINUE
      J=NUMCEL(I)                                                     PRB0320
      M1=NB(1,J)
      M2=0
      IF(MSP.GE.2) M2=NB(2,J)
      M3=0
      IF(MSP.GE.3) M3=NB(3,J)
  140 WRITE(6,3) I,LEVEL,X,Y,ZC(I),FNB(I),LKW(J),M1,M2,M3,J
  200 CONTINUE                                                        PRB0370
      NM2=0
      IF(MSP.GE.2) NM2=NM(2)
```

```
      NM3=0
      IF(NSP.GE.3) NM3=NM(3)
      WRITE(6,4)FNA,NM(1),NM2,NM3
      RETURN
      END

      SUBROUTINE SIMPSN(A,B,L,INTGRL,PERCNT,COEFF,PIROOT,SUM1,FUN)
      REAL INTGRL
      DIMENSION COEFF(4,1)
-----------------------------------------------------------------------

        THE PURPOSE OF THIS SUBROUTINE IS TO PERFORM A SIMPSON'S RULE
        INTEGRATION.

        FORMAT

    4 FORMAT(/30H TOO MANY ITERATIONS. TEST IS E15.7,14H, INTEGRAL IS E1
     15.7,14H IS THE RANGE E15.7,4H TO E15.7)
-----------------------------------------------------------------------
      PREV=0.0
      N=4
      N1=3
      G=.5*(B-A)+A
      SUME=FUN(G,PIROOT,L,COEFF)
      SUMO=0.0
      K=0
  235 SUME=SUME+SUMO
      N1=N1+K
      K=N1+1
      IF(K.LT.5000) GO TO 237
      WRITE(6,4) TEST,PREV,A,B
      STOP
  237 SUMO=0.0
      DO 240 I=1,N1,2
      G=(I*(B-A))/K+A
  240 SUMO=SUMO+FUN(G,PIROOT,L,COEFF)
      INTGRL=(SUM1+4.*SUMO+2.*SUME)/(3.*K)
      TEST=ABS(2.-4.*PREV/(INTGRL+PREV))
      PREV=INTGRL
      IF(TEST.GT.PERCNT) GO TO 235
      RETURN
      END

      FUNCTION FNCTN(X,PIROOT,L,COEFF)
      DIMENSION COEFF(4,1)
-----------------------------------------------------------------------
        THE PURPOSE OF THIS FUNCTION IS TO EVALUATE THE INTEGRAND USED IN
        THE SIMPSON-S RULE INTEGRATION ROUTINE.
-----------------------------------------------------------------------
      A=COEFF(1,L)
      B=COEFF(2,L)
      C=COEFF(3,L)
      D=COEFF(4,L)
      AA=4.*(A-B)*(A*X+C)*X+C*C-4.*B*D
```


C-2

```
      IF(AA.LT.0.) AA=0.
      FNCTN=SQRT(AA)
      RETURN
      END

      SUBROUTINE HEIGHT(X,R,L,COEFF,I)
      DIMENSION COEFF(4,1),DBG2(2,3)
      DATA DBG2/'SBTR','CT  ','GAS ','    ',' RUN','    '/
-----------------------------------------------------------------------
       THE PURPOSE OF THIS SUBROUTINE IS TO COMPUTE THE R-COORDINATE OF
       THE BODY CURVE AT ANY GIVEN X-COORDINATE, IN THE X-R PLANE.
-----------------------------------------------------------------------
    2 FORMAT(//' MESSAGE ',I2,E17.8,3X,2A4)
      ARG=-((COEFF(1,L)*X+COEFF(3,L))*X+COEFF(4,L))/COEFF(2,L)
      IF(ARG.GE.0.) GO TO 100
      WRITE(6,2)L,X,(DBG2(M,I),M=1,2)
      ARG=0.
  100 R=SQRT(ARG)
      RETURN
      END

      FUNCTION FNCTM(ARG,PIROOT,L,COEFF)
      DIMENSION COEFF(4,1)
      D=0.
      IF(ABS(ARG).LT.10.) D=EXP(-ARG*ARG)/PIROOT
      E=0.
      IF(ARG.GT.-10.) E=ARG*ERRF(ARG)
      TEM=.5*(D+E)
      FNCTM=TEM
      RETURN
      END

      SUBROUTINE CELL(TH,LEV,A,B,K,KH,XO,I,J,DELANG,NWEDGE,XC,YC,ZC,FNB)
      DIMENSION DELANG(1),NWEDGE(1),XC(1),YC(1),ZC(1),FNB(1)
      COMMON /THIRD/PI
-----------------------------------------------------------------------

       THE PURPOSE OF THIS SUBROUTINE IS TO
           1. COMPUTE THE VOLUME OF EACH CELL (ALL 3 POSSIBLE LEVELS)
              AND STORE THE RESULT IN THE ARRAY CALLED 'FNB'.
           2. COMPUTE THE X, R, AND THETA COORDINATES OF THE CENTER OF
              EACH CELL (ALL 3 POSSIBLE LEVELS) AND STORE THE RESULTS IN
              ARRAYS CALLED 'XC', 'YC', AND 'ZC'.

-----------------------------------------------------------------------
      INDEX=I+J
      ZO=0.
      DO 120 MT=1,LEV
      ICNT=NWEDGE(MT)
      ANGLE=DELANG(MT)
      FACTOR=ANGLE/180.*PI*B*B*A
      Z=ZO-.5*ANGLE
      DO 110 L=1,ICNT
      Z=Z+ANGLE
      Y=-.5*B
```

```
      DO 110 N=1,KH
      X=XO-.5*A
      Y=Y+B
      DO 110 M=1,K
      X=X+A
      INDEX=INDEX+1
      XC(INDEX)=X
      YC(INDEX)=Y
      ZC(INDEX)=Z
  110 FNB(INDEX)=FACTOR*(2*N-1)
      ZO=TH
  120 CONTINUE
      RETURN
      END

      SUBROUTINE ZERO(NWIDE,NHI,NBEG,NLONG,MUP,NAREA,ICNT,FNB)
      DIMENSION FNB(1)
-------------------------------------------------------------------

       THIS SUBROUTINE SETS THE SIZES TO ZERO OF THOSE CELLS WHICH ARE
       TO BE SUBDIVIDED INTO SMALLER CELLS.

-------------------------------------------------------------------
      NGO=NBEG+1
      NSTOP=NBEG+NLONG
      DO 110 N=NGO,NSTOP
      DO 110 M=1,MUP
      DO 110 L=1,ICNT
      INDEX=NWIDE*(NHI*(L-1)+M-1)+N+NAREA
  110 FNB(INDEX)=0.0
      RETURN
      END

      SUBROUTINE SBTRCT(NGO,NTEMP,NSTOP,BWIDTH,BHITE,DELANG,XC,YC,FNB,XL
     1IM,COEFF)
      DIMENSION DELANG(1),XC(1),YC(1),FNB(1),XLIM(1),COEFF(4,1)
      COMMON /THIRD/PI,NBEG
-------------------------------------------------------------------

       THIS SUBROUTINE SUBTRACTS FROM EACH CELL SIZE ('FNB' ARRAY) THAT
       PORTION OCCUPIED BY THE BODY.

-------------------------------------------------------------------
      FACTOR=PI*DELANG(1)/180.
      DO 150 N=NGO,NSTOP
      IF(FNB(N).LE.0.) GO TO 150
      DFNB=.005*FNB(N)
      SLICE=.01*BWIDTH
      IF(N.GT.NTEMP) FACTOR=PI*DELANG(2)/180.
      X=XC(N)-.5*(BWIDTH+SLICE)
      YBOT=YC(N)-.5*BHITE
      YTOP=YBOT+BHITE
      DO 130 M=1,100
      X=X+SLICE
      IF(X.LE.XLIM(2)) GO TO 130
```

```
      DO 120 L=1,NREG                                                   SBCT230
      IF(X.LT.XLIM(L+2)) GO TO 125                                      SBCT240
  120 CONTINUE                                                          SBCT250
      GO TO 130                                                         SBCT260
  125 CALL HEIGHT(X,YBODY,L,COEFF,1)                                    SBCT270
      IF(YBODY.LE.YBOT) GO TO 130                                       SBCT280
      YTEMP=YTOP                                                        SBCT290
      IF(YBODY.LT.YTOP) YTEMP=YBODY                                     SBCT300
      FNB(N)=FNB(N)-SLICE*(YTEMP*YTEMP-YBOT*YBOT)*FACTOR                SBCT310
  130 CONTINUE                                                          SBCT320
      IF(FNB(N).LT.DFNB) FNB(N)=0.                                      SBCT330
  150 CONTINUE                                                          SBCT340
      RETURN                                                            SBCT350
      END                                                               SBCT360

      SUBROUTINE IMPACT(RM,G1,G2,G3,ET,EI,PHI,CHI,ETA,XM,CIM)
      COMMON/THIRD/PI
      IF(PHI.EQ.0.) GO TO 20
      IF(CHI.EQ.0.) GO TO 20
      DF=PHI*CHI-1.
      DS=PHI*(2.-.5*ETA)-1.
      E=ET+EI
   10 X=RAND(0)
      IF(X.EQ.0.0) GO TO 10
      XT=X**DF*(1.-X)**DS
      IF(XT.GT.XM) GO TO 15
      CIM=CIM+XT
      IF(CIM.LT.XM) GO TO 10
      CIM=CIM-XM
   15 ET=(1.-PHI)*ET+(1.-X)*PHI*E
      EI=(1.-PHI)*EI+X*PHI*E
   20 GP=SQRT(ET/RM)
      EP=2.*PI*RAND(0)
      CSX=2.*RAND(0)-1.
      SSX=SQRT(1.-CSX**2)
      G1=GP*CSX
      G2=GP*SSX*COS(EP)
      G3=GP*SSX*SIN(EP)
      RETURN
      END

      SUBROUTINE GAS(NWEDG,THETAZ,DELANG,NWEDGE,BTA,C1,DPA,NM,RLD,LWF,FNGAS0010
     1B,DB,NB,NBF,LPF,PAU,PAV,PAW,PAX,PAY,PAZ,XLIM,COEFF,LM,I2,I3,LARGE,GAS0020
     2MNM,MNB,DEBUG1,LCOL,NUMCEL,IP,ER,CHI,CNG,CMG,I,LB,NBM,NBN)
      INTEGER*2 LM(I,1),LPF(I,1),LCOL(I,1),LB(1),NBM(I,1),NBN(1)
      INTEGER*2 NB,NBF,NUMCEL
      INTEGER Q                                                         GAS0050
      LOGICAL DUMP,DEBUG1                                               GAS0060
      DIMENSION DELANG(2),NWEDGE(2),NUMCEL(1)                           GAS0070
      DIMENSION BTA(1),C1(1),DPA(1),NM(1),RLD(1),LWF(1),FNB(1),CHI(1)
      DIMENSION DB(I,1),NB(I,1),NBF(I,1),PAU(I,1),PAV(I,1),PAW(I,1)
      DIMENSION PAX(I,1),PAY(I,1),PAZ(I,1),ER(I,1),COEFF(4,1),XLIM(1)
      DIMENSION CNG(1),CMG(1)
      COMMON /FIRST/NL,NW,NH,MW,MH,LW,LH,NXA,NXB                        GAS0110
      COMMON /SECND/BW,BH,BWB,BHB,BWC,BHC,XLB,XLC                       GAS0120
```

ORIGINAL PAGE IS
OF POOR QUALITY


```
      COMMON /[illegible]/[illegible],COSANG
      COMMON /[illegible]/[illegible],LL
C---------------------------------------------------------------------
C
C     THE PURPOSE OF THIS SUBROUTINE IS TO
C        1. COMPUTE THE INITIAL VELOCITY OF EACH MOLECULE.
C           THE [illegible] ARE 'PAU', 'PAV', AND 'PAW'.
C        2. COMPUTE THE INITIAL POSITION OF EACH MOLECULE.
C           THE [illegible] ARE 'PAX', 'PAY', AND 'PAZ'.
C        3. [illegible] THE [illegible] POPULATIONS-
C           [illegible]
C           [illegible]
C        4. [illegible]
C           [illegible]
C        5. [illegible]
C           [illegible]
C
C---------------------------------------------------------------------
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
  [illegible] [illegible]
      [illegible]
  [illegible] [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      PAU(NT,N)=[illegible]
      PAV(NT,N)=[illegible]
      PAW(NT,N)=[illegible]
      ER(NT,N)=0.0
      IF(CHI(NT).LE.-1.) GO TO 113
  125 X=9.*RAND(0)
      IF(X.EQ.0.0) GO TO 125
      XT=X**CHI(NT)*EXP(-X)
      IF(XT.GE.CNG(NT)) GO TO 126
      CNG(NT)=CNG(NT)+XT
      IF(CNG(NT).LT.CRG(NT)) GO TO 125
      CNG(NT)=CNG(NT)-CRG(NT)
  126 ER(NT,N)=X
  130 Y=YR*RAND(0)
  140 R=RM*RAND(0)
      LA=0
  150 LA=LA+1
      IF(R.GT.RLD(LA)) GO TO 150
      A=LNF(LA)
```

```
      C9=C9+R/A
      IF(C9.LT.RWFM) GO TO 140
      C9=C9-RWFM
      IF(X.LE.XLIM(2)) GO TO 159
      DO 152 L=1,NREG
      IF(X.LT.XLIM(L+2)) GO TO 154
  152 CONTINUE
      GO TO 159
  154 CALL HEIGHT(X,YBODY,L,COEFF,2)
      IF(R.LT.YBODY) GO TO 130
  159 PAX(MT,N)=X
      D=PI*RAND(0)
      PAY(MT,N)=R*COS(D)
      PAZ(MT,N)=R*SIN(D)
      TANG=180.*(1.-D/PI)
      IWDGE=TANG/DELANG(1)
      IF((IWDGE.GE.NWEDGE(1)).AND.(DELANG(2).NE.0.)) IWDGE=(TANG-THETAZ)
     1/DELANG(2)+NWEDGE(1)
      IF(IWDGE.GE.NWEDG) IWDGE=NWEDG-1
      L=X/BW+1.
      IF(L.GT.NW) L=NW
      M=R/BH
      IF(M.GE.NH) M=NH-1
      K=(IWDGE*NH+M)*NW+L
      IF(K.LE.NXA) GO TO 160
      WRITE(6,2)L,M,K,MT,N,DELANG,NWEDGE,NWEDG,D,TANG,IWDGE,NH,NW
      IF(DUMP) CALL ABEND(11)
      STOP
  160 IF(NL.EQ.1) GO TO 162
      IF(IWDGE.GE.NWEDGE(1)) GO TO 162
      IF(FNB(K).GT.0.) GO TO 162
      L=(X-XLB)/BWB+1.
      IF(L.GT.MW) L=MW
      M=R/BHB
      IF(M.GE.MH) M=MH-1
      K=(IWDGE*MH+M)*MW+L+NXA
      IF(K.LE.NXA+NXB) GO TO 161
      WRITE(6,2)L,M,K,MT,N,DELANG,NWEDGE,NWEDG,D,TANG,IWDGE,MH,MW,NXA
      IF(DUMP) CALL ABEND(12)
      STOP
  161 IF(NL.EQ.2) GO TO 162
      L=(X-XLC)/BWC+1.
      IF(FNB(K).GT.0.) GO TO 162
      IF(L.GT.LW) L=LW
      M=R/BHC
      IF(M.GE.LH) M=LH-1
      K=(IWDGE*LH+M)*LW+L+NXA+NXB
      IF(K.LE.NBX) GO TO 164
      WRITE(6,2)L,M,K,MT,N,DELANG,NWEDGE,NWEDG,D,TANG,IWDGE,LH,LW,NXA,NX
     1B
      IF(DUMP) CALL ABEND(13)
      STOP
  162 IF(FNB(K).GT.0.) GO TO 164
      WRITE(6,3)L,M,K,MT,N,FNB(K)
      IF(DUMP) CALL ABEND(14)
```

```
      STOP                                                           GAS1120
  164 Q=NUMCEL(K)                                                    GAS1130
      IF(Q.GT.0) GO TO 165                                           GAS1140
      WRITE(6,2)Q,L,M,K,N,DELANG,NWEDGE,NWEDG,D,TANG,INDGE,LH,LW,NXA,NXBGAS1150
     1,X,R                                                           GAS1160
      IF(DUMP) CALL ABEND(15)                                        GAS1170
      STOP                                                           GAS1180
  165 J=NB(MT,Q)+1                                                   GAS1190
      KW=LWF(LA)                                                     GAS1200
      LPF(MT,N)=KW                                                   GAS1210
      LCOL(MT,N)=0                                                   GAS1220
      LLC=LLC+KW                                                     GAS1230
      IF(J.LE.MNB) GO TO 166                                         GAS1240
      IF(DEBUG1) WRITE(6,4)MT,Q,MNB                                  GAS1250
      GO TO 167                                                      GAS1260
  166 NB(MT,Q)=J                                                     GAS1270
      LB(N)=Q
      NBF(MT,Q)=NBF(MT,Q)+KW                                         GAS1290
  167 IF(LLC.LT.LL) GO TO 110                                        GAS1300
      NM(MT)=N                                                       GAS1310
      NBM(MT,1)=0
      DO 170 Q=1,NBX                                                 GAS1330
      N=NUMCEL(Q)
      IF(N.EQ.0) GO TO 170
      A=NBF(MT,N)                                                    GAS1360
      DB(MT,N)=A*DFA(MT)/FNB(Q)                                      GAS1370
      NBM(MT,N+1)=NBM(MT,N)+NB(MT,N)
      NBN(N)=NBM(MT,N)
  170 CONTINUE                                                       GAS1380
      NG=NM(MT)
      DO 175 N=1,NG
      Q=LB(N)
      NBN(Q)=NBN(Q)+1
      NA=NBN(Q)
  175 LM(MT,NA)=N
  180 CONTINUE                                                       GAS1390
      RETURN                                                         GAS1400
  190 LARGE=1                                                        GAS1410
      RETURN                                                         GAS1420
      END                                                            GAS1430

      SUBROUTINE FLOW(NWEDG,MNM,LARGE,BTA,C1,C7,C8,D1,D2,D3,D4,DTH,NM,SMFLO00010
     1,ST,THETA,LWF,RLD,FTH,ENTS,REMS,SSA,SSB,PAU,PAV,PAW,PAX,PAY,PAZ,LPFFLO00020
     2F,ENT,REM,LCOL,IP,ER,CHI,CNG,CMG,I)
      INTEGER*2 LPF,LCOL
      DIMENSION BTA(1),D1(1),D2(1),D3(1),D4(1),DTH(1),NM(1),SM(1),ST(1) FLO00040
      DIMENSION C1(1),C7(1),C8(1),LWF(1),RLD(1),THETA(1),SSA(2,1)    FLO00050
      DIMENSION SSB(2,1),PAU(I,1),PAV(I,1),PAW(I,1),PAX(I,1),PAY(I,1) FLO00060
      DIMENSION PAZ(I,1),LPF(I,1),ENT(2,3,6,1),REM(2,3,6,1),LCOL(I,1)
      DIMENSION ENTS(3,1),REMS(3,1),FTH(3,1),ER(I,1),CHI(1),CNG(1),
     1CMG(1)
      COMMON /THIRD/PI                                               FLO00100
      COMMON /FORTH/NBX,RM,XR                                        FLO00110
C-----------------------------------------------------------------------FLO00120
C     THE PURPOSE OF THIS SUBROUTINE IS TO ADD A NEW BATCH OF MOLECULESFLO00130
```

```
      TO THE SAMPLE THROUGH THE UPSTREAM BOUNDARY.                        FLO0140
--------------------------------------------------------------------------FLO0150
      DO 370 MT=1,IP
      ARG=SN(MT)                                                          FLO0170
      XGO=0.                                                              FLO0180
      E=1.                                                                FLO0190
      DO 180 NT=1,2                                                       FLO0200
      SM=AMAX1(0.,ARG-4.)                                                 FLO0210
      SSM=AMAX1(0.,ARG)                                                   FLO0220
      TEMPC=0.                                                            FLO0230
      DO 170 LA=1,6                                                       FLO0240
      TEMPB=RLD(LA)*RLD(LA)                                               FLO0250
      AY = TEMPB-TEMPC                                                    FLO0260
      C = TEMPC                                                           FLO0270
      TEMPC=TEMPB                                                         FLO0280
      DO 170 K=1,NWEDG                                                    FLO0290
      AM=ENT(NT,MT,LA,K)+REM(NT,MT,LA,K)                                  FLO0300
      M=AM                                                                FLO0310
      AMM=M                                                               FLO0320
      REM(NT,MT,LA,K)=AM-AMM                                              FLO0330
      IF(M.EQ.0) GO TO 170                                                FLO0340
      DY =AY/AMM                                                          FLO0350
      DO 160 N=1,M                                                        FLO0360
      IF(NM(MT).GE.MNM) GO TO 380
      NM(MT)=NM(MT)+1                                                     FLO0370
      NMX=NM(MT)                                                          FLO0390
      R = SQRT(C + DY*(N+RAND(0) -1.))                                    FLO0400
      D=(THETA(K)+RAND(0)*DTH(K))*PI/180.                                 FLO0410
      PAY(MT,NMX)=R*COS(D)                                                FLO0420
      PAZ(MT,NMX)=R*SIN(D)                                                FLO0430
      LPF(MT,NMX)=LWF(LA)                                                 FLO0440
      LCOL(MT,NMX)=0                                                      FLO0450
  130 V=SM+RAND(0)*(SSM+4.-SM)                                            FLO0460
      C1(MT)=C1(MT)+2.*V*EXP(SSB(NT,MT)+2.*ARG*V-V*V)/SSA(NT,MT)          FLO0470
      IF(C1(MT).LT.1.) GO TO 130                                          FLO0480
      C1(MT)=C1(MT)-1.                                                    FLO0490
      PAU(MT,NMX)=E*V/BTA(MT)                                             FLO0500
  140 V=8.*RAND(0)-4.                                                     FLO0510
      C7(MT)=C7(MT)+EXP(-V*V)                                             FLO0520
      IF(C7(MT).LT.1.) GO TO 140                                          FLO0530
      C7(MT)=C7(MT)-1.                                                    FLO0540
      PAV(MT,NMX)=(V+ST(MT))/BTA(MT)                                      FLO0550
  150 V=8.*RAND(0)-4.                                                     FLO0560
      C8(MT)=C8(MT)+EXP(-V*V)                                             FLO0570
      IF(C8(MT).LT.1.) GO TO 150                                          FLO0580
      C8(MT)=C8(MT)-1.                                                    FLO0590
      PAW(MT,NMX)=V/BTA(MT)                                               FLO0600
      ER(MT,NMX)=0.0
      IF(CHI(MT).LE.-1.) GO TO 160
  125 X=9.*RAND(0)
      IF(X.LE.0.0) GO TO 125
      XT=X**CHI(MT)*EXP(-X)
      IF(XT.GE.CMG(MT)) GO TO 126
      CNG(MT)=CNG(MT)+XT
      IF(CNG(MT).LT.CMG(MT)) GO TO 125
```

```
      CNG(MT)=CNG(MT)-CMG(MT)
  126 ER(MT,NMX)=X
  160 PAX(MT,NMX)=XGO
  170 CONTINUE                                                          FL00610
      ARG=-ARG                                                          FL00620
      XGO=XR                                                            FL00630
      E=-1.                                                             FL00640
  180 CONTINUE                                                          FL00650
      DO 370 K=1,NWEDG                                                  FL00660
      AM=ENTS(MT,K)+REMS(MT,K)                                          FL00670
      M=AM                                                              FL00680
      AMM=M                                                             FL00690
      REMS(MT,K)=AM-AMM                                                 FL00700
      IF(M.EQ.0) GO TO 370                                              FL00710
      DX=XR/AMM                                                         FL00720
      DO 365 N=1,M                                                      FL00730
      IF(NM(MT).GE.MNM) GO TO 380                                       FL00740
      NM(MT)=NM(MT)+1
      NMX=NM(MT)                                                        FL00750
      PAX(MT,NMX)=(N-1.+RAND(0))*DX                                     FL00770
  330 TH=(THETA(K)+RAND(0)*DTH(K))*PI/180.                              FL00780
      A=COS(TH)                                                         FL00790
      B=SIN(TH)                                                         FL00800
      SM=ST(MT)*A                                                       FL00810
      C=0.                                                              FL00820
      IF(ABS(SM).LT.10.) C=EXP(-SM*SM)                                  FL00830
      D=0.                                                              FL00840
      IF(SM.GT.-10.) D=SQRT(PI)*SM*ERRF(SM)                             FL00850
      D1(MT)=D1(MT)+(C+D)/FTH(MT,K)                                     FL00860
      IF(D1(MT).LT.1.) GO TO 330                                        FL00870
      D1(MT)=D1(MT)-1.                                                  FL00880
      PAY(MT,NMX)=-RM*A                                                 FL00890
      PAZ(MT,NMX)=RM*B                                                  FL00900
      VNM=.5*SM+SQRT(.25*SM*SM+.5)                                      FL00910
      VM=AMAX1(0.,SM-4.)                                                FL00920
  340 V=VM+RAND(0)*(SM+4.-VM)                                           FL00930
      D2(MT)=D2(MT)+V*EXP(VNM*(VNM-2.*SM)-V*(V-2.*SM))/VNM              FL00940
      IF(D2(MT).LT.1.) GO TO 340                                        FL00950
      D2(MT)=D2(MT)-1.                                                  FL00960
      VN=V                                                              FL00970
  350 V=8.*RAND(0)-4.                                                   FL00980
      D3(MT)=D3(MT)+EXP(-V*V)                                           FL00990
      IF(D3(MT).LT.1.) GO TO 350                                        FL01000
      D3(MT)=D3(MT)-1.                                                  FL01010
      VT1=SN(MT)+V                                                      FL01020
  360 V=8.*RAND(0)-4.                                                   FL01030
      D4(MT)=D4(MT)+EXP(-V*V)                                           FL01040
      IF(D4(MT).LT.1.) GO TO 360                                        FL01050
      D4(MT)=D4(MT)-1.                                                  FL01060
      VT2=ST(MT)*B+V                                                    FL01070
      PAU(MT,NMX)=VT1/BTA(MT)                                           FL01080
      PAV(MT,NMX)=(VN*A+VT2*B)/BTA(MT)                                  FL01090
      PAW(MT,NMX)=(-VN*B+VT2*A)/BTA(MT)                                 FL01100
      LCOL(MT,NMX)=0                                                    FL01110
      ER(MT,NMX)=0.0                                                    FL01120
```

```
      IF(CHI(MT).LE.-1.) GO TO 365
  225 X=9.*RAND(0)
      IF(X.EQ.0.0) GO TO 225
      XT=X**CHI(MT)*EXP(-X)
      IF(XT.GE.CMG(MT)) GO TO 226
      CNG(MT)=CNG(MT)+XT
      IF(CNG(MT).LT.CMG(MT)) GO TO 225
      CNG(MT)=CNG(MT)-CMG(MT)
  226 ER(MT,NMX)=X
  365 LPF(MT,NMX)=LWF(6)                                                FLO1130
  370 CONTINUE                                                          FLO1140
      RETURN                                                            FLO1150
  380 LARGE=2                                                           FLO1160
      RETURN                                                            FLO1170
      END                                                               FLO1180

      SUBROUTINE COLIDE(CN,CM,WTM,DB,DBA,NB,NCOL,LCOL,PAU,PAV,PAW,ER,T,
     1LM,MT,I2,I3,NUMCEL,ETA,PHI,CHI,CN8,NP,LPF,LKW,NBP,NBM)
      INTEGER TIME                                                      COL0030
      INTEGER*2 LM(NP,1),    LCOL(NP,1),LPF(NP,1),LKW(1)
      INTEGER*2 NBM,NB,NBP,NUMCEL
      DIMENSION CN(3,3,1),CM(3,3,1),WTM(1),DB(NP,1),DBA(NP,1),NB(NP,1)
      DIMENSION NCOL(3,1),T(NP,NP,1),NUMCEL(1),ETA(3,1),PHI(3,1),CHI(1)
      DIMENSION PAU(NP,1),PAV(NP,1),PAW(NP,1),ER(NP,1),CN8(3,1),WA(2)
      DIMENSION NBP(NP,1),NBM(NP,1)
      COMMON /FORTH/NBX                                                 COL0080
      COMMON /FIFTH/ND,TIME,DTM                                         COL0090
    1 FORMAT(' TIME = ',F9.4,' COLL. TIMES = ',2F9.4,' NUMBERS = ',2I5/)
    2 FORMAT(' COLIDE REACHED LINE 160 IN BOX NUMBER =', I5,' AT CPU TIM
     1E =',F9.4/' VR= ',E12.4,' REL VEL G = ',3E12.4,' EI = ',E12.4)
    3 FORMAT(' COLIDE REACHED LINE 165 IN BOX NUMBER =', I5,' AT CPU TIM
     1E =',F9.4/' VR= ',E12.4,' REL VEL G = ',3E12.4,' EI = ',E12.4)
C-----------------------------------------------------------------------COL0100
C     THE PURPOSE OF THIS SUBROUTINE IS TO ADVANCE THE ELAPSED TIMES INCOL0110
C     CELLS BY AN AMOUNT APPROXIMATELY EQUAL TO THE PRE-SELECTED COLLISCOL0120
C     TIME. THERE ARE FOUR TIMES FOR EACH CELL, SAVED IN AN ARRAY CALLECOL0130
C     'T', CORRESPONDING TO THE FOUR TYPES OF MOLECULAR COLLISIONS WHICCOL0140
C     CAN OCCUR. TO ADVANCE THE VARIOUS TIMES, AN APPROPRIATE NUMBER OFCOL0150
C     THE CORRESPONDING MOLECULAR COLLISIONS IS COMPUTED. THE ACTUAL   COL0160
C     MOLECULES TO COLLIDE ARE SELECTED AT RANDOM, AND THEIR VELOCITY VCOL0170
C     DIRECTIONS AFTER COLLISION ARE SELECTED AT RANDOM.               COL0180
C-----------------------------------------------------------------------COL0190
      AIME=DTM*TIME
      DO 240 MTA=1,MT
      DO 230 MTB=1,MTA
      D = WTM(MTA) + WTM(MTB)
      WA(MTA)=WTM(MTA)/D
      WA(MTB)=WTM(MTB)/D
      RM=WTM(MTA)*WTM(MTB)/D
      CHT=CHI(MTA)+CHI(MTB)+2.0
      PHT=PHI(MTA,MTB)
      ETT=ETA(MTA,MTB)
      DO 220 M=1,NBX
      N=NUMCEL(M)
      IF(N.LE.0) GO TO 220
```

```
      AKW=LKW(N)
      IF(T(MTA,MTB,N).LT.AIME) GO TO 100
      IF(T(MTB,MTA,N).GE.AIME) GO TO 220
  100 NA=NB(MTA,N)*NB(MTB,N)
      IF(MTA.EQ.MTB) NA=(NA-NB(MTA,N))/2.
      IF(NA.LT.1) GO TO 220
      KS=0
  120 KC=0
      CPUT=ELTIME(0)
      KS=KS+1
      IF(KS.GT.NA) GO TO 220
  130 KC=KC+1
      IF(KC.GT.NA) GO TO 220
  135 I=NB(MTA,N)*RAND(0)+1+NBM(MTA,N)
      IF(I.GT.NBM(MTA,N+1)) I=NBM(MTA,N+1)
      J=LM(MTA,I)
      CR=LPF(MTA,J)/AKW
      IF(CR.GT.0.99) GO TO 140
      IF(RAND(0).GT.CR) GO TO 135
  140 K=NB(MTB,N)*RAND(0)+1+NBM(MTB,N)
      IF(K.GT.NBM(MTB,N+1)) K=NBM(MTB,N+1)
      IF(MTA.EQ.MTB.AND.I.EQ.K) GO TO 140
      L=LM(MTB,K)
      CR=LPF(MTB,L)/AKW
      IF(CR.GT.0.99) GO TO 145
      IF(RAND(0).GT.CR) GO TO 140
  145 CONTINUE
      GM1=WA(MTA)*PAU(MTA,J)+WA(MTB)*PAU(MTB,L)
      GM2=WA(MTA)*PAV(MTA,J)+WA(MTB)*PAV(MTB,L)
      GM3=WA(MTA)*PAW(MTA,J)+WA(MTB)*PAW(MTB,L)
      G1=PAU(MTA,J)-PAU(MTB,L)
      G2=PAV(MTA,J)-PAV(MTB,L)
      G3=PAW(MTA,J)-PAW(MTB,L)
      GS=G1**2+G2**2+G3**2
      IF(GS.LT.1.0E-8) GO TO 130
      ET=RM*GS
      EI=ER(MTA,J)+ER(MTB,L)
      VR=GS**(.5-ETT/2.)
      IF(VR.GE.CM(MTA,MTB,1)) GO TO 160
      CN(MTA,MTB,1)=CN(MTA,MTB,1)+VR
      IF(CN(MTA,MTB,1).LT.CM(MTA,MTB,1)) GO TO 130
      CN(MTA,MTB,1)=CN(MTA,MTB,1)-CM(MTA,MTB,1)
  160 CONTINUE
      CPUT=ELTIME(0)
      IF(M.EQ.1196) WRITE(6,2) M,CPUT,VR,G1,G2,G3,EI
      CALL IMPACT(RM,G1,G2,G3,ET,EI,PHT,CHT,ETT,CM(MTA,MTB,2),CN(MTA,MTB
     1,2))
  165 CONTINUE
      CPUT=ELTIME(0)
      IF(M.EQ.1196) WRITE(6,3) M,CPUT,VR,G1,G2,G3,EI
      IF(PHT.EQ.0.) GO TO 175
      X1=0.0
      IF(CHI(MTA).EQ.-1.) GO TO 175
      X1=1.0
      IF(CHI(MTB).EQ.-1.) GO TO 175
```

```
170 X1=RAND(0)
    IF((CHI(MTA).EQ.0.).AND.(CHI(MTB).EQ.0.)) GO TO 175
    XT=X1**CHI(MTA)*(1.-X1)**CHI(MTB)
    IF(XT.GT.CM(MTA,MTB,3)) GO TO 175
    CN(MTA,MTB,3)=CN(MTA,MTB,3)+XT
    IF(CN(MTA,MTB,3).LT.CM(MTA,MTB,3)) GO TO 170
    CN(MTA,MTB,3)=CN(MTA,MTB,3)-CM(MTA,MTB,3)
175 CONTINUE
    C=DBA(MTA,N)
    D=DBA(MTB,N)
    IF(C.EQ.0.0) C=DB(MTA,N)
    IF(D.EQ.0.0) D=DB(MTB,N)
    IF(T(MTA,MTB,N).GE.AIME) GO TO 180
    PAU(MTA,J)=GM1+WA(MTB)*G1
    PAV(MTA,J)=GM2+WA(MTB)*G2
    PAW(MTA,J)=GM3+WA(MTB)*G3
    IF(PHT.GT.0.) ER(MTA,J)=X1*EI
    LCOL(MTA,J)=1
    NCOL(MTA,MTB)=NCOL(MTA,MTB)+1                                  COL0810
    T(MTA,MTB,N)=T(MTA,MTB,N)+CN8(MTA,MTB)*LPF(MTA,J)/NBF(MTA,N)/D/VR
    IF(MTA.EQ.MTB) GO TO 190
180 IF(T(MTB,MTA,N).GE.AIME) GO TO 210
190 PAU(MTB,L)=GM1-WA(MTA)*G1
    PAV(MTB,L)=GM2-WA(MTA)*G2
    PAW(MTB,L)=GM3-WA(MTA)*G3
    IF(PHT.GT.0.) ER(MTB,L)=(1.-X1)*EI
    LCOL(MTB,L)=1
    NCOL(MTB,MTA)=NCOL(MTB,MTA)+1                                  COL0920
    T(MTB,MTA,N)=T(MTB,MTA,N)+CN8(MTB,MTA)*LPF(MTB,L)/NBF(MTB,N)/C/VR
210 CONTINUE
    IF(M.EQ.1196) WRITE(6,1) AIME,T(MTA,MTB,N),T(MTB,MTA,N),NBF(MTA,N)
   1,NBF(MTB,N)
    IF(T(MTA,MTB,N).LT.AIME.OR.T(MTB,MTA,N).LT.AIME) GO TO 120
220 CONTINUE
230 CONTINUE
240 CONTINUE
    RETURN
    END

    SUBROUTINE MOVE(KSWCH,AKN,MJ,NS,NWEDG,THETAZ,XSTART,I2,I3,I4,I5,DEMOV0010
   1LANG,NWEDGE,BTA,C2,C3,DPA,FL,HTI,HTR,JNT,KNM,NM,XCB,XLIM,MS,IWS,NTMOV0020
   2CF,NTCV,FV,CTI,CTR,CNI,CNR,ALPHA,SIGMA,COEFF,HTS,HTSI,NTS,NTSF,UTLMOV0030
   3,UTT,VTS,PAU,PAV,PAW,PAX,PAY,PAZ,LPF,LCOL,TB,IP,ER,CHI,CNG,CMG,I,
   4UTLI,UTTI,VTSI)
    INTEGER*2 LPF(I,1),LCOL(I,1)
    INTEGER SWTCH,TIME                                             MOV0060
    LOGICAL DUMP                                                   MOV0070
    REAL LAM,MU,NU                                                 MOV0080
    DIMENSION DELANG(1),NWEDGE(1),BTA(1),C2(1),C3(1),FL(1),HTI(1)  MOV0090
    DIMENSION HTR(1),TB(1),XCB(1),ALPHA(3,1),SIGMA(3,1),COEFF(4,1)
    DIMENSION PAU(I,1),PAV(I,1),PAW(I,1),CTI(3,1),CTR(3,1)
    DIMENSION CNI(3,1),CNR(3,1),DPA(1),JNT(1),XLIM(1),KNM(1),NM(1)
    DIMENSION HTS(3,I2,I3),HTSI(3,I2,I3),NTS(3,I2,I3),NTSF(3,I2,I3)
    DIMENSION UTLI(3,I2,I3),UTTI(3,I2,I3),VTSI(3,I2,I3)
    DIMENSION UTL(3,I2,I3),UTT(3,I2,I3),VTS(3,I2,I3),NTCF(3,I4)
```

```
      DIMENSION PAX(I,1),PAY(I,1),PAZ(I,1),IWS(1),MS(1)
      DIMENSION NTCV(3,I4,2,I5,3),FV(3,I4,2,I5,3)
      DIMENSION ER(I,1),CNG(1),CMG(1),CHI(1)
      COMMON /THIRD/PI,NREG
      COMMON /FORTH/NBX,RM,XR,DUMP
      COMMON /FIFTH/ND,TIME,DTM
      COMMON /SVNTH/LAM,MU,NU,MT,N,J,XI,YI,ZI,TUSE
      NAMELIST/CHECK/TIME,X,Y,Z,DX,DY,DZ,TLEFT,RADS,RMS,XR
C-----------------------------------------------------------------------
C       THE PURPOSE OF THIS SUBROUTINE IS TO ADVANCE THE SPATIAL POSITION
C       OF ALL THE MOLECULES BY AN AMOUNT APPROPRIATE TO THEIR CURRENT VE
C       LOCITIES AND THE PRE-SELECTED COLLISION TIME.
C-----------------------------------------------------------------------
    2 FORMAT(27H SOMETHING IS WRONG IN MOVE/3E20.7,4I7,E20.7)
      NAREA=NREG+3
      RMS=RM**2
      DO 150 MT=1,IP
      N=KNM(MT)
   10 N=N+1
      TLEFT=DTM
      IF(KSWCH.EQ.1) TLEFT=TLEFT*RAND(0)
      IF(N.GT.NM(MT)) GO TO 150
   15 LAM=PAU(MT,N)
      IF(LAM.EQ.0.) LAM=.0000001
      MU=PAV(MT,N)
      NU=PAW(MT,N)
      XI=PAX(MT,N)
      YI=PAY(MT,N)
      ZI=PAZ(MT,N)
      DX=TLEFT*LAM
      DY=TLEFT*MU
      DZ=TLEFT*NU
      X=XI+DX
      Y=YI+DY
      Z=ZI+DZ
      RADS=Y**2+Z**2
      IF((RADS.GT.2.*RMS).OR.(ABS(X).GT.2.*XR)) WRITE(6,CHECK)
      IF(RADS.GT.RMS) GO TO 100
      RAD=SQRT(RMS)
      KEY=ABS(LAM)/LAM+.5
      DO 60 L=1,NAREA
      IF(XI-XLIM(L))65,55,60
   55 J=L+KEY-2
      GO TO 70
   60 CONTINUE
      WRITE(6,2)DTM,XI,LAM,N,KEY,NAREA,L,XLIM(L)
      IF(DUMP) CALL ABEND(16)
      STOP
   65 J=L-2
   70 K=J+KEY+1
      IF((K.EQ.0).OR.(K.EQ.NAREA+1)) GO TO 100
      TUSE=(XLIM(K)-XI)/LAM
      XTEMP=XLIM(K)
      IF(TUSE.LE.TLEFT) GO TO 75
      TUSE=TLEFT
```

```
      XTEMP=XI+TUSE*LAM                                                 MOV0650
   75 IF((J.EQ.0).OR.(J.EQ.NREG+1)) GO TO 85                            MOV0660
      CALL INTERS(AKN,MJ,NS,NWEDG,SWTCH,THETAZ,XSTART,I2,I3,I4,I5,DELANGMOV0670
     1,NWEDGE,BTA,C2,C3,DFA,FL,HTI,HTR,JNT,TB,XCB,CTI,CTR,CNI,CNR,ALPHA,MOV0680
     2SIGMA,COEFF,HTS,HTSI,NTS,NTSF,UTL,UTT,VTS,MS,IWS,NTCF,NTCV,FV,PAU,MOV0690
     3PAV,PAW,LPF,LCOL,IP,ER,CHI,CNG,CMG,I,UTLI,UTTI,VTSI)
      IF(SWTCH.EQ.1) GO TO 90                                           MOV0710
   85 XI=XTEMP                                                          MOV0720
      YI=YI+TUSE*MU                                                     MOV0730
      ZI=ZI+TUSE*NU                                                     MOV0740
   90 PAX(MT,N)=XI                                                      MOV0750
      PAY(MT,N)=YI                                                      MOV0760
      PAZ(MT,N)=ABS(ZI)                                                 MOV0770
      PAW(MT,N)=ABS(ZI)/ZI*PAW(MT,N)                                    MOV0780
      TLEFT=TLEFT-TUSE                                                  MOV0790
      IF(TLEFT.GT.0.) GO TO 15                                          MOV0800
      GO TO 10                                                          MOV0810
  100 NZ=NM(MT)                                                         MOV0820
      PAX(MT,N)=PAX(MT,NZ)                                              MOV0830
      PAY(MT,N)=PAY(MT,NZ)                                              MOV0840
      PAZ(MT,N)=PAZ(MT,NZ)                                              MOV0850
      PAU(MT,N)=PAU(MT,NZ)                                              MOV0860
      PAV(MT,N)=PAV(MT,NZ)                                              MOV0870
      PAW(MT,N)=PAW(MT,NZ)                                              MOV0880
      ER(MT,N)=ER(MT,NZ)
      LPF(MT,N)=LPF(MT,NZ)                                              MOV0890
      LCOL(MT,N)=LCOL(MT,NZ)                                            MOV0900
      N=N-1                                                             MOV0910
      NM(MT)=NM(MT)-1                                                   MOV0920
      GO TO 10                                                          MOV0930
  150 CONTINUE                                                          MOV0940
      RETURN                                                            MOV0950
      END                                                               MOV0960

      SUBROUTINE ACCUM(I2,I3,FNB,NB,PAU,PAV,PAW,ER,TMP,TRP,XV,YV,ZV,LM,
     1IP,I,LPF,NBF,NBM)
      INTEGER*2 LM(I,1),LPF(I,1),NBM(I,1)
      INTEGER*2 NB,NBF
      DIMENSION FNB(1),NB(I,1),PAU(I,1),PAV(I,1),PAW(I,1),TMP(I,1)
      DIMENSION XV(I,1),YV(I,1),ZV(I,1),ER(I,1),TRP(I,1),NBF(I,1)
      COMMON /FORTH/NBX                                                 ACUM050
C---------------------------------------------------------------------ACUM060
C     THE PURPOSE OF THIS SUBROUTINE IS TO ACCUMULATE TEMPERATURES,     ACUM070
C     VELOCITIES, AND DENSITIES IN VARIOUS ARRAYS FOR DETERMINING THE   ACUM080
C     AVERAGE FLOW FIELD PROPERTIES AFTER STEADY-STATE HAS BEEN REACHEDACUM090
C---------------------------------------------------------------------ACUM100
      N=0                                                               ACUM110
      DO 180 K=1,NBX                                                    ACUM120
      IF(FNB(K).LE.0.) GO TO 180                                        ACUM130
      N=N+1                                                             ACUM140
      DO 110 MT=1,IP
      XV(MT,N)=0.0                                                      ACUM160
      YV(MT,N)=0.0                                                      ACUM170
      ZV(MT,N)=0.0                                                      ACUM180
      TMP(MT,N)=0.                                                      ACUM190
```

```
      TRP(MT,N)=0.0                                                  ACUM200
      TTX=0.                                                         ACUM210
      TTY=0.                                                         ACUM220
      TTZ=0.
      TTR=0.0                                                        ACUM230
      M=NB(MT,N)                                                     ACUM240
      IF(M.LT.1) GO TO 110                                           ACUM250
      U=0.                                                           ACUM260
      V=0.                                                           ACUM270
      W=0.                                                           ACUM280
      DO 100 L=1,M
      NA=NBM(MT,N)+L
      J=LM(MT,NA)
      AKW=LPF(MT,J)                                                  ACUM300
      PU=PAU(MT,J)                                                   ACUM310
      PV=PAV(MT,J)                                                   ACUM320
      PW=PAW(MT,J)
      U=U+PU*AKW
      V=V+PV*AKW
      W=W+PW*AKW
      TTR=TTR+ER(MT,J)*AKW
      TTX=TTX+PU*PU*AKW
      TTY=TTY+PV*PV*AKW
  100 TTZ=TTZ+PW*PW*AKW
      M=NBP(MT,N)                                                    ACUM390
      XV(MT,N)=U/M                                                   ACUM400
      YV(MT,N)=V/M                                                   ACUM410
      ZV(MT,N)=W/M                                                   ACUM420
      TMP(MT,N)=(TTX+TTY+TTZ)/M
      TRP(MT,N)=TTR/M                                                ACUM430
  110 CONTINUE                                                       ACUM440
  130 CONTINUE                                                       ACUM450
      RETURN                                                         ACUM460
      END
      SUBROUTINE AVRGE(FNB,DB,DBA,NB,NBT,XV,YV,ZV,XVA,YVA,ZVA,TMP,TMPA,
     1TRP,TRPA,IP,I,NBP,NBS)
      INTEGER*2 NB,NBT,NBF,NBS
      DIMENSION FNB(1),DB(I,1),DBA(I,1),NB(I,1),NBT(I,1),TMP(I,1)
      DIMENSION TMPA(I,1),XV(I,1),XVA(I,1),YV(I,1),YVA(I,1),ZV(I,1)
      DIMENSION ZVA(I,1),TRP(I,1),TRPA(I,1),NBF(I,1),NBS(I,1)        AVG0050
      COMMON /FORTH/NBX                                              AVG0060
C-----------------------------------------------------------------   AVG0070
C      THE PURPOSE OF THIS SUBROUTINE IS TO COMPUTE THE AVERAGE FLOW  AVG0080
C      FIELD PROPERTIES.                                             AVG0090
C-----------------------------------------------------------------   AVG0100
      N=0                                                            AVG0110
      DO 110 M=1,NBX                                                 AVG0120
      IF(FNB(M).LE.0.) GO TO 110                                     AVG0130
      N=N+1
      DO 100 MT=1,IP                                                 AVG0150
      A=NBT(MT,N)
      B=NBF(MT,N)                                                    AVG0170
      C=A+B                                                          AVG0180
      NBT(MT,N)=C
```

```
      NBS(MT,N)=NBS(MT,N)+NB(MT,N)
      IF(C.LT.1.) GO TO 100
      DBA(MT,N)=(DBA(MT,N)*A+DB(MT,N)*B)/C
      XVA(MT,N)=(XVA(MT,N)*A+XV(MT,N)*B)/C
      YVA(MT,N)=(YVA(MT,N)*A+YV(MT,N)*B)/C
      ZVA(MT,N)=(ZVA(MT,N)*A+ZV(MT,N)*B)/C
      TMPA(MT,N)=(TMPA(MT,N)*A+TMP(MT,N)*B)/C
      TRPA(MT,N)=(TRPA(MT,N)*A+TRP(MT,N)*B)/C
  100 CONTINUE
  110 CONTINUE
      RETURN
      END

      SUBROUTINE DRAG(AKN,MJ,NS,NWEDG,THETAZ,ISTART,I2,I3,I4,I5,DELANG,
     1NWEDGE,BTA,C2,C3,DFA,FL,HTI,HTR,TB,XCB,CTI,CTR,CNI,CNR,ALPHA,SIGMA
     2,COEFF,HTS,HTSI,NTS,NTSP,UTL,UTT,VTS,MS,IWS,NTCP,NTCV,FV,PAU,PAV,P
     3AW,LPF,LCOL,IP,ER,CHI,CNG,CMG,I,UTLI,UTTI,VTSI)
      INTEGER*2 LPF,LCOL
      INTEGER TIME,TST
      REAL LAM,MU,NU,JAY,KAY
      DIMENSION DELANG(1),NWEDGE(1),BTA(1),C2(1),C3(1),FL(1),HTI(1)
      DIMENSION HTR(1),TB(1),XCB(1),ALPHA(3,1),SIGMA(3,1),COEFF(4,1)
      DIMENSION LPF(I,1),PAU(I,1),PAV(I,1),PAW(I,1),CTI(3,1),CTR(3,1)
      DIMENSION HTS(3,I2,I3),HTSI(3,I2,I3),NTS(3,I2,I3),NTSP(3,I2,I3)
      DIMENSION UTL(3,I2,I3),UTT(3,I2,I3),VTS(3,I2,I3),LCOL(I,1)
      DIMENSION UTLI(3,I2,I3),UTTI(3,I2,I3),VTSI(3,I2,I3)
      DIMENSION CNI(3,1),CNR(3,1),DFA(1),IWS(1),MS(1),NTCP(3,I4)
      DIMENSION        NTCV(3,I4,2,I5,3),FV(3,I4,2,I5,3)
      DIMENSION ER(I,1),CNG(1),CMG(1),CHI(1)
      COMMON /THIRD/PI
      COMMON /FIFTH/ND,TIME,DTM,TI,ITS,ITP,TST
      COMMON /SVNTH/LAM,MU,NU,MT,N,J,XCL,YCL,ZCL
C-----------------------------------------------------------------------
C       THE PURPOSE OF THIS SUBROUTINE IS TO ACCUMULATE THE DRAG AND HEAT
C       TRANSFER INCREMENTS ON THE BODY CONTRIBUTED BY EACH MOLECULE WHIC
C       COLLIDES WITH THE BODY. IN ADDITION, EACH MOLECULE WHICH COLLIDES
C       WITH THE BODY IS ASSIGNED AN APPROPRIATE NEW VELOCITY (OF REFLECT
C       WHICH IS USED TO CONTINUE ITS SPATIAL TRANSLATION (IN SUBROUTINE 
C-----------------------------------------------------------------------
      CALL NORMAL(EYE,JAY,KAY,ONE,COEFF)
      AKW=LPF(MT,N)
      RAD=SQRT(YCL*YCL+ZCL*ZCL)
      ARG=YCL/RAD
      TANG=180.*(1.-ARCCOS(ARG)/PI)
      IWDG=TANG/DELANG(1)+1.
      IF((IWDG .GT.NWEDGE(1)).AND.(DELANG(2).NE.0.)) IWDG =(TANG-THETAZ)
     1/DELANG(2)+NWEDGE(1)+1
      IF(IWDG.LT.1) IWDG=1
      IF(IWDG.GT.NWEDG) IWDG=NWEDG
      D=(LAM*LAM+MU*MU+NU*NU)*AKW
      G=ER(MT,N)*AKW
      H=G
      DO 100 M=1,ND
      IF(XCL.LT.XCB(M)) GO TO 110
  100 CONTINUE
```

```
110 UI=LAM                                                            DRG0360
    WI=(NU*JAY-MU*KAY)/ONE                                            DRG0370
    VID=LAM*EYE+MU*JAY+NU*KAY                                         DRG0380
    UID=LAM*ONE-EYE*(MU*JAY+NU*KAY)/ONE                               DRG0390
    E=RAND(0)                                                         DRG0400
    IF(E.LT.SIGMA(MT,M)) GO TO 115                                    DRG0410
    VRD=-VID                                                          DRG0420
    URD=UID                                                           DRG0430
    WR=WI                                                             DRG0440
    GO TO 125                                                         DRG0450
115 V=4.*RAND(0)                                                      DRG0460
    C2(MT)=C2(MT)+.544331*V*V*V*EXP(1.5-V*V)                          DRG0470
    IF(C2(MT).LT.1.) GO TO 115                                        DRG0480
    C2(MT)=C2(MT)-1.                                                  DRG0490
    IF(NTSF(MT,M,IWDG).NE.0) GO TO 117                                DRG0500
    ATR=ALPHA(MT,M)*TB(M)/SIGMA(MT,M)                                 DRG0510
    GO TO 118                                                         DRG0520
117 ATR=ALPHA(MT,M)*TB(M)/SIGMA(MT,M)+(1.-ALPHA(MT,M)/SIGMA(MT,M))*HTSDRG0530
   1I(MT,M,IWDG)/NTSF(MT,M,IWDG)/(3.+CHI(MT))
118 ABR=SQRT(ATR)                                                     DRG0550
    V=V*ABR/BTA(MT)                                                   DRG0560
120 A=RAND(0)                                                         DRG0570
    C3(MT)=C3(MT)+A                                                   DRG0580
    IF(C3(MT).LT.1.) GO TO 120                                        DRG0590
    C3(MT)=C3(MT)-1.                                                  DRG0600
    B=SQRT(1.-A*A)                                                    DRG0610
    C=2.*PI*RAND(0)                                                   DRG0620
    VRD=V*A                                                           DRG0630
    URD=V*B*COS(C)                                                    DRG0640
    WR=V*B*SIN(C)                                                     DRG0650
    IF(CHI(MT).EQ.-1.) GO TO 125
122 X=9.*RAND(0)
    IF(X.EQ.0.0) GO TO 122
    XTEMP=1.0
    IF(CHI(MT).NE.0.0) XTEMP=X**CHI(MT)
    CNG(MT)=CNG(MT)+XTEMP*EXP(-X)
    IF(CNG(MT).LT.CMG(MT)) GO TO 122
124 CONTINUE
    CNG(MT)=CNG(MT)-CMG(MT)
    IF(CNG(MT).GE.CMG(MT)) GO TO 124
    ER(MT,N)=X*ATR
    H=ER(MT,N)*AKW
125 UR=EYE*VRD+ONE*URD                                                DRG0660
    PAU(MT,N)=UR                                                      DRG0670
    PAV(MT,N)=JAY*VRD-(KAY*WR+EYE*JAY*URD)/ONE                        DRG0680
    PAW(MT,N)=KAY*VRD+(JAY*WR-EYE*KAY*URD)/ONE                        DRG0690
    IF(TIME.LE.TST) RETURN
    IF(TI.GT.0.) GO TO 130                                            DRG0710
    TI=TST*DTM
130 XMZ=(XCL-XSTART)*AKN                                              DRG0730
    YMZ=RAD*AKN                                                       DRG0740
    B=(URD*URD+VRD*VRD+WR*WR)*AKW                                     DRG0750
    UTI=UID*UID+WI*WI                                                 DRG0760
    UYI=-(UID*EYE*JAY+WI*KAY)/ONE                                     DRG0770
    UYR=-(URD*EYE*JAY+WR*KAY)/ONE                                     DRG0780
```

```
      FL(MT)=FL(MT)+AKW*DFA(MT)                                         DRG0790
      HTI(MT)=HTI(MT)+D+G
      HTR(MT)=HTR(MT)-B-H
      CTI(MT,1)=CTI(MT,1)+UID*ONE*AKW                                   DRG0820
      CTI(MT,2)=CTI(MT,2)+UYI*AKW                                       DRG0830
      CTI(MT,3)=CTI(MT,3)+(XMZ*UYI-YMZ*UID*ONE)*AKW                     DRG0840
      CNI(MT,1)=CNI(MT,1)+VID*EYE*AKW                                   DRG0850
      CNI(MT,2)=CNI(MT,2)+VID*JAY*AKW                                   DRG0860
      CNI(MT,3)=CNI(MT,3)+(XMZ*JAY-YMZ*EYE)*VID*AKW                     DRG0870
      CTR(MT,1)=CTR(MT,1)-URD*ONE*AKW                                   DRG0880
      CTR(MT,2)=CTR(MT,2)-UYR*AKW                                       DRG0890
      CTR(MT,3)=CTR(MT,3)-(XMZ*UYR-YMZ*URD*ONE)*AKW                     DRG0900
      CNR(MT,1)=CNR(MT,1)-VRD*EYE*AKW                                   DRG0910
      CNR(MT,2)=CNR(MT,2)-VRD*JAY*AKW                                   DRG0920
      CNR(MT,3)=CNR(MT,3)-(XMZ*JAY-YMZ*EYE)*VRD*AKW                     DRG0930
      NTS(MT,M,IWDG)=NTS(MT,M,IWDG)+1                                   DRG0940
      NTSF(MT,M,IWDG)=NTSF(MT,M,IWDG)+AKW                               DRG0950
      UTLI(MT,M,IWDG)=UTLI(MT,M,IWDG)+UID*AKW
      UTL(MT,M,IWDG)=UTL(MT,M,IWDG)+(UID-URD)*AKW                       DRG0960
      UTTI(MT,M,IWDG)=UTTI(MT,M,IWDG)+ WI*AKW
      UTT(MT,M,IWDG)=UTT(MT,M,IWDG)+(WI-WR)*AKW                         DRG0970
      VTSI(MT,M,IWDG)=VTSI(MT,M,IWDG)-VID*AKW
      VTS(MT,M,IWDG)=VTS(MT,M,IWDG)+(VRD-VID)*AKW                       DRG0980
      HTSI(MT,M,IWDG)=HTSI(MT,M,IWDG)+D+G
      HTS(MT,M,IWDG)=HTS(MT,M,IWDG)+D-B+G-H
      IF(NS.EQ.0) RETURN                                                DRG1010
      NC=(2*LCOL(MT,N))/(1+LCOL(MT,N))+1                                DRG1020
      DO 160 L=1,NS
      MTEST=MS(L)
      IWT=IWS(L)
      IF((M.NE.MTEST).OR.(IWDG.NE.IWT)) GO TO 160                       DRG1060
      IF(NC.EQ.1) NTCF(MT,L)=NTCF(MT,L)+AKW
      DO 145 JJ=1,MJ                                                    DRG1080
      IF(ABS(VID).GT.FV(MT,L,NC,JJ,1)) GO TO 135
      NTCV(MT,L,NC,JJ,1)=NTCV(MT,L,NC,JJ,1)+AKW
  135 IF(UID.GT.FV(MT,L,NC,JJ,2)) GO TO 140
      NTCV(MT,L,NC,JJ,2)=NTCV(MT,L,NC,JJ,2)+AKW
  140 IF(WI*ABS(ZCL)/ZCL.GT.FV(MT,L,NC,JJ,3)) GO TO 145
      NTCV(MT,L,NC,JJ,3)=NTCV(MT,L,NC,JJ,3)+AKW
  145 CONTINUE                                                          DRG1150
  160 CONTINUE                                                          DRG1160
      RETURN                                                            DRG1170
      END                                                               DRG1180

      SUBROUTINE INTERS(AKN,MJ,NS,NWEDG,SWTCH,THETAZ,XSTART,I2,I3,I4,I5,INT0010
     1DELANG,NWEDGE,BTA,C2,C3,DFA,FL,HTI,HTR,JNT,TB,XCB,CTI,CTR,CNI,CNR,INT0020
     2ALPHA,SIGMA,COEFF,HTS,HTSI,NTS,NTSF,UTL,UTT,VTS,MS,IWS,NTCF,NTCV,FINT0030
     3V,PAU,PAV,PAW,LPF,LCOL,IP,ER,CHI,CNG,CMG,I,UTLI,UTTI,VTSI)
      INTEGER*2 LPF,LCOL                                                INT0050
      INTEGER SWTCH                                                     INT0060
      REAL LAM,MU,NU,LINEAR                                             INT0070
      DIMENSION DELANG(1),NWEDGE(1),BTA(1),C2(1),C3(1),FL(1),HTI(1)     INT0080
      DIMENSION HTR(1),TB(1),XCB(1),ALPHA(3,1),SIGMA(3,1),COEFF(4,1)
      DIMENSION LPF(I,1),PAU(I,1),PAV(I,1),PAW(I,1),CTI(3,1),CTR(3,1)
      DIMENSION HTS(3,I2,I3),HTSI(3,I2,I3),NTS(3,I2,I3),NTSF(3,I2,I3)
```

ORIGINAL PAGE IS
OF POOR QUALITY


```
      DIMENSION UTL(3,I2,I3),UTT(3,I2,I3),VTS(3,I2,I3),LCOL(I,1)
      DIMENSION UTLI(3,I2,I3),UTTI(3,I2,I3),VTSI(3,I2,I3)
      DIMENSION CNI(3,1),CNR(3,1),DPA(1),IWS(1),MS(1),NTCP(3,I4)
      DIMENSION JNT(1),NTCV(3,I4,2,I5,3),PV(3,I4,2,I5,3)
      DIMENSION ER(I,1),CNG(1),CMG(1),CHI(1)
      COMMON /SYNTH/LAM,MU,NU,MT,N,J,XI,YI,ZI,TUSE
C-----------------------------------------------------------------------
C     THE PURPOSE OF THIS SUBROUTINE IS TO DETERMINE FOR EACH MOLECULAR
C     TRAJECTORY IF THERE IS AN INTERSECTION OF THE TRAJECTORY WITH THE
C     BODY SURFACE.
C-----------------------------------------------------------------------
      SWTCH=0
      A=COEFF(1,J)
      B=COEFF(2,J)
      C=COEFF(3,J)
      D=COEFF(4,J)
      ONE=A*LAM
      TWO=B*MU
      TRE=B*NU
      FUR=.5*C*LAM
      SQUARE=ONE*LAM+TWO*MU+TRE*NU
      LINEAR=ONE*XI+TWO*YI+TRE*ZI+FUR
      CONST=(A*XI+C)*XI+B*(YI*YI+ZI*ZI)+D
      IF(SQUARE.EQ.0.) GO TO 150
      DISCR=LINEAR*LINEAR-SQUARE*CONST
      IF(DISCR.LT.0.) RETURN
      SDISC = SQRT(DISCR)
      TYME = (-LINEAR-SDISC)/SQUARE
      GO TO 250
  150 IF(LINEAR.EQ.0.) RETURN
      TYME=-.5*CONST/LINEAR
  250 IF(TYME.GT.TUSE) RETURN
      IF(TYME.LE.0.) RETURN
      JNT(MT)=JNT(MT)+1
      XI=XI+LAM*TYME
      YI=YI+MU*TYME
      ZI=ZI+NU*TYME
      CALL DRAG(AKN,MJ,NS,NWEDG,THETAZ,XSTART,I2,I3,I4,I5,DELANG,NWEDGE,
     1BTA,C2,C3,DPA,PL,HTI,HTB,TB,XCB,CTI,CTR,CNI,CNR,ALPHA,SIGMA,COEFF,
     2HTS,HTSI,NTS,NTSP,UTL,UTT,VTS,MS,IWS,NTCP,NTCV,PV,PAU,PAV,PAW,LPF,
     3LCOL,IP,ER,CHI,CNG,CMG,I,UTLI,UTTI,VTSI)
      LCOL(MT,N)=2
      TUSE=TYME
      SWTCH=1
      RETURN
      END

      SUBROUTINE NORMAL(EYE,JAY,KAY,ONE,COEFF)
      REAL LAM,MU,NU,JAY,KAY
      DIMENSION COEFF(4,1)
      COMMON /SYNTH/LAM,MU,NU,MT,N,J,XCL,YCL,ZCL
      DFDX=2.*COEFF(1,J)*XCL+COEFF(3,J)
      DFDY=2.*COEFF(2,J)*YCL
      DFDZ=2.*COEFF(2,J)*ZCL
      DENOM=SQRT(DFDX*DFDX+DFDY*DFDY+DFDZ*DFDZ)
```

```
      EYE=DFDX/DENOM                                                   NORM090
      JAY=DFDY/DENOM                                                   NORM100
      KAY=DFDZ/DENOM                                                   NORM110
      ONE=SQRT(JAY*JAY+KAY*KAY)                                        NORM120
      RETURN                                                           NORM130
      END                                                              NORM140

      FUNCTION ARCCOS(ARG)                                             ARCS010
      COMMON /THIRD/PI                                                 ARCS020
      IF(ARG)30,10,20                                                  ARCS030
   10 A=.5*PI                                                          ARCS040
      GO TO 40                                                         ARCS050
   20 A=ATAN(SQRT(1.-ARG*ARG)/ARG)                                     ARCS060
      GO TO 40                                                         ARCS070
   30 A=PI+ATAN(SQRT(1.-ARG*ARG)/ARG)                                  ARCS080
   40 ARCCOS=A                                                         ARCS090
      RETURN                                                           ARCS100
      END                                                              ARCS110

      FUNCTION ERRF(SS)                                                ERRF010
      ERRF=ERFC(-SS)
      RETURN                                                           ERRF050
      END                                                              ERRF060

      SUBROUTINE PRINT1(DT,COSANG,SINANG,RMA,RNU,DRF,PCF,HTF,FL,HTI,HTR,
     1CTI,CTR,CNI,CNR)
      DIMENSION DD(3),WD(2,5),PP(4,4),QQ(4,4),RR(4,4),SS(4,4),TT(4,4)
      DIMENSION UU(4,4),P1(4,4),Q1(4,4),R1(4,4),PA(4),PB(4),PC(4)
      DIMENSION FL(1),HTI(1),HTR(1),CTI(3,1),CTR(3,1),CNI(3,1),CNR(3,1)
      DIMENSION RMA(1),RNU(1)
      DATA WD/'X-PO','RCE ','Y-PO','RCE ','Z-MO','MENT','DRAG','    ','LPT10070
     1IFT','    '/                                                      PT10080
---------------------------------------------------------------------PT10090
       THE PURPOSE OF THIS SUBROUTINE IS TO PRINT OUT THE GROSS SURFACE PT10100
       COEFFICIENTS OF THE BODY.                                        PT10110
---------------------------------------------------------------------PT10120
                                                                      PT10130
                                                                      PT10140
                                                                      PT10150
                 FORMATS                                              PT10160
                                                                      PT10170
                                                                      PT10180
    1 FORMAT(//1X,50('*'),' GROSS SURFACE COEFFICIENTS ',50('*')/' MOLEC
     1ULAR WEIGHT',12X,F8.3,3(19X,F8.3)/25X,
     2  'INC.     REF.     TOT.       INC.     REF.     TOT.       INC.
     3REF.     TOT.     INC.     REF.     TOT.')
   10 FORMAT(' NUMBER FLUX         ', 4(F9.3,18X))
   12 FORMAT(1X,2A4,2X,'SHEAR      ',4(3F8.3,3X))
   14 FORMAT(11X,'PRESSURE   ',4(3F8.3,3X))
   16 FORMAT(11X,'TOTAL      ',4(3F8.3,3X)/)
   18 FORMAT(' HEAT TRANSFER',7X,4(3F8.3,3X)/)
                                                                      PT10280
*********************************************************************PT10290
                                                                      PT10300
                                                                      PT10310
```

```
      RMR=0.0
      DO 50 MT=1,3
      DD(MT)=RMA(MT)*RNU(MT)*DRF/DT
   50 RMR=RMR+RMA(MT)*RNU(MT)
      WRITE(6,1)  (RMA(MT),MT=1,3),RMR
      PF=FL(1)*FCF/DT
      QF=FL(2)*FCF/DT
      RF=FL(3)*FCF/DT
      SF=PF+QF+RF
      WRITE(6,10) PF,QF,RF,SF
      DO 200 I=1,3
      PP(4,I)=0.0
      QQ(4,I)=0.0
      RR(4,I)=0.0
      SS(4,I)=0.0
      TT(4,I)=0.0
      UU(4,I)=0.0
      P1(4,I)=0.0
      Q1(4,I)=0.0
      R1(4,I)=0.0
      DO 150 MT=1,3
      PP(MT,I)=CTI(MT,I)*DD(MT)/RMR
      QQ(MT,I)=CTR(MT,I)*DD(MT)/RMR
      SS(MT,I)=CNI(MT,I)*DD(MT)/RMR
      TT(MT,I)=CNR(MT,I)*DD(MT)/RMR
      P1(MT,I)=PP(MT,I)+SS(MT,I)
      Q1(MT,I)=QQ(MT,I)+TT(MT,I)
      RR(MT,I)=PP(MT,I)+QQ(MT,I)
      UU(MT,I)=SS(MT,I)+TT(MT,I)
      R1(MT,I)=P1(MT,I)+Q1(MT,I)
      PP(4,I)=PP(4,I)+PP(MT,I)
      QQ(4,I)=QQ(4,I)+QQ(MT,I)
      RR(4,I)=RR(4,I)+RR(MT,I)
      SS(4,I)=SS(4,I)+SS(MT,I)
      TT(4,I)=TT(4,I)+TT(MT,I)
      UU(4,I)=UU(4,I)+UU(MT,I)
      P1(4,I)=P1(4,I)+P1(MT,I)
      Q1(4,I)=Q1(4,I)+Q1(MT,I)
      R1(4,I)=R1(4,I)+R1(MT,I)
  150 CONTINUE
      WRITE(6,12) (WD(J,I),J=1,2),(PP(K,I),QQ(K,I),RR(K,I),K=1,4)
      WRITE(6,14) (SS(K,I),TT(K,I),UU(K,I),K=1,4)
      WRITE(6,16) (P1(K,I),Q1(K,I),R1(K,I),K=1,4)
  200 CONTINUE
      AA=COSANG
      BB=SINANG
      DO 300 I=4,5
      DO 250 K=1,4
      PP(K,4)=AA*PP(K,1)+BB*PP(K,2)
      QQ(K,4)=AA*QQ(K,1)+BB*QQ(K,2)
      RR(K,4)=AA*RR(K,1)+BB*RR(K,2)
      SS(K,4)=AA*SS(K,1)+BB*SS(K,2)
      TT(K,4)=AA*TT(K,1)+BB*TT(K,2)
      UU(K,4)=AA*UU(K,1)+BB*UU(K,2)
      P1(K,4)=AA*P1(K,1)+BB*P1(K,2)
```

```
      Q1(K,4)=AA*Q1(K,1)+BB*Q1(K,2)
  250 R1(K,4)=AA*R1(K,1)+BB*R1(K,2)
      WRITE(6,12) (WD(J,I),J=1,2),(PP(K,4),QQ(K,4),RR(K,4),K=1,4)
      WRITE(6,14) (SS(K,4),TT(K,4),UU(K,4),K=1,4)
      WRITE(6,16) (P1(K,4),Q1(K,4),R1(K,4),K=1,4)
      AA=-SINANG
      BB=COSANG
  300 CONTINUE
      HD=HTF/DT
      PA(4)=0.0
      PB(4)=0.0
      PC(4)=0.0
      DO 400 MT=1,3
      PA(MT)=HTI(MT)*RMA(MT)*RNU(MT)*HD/RMR
      PB(MT)=HTR(MT)*RMA(MT)*RNU(MT)*HD/RMR
      PC(MT)=PA(MT)+PB(MT)
      PA(4)=PA(4)+PA(MT)
      PB(4)=PB(4)+PB(MT)
      PC(4)=PC(4)+PC(MT)
  400 CONTINUE
      WRITE(6,18)  (PA(I),PB(I),PC(I),I=1,4)
      RETURN
      END

      SUBROUTINE PRINT2(AKN,XSTART,DT,RNU,RMA,DPF,FCF,HTF,UTLI,UTTI,VTSI
     1,HTSI,DELANG,NWEDGE,XS,XCB,YCB,HTS,NTS,NTSF,UTL,UTT,VTS,I2,I3,IP)
      DIMENSION RMA(1),RNU(1),DELANG(1),NWEDGE(1),XS(1),XCB(1),YCB(1)
      DIMENSION HTS(3,I2,I3),NTS(3,I2,I3),NTSF(3,I2,I3),UTL(3,I2,I3)
      DIMENSION UTT(3,I2,I3),VTS(3,I2,I3),UTLI(3,I2,I3),UTTI(3,I2,I3)
      DIMENSION VTSI(3,I2,I3),HTSI(3,I2,I3)
      COMMON /FIFTH/ND
---------------------------------------------------------------------
      THE PURPOSE OF THIS SUBROUTINE IS TO PRINT OUT THE DISTRIBUTION
      ON SURFACE OF THE SURFACE COEFFICIENTS
---------------------------------------------------------------------



                   FORMATS

    8 FORMAT(//1X,45('*'),' DISTRIBUTION ON SURFACE ',45('*')/71X,'INC.
     1   TOT.    INC.    TOT.    INC.    TOT.'/11X,'SEGMENT GEOMETRY',
     214X,'MOL.    MOLE  SAMP    NUM.    SKIN    SKIN    PRES-   PRES-
     3 HEAT    HEAT'/' NO.  CENTER    DELX    CENTER  DELANG',4X,'WGHT.  F
     4RACT.',10X,
     5        'FLUX   FRCTN   FRCTN   SURE    SURE    TRNSF   TRNSF')
   10 FORMAT(1X,I3,F8.3,F7.3,F9.3,F8.3,1X,2F8.4,I6,7F8.4)
   11 FORMAT(37X,F8.4,'  1.0000',I6,7F8.4)

*********************************************************************


      RMR=0.0
      DO 50 MT=1,IP
```

```
   50 RMR=RMR+RMA(MT)*RNU(MT)
      WRITE(6,8)
      I=0                                                               PT20280
      DO 110 N=1,ND                                                     PT20320
      DTY=DT*YCB(N)/180.                                                PT20330
      P=XS(N)                                                           PT20340
      Q=2.*((XCB(N)-XSTART)*AKN-XS(N))                                  PT20350
      ANGLE=0.                                                          PT20360
      R=0.                                                              PT20370
      J=0                                                               PT20380
      DO 105 L=1,2
      R=R+.5*ANGLE                                                      PT20400
      ANGLE=DELANG(L)
      R=R-.5*ANGLE                                                      PT20420
      ICNT=NWEDGE(L)
      IF(ICNT.EQ.0) GO TO 105
      PMLT=FCP/(DTY*ANGLE)                                              PT20440
      QMLT=DRF/(DTY*ANGLE)                                              PT20450
      SMLT=HTF/(DTY*ANGLE)                                              PT20460
      DO 100 K=1,ICNT                                                   PT20470
      R=R+ANGLE                                                         PT20480
      I=I+1                                                             PT20490
      J=J+1                                                             PT20500
      M3=0
      P3=0.0
      Q3=0.0
      Q4=0.0
      R3=0.0
      R4=0.0
      S3=0.0
      S4=0.0
      DO 90 MT=1,IP
      M1=NTS(MT,N,J)
      M3=M3+M1
      P1=NTSF(MT,N,J)*PMLT*RNU(MT)
      P3=P3+P1
      Q1=SQRT(UTLI(MT,N,J)**2+UTTI(MT,N,J)**2)*RNU(MT)*RMA(MT)*QMLT/RMR
      Q2=SQRT( UTL(MT,N,J)**2+ UTT(MT,N,J)**2)*RNU(MT)*RMA(MT)*QMLT/RMR
      Q3=Q3+Q1
      Q4=Q4+Q2
      R1=VTSI(MT,N,J)*RNU(MT)*RMA(MT)*QMLT/RMR
      R2= VTS(MT,N,J)*RNU(MT)*RMA(MT)*QMLT/RMR
      S1=HTSI(MT,N,J)*RNU(MT)*RMA(MT)*SMLT/RMR
      S2= HTS(MT,N,J)*RNU(MT)*RMA(MT)*SMLT/RMR
      R3=R3+R1
      R4=R4+R2
      S3=S3+S1
      S4=S4+S2
   90 WRITE(6,10)I,P,Q,R,ANGLE,RMA(MT),RNU(MT),M1,P1,Q1,Q2,R1,R2,S1,S2
      WRITE(6,11)RMR,M3,P3,Q3,Q4,R3,R4,S3,S4
  100 CONTINUE                                                          PT20660
  105 CONTINUE
  110 CONTINUE                                                          PT20670
      RETURN                                                            PT20680
      END                                                               PT20690
```

```
:
      SUBROUTINE PRINT3(         IP,MJ,NS,NWEDG,I2,I3,I4,I5, RMA,XS,IWS,
     1MS,TANGN,NTSF,NTCF,NTCV,FV)                                       PT30020
      DIMENSION  RMA(1),XS(1),IWS(1),MS(1),TANGN(1),NTSF(3,I2,I3)
      DIMENSION NTCF(3,I4),  NTCV(3,I4,2,I5,3),FV(3,I4,2,I5,3),QUO(3,3) PT3
:                                                                       PT30050
:---------------------------------------------------------------------- PT30060
:                                                                       PT30070
:              FORMATS                                                  PT30080
                                                                        PT30090
    2 FORMAT(//1X,40('*'),' MOMENTS OF INCIDENT DISTRIBUTION FUNCTIONS 'PT30100
     1,40('*'))                                                         PT30110
    6 FORMAT(/21X,I5,' UNCOLLIDED MOLECULES',F8.4,27X,I5,' COLLIDED MOLE
     1CULES',F8.4/)
    8 FORMAT(I5,4X,A3,12(1X,F9.4))                                      PT30170
   10 FORMAT(12X,12(1X,F9.4))                                           PT30180
   12 FORMAT(1H )                                                       PT30190
                                                                        PT30200
                                                                        PT30210
***********************************************************************PT30220
                                                                        PT30230
      WRITE(6,2)                                                        PT30240
      DO 155 I=1,NS
      MR=MS(I)                                                          PT30260
      ITT=IWS(I)                                                        PT30270
      N=(MR-1)*NWEDG+ITT                                                PT30280
      DO 150 MT=1,IP
      A=0.                                                              PT30370
      B=1.                                                              PT30380
      IC=NTCF(MT,I)
      ID=NTSF(MT,MR,ITT)-IC
      E=NTSF(MT,MR,ITT)
      IF(E.LE.0.) GO TO 110                                             PT30400
      A=IC/E
      B=1.-A
  110 WRITE(6,6) IC,A,ID,B
      E=NTCF(MT,I)                                                      PT30450
      DO 121 NC=1,2                                                     PT30460
      DO 120 K=1,3                                                      PT30470
      QUO(NC,K)=0.                                                      PT30480
      IF(E.EQ.0.) GO TO 120                                             PT30490
      QUO(NC,K)=NTCV(MT,I,NC,1,K)/E                                     PT30500
  120 CONTINUE                                                          PT30510
      E=NTSF(MT,MR,ITT)-NTCF(MT,I)                                      PT30520
  121 CONTINUE                                                          PT30530
      WRITE(6,8) N,    RMA(MT),FV(MT,I,1,1,1),QUO(1,1),FV(MT,I,1,1,2),QUO
     1(1,2),FV(MT,I,1,1,3),QUO(1,3),FV(MT,I,2,1,1),QUO(2,1),FV(MT,I,2,1,PT30550
     22),QUO(2,2),FV(MT,I,2,1,3),QUO(2,3)                               PT30560
      IF(MJ.EQ.1) GO TO 150                                             PT30570
      DO 140 J=2,MJ                                                     PT30580
      E=NTCF(MT,I)                                                      PT30590
      DO 131 NC=1,2                                                     PT30600
      DO 130 K=1,3                                                      PT30610
      QUO(NC,K)=0.                                                      PT30620
      IF(E.EQ.0.) GO TO 130                                             PT30630
```

```
      QUO(NC,K)=NTCV(MT,I,NC,J,K)/E                                     PT30640
  130 CONTINUE                                                          PT30650
      E=NTSP(MT,MR,ITT)-NTCP(MT,I)                                      PT30660
  131 CONTINUE                                                          PT30670
  140 WRITE(6,10)   FV(MT,I,1,J,1),QUO(1,1),FV(MT,I,1,J,2),QUO(1,2),FV(MTPT30680
     1,I,1,J,3),QUO(1,3),FV(MT,I,2,J,1),QUO(2,1),FV(MT,I,2,J,2),QUO(2,2)PT30690
     2,FV(MT,I,2,J,3),QUO(2,3)                                          PT30700
  150 WRITE(6,12)                                                       PT30710
  155 CONTINUE
      RETURN                                                            PT30720
      END                                                               PT30730

      SUBROUTINE PRINT4(MSP,CHI,RNU,I,TRP,NUMCEL,FDN,WTM,DB,NS,TMP,XV,
     1YV,ZV,KS,NB,IC,YC,ZC,LEV,LKW)
      INTEGER*2 LKW(1)
      INTEGER*2 NB,NUMCEL,NS
      DIMENSION FDN(1),RNU(1),CHI(1),WTM(1),NUMCEL(1),TMP(I,1),TRP(I,1)
      DIMENSION DB(I,1),NB(I,1),XV(I,1),YV(I,1),ZV(I,1),DBT(3),NS(I,1)
      DIMENSION IC(1),YC(1),ZC(1),LEV(1)
      COMMON /FORTH/NBX                                                 PT50050
C-----------------------------------------------------------------------PT50060
C      THE PURPOSE OF THIS SUBROUTINE IS TO PRINT OUT THE INSTANTANEOUS PT50070
C      FLOW-FIELD PROPERTIES.                                           PT50080
C-----------------------------------------------------------------------PT50090
C                                                                       PT50100
C                                                                       PT50110
C                                                                       PT50120
C                     FORMATS                                           PT50130
C                                                                       PT50140
C                                                                       PT50150
    1 FORMAT(//1X,45('*'),' INSTANTANEOUS FLOW FIELD INFORMATION ',45('*PT50160
     1'))                                                               PT50170
    2 FORMAT(/2X,'LEVEL=',I3,3X,'WEIGHTING FACTOR(MAX)=',I3,3X,'WEDGE AN
     1GLE =',F7.2,' DEGREES',3X,' RADIAL POSITION =',E11.3/2X,'BOX# X PO
     2SITION  SAMP DENSITY MACH NO  X VEL.  Y VEL.  Z VEL.  T(KIN)  T(RO
     3T)   TEMP.',14X,'MOLE FRACTIONS')
    3 FORMAT(//1X,46('*'),' ACCUMULATED FLOW FIELD INFORMATION ',46('*')
     1)
    4 FORMAT(1X,I4,E11.3,I6,8F8.3,3X,3E11.3)
C                                                                       PT50220
C***********************************************************************PT50230
C                                                                       PT50240
C                                                                       PT50250
      IF(KS.EQ.0) WRITE(6,1)
      IF(KS.NE.0) WRITE(6,3)
      DO 40 MT=1,3
   40 DBT(MT)=0.0
      FDA=0.
      CHT=0.
      DO 50 MT=1,MSP
      CHT=CHT+CHI(MT)*RNU(MT)
   50 FDA=FDA+FDN(MT)*WTM(MT)
      YCT=0.0
      ZCT=0.0
      LEVEL=1
```

```
                                                              PT50290
      DO 110 M=1,NBX
      N=NUMCEL(M)
      IF(N.LE.0) GO TO 110
      IF((ZC(M).EQ.ZCT).AND.(YC(M).EQ.YCT)) GO TO 52
      ZCT=ZC(M)
      YCT=YC(M)
      IF(M.GE.LEV(1)) LEVEL=2
      IF(M.GE.LEV(2)) LEVEL=3
      WRITE(6,2) LEVEL,LKW(N),ZCT,YCT
   52 XCT=XC(M)
      NSAMP=0
      DBA=0.
      XVM=0.
      YVM=0.
      ZVM=0.
      TMPM=0.
      TRPM=0.
      E=0.
      F=0.
      DO 100 MT=1,MSP
      NSAMP=NSAMP+NS(MT,N)
      XVM=XVM+XV(MT,N)*RNU(MT)*WTM(MT)*NB(MT,N)
      YVM=YVM+YV(MT,N)*RNU(MT)*WTM(MT)*NB(MT,N)
      ZVM=ZVM+ZV(MT,N)*RNU(MT)*WTM(MT)*NB(MT,N)
      DBA=DBA+DB(MT,N)*WTM(MT)
      TMPM=TMPM+WTM(MT)*RNU(MT)*TMP(MT,N)*NB(MT,N)
      TRPM=TRPM+RNU(MT)*NB(MT,N)*TRP(MT,N)
      E=E+WTM(MT)*RNU(MT)*NB(MT,N)
  100 F=F+RNU(MT)*NB(MT,N)
      DBA=DBA/FDA
      IF(E.EQ.0.0) GO TO 55
      XVM=XVM/E
      YVM=YVM/E
      ZVM=ZVM/E
      VS=XVM**2+YVM**2+ZVM**2
      TMPM=TMPM/E-VS
      TRPM=TRPM/F
   55 CONTINUE
      TTM=(TMPM+TRPM)/(2.5+CHT)
      TMPM=TMPM/1.5
      IF(CHT.NE.-1.) TRPM=TRPM/(1.+CHT)
      AMS=SQRT(VS)
      IF(TTM.GT.0.) AMS=SQRT((5.+2.*CHT)*VS/(TTM*(3.5+CHT)))
      CCZ=COS(ZCT/57.29578)
      SCZ=SQRT(1.-CCZ**2)
      RVM=ZVM*SCZ-YVM*CCZ
      TVM=YVM*SCZ+ZVM*CCZ
      DO 60 MT=1,MSP
      DBT(MT)=RNU(MT)*NB(MT,N)
      IF(F.NE.0.) DBT(MT)=DBT(MT)/F
   60 CONTINUE
      WRITE(6,4) M,XCT,NSAMP,DBA,AMS,XVM,RVM,TVM,TMPM,TRPM,TTM,(DBT(J),
     1J=1,3)                                                  PT50470
  110 CONTINUE                                                PT50480
      RETURN
```

```
     END                                                          PT50490
/LKED.SYSLMOD DD DSN=U.GKBSPACE.ROTATION(FORTH),
/      DISP=OLD,UNIT=3350,VOL=SER=RES101,SPACE=
LKED.SYSIN DD *
      INCLUDE SYSLMOD(FORTH)
      ENTRY MAIN
/ EXEC COMPRESS,DSN='U.GKBSPACE.ROTATION',RLSE=RLSE
```

APPENDIX B

MASTER'S THESIS OF Y.P. TSAI

COLLISION INDUCED VIBRATIONAL TRANSITION PROBABILITIES IN DIATOMIC MOLECULES

ABSTRACT

In order to improve the Monte-Carlo Direct Simulation calculations for hypersonic flow in the transition regime, we have to incorporate the effects due to vibrational non-equilibrium and potenial dissociation for a diatomic gas. The state-to-state transition probabilities are desired. In this paper, we model the diatonic molecule as a harmonic oscillator which collides with another molecule collinearly. Two different methods have been developed, the first one is the semi-classical treatment and the second one is the fully quantum mechanical approach. The interaction potential between two molecules is assumed to be the Lennard-Jones 12-6 interaction law which is a small perturbation to the colliding system. Some numerical results of the state-to-state transition probabilities and comparisons are presented in Chapter 4. Discussions, which present the important aspects of this kind of problem for further study, are made in Chapter 5.

Chapter 1 Introduction

The characteristic flow in a highly rarefied gas is called "free molecular flow". In this regime the mean free path is large compared to the characteristic dimensions of an aerodynamic body in the flow; and molecules that impinge on the body, and are then reemitted from it will, in general, be far away from the body before they strike another molecule. The characteristic flow in a moderately rarefied gas is called "slip flow". The flow regime intermediate between slip and molecular flow is known as the "transition flow regime". It corresponds to densities for which the mean free path has the same general order of magnitude as the characteristic dimension of the flow field. There is a dimensionless parameter called the Knudsen number Kn, introduced to serve as a criterion for determining the relative importance of these rarefaction effects.

where
$$Kn = \frac{\lambda}{d} \sim \frac{M}{Re}$$

and

λ = molecular mean free path

d = characteristic dimension of vehicle

M = Mach Number

Re = Reynolds Number

Free molecular flow is usually defined as that flow for which $Kn>10$. Slip flow is characterized by a Knudsen number of a few per cent, $0.01<Kn<0.1$, and the intermediate transition regime corresponds to Knudsen number in the range $0.1<Kn<10$.

These values are, of course, arbitrary. Since the Knudsen number is defined as the ratio of particle mean free path to body size, it therefore increases with altitude for a fixed size body travelling through the atmosphere.

Generally, for a body of the order of a meter at altitudes 150 Kilometers above the earth's surface, the mean free path of the particles is much larger than vehicle size (here the free stream mean free path $\lambda_\infty \approx$ 40m). Hence the particle-particle collision process in the vicinity of the vehicle need not be considered and the relation of measurement on the surface to atmosphereic properties is explained in a relatively straightforward manner through free molecular theory. Below 90Km, (here the free stream mean free path $\lambda_\infty \simeq$ 2.5 cm), the Knudsen number is so small that the fluid can be treated as a continuum with limited influence of transport properties and slip-flow boundary conditions. However, between these two zones, i.e. in the lower region of the thermosphere, is what we called the "transition flow regime" in which neither of the limiting theories is applicable.

Flight at very high altitudes often involves extremely high velocities and resulting high gas temperature. At velocities which correspond to effective temperature of the order of a few thousand degrees Kelvin, the so called "real gas" effects associated with vibration, dissociation, and ionization of the gas molecules can begin to be of importance. We are interested in atmospheric entry of a vehicle such as the space shuttle. The Mach number is generally above 20 in passing through "transition flow regime" and the flow is hypersonic there. At such high speeds, any significant number of collisions between incoming particles and those reflected from the body can produce extreme changes in the environment near the surface of the vehicle. Measurements on the surface are thus strongly affected making inference of conditions in the ambient atmosphere extremely difficult. For the purpose of data interpretation, previous calculation based on direct simulation Monte-Carlo computer technique have been developed by G.A. Bird(1). By using the Monte-Carlo method, the real gas molecules are replaced by their statistical models, and the motion of one or more of the chosen particles is traced by the computer. In the "Bird" method the real gas molecules are simulated by several thousand modeled molecules, rigid spheres in the simplest version(1). Theoretical calculations of the heat transfer and aerodynamic characteristics of a body submerged in the transition flow regime may be carried out in this way. Modifications involving more realistic interaction laws have been also carried out. Molecules, however, contain internal structure, which is important primarily for its effect on the energy content of the flowing gas. Being composed of nuclei and electrons that have motion relative to the center of mass of the molecule, the molecules can possess rotational and vibrational as well as electronic internal states. Therefore, translational energy is not necessarily conserved in all collisions. The hypersonic flow past the sharp leding edge of a flat plate incorporating the effects of rotational non-equilibrium for a diatomic gas was studied by D.I. Pullin, J.K. Harvey, and G.K. Bienkowski(2), and subsequently applied to other blunt body problems(3,4). A different model was used to incorporate the same effect in other works such as references 5 and 6. At the present stage, we want to improve the Monte-Carlo calculations by including the effects due to the vibrational degree of freedom of diatomic molecules.

It is convenient to divide vibrational energy exchange into two cases: the V-V process, in which the total vibrational quantum number of the system is unchanged, and the V-T process in which energy is exchanged between translation and vibration without conserving the vibrational quantum. For harmonic oscillators in the V-V process, the amount of vibrational energy lost by one molecule is gained by the other and no vibrational-translational energy transfer occurs. Considerable interest has been shown in the details of inelastic molecular collisions. The treatment of scattering between particles with internal structure is capable of producing differential and total cross section for state-to-state transitions. Our purpose is to calculate scattering cross sections for different transitions with known initial and final states as a function of collision energy or initial relative velocity. We emphasize here that we need state-to-state transition cross sections rather than some overall "rates of transition" (rate coefficients) which are averaged over an equilibrium velocity distribution for the relative translational motion (e.g. Maxwellian distribution). The reasons are twofold:

1. We are dealing with a hypersonic flow system in a highly non-equilibrium state with a non-Maxwellian distribution of relative velocities of collision. The relative contributions of different energy molecules to the overall rates may therefore be drastically different than in the equilibrium state. This effect is accentuated by the steep rise of cross-sections with energy coupled to the generally decreasing magnitude of the distribution function with energy. This unifies that small changes in the fraction of molecules with high energies due to the non-equilibrium aspect of the flow can have extreme effects on inelastic processes without corresponding effects on mean properties such as density or fluid momentum.

2. The Direct Simulation Monte Carlo Code (DSMCC) consists of tracing a set of "test" molecules through a designated volume surrounding the body. The velocities and internal states of the simulated molecules are altered on the basis of collisions computed (as determined by local

collision probabilities) at fixed time steps. The positions are then advanced to new values on the basis of motion through the time step increments. This detailed computation of individual molecular collisions requires, in principle, state to state transition probabilities in order to incorporate the inelastic energy exchange into the computation of molecular velocities and internal states after collision. While the results we desire must ultimately be consistent with overall measured rates the level of detail necessary within the program is well beyond the level of availability of experimental data.

Early theoretical studies of vibrational, rotational, and translational energy transfer in collisions were based on approximate analytical solutions to the quantum mechanical and classical equations of motion. The method of Zener (7), later to become known as the distorted wave method, and the Born (8) approximation are leading examples of approximate solutions to quantum mechanical collision problems based on first order perturbation theory. A more detailed literature review on previous work in the field of vibrational collisions is given in the next Chapter.

Instead of doing an approximate treatment of a three-dimensional realistic system, in this work, we do an exact numerical treatment of a simpler model one-dimensional system which has some important features in common with the real one. A calculation of transition probabilities for vibrational-vibrational-translational energy transfer in a collision of two diatomic molecules is to be presented. This simple collision model is approximate, utilizing a collinear collision of harmonic oscillators with an exponential repulsion between center atoms and no chemical reaction between the molecules. It may be argued that the configuration allowing the most efficient transfer of energy between translation and vibration is that in which the atoms are collinear. Collinear or head-on collisions make the most significant contribution to the transition probability. The averaged probability is equal to the probability of excitation in a head-on collision times a "steric factor" smaller than unity which takes account of unfavorable trajectories. For homonuclear molecules it is usually taken as $\frac{1}{3}$ (the average of $\cos^2\theta$ taken over a sphere). A more detailed theory for the steric factor of linear molecules has been propounded by Herzfeld (9). It is our belief that an accurate treatment of a collinear model is of more worth than an approximate result for the three-dimensional problem. The latter approach frequently contains errors which are difficult to estimate. The general magnitudes and trends of the transition probability obtained by this restricted treatment can show us qualitatively, or semi-quantitatively, some characteristic features of the problem.

Two different methods for calculating transition probabilities are discussed in this paper. One is a semi-classicial approach, and the other one is a fully quantum mechanical treatment. In the semi-classical calculation,, it is assumed that the vibrational amplitude of the harmonic oscillators are small and so the molecular oscillatioons do not greatly affect the external classical collision trajectory. The trajectory can be calculated from the classical equation of motion. The classical trajectory is assumed to define a time-dependent perturbation potential for the colliding system and quantum theory is used to derive the transition probability. Essentially, the Schrodinger equation is solved subject to certain initial conditions according to time-dependent perturbation theory. A detailed discussion about this theory can be found in the book Quantum Mechanic by Schiff (10).

In the quantum mechanical calculation of transition probability, the wavefunction of the whole system is expanded in terms of the complete set of eigenfunctions of vibrational states of the diatomic molecule. With the aid of the orthonormality property of these eigenfunctions, a set of coupled second order differential equations is obtained for the translational wavefunctions. The transition probability is given by the solution to this set of equations in the asymptotic region, subject to appropriate boundary condition. Several different numerical methods for solving the set of coupled equations have been developed by Diestler and Mckoy (11).

Riley and Kuppermann (12), and Gutshick et al (13). Solving a problem of quantum scattering between two diatomic molecules is then reduced to the task of finding a good numerical scheme for integrating a system of coupled differential equations acccurately. In this work, we use IBM IMSL ROUINE DGEAR to solve for the scattered wavefunction in the asumptotic region and then calculate the transition probability.

In Chapter 2, a brief review of previous semiclassical and quantum mechanical methods on vibrational collisions will be given so that we can identify the new points in the current work in Chapter 3. Chapter 3 is devoted to a discussion of the general theory. We derive some equations and expressions there which make a numerical algorithm feasible. In Chapter 4, theoretical results of state-to-state transition probability specifically for N_2-N_2 collisions at different relative velocities are presented. Comparison is made with published results. Finally, we discuss some important problems related to inelastic molecular collisions which deserve further study because they make extension to a more realistic treatment of molecular scattering possible.

Chapter 2 Literature Review

In 1931, Oldenberg (14) discussed molecular collision processes qualitively to show the persistence of the rotational and vibrational motion. Zener (7) was the first to give a detailed mathematical treatment for collisions in which molecular vibrations are excited or de-excited. He restricted himself to collinear collisions between a diatomic molecule and an atom. His theory was based on the distorted wave method which includes direct transition from the initial state to the final state and assumes that the probability of transition is small. It is a perturbation method and cannot treat strongly coupled system. Takayanagi (15) then extended Zener's one-dimensional treatments to three-dimensional collisions. In order to save computational labor, the modified wave number approximation was introduced. Meanwhile, Schwartz, Slawsky and Herzfeld (16) gave a mathematical formulation for the vibrational transitions based on the distorted wave approximation due to Jackson and Mott (17), simplified by the modified wave number approximation due to Takayanagi, in diatom-diatom collisions. Their formulation is referred to as the SSH theory now and is a quantum mechanical result. For purpose of future reference, we describe SSH theory in more detail.

Consider the head-on collision between two diatomic molecules AB and CD (assume harmonic oscillators). For exponential intermolecular interaction between nearest atoms B and C, one has,

$$V = V_o \exp(-\alpha r_{BC})$$
$$= V_o \exp[-\alpha(R-\lambda_1 r_1-\lambda_2 r_2)] \qquad (2\text{-}1)$$

where R is the distance between centers of mass of two molecules, r_1 and r_2 are vibrational coordinates, $\frac{1}{\alpha}$ is the range of the potential and V_o is a constant.

$$\lambda_1 = \frac{m_A}{m_A + m_B} \quad , \quad \lambda_2 = \frac{m_D}{m_C + m_D}$$

m_i is the mass of the i-th atom. Solving the Schrodinger equation, the transition probability is given in the closed analytic form:

$$P(n_1n_2 \rightarrow n_1'n_2') = \frac{\pi^2}{4}|V_1(n_1'n_1)|^2 \; |V_2(n_2'n_2)|^2 \; \{\frac{\Delta q^2}{\cosh \pi q' - \cosh \pi q}\}^2 . \qquad (2\text{-}2)$$
$$\sinh \pi q' . \sinh \pi q$$

where

$$V_1(n_1'n_1) = \int_{-\infty}^{+\infty} Z_1(n_1',r)\exp[\alpha\lambda_1(r-r_{e1})Z_1(n_1,r)dr \qquad (2\text{-}3)$$

$$V_2(n_2'n_2) = \int_{-\infty}^{+\infty} Z_2(n_2',r)\exp[\alpha\lambda_2(r-r_{e2})Z_2(n_2,r)dr \qquad (2\text{-}4)$$

$Z_i(n_j,r)$ is the vibrational wave function for harmonic oscillator "i" in quantum state n_j; r_{e1} and r_{e2} are equilibrium separations for AB and CD respectively, and

$$q = q_{n_1n_2} = \frac{2kn_1n_2}{\alpha} \qquad (2\text{-}5)$$

$$q' = q_{n_1'n_2'} = \frac{2kn_1'n_2'}{\alpha} \qquad (2\text{-}6)$$

$$\Delta q^2 = q'^2 - q^2 = -\left(\frac{8\mu}{h^2\alpha^2}\right)\left[\varepsilon_1(n_1') + \varepsilon_2(n_2') - \varepsilon_1(n_1) - \varepsilon_2(n_2)\right] \quad (2\text{-}7)$$

The wave number of relative motion before and after collision are k_{n1n2} and $k_{n1',n2'}$, which satisfy the energy conservation law:

$$\varepsilon_1(n_1) + \varepsilon_2(n_2) + \frac{h^2 k n_1^2 n_2}{2\mu} = \varepsilon_1(n_1') + \varepsilon_2(n_2') + \frac{h^2 k^2 n_1' n_2'}{2\mu} \quad (2\text{-}8)$$

u is the reduced mass of the whole system; $\varepsilon_i(n)$ is the energy of molecule "i" in the n-th vibration state. For the special case, V-V transition, in which $n_1+n_2=n_1'+n_2'$ and $\Delta q=0$. Applying L'Hospital's rule, we get:

$$\lim_{q'\to q} \frac{q'^2 - q^2}{\cosh \pi q' - \cosh \pi q} = \frac{2q}{\pi \sinh \pi q}$$

Then the transition probability for V-V process becomes

$$p(n_1 n_2 \to n_1' n_2') = |V_1(n_1' n_1)|^2 |V_2(n_2' n_2)|^2 q^2 \quad (2\text{-}9)$$

SSH theory has been most widely used for quantitative comparison with experimental measurements of vibrational relaxation, but the coupling between rotation has been ignored.

Zelechow, Rapp and Sharp (ZRS) (18) have developed a semi-classical method for calculating transition probabilities for V-V and V-T energy transfer in a collision of two diatomic molecules. Their basic assumption are:

(1) The perturbation potential is linearized in the oscillator coordinates.

(2) The collision velocity is not too high (e.g. the upper limit for N_2-N_2 collision is 10Km/sec) and the collision induced time-varying force constant $k'(t)$ is small compared to k, the characteristic force constant of the molecule.

Under these two conditions, Kerner (19) method can be applied to solve the Schrodinger equation and closed form analytical results are obtained. However, this approach restricts itself to the transitions of processes of symmetric type only. The general formula is:

$$AB(n)+BA(m) \longrightarrow AB(n')+BA(m')$$

where n,m are vibration quantum numbers before collision and n', m' are that after collision. The collision is symmetric in the sense that the two B atoms are in the center.

The development of the high speed electronic computer has made it possible to solve the collision problem by direct numerical techniques. T.E. Sharp and D. Rapp (20) have calculated the vibrational transition probabilities for collisions between a diatomic molecule and an atom. In their semi-classical treatment, an N-state approximation method is used, in which the total wave function is expanded in terms of N eigenfunctions of stationary states of the system including the initial, final and all energetically intervening states. A Runge-Kutta single-step integration method is employed in the computation program. Generally, the value of N needed in expanding the total wave function increases with collisioon velocity. An "exact" solution for any transition probability $P_{j \to k}$ is reached when the addition of more states to the computation results in no significant change in $P_{j \to k}$. We extend this method to collisions between two diatomic molecules (Chapter 3-A).

In quantum mechanical treatment of collisions between two diatomic molecules AB and CD, taking B and C as the inner atoms of the system, the total wave function is expanded in terms of normalized vibrational wave functions $Z_{AB}(n_1, r_1)$ and $Z_{CD}(n_2, r_2)$. That is:

$$\psi = \sum_{n_1} \sum_{n_2} f_{n_1 n_2}(R)\, Z_{AB}(n_1, r_1)\, Z_{CD}(n_2, r_2)$$

Inevitably, we have to solve a system of coupled differential equations, which are equivalent to the Schrodinger equation, of the following form (detailed discussions will be given in Chapter 3-C).

$$\frac{h^2}{2\mu} \quad \frac{d^2}{dR^2} + k^2_{n_1' n_2'} \quad f_{n_1 n_2}(R) = \sum_{n_1} \sum_{n_2} \langle n_1 n_2' | V | n_1 n_2 \rangle f_{n_1 n_2}(R) \quad (2\text{-}10)$$

where $\langle n_1' n_2' | V | n_1 n_2 \rangle$ is the matrix element. In principle if we can obtain the solution to equation (2-10) with the asymptotic form

$$f_{n_1'n_2'}(R) \to 0 \qquad R \to -\infty$$

$$f_{n_1'n_2'}(R) \to \delta_{n_1'n_1}\delta_{n_2'n_2}\exp(-ik_{n_1'n_2'}R) + A_{n_1'n_2';\,n_1n_2}\exp(+ik_{n_1'n_2'}R) \qquad R\to+\infty \tag{2-11}$$

the probability per collision for transition $(n_1,n_2) \longrightarrow (n_1',n_2')$ will be given by:

$$P(n_1n_2\to n_1'n_2') = \frac{k_{n_1'n_2'}}{k_{n_1n_2}}\left|A_{n_1'n_2';n_1n_2}\right|^2 \tag{2-12}$$

A number of numerical methods (7-9) have been proposed for solving the systems of equations (2-10). However, due to the rapidly oscillating wavelike solutions to the Schrodinger equation, the numerical technique is not straightforward. Riley (12) developed the initial-value technique with periodic "reorthogonalization". Gadschick et al. (13), on the other hand introduced a technique of integration using Dirichlet boundery condition and sinple one-step Euler integration. A new method for constructing wave function for bound states and scattering has been proposed by Roy G. Gordon (21), perhaps this procedure can save much computer time. Our quantum mechanical treatment of this molecule scattering problem is similar to the method due to Riley and Kuppermann (12). It is relatively simple and straightforward, but in our procedures, the virtual states (energetically inacessible) are not included in the total wavefunction expansion.

Chapter 3 Theory

A. General Formalism

The collision model is shown in Fig. 3-1.

Fig. 3-1 Collision Coordinates

This figure is the collinear collision configuration between two diatoms AB and CD. Assuming that CD is the target, and AB is the incident projectile from right. The laboratory coordinates of A, B, C and D are x_A, x_B, x_C and x_D; their masses are m_A, m_B, m_C and m_D respectively. Let V_{AB} and V_{CD} be the binding potential of molecules AB and CD. The short range interaction is assumed to be a sum of interatomic interactions,

$$V_{INT} = V_{AC}(x_A-x_C) + V_{AD}(x_A-x_D) + V_{BD}(x_B-x_D) + V_{BC}(x_B-x_C)$$

The interatomic potentials are exponentially decreasing functions, so that for the collinear configuration under consideration, only the term $V_{BC}(x_B-x_C)$ is important and thus

$$V_{INT} \cong V_{BC}(x_B-x_C)$$

The Schrodinger equation for the system is:

$$\left\{- \frac{h^2}{2M_A} \frac{\partial^2}{\partial x_A^2} - \frac{h^2}{2M_B} \frac{\partial^2}{\partial x_B^2} - \frac{h^2}{2M_C} \frac{\partial^2}{\partial x_C^2} - \frac{h^2}{2M_D} \frac{\partial^2}{\partial x_D^2} + V_{AB}(x_A - x_B) + V_{CD}(x_C - x_D)\right.$$

$$\left. + V_{INT}(x_B - x_C) \right\} \psi(x_A, x_B, x_C, x_D) = E_{TOT}\ \psi(x_A, x_B, x_C, x_D) \quad (3\text{-}1)$$

We designate the distance between the centers of mass of two molecules as R. In molecule AB the distance between the atoms is x; in CD it is y. x and y are internal coordinates. i.e.

$$R = \frac{m_A x_A + m_B x_B}{m_A + m_B} - \frac{m_C x_C + m_D x_D}{m_C + m_D}$$

$$x = x_A - x_B$$

$$y = x_c - x_D$$

Let $R_{CM} = \dfrac{m_A x_A + m_B x_B + m_C x_C + m_D x_D}{m_A + m_B + m_C + m_D}$ which is the coordinate of center of mass of the whole system. In terms of the new coordiniates (x, y, R, R_{CM}) the Schrodinger equation becomes:

$$\left\{- \frac{h^2}{2\mu_{AB}} \frac{\partial^2}{\partial x^2} - \frac{h^2}{2\mu_{CD}} \frac{\partial^2}{2y^2} - \frac{h^2}{2\mu} \frac{\partial^2}{\partial R^2} - \frac{h^2}{2M} \frac{\partial^2}{\partial R_{CM}^2} + V_{AB}(x) + V_{CD}(y)\right. \quad (3\text{-}2)$$

$$\left. + V_{INT}\ (R - \gamma_{AB}x - \gamma_{CD}y) \right\} \psi(x, y, R, R_{CM}) = E_{TOT}\ \psi(x, y, R, R_{CM})$$

where $M = m_A + m_B + m_c + m_D$ = total mass of the system.

$\mu = \dfrac{(m_{A+} m_B)\ (m_C + m_D)}{M}$ = reduced mass of the system

$$\mu_{AB} = \frac{m_A m_B}{m_A+m_B} \quad = \text{reduced mass of the molecule AB}$$

$$\mu_{CD} = \frac{m_C m_D}{m_C+m_D} \quad = \text{reduced mass of the molecule CD}$$

$$\gamma_{AB} = \frac{m_A}{m_A+m_B}, \qquad \gamma_{CD} = \frac{m_D}{m_C+m_D}$$

Since there is no external force applied to the system, the center of mass of the system moves like a free particle and its motion can be described by a plane wave $e^{iK_{CM}R_{CM}}$, and the energy of the center of mass, $T_{CM} = \frac{h^2K_{CM}^2}{2M}$, is also a constant of motion. This does not affect the energy transfer and need not be considered further. We can remove the R_{CM}-dependent part of the wave function (x,y,R,R_{CM}) by separation of variables.

Let

$$\Psi(x,y,R,R_{CM}) = \psi(x,y,R)\, e^{iK_{CM}R_{CM}}, \quad E_{TOT} = E + T_{CM}$$

Substituting these into equation (3-2), we arrive at a Schrodinger equation concerning the internal coordinates x,y and the relative motion R of thee two colliding molecules as follows:

$$\left\{-\frac{h^2}{2\mu_{AB}}\frac{\partial^2}{\partial x^2} - \frac{h^2}{2\mu_{CD}}\frac{\partial^2}{\partial y^2} - \frac{h^2}{2\mu}\frac{\partial^2}{\partial R^2} + V_{AB}(x) + V_{CD}(y) + V_{INT}(R-\gamma_{AB}x-\gamma_{CD}y)\right\}$$

$$\psi(x,y,R) = E\psi(x,y,R) \tag{3-3}$$

Define

$$X=x-x_{eq}$$

$$Y=y-y_{eq}$$

where x_{eq} and y_{eq} are equilibrium separations of AB and CD. Then X and Y are displacements from equilibrium of each oscillator. Now we introduce harmonic bonds (intramolecular potential) into AB and CD with force constants k_{AB} and k_{CD}, hence

$$V_{AB} = 1/2\ k_{AB}X^2,$$

$$V_{CD} = 1/2\ k_{CD}Y^2,$$

A conventional representation of the intermolecular potential energy curve is given by the Lennard-Jones 12-6 equation. Since the elementary models for energy transfer are based on expontial potential, the exponential function $V_{INT}(x_B-x_C) \sim e^{-(x_b-x_C)/L} - \varepsilon$ must be fitted to the Lennard-Jones potential (Appendix 1), where L is a parameter characterizing the range of the interaction. Landau and Teller (22) assumed that only the short range repulsive part of the intermolecular potential is steep enough to influence energy transfer, so that the long-range attractive potential ε can be neglected. The molecular interaction is then assumed to be an exponential repulsion between atoms C and B.

Let v = initial relative velocity

$E = 1/2\mu v_0^2$ = initial relative kinetic energy

$$\bar{R} = R - R_T$$

where R_T is the distance at the classical turning point. The potential energy in equation (3-3) may be expressed as

(16):

$$V_{INT}(X,Y,\bar{R}) = E_o \exp\left[\frac{-1}{L}(\bar{R}-\gamma_{AB}X-\gamma_{CD})\right] \tag{3-4}$$

Equation (3-3) can be written in the form:

$$\left\{\frac{-h^2}{2\mu_{AB}}\frac{\partial^2}{\partial x^2} - \frac{-h^2}{2\mu_{CD}}\frac{\partial^2}{\partial Y^2} - \frac{h^2}{2\mu}\frac{\partial^2}{\partial \bar{R}^2} + \frac{1}{2}k_{AB}x^2 + \frac{1}{2}k_{CD}Y^2\right.$$

$$\left. + E_o\exp\left[\frac{-1}{L}(\bar{R}-\gamma_{AB}x-\gamma_{AB}x-\gamma_{CD}Y)\right]\right\}\psi(X,Y,\bar{R}) = E\psi(X,Y,\bar{R}) \tag{3-5A}$$

or

$$\left\{-\frac{h^2}{2\mu_{AB}}\frac{\partial^2}{\partial x^2} - \frac{h^2}{2\mu_{CD}}\frac{\partial}{\partial Y^2} + \frac{1}{2}k_{AB}X^2 + \frac{1}{2}k_{CD}Y^2 + E_o\exp\left(\frac{-1}{L}(\bar{R}-\gamma_{AB}X-\gamma_{CD}Y)\right)\right\}$$

$$\psi(x,Y,\bar{R}) = ih\frac{\partial}{\partial t}\psi(x,Y,\bar{R}) \tag{3-5B}$$

We shall solve the time-independent Schrodinger equation (3-5A) by purely quantum mechanical method and the time-dependent Schrodinger equation (3-5B) by semi-classical method.

B. Semi-classical Calculation

Since the deBroglie wavelength of the relative motion of two molecules is usually very small compared with atomic dimensions (e.g. for N_2-N_2 collisions at velocity 5Km/sec, the deBroglie wavelength is of the order 10^{-15} cm, however the dimension of N_2 molecules is of the order of a few Å), it is a fairly good approximation to use the classical trajectory for relative motion. The classical equations of motion are:

$$\mu \frac{d^2\bar{R}}{dt^2} = - \frac{\partial}{\partial \bar{R}} V_{INT}(X,Y,\bar{R}) \qquad (3\text{-}6A)$$

$$\mu_{AB} \frac{d^2X}{dt^2} = - k_{AB}X - \frac{\partial}{\partial X} V_{INT}(X,Y,\bar{k}) \qquad (3\text{-}6B)$$

$$\mu_{CD} \frac{d^2Y}{dt^2} = - k_{CD}Y - \frac{\partial}{\partial Y} V_{INT}(X,Y,\bar{R}) \qquad (3\text{-}6C)$$

Generally, the incident energy is much larger than the change in the vibrational energy, or in other words, due to small transition probabilities, only a small fraction of the translational energy is transferred to vibrational energy. We may then assume that during the collision, the vibrational amplitudes of the oscillators are not driven to large values, that means

$$X \ll L, \qquad Y \ll L. \qquad (3\text{-}7)$$

In a series calculation given by Wolfberg and Kelley (23), we can see that conditions (3-7) are justifiable. Wolfberg and Kelley have calculated the energy transfer for collisions involving two harmonic oscillators via an exponential collision with L=0.22Å. Other parameters and data are: $m_A=m_B=m_C=m_D=12$a.m.u., angular frequency $=2.3\times10^{14}$ sec^{-1} (cf: for N_2-N_2 collisions $m_A=m_B=m_C=m_D=14$ a.m.u.,$=4.45\times10^{14}sec^{-1}$), the initial energy E_o= 5.078 ev (corresponds to v_o = 9Km/sec), then the vibrational energy transferred to each diatomic molecule is $\Delta E_{CD} = 1.78\times10^{-3}$ ev. Obviously, both ΔE_{AB} and ΔE_{CD} are much less then E_o.

The vibrational amplitude never exceeds 0.007 Å, and since $L \approx 0.22$ Å the conditions (3-7) are fairly well satisfied. At low velocities (still high enough so that the deBroglie wavelength is much less than the atomic dimension), the energy transfer becomes smaller because the oscillator can readjust adiabatically to the perturbation caused by the incident particle. Conditions (3-7) are satisfied even better. Under conditions (3-7), the molecular oscillations do not greatly affect the external classical collision trajectory. Therefore, one can neglect the motion in X and Y in treating the motion in R. Equations (3-6A), (3-6B), and (3-6C) may then be replaced by:

$$\mu \frac{d^2\bar{R}}{dt^2} = - \frac{\partial}{\partial \bar{R}} V_{INT}(\bar{R}, X=0,\ Y=0) \qquad (3\text{-}8)$$

Solving equations (3-8) with V_{INT} given by (3-4), one finds that the trajectory R(t) satisfies the relation

$$\exp(-\frac{\bar{R}(t)}{L}) = \operatorname{sech}^2 (\frac{u_o t}{2L}) \qquad (3\text{-}9)$$

R(t) is then inserted into the interaction potential function $V_{INT}(X,Y,\bar{R})$ to obtain $V_{INT}(X,Y,t)$. In this semi-classical treatment, $V_{INT}(X,Y,t)$ is used as a transition inducing perturbation acting upon a quantum mechanical harmonic oscillator. Finally, from equation (3-5B) we get the time-dependent Schrodinger equation to be solved numerically by first order time-dependent perturbation theory.

$$\left\{-\frac{h^2}{2\mu_{AB}}\frac{\partial^2}{\partial X^2}+\frac{1}{2}k_{AB}X^2-\frac{h^2}{2\mu_{CD}}\frac{\partial^2}{\partial Y^2}+\frac{1}{2}k_{CD}Y^2+E_o\exp\left(\frac{\gamma_{AB}X+\gamma_{CD}Y}{L}\right)\right. \tag{3-10}$$

$$\left.\operatorname{sech}^2\left(\frac{u_C t}{2L}\right)\right\}\psi(X,Y,t)=ih\frac{\partial}{\partial t}\psi(X,Y,t)$$

There is one more comment about the assumptions of this semi-classical model. In principle, if the deBroglie wavelength associated with the relative motion is much less than the atomic dimensions, it is a good approximation to use classical mechanics for solving the relative trajectory. As just mentioned before, if the criteria (3-7) are fulfilled, in other words, each of the oscillation amplitudes of the two colliding diatomic molecules is very small compared to the characteristic range of the interaction potential, a further simplification can be made in solving for $\bar{R}(t)$, the relative trajectory. These procedures correspond to the "approximate" classical method, because the trajectory $\bar{R}(t)$ is determined with X and Y set equal to zero. As a result, these calculations do not include the conservation of energy, and E is assumed to remain as the energy in coordinate $\bar{R}$, regardless of how much excitation occurs in the oscillators. For more accurate semi-classical calculations, one must solve equations (3-6A), (3-6B), and (3-6C) for "exact" classical trajectory $\bar{R}(t)$ to be used as a time-dependent perturbation. Prior to the work of Wolfsberg and

Kelley (23), it has been thought that the criteria (3-7) are automatically satisfied for all low-velocity collisions. Actually, this is not the case Wolfsberg and Kelley proved that the approximate classical method should be limited to collisions between a light particle and a heavy oscillator.

For our case of N_2-N_2 collision, Wolfsberg and Kelley's requirement is satisfied. The energy transferred to each diatomic species is very small compared to the kinetic energy of relative motion. Both molecules are negligibly distorted during collision approach and the molecular oscillations never deviate substantially from their equilibrium configurations. The semi-classical approach is a good approximation even within the high velocity range in which we are interested. The reason is that as v_o increases, the deBroglie wavelength characterizing the relative mootion in R is small compared to the distance over which the interaction potential varies significantly in R, i.e. $L.k_{n1n2} >> 1$. This implies that both q and q' defined by equations (2-5) and (2-6) are much greater than unity. In this limit, q >> 1 and q' >> 1, the quantum mechanical results of equation (2-2) are reduced to the classical results of Landau and Teller (22).

Let ω_{AB} and ω_{CD} be the angular frequencies for the AB and CD molecules, their isolated Hamiltonians are $H^{(AB)}(X)$ and $H^{(CD)}(Y)$ respectively, then

$$\omega_{AB} = \frac{k_{AB}}{\mu_{AB}}$$

$$\omega_{CD} = \frac{k_{CD}}{\mu_{CD}}$$

$$H^{(AB)}(X) = -\frac{h^2}{2\mu_{AB}}\frac{\partial^2}{\partial X^2} + \frac{1}{2}\mu_{AB}\omega_{AB}^2 X^2$$

$$H^{(CD)}(Y) = -\frac{h^2}{2\mu_{CD}}\frac{\partial^2}{\partial Y^2} + \frac{1}{2}\mu_{CD}\omega_{CD}^2 Y^2$$

Suppose the individual eigenfunctions of $H^{(AB)}(X)$ and $H^{(CD)}(Y)$ are $\phi_n^{(AB)}(X)$ and $\phi_j^{(CD)}(Y)$, we have:

$$H^{(AB)}(X)\phi_n^{(AB)}(X) = (n+\tfrac{1}{2})\ h\omega_{AB}\phi_n^{(AB)}(X) \qquad n = 0,1,2...$$

$$H^{(CD)}(Y)\phi_j^{(CD)}(Y) = (j+\tfrac{1}{2})h\omega_{CD}\phi_j^{(CD)}(Y) \qquad j = 0,1,2,...$$

Set

$$\phi_{nj}(X,Y) = \phi_n^{(AB)}(X)\ \phi_j^{(CD)}(Y) \tag{3-11}$$

$$W_{ij} = (n+\tfrac{1}{2})\ h\omega_{AB} + (j+\tfrac{1}{2})h\omega_{CD} \tag{3-12}$$

$$H_o(X,Y) = H^{(AB)}(X) + H^{(CD)}(Y) \tag{3-13}$$

$H_o(X,Y)$ is the unperturbated Hamiltonian of the system. It is obvious that $H_o(X,Y)\phi_{nj}\ (X,Y) = W_{nj}\phi_{nj}(X,Y)$ (3-14) W_{ij} are eigenvalues of the unperturbed Hamiltonian $H_o(X,Y)$. The solutuion to the equation (3-10), i.e. the total wavefunction for the system of two oscillators $\psi(X,T,t)$ can be expanded in terms of the individual harmonic oscillator wavefunctions $\phi_n^{(AB)}(X)$ and $\phi_j^{(CD)}(Y)$.

$$\psi(X,Y,t) = \sum_n \sum_j a_{nj}(t)\phi_n^{(AB)}(X)\ \phi_j^{(CD)}(Y)\ e^{-i(n+\frac{1}{2})\omega_{AB}t}e^{-i(j+\frac{1}{2})\ \omega_{CD}t} \tag{3-15}$$

where the expansion coefficients a_{nj} depend on time. If the oscillators AB and CD are initially in state N and J respectively, then the initial conditions are:

$$a_{nj}(-\infty) = \delta_{nN}\delta_{jJ} \qquad \begin{matrix} n = 0,1,2,\ldots \\ j = 0,1,2,\ldots \end{matrix}$$

(Appendix 2)

The probability of the system ending up with AB in state Q and CD in state K is

$$P_{NJ\to QK} = |a_{QK}(+\infty)|^2$$

We are now in the position to solve for the expansion coefficients a(t). Substituting equation (3-15) into Schrodinger equation (3-10), and using the relations (3-11)⟶(3-14), we find:

$$(i\hbar)\sum_{nj}\sum \frac{d}{dt}\ a_{nj}(t)\phi_n^{(AB)}(X)\phi_j^{(CD)}(Y)\ e^{-i(n+\frac{1}{2}\ \omega_{AB}t}e^{-i(j+\frac{1}{2})\omega_{CD}t}$$

$$= E_o\ \exp\ (\frac{\gamma_{AB}X+\gamma_{CD}Y}{L})\ \operatorname{sech}^2(\frac{u_o t}{2L})\ \sum_{nj}\sum\ a_{nj}(t)\phi_n^{(AB)}(X)\phi_j^{(CD)}(Y)\ e^{-i(n+\frac{1}{2})\omega_{AB}t}\ e^{-i(j+\frac{1}{2})\omega_{CD}t} \tag{3-16}$$

Equation (3-16) is multiplied by $\phi_{n'}^{(AB)*}(X)\phi_{j'}^{(CD)*}(Y)$ on both side, where $\phi_{n'}^{(AB)*}$ is the complex conjugate function of $\phi_{n'}^{(AB)}$ and $\phi_j^{(CD)*}(Y)$ is the complex conjugate of $\phi_j^{(CD)}(Y)$, and then integrated. Equation (3-16) then becomes:

$$(i\hbar)\sum_{n}\sum_{j} \frac{d}{dt} a_{nj}(t)\delta_{nn'}\delta_{jj'}\ e^{-i(n+\frac{1}{2})\omega_{AB}t}\ e^{-i(j+\frac{1}{2})\omega_{CD}t}$$

$$= E_o \operatorname{sech}^2\left(\frac{u_o t}{2L}\right) \sum_{n}\sum_{j} a_{nj}(t)\ \left[\int_{-\infty}^{+\infty} \phi_{n'}^{(AB)^*}(X)e^{\gamma_{AB}X}\phi_n^{(AB)}(X)dX\right] \tag{3-17}$$

$$\left[\int_{-\infty}^{+\infty}\phi_j^{(CD)*}(Y)e^{\gamma_{CD}Y}\phi_j^{(CD)}(Y)dY\right]\ e^{-i(n+\frac{1}{2})\omega_{AB}t}e^{-i(j+\frac{1}{2})\omega_{CD}t}$$

where use has been made of the orthonormal property for the set of harmonic oscillator energy eigenfunctions $\phi_n^{(AB)}(X)$ and $\phi_j^{(CD)}(Y)$.

$$\int_{-\infty}^{+\infty} \phi_{n'}^{(AB)*}(X)\ \phi_n^{(AB)}(X)dX = \delta_{nn'}$$

$$\int_{-\infty}^{+\infty} \phi_{j'}^{(CD)*}(Y)\ \phi_j^{(CD)}(Y)dY = \delta_{jj'}$$

Let $$U_{n'n} = \int_{-\infty}^{+\infty} \phi_{n'}^{(AB)*}(X)e^{\gamma_{AB}X}\phi_n^{AB}(X)\ dX$$

$$V_{j'j} = \int_{-\infty}^{+\infty} \phi_{j'}^{(CD)*}(Y)e^{\gamma_{CD}Y}\phi_j^{(CD)}(Y)dY$$

$U_{n'n}$ and $V_{j'j}$ are the coupling terms between states, called matrix elements. Equations (3-17) can be written now as:

$$\frac{d}{dt} a_{n'j'}(t) = \left(\frac{E_o}{i\hbar}\right) \operatorname{sech}^2\left(\frac{u_o t}{2L}\right) \sum_{n}\sum_{j} a_{nj}(t)\ U_{n'n}V_{j'j}\ e^{-i(n+\frac{1}{2})\omega_{AB}t}\ e^{-i(j+\frac{1}{2})\omega_{CD}t} \tag{3-18}$$

Equation (3-18) is a set of coupled first order differential equations subject to certain initial conditions. Actually, this set of differential equations is equivalent to the Schrodinger equation (3-10). Next, let

$$a_{nj}(t) = A_{nj}(t) + iB_{nj}(t) \tag{3-19}$$

where $A_{nj}(t)$ is the real part of $a_{nj}(t)$, and $B_{nj}(t)$ is the imaginary part of $a_{nj}(t)$. Both $A_{nj}(t)$ and $B_{nj}(t)$ are real functions of time t. We also know that $U_{m'n}$, V_{jj} are real, and

$$e^{i(n+\frac{1}{2})\omega_{AB}t}e^{i(j+\frac{1}{2})\omega_{CD}t} = \cos\ [(n+\tfrac{1}{2})\omega_{AB}t + (j+\tfrac{1}{2})\omega_{CD}t] + i\sin\ [(n+\tfrac{1}{2})\omega_{AB}t + (j+\tfrac{1}{2})\omega_{CD}t] \tag{3-20}$$

Inserting equations (3-19) and (3-20) into equation (3-18), with some algebra, the real part and imaginary part on each side of the equation must be equal separately. So we obtain a set of coupled differential equations for A's and B's:

$$\frac{d}{dt} A_{n'j'}(t) = \frac{E_o}{h} \operatorname{sech}^2 \left(\frac{u_o t}{2L}\right) \sum_{nj}\sum \{A_{nj}(t)\ \sin[(n'-n)\omega_{AB}t + (j'-j)\omega_{CD}t] + B_{nj}(t)\ \cos\ [(n'-n)\omega_{AB}t + (j'-j)\omega_{CD}t]\}\quad U_{n'n}V_{j'j} \tag{3-21}$$

$$\frac{d}{dt}B_{n'j'}(t) = \frac{E_o}{h} \operatorname{sech}^2\left(\frac{v_o t}{2L}\right) \sum_{nj}\sum \{B_{nj}(t)\ \sin\ [(n'-n)\omega_{AB}t + (j'-j)\omega_{CD}t] - A_{nj}(t)\cos\ [(n'-n)\omega_{AB}t + (j'j)\omega_{CD}t]\}\ U_{n'n}V_{n'j}$$

$A_{rs}\ (+\infty)$ and $B_{rs}\ (+\infty)$ are to be found, hence

$$|a_{rs}(+\infty)|^2 = |A_{rs}(+\infty)|^2 + |B_{rs}(+\infty)|^2,$$

the desired transition probability.

In Chapter 2, we mentioned that in the work of ZRS (18), they treated collinear collisions between two diatomic molecules with symmetric configuration only. Therefore, γ_{AB} is equal to γ_{CD} i.e.:

$$\gamma_{AB} = \gamma_{CD} = \gamma$$

The intermolecular potential function in equation (3-10), $E_o \text{sech}^2 (\frac{v_o t}{2L}) \exp (\frac{\gamma_{AB} X + \gamma_{CD} Y)}{L})$is expanded in a Taylor series to the second order of X and Y;

$$E_o \text{sech}^2 (\frac{v_o t}{2L}) \exp [\frac{\gamma}{L} (x+Y)] = E_o \text{sech}^2 (\frac{v_o t}{2L}) \{1 + \frac{\gamma}{L} (X+Y) + \frac{1}{2} (\frac{\gamma}{L})^2 (x+Y)^2\} \quad (3\text{-}22)$$

neglecting high order terms in X,Y. Equation (3-10), with interaction potential given by equation (3-22), can be solved analytically by applying Kerner (19) method. Our semi-classical approach differs from the method of ZRS in that we do not expand the potential function $E_o \text{ sech}^2 (\frac{v_o t}{2L}) \exp (\frac{\gamma_{AB} X + \gamma_{CD} Y}{L})$ into a Taylor series. We make a direct numerical integration of the coupled equations (3-21), which are equivalent to equation (3-10). It is appropriate to carry out classical calculations in which the approximate equation (3-8) is used, but the potential expansion (3-22) is not carried out. Numerical comparison between these two methods for N_2-N_2 collision will be given in Chapter 4.

C. Quantum Mechanical Calculation

To clarify the presentation of the theory, we shall use a scaled Schrodinger equation. Define (13):

$$y^* = \left(\frac{\mu_{CD} k_{CD}}{h^2}\right)^{\frac{1}{4}} Y \qquad (3\text{-}23A)$$

$$x^* = \left(\frac{\mu_{AB} k_{AB}}{h^2}\right)^{\frac{1}{4}} X \qquad (3\text{-}23B)$$

Inserting equations (3-23A) and (3-23B) into equation (3-5A), it becomes:

$$\{\frac{1}{2} h\omega_{CD}\left(-\frac{\partial^2}{\partial y^{*2}} + y^{*2}\right) + \frac{1}{2} h\omega_{AB}\left(-\frac{\partial^2}{\partial x^{*2}} + x^{*2}\right) - \frac{h^2}{2\mu}\frac{\partial^2}{\partial \bar{R}^2}$$

$$+ E_o \exp\left[\frac{-1}{L}\gamma_{CD}\left(\frac{h^2}{\mu_{CD}k_{CD}}\right)^{\frac{1}{4}}\left(\frac{1}{\gamma_{CD}}\left(\frac{\mu_{CD}k_{CD}}{h^2}\right)^{\frac{1}{4}}\bar{R} - y^* - \frac{\gamma_{AB}}{\gamma_{CD}}\left(\frac{\mu_{CD}k_{CD}}{\mu_{AB}k_{AB}}\right)^{\frac{1}{4}} x^*\right)\right]\}$$

$$\psi(x^*, y^*, \bar{R}) = E\psi\ (x^*, y^*, \bar{R}) \qquad (3\text{-}24A)$$

Dividing both sides by 1/2h $_{CD}$, equation (3-24) looks simpler. If we define:

$$\omega = \frac{\omega_{AB}}{\omega_{CD}}, \quad E^* = \frac{1}{\frac{1}{2}}\frac{E}{h\omega_{CD}} \quad r^* = \frac{1}{\gamma_{CD}}\left(\frac{\mu_{CD}k_{CD}}{h^2}\right)^{\frac{1}{4}} \bar{R}$$

$$L^* = \frac{1}{L}\gamma_{CD}\left(\frac{h^2}{\mu_{CD}k_{CD}}\right)^{\frac{1}{4}} = \frac{1}{L}\ \frac{m_D}{m_C+m_D}\left(\frac{h^2}{\mu_{CD}k_{CD}}\right)^{\frac{1}{4}}$$

$$\frac{1}{m} = \frac{h^2}{\mu\omega_{CD}}\left(\frac{\mu_{CD}k_{CD}}{h^2}\right)^{\frac{1}{2}}\frac{1}{\gamma_{CD}^2} = \frac{\mu_{CD}}{\mu}\left(\frac{m_C+m_D}{m_D}\right)^2 = \frac{m_C M}{(m_A+m_B)m_D}$$

$$\beta = \frac{\gamma_{AB}}{\gamma_{CD}}\left(\frac{k_{CD}\mu_{CD}}{k_{AB}\mu_{AB}}\right)^{\frac{1}{4}} = \frac{m_C}{m_B}\left(\frac{\mu_{AB}}{\mu_{CD}}\right)^{\frac{1}{2}}\frac{1}{\omega^{1/2}}$$

The new equation (3-24) is dimensionless, namely;

$$\{(-\frac{\partial}{\partial y^{*2}} + y^{*2}) + (-\frac{\partial^2}{\partial x^{*2}} + x^{*2}) - \frac{1}{\bar{\bar{m}}}\frac{\partial^2}{\partial j^{*2}} + (\frac{E_o}{\frac{1}{2}h\omega_{CD}})\exp[-L^*(r^*-y^* -\beta x^*)]\}\ \psi\ (x^*,y^*,r^*) = E^*\ \psi\ (x^*,y^*,r^*) \qquad (3\text{-}24B)$$

Since the constant coefficient $\frac{E_o}{1/2h\omega_{CD}}$ can be absorbed into the argument of the exponential function exp $(-L^*(r^*-y^*-\beta x^*))$, and the operator $\frac{\partial}{\partial r^{*2}}$ is invariant under the transformation

$$r^* \longrightarrow r^* + \text{any constant},$$

we can choose $\frac{E_o}{1/2h\omega_{CD}} = 1$ with no loss of generality. The notation x^*, y^* and r^* refer to dimensionless variables, we now drop the "*" for convenience. Hence the scaled Schrodinger equation is:

$$\{(-\frac{\partial^2}{\partial y^2} + y^2) + \omega\ (-\frac{\partial^2}{\partial x^2} + x^2) - \frac{1}{\bar{\bar{m}}}\frac{\partial^2}{\partial r} + \exp\ [-L^*(r-y-\beta x)]\}\psi(x,y,r) = E^*\ \psi\ (x,y,r) \qquad (3\text{-}25)$$

The system is specified by the five dimensionless parameters. They are ω, $\bar{m}$, β, L^* and E^*, E^* is energy of the system in term of ground state energy of molecule CD. The dimensionless form of the unperturbated Hamiltonian H_o is:

$$H_o(x,y) = (-\frac{\partial^2}{\partial y^2} + y^2) + \omega\ (-\frac{\partial}{\partial x^2} + x^2)$$

with eigenfunctions $_{ij}(x,y)$ and eigenvalues W_{ij}, then

$$H_o(x,y)\ \phi_{ij}\ (x,y) = W_{ij}\ \phi_{ij}\ (x,y) \qquad \begin{array}{l} i = 0,1,2,\ldots \\ j = 0,1,2,\ldots \end{array}$$

where $$\phi_{ij}(xx,y) = \phi_i(y)\phi_j(x)$$
$$W_{ij} = (2i+1) + (2j+1)\,\omega$$

$\phi_{ij}(x,y)$ is the product of the individual harmonic oscillator wavefunction which indicates that molecule AB with internal coordinate x is the vibrational state j, and molecule CD with internal coordinate y is in the state i. Let the system be in a particular initial state (n_o, m_o). We can expand the total stationary scattering wavefunction $\psi_{nomo}(x,y,r)$, i.e. the solution to equation (3-25) in terms of $\phi_{nm}(x,y)$ because they form a complete set.

$$\psi_{n_om_o}(x,y,r) = \sum_{n=o}^{N-1} \sum_{m=o}^{M-1} f_{nm,nomo}(r)\ \phi_n(y)\phi_m(x) \qquad (3\text{-}26)$$

Substituting equation (3-26) into equation (3-25) get:

$$\sum_{n=0}^{N=1} \sum_{m=0}^{M-1} \{W_{nm} f_{nm,n_om_o}(r)\ \phi_n(y)\ \phi_m(x) - \frac{1}{\bar{m}}\ \phi_n(y)\ \phi_m(x)\ \frac{d^2}{dr^2}\ f_{nm,n_om_o}(r)$$
$$+ \exp\ [-L^*(r-y-\beta x)]\ f_{nm,n_om_o}(r)\ \phi_n(y)\ \phi_m(x)\}$$
$$= E^* \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} f_{nm',n_om_o}(r)\ \phi_n(y)\ \phi_m(x) \qquad (3\text{-}27)$$

where we have used the relation

$$H_o(x,y)\ \phi_{ij}(x,y) = W_{ij}\ \phi_{ij}(x,y)$$

When equation (3-27) is multiplied by $\phi^*_{n'}(y)\phi^*_{m'}(x)$ on both sides and integrated over x and y, we have:

$$\sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \{W_{nm} f_{nm,n_om_o}(r) \langle\phi_{n'm'}(x,y)\rangle - \frac{1}{\bar{m}}\ \frac{d^2}{dy^2}\ f_{nm,n_om_o}(r) \langle\phi_{n'm'}(x,y)|\phi_{nm}(x,y)\rangle$$

$$+ f_{n_om_o,nm}(r)\ \langle\phi_{n'm'}(x,y)|\exp\ [-L^*(r-y-\beta x)]|\phi_{nm}(x,y)\rangle\} \quad (3\text{-}28)$$

$$= E^* \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} f_{nm,n_om_o}(r) \langle\phi_{n'm'}(x,y)|\phi_{nm}(x,y)\rangle$$

where $$\langle \phi_{n'm'}(x,y) | \phi_{nm}(x,y) \rangle \equiv \int_{-\infty}^{+\infty} \phi_{n'}(y)\ \phi_n(y)dy \int_{-\infty}^{+\infty} \phi_{m'}(x)\phi_m(x)dx$$

$$= \delta_{n'm}\delta_{m'm} \qquad (3\text{-}29)$$

$$\langle \phi_{n'm'}(x,y) | \exp\ [-L^*(r-y-\beta x)] | \phi_{nm}(x,y) \rangle$$

$$= \exp\ (-L^* r) \int_{-\infty}^{+\infty} \phi_{n'}(y)\ \exp\ (L^* y)\ \phi_n(y)\ dy \int_{-\infty}^{+\infty} \phi_{m'}(x)\ \exp\ (L^*\beta x)\ \phi_m(x)\ dx$$

Defining: $$V_{n'm',nm}(r) = \bar{m} \langle \phi_{n'm'}(x,y) | \exp\ [-L^*(r-y-\beta x)] | \phi_{nm}(x,y) \rangle \qquad (3\text{-}30)$$

and using equation (3-29) and (3-30), equation (3-28) becomes

$$\bar{m}W_{n'm'}f_{n'm',n_om_o}(r) - \frac{d^2}{dr^2} f_{n'm',n_om_o}(r) + \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} V_{n'm',nm}(r)\ f_{nm,n_om_o}(r)$$

$$= \bar{m}\ E^*\ f_{n'm',n_om_o}(r)$$

Rearranging this equation, we get

$$\frac{d^2}{dr^2} f_{n'm',n_om_o}(r) = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} V_{n'm',nm}(r)\ f_{nm,n_om_o}(r) - \bar{m}\ (E^*-W_{n'm'})\ f_{n'm',n_om_o}(r) \qquad (3\text{-}31)$$

Define

$$k_{ij} = \sqrt{\bar{m}\ (E^* - W_{ij})}$$

$$= \sqrt{\bar{m}\ (E-(2i+1)-(2j+1))} \qquad (3\text{-}32)$$

N and M are the number of states of CD and AB included in the expansion. We introduce (9)

$i = n + m.N \quad n = 0,1,2,...N-1 \quad m = 0,1..M-1$

$j = n + m.N \quad n = 0,1,2,...N-1 \quad m = 0,1..M-1$

$k = n + m.N \quad n = 0,1,2,...N-1 \quad m = 0,1..M-1$

to indicate the states of the system. For example, i = (n,m), j = (n,m) and k = (n,m) etc. By incorporating the definition of k, equation (3-32), we obtain a system of coupled second order ordinary differential equations in matrix form which is equivalent to the Schrodinger equation (3-32).

$$\frac{d^2}{dr^2} \vec{F}(r) = (\vec{V}-\vec{K}^2)\ \vec{F} \tag{3-33}$$

where $(\vec{F}(r))_{ij} = (\vec{F}(r)_{i(n,m)j(n'm')} = f_{nm,n'm'}(r)$

$(\vec{V}(r))_{ij} = (\vec{V}(r)_{i(n,m)j(n'm')} = V_{nm,n'm'}(r)$

and $(\vec{K})_{ij} = (\vec{K})_{i(n,m)j(n'm')} = k_{nm}\delta_{mn'}\delta_{mm'}$

In the asymptotic region where r is very large, V(r) tends to zero, so integrating equation (3-33), we get

$$\lim_{r\to\infty} \vec{F}(r) = e^{-i\vec{K}r}G + e^{i\vec{K}r}\ \vec{J} \tag{3-34}$$

Equation (3-34) is the asymptotic form of $\vec{F}(r)$ at large r. In principle if $\vec{G}$ and $\vec{J}$ are determined, then the transition probability from state $i = i(n_o,m_o)$ to state $j = j(n,m)$ is given by:

$$P_{ij} = |(\vec{J}\ \vec{G}^{-1})_{ji}|\ \frac{k_j}{k_i} \tag{3-35}$$

where $\vec{G}^{-1}$ is the inverse matrix of $\vec{G}$. However, it is rather difficult to find matrices $\vec{G}$ and $\vec{J}$ in a straightforward manner. We go to the following alternative way:

Set

$$\frac{d}{dr}\vec{F}(r) = \vec{E}(r)$$
$$\frac{d}{dr}\vec{E}(r) = (\vec{V}(r) - \vec{K}^2)\ \vec{F}(r) \tag{3-36}$$

Letting r_o be some point in the asymptotic regime, the asymptotic form of $\vec{F}(r)$ may be also written as

$$\lim_{r\to\infty} \vec{F}(r) = e^{-i\vec{K}(r-r_o)}\,\vec{G} + e^{i\vec{K}(r-r_o)}\,\vec{J} \qquad (3\text{-}37)$$

and then,

$$\lim_{r\to\infty} \vec{E}(r) = \lim_{r\to\infty} \frac{d\vec{F}}{dr} = -i\vec{K}e^{-i\vec{K}(r-r_o)}\,\vec{G} + i\vec{K}e^{i\vec{K}(r-r_o)}\,\vec{J}.$$

So, as the point $r = r_o$ in the asymptotic region, it is obvious that:

$$\vec{F}\,(r_o) = \vec{G} + \vec{J}\,. \qquad (3\text{-}38A)$$

$$\vec{E}\,(r_o) = -i\vec{K}.\vec{G} + i\vec{K}.\vec{J} \qquad (3\text{-}38B)$$

These two relations will be used later. From equation (3-37) we have:

$$\lim_{r\to\infty} \vec{F}(r)\vec{G}^{-1}e^{-i\vec{K}r_o} = e^{-i\vec{K}r} + e^{i\vec{K}r}\,(e^{-i\vec{K}r_o}\,\vec{J}\vec{G}^{-1}e^{-i\vec{K}r_o}),$$

then the transition probabilities are:

$$P_{ij} = |(e^{-i\vec{K}r_o}\,\vec{J}\vec{G}^{-1}\,e^{-i\vec{K}r_o})_{ji}|^2\,\frac{k_j}{k_i} \qquad (3\text{-}39)$$

Since k_j is real for open channels, that is to say the incoming particle has sufficient energy to excite the bound particle to any of its lowest N eigenstates, matrix $\vec{K}$ is real too, and these are the only observed ones. In equation (3-39) since $\vec{K}$ is real, we end up with

$$P_{ij} = (\vec{J}\vec{G}^{-1})_{ji}\,\frac{k_j}{k_i}$$

This result is exactly the same as the probabilities obtained based on the asymptotic form oof $\vec{F}(r)$ in equation (3-34). We can find $\vec{G}$ and $\vec{J}$ in terms of $\vec{F}(r_o)$, and $\vec{E}(r_o)$ with no difficult, since $\vec{G}$ and $\vec{J}$ are related to $\vec{F}(r_o)$, $\vec{E}(r_o)$ by equation (3-38A) and (3-38B). Solving for $\vec{G}$ and we obtain:

$$2\vec{G} = \vec{F}(r_o) + i\vec{K}^{-1}\vec{E}(r_o)$$

$$2\vec{J} = \vec{F}(r_o) - i\vec{K}^{-1}\vec{E}(r_o)$$

Defining $\vec{S} = \text{Re } \vec{S} + i.\text{Im } \vec{S} = \vec{J}\vec{G}^{-1}$.

where Re $\vec{S}$ is the real part of $\vec{S}$ and Im $\vec{S}$ is the imaginary part of $\vec{S}$, (both are real) then

$$|\vec{J}\vec{G}^{-1}|^2 = |\vec{S}|^2 = (\text{Re } \vec{S})^2 + (\text{Im } \vec{S})^2$$

We must find Re $\vec{S}$ and Im $\vec{S}$ in terms of $\vec{F}(r_o)$, $\vec{E}(r_o)$ and $\vec{K}$. We know that (19) given three matrices $\vec{A}, \vec{X}$, and $\vec{Y}$ such that

$$\vec{A} = \vec{X} + i\vec{Y}$$

where $\vec{X}$ and $\vec{Y}$ are real, if $\begin{pmatrix} X & Y \\ -Y & X \end{pmatrix}^{-1} = \begin{pmatrix} Z & W \\ -W & Z \end{pmatrix}$

then the inverse matrix of A is

$$\vec{A}^{-1} = \vec{Z} + i\vec{W}.$$

In our case $\vec{F}(r_o)$, $\vec{E}(r_o)$ and $\vec{K}$ are all real, we may define

$$\vec{D} = \vec{K}_{-1}\vec{E}(r_o)$$

so that equation (3-40) becomes:

$$2\vec{G} = \vec{F}(r_o) + i\vec{D}$$

$$2\vec{J} = \vec{F}(r_o) - i\vec{D}$$

Here we want to find the inverse of $\vec{G}$, then set:

$$\begin{pmatrix} \vec{F}(r_o) & \vec{D} \\ -\vec{D} & \vec{F}(r_o) \end{pmatrix}^{-1} = \begin{pmatrix} \vec{Z} & \vec{W} \\ -\vec{W} & \vec{Z} \end{pmatrix} \qquad (3\text{-}41)$$

and solve for $\vec{Z}$, $\vec{W}$ in terms of $\vec{F}(r_o)$ and $\vec{D}$. Equation (3-41) means that $\begin{pmatrix} \vec{F}(r_o) & \vec{D} \\ -\vec{D} & \vec{F}(r_o) \end{pmatrix} \begin{pmatrix} \vec{Z} & \vec{W} \\ -\vec{W} & \vec{Z} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

Or equivalently

$$\vec{F}(r_o)\vec{Z} - \vec{D}\vec{W} = 1$$

$$\vec{F}(r_o)\vec{W} + \vec{D}\vec{W} = 0$$

From the second equation of (3-42),

$$\vec{W} = -(\vec{F}(r_o))_{-1}\overleftrightarrow{DZ}$$

Substituting $\vec{W}$ into the first equation of (3-42), get:

$$(\vec{F}(r_o) + \vec{D}(\vec{F}(r_o))^{-1}\vec{D})\vec{Z} = 1$$

hence $\quad \vec{Z} = (\vec{F}(r_o) + \vec{D}(\vec{F}(r_o))^{-1}\vec{D})^{-1}$

and $\quad \vec{W} = -(\vec{F}(r_o))^{-1}\vec{D}(\vec{F}(r_o) + \vec{D}(\vec{F}(r_o))^{-1}\vec{D})^{-1}$

in turn, $\vec{G} = 2((\vec{F}(r_o)+\vec{D}(\vec{F}(r_o))\ \vec{D})^{-1} - i(\vec{F}(r_o))^{-1}\vec{D}(\vec{F}(r_o)+\vec{D}(\vec{F}(r_o))^{-1}D)^{-1})$

The derived result is clear now:

$$\overleftrightarrow{J}\vec{G}^{-1} = (\vec{F}(r_o) - i\vec{D})((\vec{F}(r_o)+\vec{D}(\vec{F}(r_o))^{-1}\ \vec{D})^{-1} - i(\vec{F}(r_o))^{-1}$$
$$\vec{D}(\vec{F}(r_o)+\vec{D}(\vec{F}(r_o))^{-1}\ \vec{D})^{-1}) = \text{Re}\ \vec{S} + i\text{Im}\ \vec{S} \qquad (3\text{-}43)$$

Since the real part and imaginary part on each side of equation (3-43) are equal to each other respectively, we have:

$$\text{Re}\ \vec{S} = (\vec{F}(r_o) - \vec{D}(\vec{F}(r_o))^{-1}\ \vec{D})(\vec{F}(r_o)+\vec{D}(\vec{F}(r_o))^{-1}D)^{-1}$$
$$\text{IM}\ \vec{S} = -(2\vec{D})(\vec{F}(r_o)+\vec{D}(\vec{F}(r_o))^{-1}\ \vec{D})^{-1} \qquad (3\text{-}44)$$

where $\quad \vec{D} = \vec{K}^{-1}\vec{E}(r_o)$.

The transition probability from state $i = (n_o, m_o)$ to state $j = (n,m)$ is:

$$P_{i(n_o,m_o)\to j(n,m)} = ((\text{Re}\ \vec{S})^2_{ji} + (\text{Im}\ \vec{S})^2_{ji})\ \frac{k_j}{k_i}$$

where Re $\vec{S}$ and Im $\vec{S}$ are given by equation (3-44). This completes the basic principle in calculating transition probability for one-dimensional scattering problem within the quantum mechanical approach. Discussions of boundary conditions in integrating equation (3-36) will be made in the next Chapter.

The original Schrodinger equation has rapidly oscillating wavelike solutions which are difficult to represent numerically. The integration of equation (3-33) is numerically unstable, unless special algorithms are used. Secrest and Johnson, ( ) in their exact quantum mechanical treatment of the one-dimensional scattering problem, convert the coupled differential equations into equivalent integral equations. The integral involved is then replaced by a quadrature sum. The resulting matrix equation is then solved indirectly by numerical method to obtain the transition probabilities. Chan et al. (24), propose a different numerical approach to this problem. It involves converting the set of coupled second-order equations for the translational wavefunctions into first-order equations in matrix form and then solving it by an expotential method developed by W. Magnus (25). The idea was first conceived by Light et al. (26). The method we just discussed for quantum mechanical calculation of transition probabilities is a direct integration of the state expanded Schrodinger equation. This treatment is similar to the method due to Riley and Kupermann (12). It is relatively simple and straightforward, but in our procedures, the virtual states are not included in the total wavefunction expansion. Roy G. Gordon (21) developed another method for integrating coupled differential equations arising in bound state and scattering problems in quantum mechanics. The wavefunctions are constructed in piecewise analytic form, to any prescribed accuracy. The chief advantage of this method is that it avoids searching for the correct initial derivatives of the wavefunction. It is claimed to be numerically very stable.

Chapter 4 Numerical Results

A. Semi-classical Results

For the specific N^2-N^2 molecular collision, the two molecules are identical, then $\omega_o = \omega_{AB} = \omega_{CD} = 4.45 \times 10^{-14}$ sec, $\gamma_{AB} = \gamma_{CD}$, and $m_A = m_B = m_C = m_D = 14.0$ a.m.u... This implies that the matrix elements $U_{n'n}$ and $V_{n'n}$ are equal. The matrix elements $U_{n'n}$ are given in Table 4-1.

As mentioned in Chapter 3-B, we may set:

$$k = j + (n-1)\,J, \qquad j = 1,2,3,\ldots \qquad n = 1,2,3,\ldots$$

where J is an integer which is the number of states of molecules CD and AB included in the expansion of the total wavefunction ψ. The integer k is used to represent the state (j,n) which means that molecule CD is in the state j and molecule AB is in the state n. For example, if we choose J = 4, there are 4x4 = 16 states involved in the expansion of total wavefunction ψ in terms of the individual harmonic oscillator wavefunctions. In other words, there are 16x2=32 coupled first order differential equations to be solved in equation (3-21). (for J greater than 4, the extension is straightforward). In general, J=N, let K=k+N.N, where k=1,2,3,...(N.N). In this way, A_{nj} and B_{nj} can be designated as:

$$Y(k) = A_{nj} \qquad k = 1,2,\ldots(N.N)$$
$$Y(K) = B_{nj} \qquad K = (N.N)+1,(N.N)+2,\ldots 2(N.N)$$

Y is then a vector of length 2(N.N). This is the suitable form for doing numerical integration of equation (3-21). In our computer program, the IBM IMSL ROUTINE DGEAR is called. On input, Y(1), Y(2),....Y(2N*N) supply initial values which are initial conditions for the system. One of the arguments in the subroutine DGEAR, TOL, must be chosen suitably. Otherwise the computer time is unecessarily long. This parameter TOL, is an estimate of the local truncation error. In a series test calculations, we choose N=11, Vo = 8 km/sec and initial state = (1.2), for three different values of TOL, 10^{-7}, 10^{-8}, and 10^{-9}. We obtain the data as shown in Table 4-2. We then choose TOL = 10^{-8}. The initial value of the step size H, is chosen small enough at the beginning of integration so that it can pass the error test (based on TOL). In the subsequent procedures H is adjusted by the routine itself, but changing in the step size always satisfies the error test. The number of states used in the total wavefunction expansion, N*N plays a very important role in integration of equation (3-21).

In principle, we have to increse N until the final transition probabilities converge to values independent of N. Table 4-3 to Table 4-5 show the transition probability as a function of N for initial state (1,2) at low, medium, and high initial relative velocity; i.e. v_o = 3 Km/sec, 6 Km/sec, and 9 Km/sec respectively. From these tables, it is obvious that for high value of velocity v_o, we need more states in the expansion of total wavefunction ψ . Generally speaking, for v_o less than 6 Km/sec, N = 7 i.e. 7x7 = 49 states expansion is good enough for initial states (1,1), (1,2), (2,2), (3,1), and (3,2). For v_o = 7 Km/sec, N should be no less than 9; and for v_o = 8 Km/sec, N should be no less than 11. For v_o = 9 Km/sec, N must be larger than 12. We also find that the value of N depends on the initial state. For example, for initial state (4.1) at velocity 6 Km/sec, only N = 9, i.e. 81 states expansion makes the transition probability converge. The integration limits are adjusted until the constraint equation (4-1) is satisfied: (for suitable N)

$$\left| \sum_{\text{final state}} P_{\text{initial state} \rightarrow \text{final state}} - 1.0 \right| < \delta \quad (4\text{-}1)$$

where δ is an arbitrary small number, we choose $\delta = 10^{-4}$ here. The integration limits depend on the initial relative velocity v_o, e.g. for v_o = 3 Km/sec, the lower limit T = -1.6×10^{-13} and the upper limit TEND = 1.6×10^{-13}; for v_o = 6 Km/sec, T = -1.0×10^{-13}, and TEND = 1.0×10^{-14}; for v_o = 9 Km/sec, T = -5.0×10^{-14}, and TEND = 5.0×10^{-14}. Condition (4-1) serves as a useful criterion on numerical calculations.

The numerical results of transition probabilities for different initial and final states as a function of v_o are obtained by the semi-classical method and shown in Tables

4-6 to 4-10. Figure 4-1 through Figure 4-5 plot the transition probabilities as a function of v_o for five initial states (1,1), (1,2), (2,2), (3.1) and (3,2). A scaling relationship can be very useful for both the theoretical and experimental analysis of molecular scattering problem. With this relationship, it is easier to write the direct Monte Carlo computation program and save computer storage space. For a systematic study of the scaling relationship, we need more data. It is too expensive to be done at this time. In Figure 4-6a and 4-6b, we plot only the scaling relationship for V-T processes (1,i) (1,i+1) at different v_o. Since the collision is symmetric, there is nothing new in the results by changing the initial state (i,j) to (j,i). We examine Figure 4-1 through Figure 4-5 and find that at low energies the probability of transferring a given number of quanta by a V-V process is much greater than the probability of converting them by a V-T process into translational energy. For V-T process, the probability increases rapidly with increasing collision energies in the low energy regime. However, the probability of V-V transfer rises less sharply with increasing collision energy. For V-V transitions involving two quanta jumps such as (3,1) (1,3) and (3,2) (1,4), the transition probability is less than that of V-V process (1,2) (2,1), which involve only one quantum jump. Generally, at low energies, the transition probabilities are very small and multiple quantum transitions are assumed primarily due to stepwise transitions via single collision $j \to j+1 \to \ldots \to K-1 \to K$. At high collision energies the direct transition $j \to K$ has significant contribution to the transition probability of multiple quantum jump. For V-V-T transfer to an adjacent level, processes involving transfer of a single quantum (such as $(2,1) \to (3,1)$ and $(2,1)$ $(2,1)$) are much more probable than processes of several quanta such as $(1,2) \to (3,1)$. Likewise, transitions $(3,2)$ $(4,2)$, $(3,2) \to (2,2)$ and $(3,2) \to (3,1)$ are much more probable than the transition $(3,2) \to (2,1)$. In Figure 4-6a and 4-6b we notice that the general trend of the scaling relationship for transitions

$$(1,i) \longrightarrow (1,i+1),$$

seems to be a weak v_o-dependent function. We need more data for further analysis.

A useful check on numerical results is provided by time-reversal invariance (which leads to the principle of detailed balance). Stated classically, the principle implies that a system executes its motion in reverse if time is allowed to run backward. In quantum scattering processes this means that $P_{ij} = P_{ji}$, i.e., the probability of a transition for state i to state j is equal to that for transition for state j to i. For example, check table 4-8 and table 4-9, we have $P_{(2,2) \to (3,1)} = 0.845 \times 10^{-3}$, $P_{(3,1) \to (2,2)} = 0.843 \times 10^{-3}$, this gives

$$\frac{\left|P_{(3,1)\rightarrow(2,2)} - P_{(2,2)\rightarrow(3,1)}\right|}{P_{(3,1)\rightarrow(2,2)}} = 0.237\%, \text{ at}$$

v_o = 4 Km/sec

At v_o = 6 Km/sec, both $P_{(3,1)\rightarrow(2,2)}$ and $P_{(2,2)\rightarrow(3,1)}$ are equal to 0.121. The principle of time reversal invariance is satisfied quite well.

B. Quantum Mechanical Results and Comparison

Referring to Chapter 3-C, it is clear that the numerical procedures for calculating transition probability by the quantum mechanical method are as follows:

(1) Integrate equation (3-36) and solve for $\vec{F}(r_o)$ and $\vec{E}(r_o)$.

(2) Form $\vec{D}$ and the expression $(\vec{F}(r_o)+\vec{D}(\vec{F}(r_o))^{-1}$ $\vec{D})^{-1}$.

(3) Construct Re $\vec{S}$ and Im $\vec{S}$.

(4) $P_{ij} = ((\text{Re } \vec{S})^2 + (\text{Im } \vec{S})^2_{ji}) \cdot \dfrac{k_j}{k_i}$

For N^2-N^2 collisions the parameters of the system are = 0.113 (corresponding to L = 0.2 A), ω = 1.0, β = 1.0, and $\bar{m}$ = 0.5. The total energy E can be assigned a suitable value which corresponds to some value of v_o. Having all these parameters, the IBM IMSL routine <u>DGEAR</u> is called to integrate equuation (3-36) and find $\vec{F}(r_o)$, $\vec{E}(r_o)$. There are four factors in this problem that can affect the numerical integration.

(1) Integration error.

(2) Number of states retained in the state expansions.

(3) Starting point of integration r_s.

(4) End point of integration r_o.

The local trunction error is conntrolled by TOL, which is one of the arguments of the subroutine DGEAR. We choose TOL = 1.0×10^{-8} here. The starting point is chosen as the point $r = r_s$ at which the largest diagonal element of $\vec{V}(r_o)$

is equal to twice of the total energy there. The starting point is just beyond the classical turning point and in the classical forbidden region. Now, it is appropriate to discuss the initial conditions for this system. We set N = M = 2, so that there are four states involved in the total wavefunction expansion (for high energy collisions we need larger N and M to get more accurate results). The initial states of the molecule AB and CD are n_o and m_o definitely at the beginning, where $n_o = 1,2$; $m_o = 1,2$. Therefore the initial value of the matrix $\vec{F}(r)$ is a unit matrix:

$$\vec{F}_o = \begin{matrix} 1.0 & 0 & 0 & 0 \\ 0 & 1.0 & 0 & 0 \\ 0 & 0 & 1.0 & 0 \\ 0 & 0 & 0 & 1.0 \end{matrix}$$

Since the point $r = r_s$ is in the classical forbidden region, the diagonal element of $\vec{V}(r_s)$ is much larger than the element of K, equation (3-33) becomes:

$$\frac{d^2\vec{F}}{dr^2} = \vec{V}(r_s).\vec{F}$$

The diagonal element of $\vec{F}$, f, satisfies the equation

$$\frac{d^2 f}{dr^2} = me^{-\alpha r_s}.f(r) \qquad \text{as } r\to-\infty$$

The asymptotic solution is then $\lim_{r\to\infty} f(r) = e^{\sqrt{me^{-\alpha r_s}}\,r}$. Recall that $E = \frac{d\vec{F}}{dr}$, this gives the initial value of matrix $\vec{E}$ at starting point r_s as:

$$\vec{E}_o = \begin{matrix} g & 0 & 0 & 0 \\ 0 & g & 0 & 0 \\ 0 & 0 & g & 0 \\ 0 & 0 & 0 & g \end{matrix}$$

where $g = \sqrt{me^{-r_s}}$. The stopping point r_o is chosen as the point where the diagonal element of $V(r_o)$ are less than $\frac{1.0}{5000}$. of the total energy. In principle, we have to increase the total number of states N,M, until the probabilities do not change significantly. We keep N=2, and M=2 in this work and calculate trnsition probabilities among states (1,1), (1,2) and (2,2). Obviously, this four-state expansion is good for low energy collisions only. Table 4-11 and Table 4-12 display the quantum transition probabilities as a function of V_o for initial states (2,1) and (1,1). In Table 4-13 through Table 4-16, we list the probabilities for transitions (1,2) $\rightarrow$ (1,1), (1,2) $\rightarrow$ (2,2), (1,2) $\rightarrow$ (2,1) and (1,1) $\rightarrow$ (1,2) by the four different methods. Since the computer expense is prohibitively large for fully quantum mechanical method, we do not have enough data for plotting purpose. Here, we show that the method just explored does work. The plots of these tables are thus given by Figure 4-7, Figure 4-8, Figure 4-9 and Figure 4-10 respectively.

We check these Figures (4-7) through (4-10) and find that for the three V-T transitions the probability obtained by the semi-classical method is almost one order of magnitude smaller than that obtained from ZRS analytic method. For V-1 process $(1,2) \rightarrow (2,1)$, the semi-classical and ZRS results are very close to each other, however, the ZRS results are always slightly larger than the semi-classical results. The SSH theory is only good for low energy collisions, if the energy is too high the probability is greater than unity. This theory breaks down there. Since we are interested in high energy collision processes of two molecules. We concentrate on the semi-classical method. We believe the semi-classical treatment can supply a reasonable estimate in calculating transition probabilities. This is very helpful because the semi-classical method can save much computer time and the numerical algorithm is relatively simple.

Chapter 5 Discussions

In this final chapter, we discuss some important problems requiring further study for vibrational energy transfer.

A. Three-Dimensional Collisions

One of the assumptions which we have made in the collinear molecular collision model is that the target molecule is struck in the direction of its axis. To avoid this assumption in the collinear treatment, we have to average over the relative orientation of the molecule at the proper stage of calculations. However, the period of rotation is usually comparable with the duration of the collision, there is no simple way to take the average. A constant steric factor is generally used.

Since the rotational energy spacing is much smaller than the vibrational spacing, appreciable rotational scattering occurs over a range of molecule-molecule separations that is considerably longer than that for which vibrational transitions are important. The coupling between rotational and translation is usually strong too, so that the rotational state generally changes before the vibrational transition occurs. When a vibrational transition takes place the corresponding energy change will appear in either translational motion, or rotational motion of molecules, or both. It is obvious that if we calculate the vibrational transition probabilities, effects of rotational motion have to be considered. In the collinear treatment however, we have assumed that the simultaneous rotational and vibrational transitions are not important and the impact parameter is zero. The realistic three-dimensional analyses that take rotational transitions into account should include the correction to the vibrational transition probability that results from the finite size of the rotational energy spacing in future work. Also, the incident particle is described by a plane wave which contains the partial waves of different orbital angular momentum (the one-dimensional model corresponding to an s-wave scattering problem). Usually, many partial waves have to be considered and this makes the problem very difficult.

B. Effect of Anharmonicity

It has been found by experience that the potential energy function of actual diatomic molecules can be represented quite accurately by a simple analytical function called Morse potential, which contains three adjustable parameters. If Morse potentials are used to describe the intramolecular forces, the diagonal matrix elements of the interaction potential which enter into the quantum theory of vibrational energy transfer are approximately but not identically equal. In the calculation given by F.H. Mies, (27) the consideration was restricted to the head-on collision between a structless incident particle and a diatomic molecule. The transition probability is found to decrease markedly when the ratio of the diagonal elements of the initial and final oscillator states is allowed to deviate even slightly from one. The deviation in turn, increases with the anharmonicity of the molecular vibrations, and an anharmonic correction factor of the order of 10^{-1} to 10^{-2} should be applieed to the generally used probability expression for atom-molecule collision. There must exist a correction factor of this kind for molecule-molecule collision.

C. Interaction Potential

Choice of a potential function to be used in calculating the transition probability is a very important task since it affects the results considerably. In the theory of inelastic molecular collisions, the scattering potential to be adopted should be simple enough to make the calculations feasible as long as the essential features of the physics of collision is not lost. This requirement is relaxed if we deal with numerical calculations. The chosen interaction potential for some pair of molecules must be relatively accurate and can be used to represent the real situation. If the intermolecular interaction is strongly orientation dependent, as in the polar gas, the molecules may take a particular orientation during the encounter. For this problem of preferential orientation, a somewhat different treatment is required.

D. Exact Classical Trajectory

In our semi-classical approximation, the classical equations are first solved to obtain the relative motion of the molecules as a function of the time. The time-dependent Schrodinger equation for the internal motion under the external perturbation is then solved to obtain the probabilities of various transitions. However, the occurrence of inelastic processes are not taken into account in solving the classical equation of motion in that the effective intermolecular potential and the effective translational energy depend on the internal state. If the incident energy is much greater than the internal energy, the influence of inelastic process on the relative motion is unimportant. It is a good approximation to ignore the internal state in calculating classical trajectory. For more rigorous calculations, an exact classical trajectory must be found in which the energy conservation law is satisfied. For high energy collisions, the semi-classical treatment is, however, a fairly good approximation. It requires less computational effort and saves much computer time.

Appendix 1 Exponential Interaction Potential

A conventional representation of the intermolecular potential energy curve is given by the Lennard-Jones 12-6 power law (28);

$$V(r) = 4\varepsilon \left(\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right) \qquad \text{(A1-1)}$$

where V(r) is the potential energy at separation r, and r is the distance between atom B and C. This is shown graphically in Figure A1-1.

is the depth of the potential well at intermolecular distance r_m, where the repulsive force $\left(\frac{\sigma}{r}\right)^{12}$ takes over the long range attractive force $\left(\frac{\sigma}{r}\right)^{6}$, and $V(r_m)$ is the minimum of the potential function V(r). σ is the separation at zero energy, when $V(r) = 0$, sometimes loosely called the "collision diameter". The exponential function

$$V_{INT}(r) = \text{constant}.e^{-r/L} - \varepsilon \qquad \text{(A1-2)}$$

must be fitted to the Lennard-Jones potential $V(r)$, equation (A1-1). Here, the choice is made that the magnitudes and slopes of the potentials are set equal at $r = r_c$. r_c is the minimum value of r. These two potentials are illustrated in Figure A1-2.

Figure A1-2. The exponential potential $V_{INT}(r)$ fitted to the Lennard Jones potential $V(r)$. The magnitudes and slopes of the two potentials are set equal to each other at the classical turning point $r = r_c$.

We deduce an approximate formula:

$$L \simeq \frac{\sigma}{17.5}$$

For N_2 molecule, $\sigma = 3.749$ Å, so $L = 0.21$ Å.

Appendix 2 The Choice of Initial Time Reference Coordinate in Quantum Scattering Process

Consider the general scattering in one-dimensional space, A flux of incoming particles with mean momentum p_o are incident from left and scattered by an arbitrary potential distribution V(x) as shown in Figure A2-1, where V(x) is finite and $V(x) \rightarrow 0$ as $x \rightarrow \pm\infty$

Figure A2-1. Particles scattered by an arbitrary potential.

For large and negative x, the wave packet with mean momentum p_o can be superposed as:

$$\psi\,(x,t) = \int_{-\infty}^{+\infty} dp \exp\,(-\alpha(p-p_o)^2) \exp\,(\frac{ipx}{n}) \exp\,(-i\,\frac{p^2}{2m\hbar}\,t)$$

$$+\int_{-\infty}^{+\infty} dpR(p) \exp\,(-\alpha(p-p_o)^2 \exp\,(-\,\frac{ipx}{\hbar}) \exp\,(-i\,\frac{p^2}{2m\hbar}\,t) \qquad \text{(A2-1)}$$

where R(p) is the reflection coefficient which is a constant over a region $\Delta p \sim \frac{1}{\sqrt{\alpha}}$. For large and positive x, the transmitted wave packet is:

$$\psi\ (x,t) = \int_{-\infty}^{+\infty} dpT(p)\exp(-\alpha(p-p_o)^2)\exp(\frac{ipx}{h})\exp(-i\ \frac{p^2t}{2mh}) \qquad \text{(A2-2)}$$

where T(p) is the transmission coefficient which is constant over a region $\Delta p \sim \frac{1}{\sqrt{\alpha}}$. Since for large $|x|$ and $|t|$, the term $\exp(i(\frac{px}{h} - \frac{p^2t}{2mh}))$ in equation (A2-1) and (A2-2) is a very rapidly varying function of momentum p, the integrals are essentially zero unless p, x, and t are such that the stationary phase conditions are satisfied:

$$p \approx p_o$$

$$\frac{\partial}{\partial p}\ (\frac{px}{h} - \frac{p^2t}{2mh}) \approx 0 \qquad \text{(A2-3)}$$

From equation (A2-3) we get:

$$x \approx \frac{p_o}{m}\ t \qquad \text{(A2-4)}$$

where $\frac{p_o}{m}$ is the classical velocity. Equation (A2-4) gives the result that t<<0, if x<<0. Let's check the second term in the right hand side of equation (A2-1). Since R(p) is approximately a constant within a width $\Delta p \sim \frac{1}{\sqrt{\alpha}}$ centered at $p = p_o$, then,

$$\int_{-\infty}^{+\infty} dpR(p)\exp(-(p-p_o)^2)\exp(-i\ \frac{px}{h})\exp(-i\ \frac{p^2t}{2mh})$$

$$= R(p_o)\ \int_{-\infty}^{+\infty} dp\exp(-(p-p_o)^2)\exp(-i(\frac{px}{h} + \frac{p^2t}{2mh})$$

Stationary phase conditions require that:

$$p \approx p_o$$

and

$$\frac{\partial}{\partial p}\left(\frac{px}{h} + \frac{p^2 t}{2mh}\right) \approx 0 \qquad \text{(A2-5)}$$

Equation (A2-5) implies that $x \approx \frac{-p_o t}{m}$, this means that reflection occurs only when t>0 because the reflected wave exists only at $x \longrightarrow -\infty$. Combining the discussions just made, we conclude that the incident particles hit the pootential at $x \approx 0$ and $t \approx 0$, and the initial conditions of the system are described at $t \longrightarrow -\infty$, the final conditions are the states at $t \longrightarrow +\infty$.

REFERENCES

1. G.A. Bird, "Shock Wave Structure in a Rigid Sphere Gas" Academic Press, Vol. 1, P. 216, New York 1965.
2. D.I. Pullin, J.K. Harvey, and G.K. Bienkowski "Hypersonic Leading Edge Flow of a Diatomic Gas by the Direct Simulation Method".

APPENDIX C

COMPUTER CODE INTERNAL

```
// JOB GKB 0367425.GKBSPACE N=WATRUN2 REG=560 T=1.0 P=100 C=0
// EXEC WATFIV
//WATFIV.FT09F001 DD DISP=OLD,DSN=U.GKBSPACE.KM115
//WATFIV.SYSIN DD DATA
$JOB            BIENKOWSKI,T=59,P=100,NOLIST
C       MAIN PROGRAM FOR MONTE CARLO 3-D ENTRANCE PROBLEM CALCULATIONS
C      OBJECTIVE OF THIS MAIN PROGRAM IS TO SET THE DIMENSIONS
C      MAIN RUNNING PROGRAM IS  *** RUN ***
C
C      FOLLOWING TWO CARDS HAVE TO BE ELIMINATED FOR NON IBM MACHINES
C*********************************************************************
      INTEGER*2 LB,NBM,NBN,NB,NBT
      INTEGER*2 LM,LCOL
C*********************************************************************
C
C      THE NEXT CARD IS ASSOCIATED WITH PRINCETON RANDOM NUMBER GENERATOR
C*********************************************************************
      COMMON/RANCOM/NRAN(4)
C*********************************************************************
C
C      THE FOLLOWING DIMENSION STATEMENTS SET THE MAJOR ARRAY DIMENSIONS
C      AND MUST BE CONSISTENT WITH THE FOLLOWING DATA CARD ------------
C
C      NSP=NUMBER OF SPECIES  - EXAMPLE BELOW NSP=1
C      NJV=NUMBER OF SUBDIVISIONS OF INPUT DISTRIBUTION FUNCTION
C          EXAMPLE BELOW NJV=22
C      NMC=NUMBER OF FINAL CELLS   -  EXAMPLE BELOW NMC=150
C      NMP=MAX NUMBER OF MOLECULES OF EACH SPECIES ALLOWED IN PROGRAM.
C          IF EXCEEDED, PROGRAM EITHER FAILS OR RESTARTS AT BEGINNING
C          WITH NUMBER REDUCED BY 10%  - EXAMPLE BELOW NMP=5000
C      NPB=MAXIMUM NUMBER IN EACH CELL - EXAMPLE NPB=150
C
      DIMENSION DBA(1,150),NB(1,150),NBT(1,150)
      DIMENSION TMP(1,150),TMPA(1,150),XV(1,150),XVA(1,150)
      DIMENSION YV(1,150),YVA(1,150),ZV(1,150),ZVA(1,150),DB(1,150)
      DIMENSION TRP(1,150),TRPA(1,150),NBM(1,150)
      DIMENSION NBN(150),T(1,1,150)
      DIMENSION LB(5000),LM(1,5000),ER(1,5000)
      DIMENSION PAU(1,5000),PAV(1,5000),PAW(1,5000)
      DIMENSION PAX(1,5000),PAY(1,5000),PAZ(1,5000),LCOL(1,5000)
      DIMENSION FNB(150),XC(150),YC(150),ZC(150)
      DIMENSION VEL(22,4,1),PFV(22,4,1)
      DATA NSP/1/,NJV/22/,NMC/150/,NMP/5000/,NPB/150/
C
C
    2 FORMAT(/17X,'NORMAL TERMINATION OF THE PROGRAM')
      NAMELIST/DIM/NSP,NJV,NMC,NMP,NPB,NRAN
C
C
      INITIALIZATION OF RANDOM NUMBER GENERATOR - PRINCETON ROUTINE
C
      NRAN(1)=0
      NRAN(2)=0
      NRAN(3)=0
      NRAN(4)=0
C
```

```
      PRINTOUT OF MAJOR ARRAY DIMENSIONS USED ABOVE

      WRITE(6,DIM)

       CALL OF MAIN OPERATING PROGRAM WHICH REQUIRES INPUTS:
       &CONTRL,&TIMES,&FLOREF,&MOLEC,&SHAPES,&GEOM,&INCUPL,&INOUT
       THESE INPUTS ARE ALL CURRENTLY IN THE NAMELIST FORMAT
       AND MAY HAVE TO BE CHANGED IF THAT CONVENTION IS NOT AVAILABLE
       BRIEF DESCRIPTION OF THE PARAMETERS FOLLOWS
 &CONTRL - ONE OCCURRENCE (NEW OR RESTART)
 PARAMETER  DEFAULT   DEFINITION OR EXPLANATION
 NAME       8 BLANKS  ANY ALPHANUMERIC NAME UP TO 8 CHARACTERS
 TITLE      24 BLNKS  ANY ALPHANUMERIC TITLE UP TO 24 CHARACTERS
 PERCNT     .001      ACCURACY IN INTEGRATION PROCEDURES
 ICOPY      1         NUMBER OF ADDITIONAL COPIES OF OUTPUT
 DUMP       .TRUE.    IF TRUE WILL CAUSE SYSTEM DUMP FOR ANY OF 12
                      PROGRAMMER DESIGNED ERROR HALTS.
 DEBUG(1)   .FALSE.   IF TRUE WILL PRINT MESSAGE WHEN CELL POP. EXCEEDS MNB
 DEBUG(2)   .FALSE.   IF TRUE WILL PRINT CPU TIME AROUND EACH PART OF LOOP
 DEBUG(3)   .TRUE.    IF TRUE WILL PRINT CPU TIME REMAINING AT END OF LOOP
 NEW        .TRUE.    IF TRUE - NEW RUN - IF FALSE - RESTART OF RUN
 SAVE       .FALSE.   IF TRUE - SNAPSHOT SAVED ON TAPE(9) FOR RESTART
 REDO       .FALSE.   IF TRUE PROGRAM WILL AUTOMATICALLY RESTART WITH 90%
                      OF TOTAL IF TOTAL CELL POPULATION EXCEEDS MNM

 &TIMES  -  ONE OCCURRENCE (NEW OR RESTART)
 PARAMETER  DEFAULT   DEFINITION OR EXPLANATION
 DTM        - - -     REAL NUMBER - FRACTION OF MEAN FREE TIME PER CYCLE
 ITS        - - -     INTEGER - NUMBER OF CYCLES PER SAMPLE
 ITP        - - -     INTEGER - NUMBER OF CYCLES BETWEEN PRINTOUTS
 TST        - - -     INTEGER - ESTIMATE OF NUMBER OF CYCLES TO STEADY STATE
 TLIM       - - -     INTEGER - TOTAL NUMBER OF CYCLES TO END OF RUN -
                      WILL TERMINATE SOONER IF CPU TIME IS TO BE EXCCEDED

 &FLOREF  -  ONE OCCURRENCE (NEW RUN ONLY)
 PARAMETER  DEFAULT   DEFINITION
 LLM        - - -     INITIAL NUMBER OF MOLECULES  LLM<MNM< OR = NMP
 MNM        - - -     MAXIMUM NUMBER OF MOLECULES PER SPECIES
 MNB        - - -     MAXIMUM NUMBER PER CELL - DIAGNOSTIC ONLY
 MSP        - - -     NUMBER OF MOLECULAR SPECIES  (MAX. IS 3)
 MET          0       IF 0 - DATA IS IN SI (METRIC) UNITS
                      IF>0 - DATA IS IN ENGLISH UNITS
 U          - - -     FLOW VELOCITY (M/SEC) OR (FT/SEC)
 ANGLE      - - -     ANGLE OF ATTACK (DEGREES)
 RNU        0.0       ARRAY GIVING MOLE FRACTIONS OF SPECIES IN FREE STREAM
 RMA        0.0       ARRAY GIVING MOLECULAR WEIGHTS OF SPECIES ABOVE
 TF         - - -     FREE STREAM TEMPERATURE (K OR R)
 DENF       - - -     FREE STREAM NUMBER DENSITY (NUM/M**3 OR NUM/FT**3)

 &MOLEC  -  ONE OCCURRENCE (NEW RUN ONLY)
 PARAMETER  DEFAULT   DEFINITION
 TRF        - - -     REFERENCE TEMPERATURE FOR MOLECULAR DATA
 DIR        0.0       CROSS-SECTIONS AT REFERENCE TEMP. (MSPXMSP)
 ETA        0.0       PARAMETERS IN DIFFUSION AND VISCOSITY LAW (MSPXMSP)
 PHI        0.0       PARAMETERS FOR ROTATIONAL RELAXATION (MSPXMSP)
```

```
CHI          0.0       ROTATIONAL DEGREE OF FREEDOM PARAMETER (NROT/2 - 1)
ACR          .001      ACCURACY IN MOLECULAR COLLISION CALCULATIONS

&SHAPES  -  ND+1 OCCURRENCES WHERE ND=NUMBER OF BODY SEGMENTS (NEW RUN)
PARAMETER  DEFAULT   DEFINITION
              FIRST OCCURRENCE
BODY(I) I>3 - -        NEED NOT BE SPECIFIED
BODY(1)      - - -     START OF BODY (XSTART) IN ARBITRARY COORDINATE
BODY(2)     0.0        TEMPERATURE AT FRONT OF TUBE IN K OR R
BODY(3)     0.0        DIAMETER OF TUBE IN METERS OR FT.
              SUBSEQUENT OCCURRENCES (ND)
BODY(1)     - - -      X COORDINATE FROM FRONT OF BODY OF THE DOWNSTREAM
                       EDGE OF THE CURRENT BODY SEGMENT
BODY(2)     - - -      TEMPERATURE AT THE BACK OF THIS BODY SEGMENT
BODY(3)     - - -      SWITCH - IF NOT 0.0 THIS IS THE LAST SHAPES CARD
BODY(I) I EVEN         ALPHA - ENERGY ACCOMODATION COEFFICIENT FOR SPECIES
BODY(J) J ODD          SIGMA - TANGENTIAL ACCOMODATION COEFF. FOR SPECIES
        I AND J < (4+2*MSP)

&GEOM  - ONE OCCURRENCE  (NEW RUN ONLY)
PARAMETER  DEFAULT   DEFINITION
NWEDG       - - -      INTEGER GIVING THE NUMBER OF WEDGES WITHIN 180 DEGREES
NW          - - -      NUMBER OF FIRST LEVEL CELLS IN X DIRECTION
NH          - - -      NUMBER OF FIRST LEVEL CELLS IN RADIAL DIRECTION

&INCOPL  -  ONE OCCURRENCE (NEW RUN ONLY) - INPUT DISTRIBUTION
PARAMETER  DEFAULT   DEFINITION
FLUXIN        1.0      FLUX INPUT IN TERMS OF FREE STREAM FLUX - ONE
                       NUMBER FOR EACH SPECIES
FCOL          0.0      FRACTION OF ARRIVING MOLECLUES THAT HAVE
                       PREVIOUSLY COLLIDED
RMP         - - -      RATIO OF "CAVITY" PRESSURE TO THE EFFECTIVE
                       PRESSURE OF THE INCOMING STREAM AT ENTRANCE
JV          - - -      THE NUMBER OF VELOCITY INTERVALS FOR DISTRIBUTION
                       FUNCTION INFORMATION
KMX           4        NUMBER OF COMPONENTS OF DISTRIBUTION
                       KMX=3 IF NO ROTATIONAL ENERGY (CHI=-1)
                       KMX=4 IF ROTATIONAL ENERGY IS INCLUDED (CHI>-1)

&INOUT  - MSP*KMX OCCURRENCES
PARAMETER  DEFAULT   DEFINITION
                       MT DESIGNATES SPECIES
                       K=1 DESIGNATES NORMAL VELOCITY
                       K=2 DESIGNATES TANGENTIAL VELOCITY IN FLOW DIRECTION
                       K=3 DESIGNATES TRANSVERSE TANGENTIAL VELOCITY
                       K=4 DESIGNATES ROTATIONAL ENERGY
VARG(J)       - - -    VELOCITY BOUNDARIES FOR DISTRIBUTION FUNCTION
                       1<J<JV VELOCITY BOUNDARIES

CURV(J)       - - -    PROBABILITY OF VELOCITY(OR ROTATIONAL ENERGY)
                       INCIDENT AT ENTRANCE BEING BELOW VARG(J)
```

A SAMPLE INPUT DECK IS GIVEN BELOW:

```
&CONTRL NAME='INTE','RNAL',TITLE=' PAR','ABOL','A AT',' 95K','M  M','ON. ',
: DEBUG=.F.,.F.,.T.,NEW=.T.,SAVE=.F.,ICOPY=0,REDO=.T. &END
&TIMES DTM=.010,ITS=5,ITP=1000,TST=400,TLIM=1000 &END
:FLOREP LLM=2000,MNM=5000,MNB=150,MSP=1,MET=0,U=7485.9,ANGLE=28.,RNU=1.,2*0.,
  RMA=28.94,0.,0.,TF=195.51,DENF=2.52E+19 &END
&MOLEC TRF=1000,DIR=3.5E-19,ETA=.104,PHI=0.0,CHI=-1.,ACR=.001 &END
&SHAPES BODY=0.0,1000.,.00235 &END
&SHAPES BODY=.0025,555.,0.0,2*1.0 &END
&SHAPES BODY=.0050,345.,0.0,2*1.0 &END
;SHAPES BODY=.0100,300.,0.0,2*1.0 &END
&SHAPES BODY=.0200,300.,0.0,2*1.0 &END
&SHAPES BODY=.0300,300.,0.0,2*1.0 &END
&SHAPES BODY=.0400,300.,0.0,2*1.0 &END
&SHAPES BODY=.0500,300.,0.0,2*1.0 &END
&SHAPES BODY=.0600,300.,0.0,2*1.0 &END
,SHAPES BODY=.0700,300.,0.0,2*1.0 &END
&SHAPES BODY=.0800,300.,0.0,2*1.0 &END
&SHAPES BODY=.0870,300.,1.0,2*1.0 &END
&GEOM NWEDG=2,NW=20,NH=3, &END

&INCUPL FLUXIN=2.1429,FCOL=1.0,RSP=0.0,JV=22,KMX=3 &END
:INOUT VARG=0.,1.,2.,3.,4.,5.,6.,7.,8.,9.,10.,11.,12.,13.,14.,15.,16.,17.,18.,
 19.,20.,21.,CURV=0.0,.070,.170,.282,.369,.459,.537,.599,.656,.710,.750,.785,
 .815,.845,.872,.900,.922,.951,.975,.988,.996,1.00, &END.
&INOUT VARG=-20.,-19.,-17.,-15.,-13.,-11.,-9.,-7.,-5.,-3.,-1.,1.,3.,5.,7.,9.,
   11.,13.,15.,17.,19.,CURV=4*0.,.003,.013,.036,.084,.149,.250,.406,.611,.762,
   .871,.932,.962,.984,.995,.999,3*1.0, &END
:INOUT VARG=-20.,-19.,-17.,-15.,-13.,-11.,-9.,-7.,-5.,-3.,-1.,1.,3.,5.,7.,9.,
 11.,13.,15.,17.,19.,CURV=4*0.0,.003,.013,.036,.084,.149,.250,.406,.611,.762,
   .871,.932,.962,.984,.995,.999,3*1.0, &END

      CALL RUN(NSP,NJV,NMC,NMP,NPB,DBA,NB,NBT,TMP,TMPA,XV,XVA,YV,
     1YVA,ZV,ZVA,T,DB,PNB,XC,YC,ZC,LM,PAU,PAV,PAW,PAX,PAY,
     2PAZ,LCOL,TRP,TRPA,ER,LB,NBM,NBN,VEL,PFV)
      WRITE(6,2)
      STOP
      END

      SUBROUTINE RUN(NSP,NJV,NMC,NMP,NPB,DBA,NB,NBT,TMP,TMPA,XV,XVA,
     1 YV,YVA,ZV,ZVA,T,DB,PNB,XC,YC,ZC,LM,PAU,PAV,PAW,PAX,PAY,PAZ,
     2LCOL,TRP,TRPA,ER,LB,NBM,NBN,VEL,PFV)
      MAIN RUNNING PROGRAM ** RUN ***  CALLS ALL OTHER SUROUTINES
 *******************************************************************

      INTEGER*2 LM,LCOL
      INTEGER*2 LB,NBM,NBN,NB,NBT
      INTEGER PRT,SAMP,TST,TLIM,TIME,Q
      LOGICAL DUMP,DEBUG(3),SAVE,NEW,REDO
      REAL INTGRL,LAM,MU,NU,JAY,KAY
      DIMENSION BTA(3),C1(3),C2(3),C3(3),C7(3),C8(3),DFA(3),FL(3)
      DIMENSION FDN(3),MTI(3),MTR(3),JNT(3),KNM(3),NM(3),SR(3)
      DIMENSION NAME(2),TITLE(6)
      DIMENSION RNU(3),RMA(3),WTM(3), CHI(3), DIR(3,3),DAM(3,3),PHI(3,3)
      DIMENSION ETA(3,3), CN8(3,3), CMG(3),CMG(3), CN(3,3,3),CM(3,3,3)
```

```
      DIMENSION CTI(3,3),CTR(3,3),CNI(3,3),CNR(3,3),SN(3),ST(3)
      DIMENSION D1(3),D2(3),D3(3),D4(3),BODY(15),DBG1(3,3),LIMIT(10)
      DIMENSION COEFF(4)
      DIMENSION XLIM(2),NCOL(3,3)
      DIMENSION XCB(18),XS(18),YCB(18),TB(18),ALPHA(3,18),SIGMA(3,18)
      DIMENSION NTS(3,18,12),UTL(3,18,12),UTT(3,18,12)
      DIMENSION VTS(3,18,12),HTSI(3,18,12),HTS(3,18,12)
      DIMENSION UTLI(3,18,12),UTTI(3,18,12),VTSI(3,18,12)
      DIMENSION ENT(3,2),REM(3,2),FLUXIN(3),FCOL(3)
      DIMENSION VARG(42),CURV(42),IFLUX(3,2)
      DIMENSION LB(NSP),NBN(NMC),NBM(NSP,NMC),LM(NSP,NSP)
      DIMENSION ER(NSP,NMP),TRP(NSP,NMC),TRPA(NSP,NMC)
      DIMENSION DBA(NSP,NMC),NB(NSP,NMC),NBT(NSP,NMC)
      DIMENSION TMP(NSP,NMC),TMPA(NSP,NMC),XV(NSP,NMC),XVA(NSP,NMC)
      DIMENSION YV(NSP,NMC),YVA(NSP,NMC),ZV(NSP,NMC),ZVA(NSP,NMC)
      DIMENSION T(NSP,NSP,NMC),DB(NSP,NMC)
      DIMENSION FNB(NMC),XC(NMC),YC(NMC),ZC(NMC)
      DIMENSION VEL(NJV,4,NSP),PFV(NJV,4,NSP)
:
      DIMENSION PAU(NSP,NMP),PAV(NSP,NMP),PAW(NSP,NMP)
      DIMENSION PAX(NSP,NMP),PAY(NSP,NMP),PAZ(NSP,NMP),LCOL(NSP,NMP)
:
:
:                                                                       RUN0460
      COMMON /RANCOM/NRAN(4),KAWLS
      COMMON /FIRST/NL,NW,NH
      COMMON /SECND/BW,BH,RMP,RMN,RMF
      COMMON /THIRD/PI,NBEG,S,SINANG,COSANG,AKN,AKT,AKN1,AKN2,AKT1,AKT2
      COMMON /FORTH/NBX,EM,XR,DUMP,C9,LL(3),LLM
      COMMON /FIFTH/ND,TIME,DTM,TI,ITS,ITP,TST,TLIM,RMA,RNU,DIR          RUN0520
      COMMON /SIXTH/RMB,XSTART,INM,MNM,MNB,NEW,SAVE,PERCNT,NSR,TR       RUN0530
      COMMON /SVNTH/LAM,MU,NU,MT,N,J,X,Y,Z,TUSE                         RUN0540
      COMMON/EIGTH/DENF,U,TF,ANGLE,TBF,CHI,PHI,ETA,WTM,DAM,VELB,XREF
      NAMELIST/CONTRL/NAME,TITLE,PERCNT,ICOPY,DUMP,DEBUG,NEW,SAVE,REDO
      NAMELIST/TIMES/DTM,ITS,ITP,TST,TLIM                               RUN0580
      NAMELIST/FLOREF/LLM,MNM,MNB,MSP,MET,U,ANGLE,RNU,RMA,TF,DENF
      NAMELIST/MOLEC/TBF,DIR,ETA,PHI,CHI,ACR
      NAMELIST/SHAPES/BODY                                              RUN0570
      NAMELIST/GEOM/NWEDG,NW,NH
      NAMELIST/INCUPL/FLUXIN,FCOL,RMP,JV,KMI
      NAMELIST/INOUT/VARG,CURV
      DATA  IC/0/,ICOPY/1/
      DATA DBG1/' GAS',' AT ','110 ','FLOW',' AT ','130 ',' RUN',' AT ',RUN0630
     1'303 '/                                                           RUN0640
      DATA LIMIT/12,9,18,500,3600,70,900,3,20,3/
      DATA TITLE/'    ','    ','    ','    ','    ','    '/           RUN0660
      DATA NAME/'    ','    '/                                          RUN0670
      DATA CPC/0.0/,CPM/0.0/,CPB/0.0/,CPJ/15.0/
:                                                                       RUN0680
:*********************************************************************RUN0690
:                                                                       RUN0700
:                                                                       RUN0710
:                   FORMATS                                             RUN0720
:                                                                       RUN0730
:                                                                       RUN0740
    1 FORMAT(1H1)                                                       RUN0750
```

```
      2 FORMAT(1H1/17X,'RARIFIED SUPERSONIC FLOW OF BINARY GAS',T74,'I')  RUN0760
      3 FORMAT('+',103X,'COPY ',I2)                                       RUN0770
      4 FORMAT(/17X,'FLOW THROUGH ALL THE BOUNDARIES'/5X,'MT',5X,'MASS MOL
     1E FR.',10X,'FCOL.    FLUXIN     FLUXES (ENT)')
      5 FORMAT(3X,I4,F9.2,F9.4,5X,4F10.4)
     25 FORMAT(/5X,'PRESSURE RATIO(INSIDE/ENTRANCE) - EITHER TYPE =',F13.5
     1,'  RMP',T76,'I'/6X,'DENSITY RATIO(INSIDE/ENTRANCE) - EITHER TYPE
     2=',F13.5,'  RMN',T76,'I'/9X,'FLUX RATIO(INSIDE/ENTRANCE) - EITHER
     3TYPE =',F13.5,'  RMF',T76,'I'/)
     26 FORMAT(//10X,'FLUX RATIOS FOR SPECIES RMA =',F7.2/2X,'BOUNDARY',
     1'    INFLUX    REFLUX   NET FLUX   NET FLUX/RHO*U'/2X,'ENTRANCE',
     24F10.4/4X,'CAVITY',4F10.4/)
     30 FORMAT('1TIME =',F6.3,60X,'RANDOM NUMBER GENERATOR HAS BEEN CALLED
     1 ',I10,' TIMES')                                                    RUN0800
     31 FORMAT(' CPU TIME LEFT- ',F8.3)                                   RUN0810
     32 FORMAT(7X,'-MOLECULES-'/3X,3I6)
     33 FORMAT(' TIME = ',F8.3,5X,'COLLISION LOOP=',F8.3,5X,'MOVE LOOP = '
     1,F8.3,5X,'TOTAL TIME = ',F8.3/21X,'2ND MOVE LOOP =',F8.3,5X,
     2'CLEANUP LOOP=',F8.3,4X,'PARTICLE NUMBERS = ',4I6)
     34 FORMAT(9X,'-MOLECULAR COLLISIONS-'/3(3I14/))
     35 FORMAT(2X,'-COLLISIONS WITH SURFACE-'/3X,3I8)
     36 FORMAT(' MAXIMUM NUMBER OF MOLECULES SO FAR- ',I6//)              RUN0880
     38 FORMAT(' EXCESS MOLECULES OCCURRED IN ',3A4)                      RUN0890
     40 FORMAT(/' SOMETHING IS WRONG WITH BOX NUMBERING IN RUN '/9I5,5E14.RUN0900
     15)
     44 FORMAT(' NB(',I2,',',I4,') POPULATION EXCEEDED ',I3,' IN MAIN AT TRUN0930
     1IME =.',F7.3)                                                       RUN0940
     50 FORMAT(///'   SNAP SAVED ON TAPE')                                RUN0950
C                                                                         RUN0960
C*************************************************************************RUN0970
C                                                                         RUN0980
      CPA=ELTIME(0)
C     CALL NOUNDF
      CALL TRAPS(0,1,1000000,1,1)
      LIMIT(4)=NMC
      LIMIT(5)=NMP
      LIMIT(6)=NPB
      LIMIT(7)=NJV
      LIMIT(10)=MSP
      KAWLS=0
      PI=3.141593                                                         RUN1030
      PIROOT=SQRT(PI)                                                     RUN1040
      MET=0
      LARGE=0                                                             RUN1060
      NL=1                                                                RUN1080
      DUMP=.TRUE.                                                         RUN1190
      DEBUG(1)=.FALSE.
      DEBUG(2)=.FALSE.
      DEBUG(3)=.TRUE.
      SAVE=.FALSE.                                                        RUN1230
      NEW=.TRUE.                                                          RUN1260
      REDO=.FALSE.                                                        RUN1240
      PERCNT=.001                                                         RUN1250
      ACR=.001
      DO 58 I=1,15
```

```
   58 BODY(I)=0.0
      DO 60 I=1,3
      RNU(I)=0.0
      RMA(I)=0.0
      CHI(I)=0.0
      FLUXIN(I)=0.0
      FCOL(I)=0.0
      LL(I)=0
      DO 59 J=1,18
      ALPHA(I,J)=1.0
   59 SIGMA(I,J)=1.0
      DO 60 K=1,3
      ETA(I,K)=0.0
      PHI(I,K)=0.0
   60 DIR(I,K)=0.0
      WRITE(6,1)                                                        RUN1330
      READ(5,CONTRL)                                                    RUN1340
      WRITE(6,CONTRL)
      IF(NEW) GO TO 103                                                 RUN1360
      REWIND 9                                                          RUN1370
      READ(9) DENF,U,XREF,TRF,KAWLS,NL,NW,NH,BW,BH,NREG,XLB,XLC,PI,ND,
     1        S,SINANG,COSANG,AKN,AKT,NBX,BM,XR,TIME,DTM,TI,ITS,ITP,TST,
     2        TLIM,RMA,RNU,DIR,XSTART,MNM,MNB,TR,BZC,CN7,DRP,FCP,FNA,
     3        HTF,INM,LLM,NAV,NMAX,NWEDG,PRT,SAMP,AKN1,AKN2,AKT1,AKT2,
     4        BTA,C1,C2,C3,C7,C8,DAM,DFA,FL,DELANG,FDN,HTI,HTR,JNT,KNM,
     5        NM,WTM,C4,VRM,NCOL,CTI,CTR,CNI,CNR,SN,
     6        ST,D1,D2,D3,D4,NRAN,VELR,RMP,RMN,RMF,IFLUX,FLUXIN,
     7        XLIM,COEFF,XCB,XS,YCB,TB,ALPHA,SIGMA,NTS,
     8        UTL,UTT,VTS,HTS,HTSI,ENT,REM,TMPA,
     9        DBA,NB,NBT,TMP,XV,XVA,YV,YVA,ZV,ZVA,T,DB,FNB,XC,YC,ZC,
     A        PAU,PAV,PAW,PAX,PAY,PAZ,LCOL,LM,
     B        ETA,PHI,CHI,CN,CM,CNG,CMG,CN8,TRP,TRPA,MSP,ANGLE,TP,
     C        UTLI,UTTI,VTSI,ER,RMB,LB,NBM,NBN,VEL,PFV,FCOL,JV
     D
      DTMO=DTM
      READ(5,TIMES)
      WRITE(6,TIMES)
      IF(DTM.EQ.DTMO) GO TO 100
      AIME=TIME*DTMO
      TIME=AIME/DTM+0.1
      DO 99 J=1,NSP
      DO 99 L=1,2
      ENT(J,L)=ENT(J,L)*DTM/DTMO
   99 CONTINUE
  100 IF(TI.GT.0.0) TST=TI/DTM
      WRITE(6,2)
      WRITE(6,4)
      WRITE(6,5)(MT,RMA(MT),RNU(MT),FCOL(MT),FLUXIN(MT),(ENT(MT,K),K=1,2
     1),MT=1,MSP)
      WRITE(6,2)                                                        RUN1540
      CALL PRINTA(NWEDG,TITLE,NAME,XCB,YCB,TB,ALPHA,SIGMA,XLIM,
     1COEFF,LIMIT,MSP)
      CALL PRINTB(FNA,MSP,FNB,NM,XLIM,XC,YC,ZC,NB,NSP)
      GO TO 280                                                         RUN1590
  103 READ(5,TIMES)
      WRITE(6,TIMES)
```


C-3

```
      READ(5,FLOREF)
      WRITE(6,FLOREF)
      READ(5,MOLEC)
      WRITE(6,MOLEC)
      IF(MSP.GT.LIMIT(10))CALL DIA  '10,LIMIT(10),MSP)
      CHIM=0.0
      RMR=0.0
      DMR=0.0
      ETT=0.0
      DO 105 M=1,MSP
      RMR=RMR+RMA(M)*RNU(M)
      CHIM=CHIM+CHI(M)*RNU(M)
  105 CONTINUE
      DO 115 K=1,MSP
      DO 115 M=1,MSP
      ETT=ETT+RNU(M)*RNU(K)*ETA(M,K)
      SRMA=SQRT(.5*RMR*(1./RMA(M)+1./RMA(K)))
  115 DMR=DMR+RNU(M)*RNU(K)*DIR(M,K)*(TRF/TP)**(ETA(M,K)/2.)*SRMA
      XREF=1./(DENF*DMR*1.414214)
      VELR=SQRT(16628.64*TP/RMR)
      IF(MET.NE.0)  VELR=SQRT(99437.92*TP/RMR)
      TMR=XREF/VELR
      S=U/VELR
      NREG=1                                                          RUN1640
      ND=1                                                            RUN1650
      READ(5,SHAPES)
      WRITE(6,SHAPES)
      XCB(1)=BODY(1)/XREF
      TB(1)=BODY(2)/TP
      XLIM(1)=XCB(1)
      XSTART=XLIM(1)
      RMB=.5*BODY(3)/XREF
      TR=TB(1)
  104 READ(5,SHAPES)                                                  RUN1670
      WRITE(6,SHAPES)
      ND=ND+1                                                         RUN1680
      IF(ND.GT.LIMIT(3))  CALL DIAG(3,LIMIT(3),ND)                    RUN1690
      XCB(ND)=BODY(1)/XREF
      TB(ND)=BODY(2)/TP
      DO 1104 M=1,MSP
      ALPHA(M,ND)=BODY(2+2*M)
 1104 SIGMA(M,ND)=BODY(3+2*M)
      IF(TB(ND).GT.TR)  TR=TB(ND)                                     RUN1770
      IF(BODY(3).EQ.0.0)  GO TO 104
      NSTEP=NREG+1                                                    RUN1800
      XLIM(NSTEP)=XCB(ND)
      COEFF(1)=0.0
      COEFF(2)=1.0
      COEFF(3)=0.0
      COEFF(4)=-RMB**2
      XR=XLIM(NSTEP)-XSTART
      AKN=1./XR
      AKT=.5/RMB
      RMFP1=TB(1)**(.5+ETT/2.)/(2.*PIROOT*S)
      RMFP2=TB(ND)**(.5+ETT/2.)/(2.*PIROOT*S)
```

```
      AKN1=AKN*RMPP1
      AKN2=AKN*RMPP2
      AKT1=AKT*RMPP1
      AKT2=AKT*RMPP2
      DO 260 N=1,3
      DO 260 M=1,3
      NCOL(N,M)=0
  260 CONTINUE
      READ(5,GEOM)
      READ(5,INCUPL)                                                   RUN2200
      WRITE(6,GEOM)
      WRITE(6,INCUPL)
      DO 116 J=1,42
      VARG(J)=0.0
  116 CURV(J)=0.0
      RMN=RSP*TB(1)/TB(ND)
      RMP=SQRT(RMN*RSP)
      RM=RMB
      BW=XR/NW
      BH=RM/NH
      IF(NWEDG.GT.LIMIT(1)) CALL DIAG(1,LIMIT(1),NWEDG)
      IF(MNM.GT.LIMIT(5)) CALL DIAG(5,LIMIT(5),MNM)                    RUN2230
      IF(MNB.GT.LIMIT(6)) CALL DIAG(6,LIMIT(6),MNB)                    RUN2240
      DELANG=180./NWEDG
      SINANG=SIN(ANGLE/180.*PI)                                        RUN2320
      COSANG=COS(ANGLE/180.*PI)                                        RUN2330
      VOL=PI*RM*RM*XR
      NBX=NW*NH*NWEDG                                                  RUN2420
      IF(NBX.GT.LIMIT(4)) CALL DIAG(4,LIMIT(4),NBX)                    RUN2470
      IF(JV.NE.LIMIT(7)) CALL DIAG(7,LIMIT(7),JV)
      BB=SQRT(TR)                                                      RUN2480
      SRMX=0.0
      DO 916 MT=1,MSP
      WTM(MT)=RMA(MT)/RMR
      BTA(MT)=SQRT(WTM(MT))
      SR(MT)=S*BTA(MT)
      SRT=SR(MT)*FLUXIN(MT)/RNU(MT)
      IF(SRT.GT.SRMX) SRMX=SRT
  916 CONTINUE
      INM=LLM*SQRT(TB(1))/PIROOT/SRMX/(1.+RMN)
      DDN=INM/VOL
      DO 140 MT=1,MSP
      FDN(MT)=RNU(MT)*DDN
      DPA(MT)=RNU(MT)
      SN(MT)=SR(MT)*COSANG                                             RUN2740
      ST(MT)=SR(MT)*SINANG                                             RUN2750
      DO 117 K=1,MSP
      DAM(K,MT)=DIR(K,MT)*(TRF/TF)**(ETA(K,MT)/2.)/DMR
      CN8(K,MT)=DDN/DAM(K,MT)*1.414214
      BT=AMIN1(BTA(K),BTA(MT))
      VR1=S+3.*(1.+SQRT(TR))/BT
      VR2=3.*SQRT((1.+2.*S**2/(5.+CHIM))*(1./WTM(K)+1./WTM(MT)))
      CM(K,MT,1)=AMAX1(VR1,VR2)
      CN(K,MT,1)=RAND(0)*CM(K,MT,1)
      DF=PHI(K,MT)*(CHI(K)+CHI(MT)+2.)-1
```

```
      DS=PHI(K,MT)*(2.-.5*ETA(K,MT))-1.0
      DO 917 N=2,3
      XPM=ACR**AMIN1(DP,DS)
      IF((DP.GT.0.).AND.(DS.GT.0.)) XPM=(DP/(DP+DS))**DP*(DS/(DP+DS))**DS
      XPN=ACR**AMAX1(DP,DS)
      IF((DP.LT.0.).AND.(DS.LT.0.)) XPN=(DP/(DP+DS))**DP*(DS/(DP+DS))**DS
      CM(K,MT,N)=XPM-XPN
      CN(K,MT,N)=RAND(0)*CM(K,MT,N)
      DP=CHI(K)
      DS=CHI(MT)
 917  CONTINUE
 117  CONTINUE
      DO 118 K=1,KMX
      READ(5,INOUT)
      WRITE(6,INOUT)
      DO 118 J=1,JV
      VEL(J,K,MT)=VARG(J)
      PFV(J,K,MT)=CURV(J)
 118  CONTINUE
      ENT(MT,1)=INM*S*DTM*AKN*FLUXIN(MT)/RNU(MT)
      ENT(MT,2)=RMN*ENT(MT,1)*SQRT(TB(ND)/TB(1))
      REM(MT,1)=0.0
      REM(MT,2)=0.0
      LL(MT)=INM*PIROOT*(1.+RMN)*SR(MT)/SQRT(TB(1))*FLUXIN(MT)/RNU(MT)
      CHT=CHI(MT)
      IF(CHT.GT.0.) CMG(MT)=CHT**CHT*EXP(-CHT)
      IF(CHT.EQ.0.) CMG(MT)=1.0
      IF(CHT.LT.0.) CMG(MT)=ACR**CHT*EXP(-ACR)
      CNG(MT)=RAND(0)*CMG(MT)
 140  CONTINUE                                                          RUN3330
      XS(1)=0.0
      DO 155 N=2,ND                                                     RUN3380
 155  XS(N)=(.5*(XCB(N)+XCB(N-1))-XSTART)*AKN                           RUN3390
      YCB(1)=0.0
      DO 160 N=2,ND
 160  YCB(N)=2.*(XCB(N)-XCB(N-1))/RMB
      CALL CELL(BW,BH,NW,NB,XSTART,DELANG,NWEDG,XC,YC,ZC,FNB)
      FNA=0.0
      DO 210 N=1,NBX
 210  FNA=FNA+FNB(N)
      NPX=NBX
 220  TIME=0                                                            RUN4250
      LARGE=0                                                           RUN4260
      SAMP=0                                                            RUN4270
      PRT=0
      NAV=0                                                             RUN4290
      AIMB=0.                                                           RUN4300
      TI=0.0                                                            RUN4310
      DT=DTM
      NMAX=0                                                            RUN4380
      DO 250 MT=1,3
      C1(MT)=RAND(0)                                                    RUN4410
      C2(MT)=RAND(0)                                                    RUN4420
      C3(MT)=RAND(0)                                                    RUN4430
      C7(MT)=RAND(0)                                                    RUN4440
```

```
      C8(MT)=RAND(0)                                                    RUN4450
      D1(MT)=RAND(0)                                                    RUN4460
      D2(MT)=RAND(0)                                                    RUN4470
      D3(MT)=RAND(0)                                                    RUN4480
      D4(MT)=RAND(0)                                                    RUN4490
      FL(MT)=0.                                                         RUN4500
      HTI(MT)=0.                                                        RUN4510
      HTR(MT)=0.                                                        RUN4520
      JNT(MT)=0                                                         RUN4530
      NM(MT)=0                                                          RUN4540
      IFLUX(MT,1)=0
      IFLUX(MT,2)=0
      DO 230 N=1,3                                                      RUN4550
      CTI(MT,N)=0.                                                      RUN4560
      CTR(MT,N)=0.                                                      RUN4570
      CNI(MT,N)=0.                                                      RUN4580
  230 CNR(MT,N)=0.                                                      RUN4590
      DO 240 N=1,ND                                                     RUN4600
      DO 240 K=1,NWEDG                                                  RUN4610
      NTS(MT,N,K)=0                                                     RUN4620
      HTSI(MT,N,K)=0.                                                   RUN4640
      UTLI(MT,N,K)=0.
      UTTI(MT,N,K)=0.
      VTSI(MT,N,K)=0.
      UTL(MT,N,K)=0.                                                    RUN4650
      UTT(MT,N,K)=0.                                                    RUN4660
      VTS(MT,N,K)=0.                                                    RUN4670
  240 HTS(MT,N,K)=0.                                                    RUN4680
      DO 245 N=1,NPX                                                    RUN4690
      NB(MT,N)=0                                                        RUN4700
      NBT(MT,N)=0                                                       RUN4720
      DBA(MT,N)=0.                                                      RUN4730
      XVA(MT,N)=0.                                                      RUN4740
      YVA(MT,N)=0.                                                      RUN4750
      ZVA(MT,N)=0.                                                      RUN4760
      TMPA(MT,N)=0.                                                     RUN4790
      TRPA(MT,N)=0.0
      DO 245 NN=1,NSP
      T(MT,NN,N)=0.0
  245 CONTINUE                                                          RUN4800
  250 CONTINUE
      FND=DDN
      DRF=2./(FND*S*S*RMB*RMB*PI)                                       RUN4930
      FCF=1./(FND*S*RMB*RMB*PI)                                         RUN4940
      HTF=.5*DRF/S
      WRITE(6,2)
      WRITE(6,4)
      WRITE(6,5) (MT,RMA(MT),RNU(MT),FCOL(MT),FLUXIN(MT),(ENT(MT,K),K=1,2
     1),MT=1,MSP)
      WRITE(6,2)                                                        RUN5000
      CALL PRINTA(NWEDG,TITLE,NAME,XCB,YCB,TB,ALPHA,SIGMA,XLIM,
     1COEFF,LIMIT,MSP)
      CALL GAS(NWEDG,DELANG,ND,BTA,C1,DFA,NM,FNB,DB,NB,NBM,NBN,
     1PAU,PAV,PAW,PAX,PAY,PAZ,XLIM,COEFF,LM,LIMIT(4),LIMIT(6),XCB,TB,
     2LARGE,MNM,MNB,DEBUG(1),LCOL,MSP,ER,CHI,CNG,CMG,MSP,LB)
```

```
      CPUTYM=TFIND(0)                                                  RUN5060
      IF(LARGE.NE.0) GO TO 345
      DO 265 I=1,MSP
  265 IF(NM(I).GT.NMAX) NMAX=NM(I)
      CALL PRINTB(FNA,MSP,FNB,NM,XLIM,XC,YC,ZC,NB,NSP)
      IF(DEBUG(2)) WRITE(6,1)
      CALL ACCUM(NMC,NPB,FNB,NB,PAU,PAV,PAW,ER,TMP,TRP,XV,YV,ZV,LM,MSP,
     1NSP,NBM)
      CPA=ELTIME(0)
      CPI=CPA                                                          RUN5130
      GO TO 340                                                        RUN5140
  280 TIME=TIME+1
      IF(TIME.NE.TST+1) GO TO 285
      TI=TST*DTM
      DO 282 MT=1,MSP
      IFLUX(MT,1)=0
  282 IFLUX(MT,2)=0
  285 LARGE=0
      CPI=ELTIME(0)                                                    RUN5170
      AIME=TIME*DTM
      DT=AIME-TI
      IF(DEBUG(1))WRITE(6,33)AIME,CPC,CPM,CPI,CPB,CPA,(NM(I),I=1,3),NMAX
                                                                       RUN5180
      PRT=PRT+1                                                        RUN5190
      SAMP=SAMP+1
      CALL COLIDE(CN,CM,WTM,DB,DBA,NB,NCOL,LCOL,PAU,PAV,PAW,ER,T,LM,MSP,
     1LIMIT(4),LIMIT(6),ETA,PHI,CHI,CN8,NSP,NBM)                       RUN5220
      KNM(1)=0                                                         RUN5230
      KNM(2)=0
      KNM(3)=0
      CPC=ELTIME(0)
      IF(DEBUG(1))WRITE(6,33)AIME,CPC,CPM,CPI,CPB,CPA,(NM(I),I=1,3),NMAX
      CALL MOVE(0,AKN,NWEDG,XSTART,LIMIT(3),LIMIT(1),LIMIT(8),LIMIT(9),
     1DELANG,BTA,C2,C3,DFA,FL,HTI,HTR,JNT,KNM,NM,XCB,XLIM,CTI,CTR,
     2CNI,CNR,ALPHA,SIGMA,COEFF,HTS,HTSI,NTS,UTL,UTT,VTS,PAU,PAV,PAW,
     3PAX,PAY,PAZ,LCOL,TB,MSP,ER,CHI,CNG,CMG,NSP,UTLI,UTTI,VTSI,IFLUX)
                                                                       RUN5280
      KNM(1)=NM(1)                                                     RUN5290
      KNM(2)=NM(2)
      KNM(3)=NM(3)
      CPM=ELTIME(0)
      IF(DEBUG(1))WRITE(6,33)AIME,CPC,CPM,CPI,CPB,CPA,(NM(I),I=1,3),NMAX
      TBI=TB(ND)
      CALL FLOW(NWEDG,MNM,LARGE,BTA,C1,C7,C8,ENT,REM,LCOL,MSP,NM,SN,ST,
     1TBI,PAU,PAV,PAW,PAX,PAY,PAZ,ER,CHI,CNG,CMG,NSP,JV,FCOL,VEL,PFV)  RUN5330
      IF(LARGE.NE.0) GO TO 345
      CPB=ELTIME(0)
      IF(DEBUG(1))WRITE(6,33)AIME,CPC,CPM,CPI,CPB,CPA,(NM(I),I=1,3),NMAX
      CALL MOVE(1,AKN,NWEDG,XSTART,LIMIT(3),LIMIT(1),LIMIT(8),LIMIT(9),
     1DELANG,BTA,C2,C3,DFA,FL,HTI,HTR,JNT,KNM,NM,XCB,XLIM,CTI,CTR,
     2CNI,CNR,ALPHA,SIGMA,COEFF,HTS,HTSI,NTS,UTL,UTT,VTS,PAU,PAV,PAW,
     3PAX,PAY,PAZ,LCOL,TB,MSP,ER,CHI,CNG,CMG,NSP,UTLI,UTTI,VTSI,IFLUX)
      CPB=CPB+ELTIME(0)
      IF(DEBUG(1))WRITE(6,33)AIME,CPC,CPM,CPI,CPB,CPA,(NM(I),I=1,3),NMAX
      DO 330 MT=1,MSP                                                  RUN5420
      DO 290 N=1,NBX                                                   RUN5450
      NB(MT,N)=0
```

```
 290 CONTINUE
     NG=NM(MT)
     N=0
 295 N=N+1
     IF(N.GT.NG) GO TO 310
     X=PAX(MT,N)
     Y=PAY(MT,N)
     Z=PAZ(MT,N)
     R=SQRT(Y*Y+Z*Z)
     TANG=180.*ATAN2(Z,-Y)/PI
     IWDGE=TANG/DELANG+1
     IF(IWDGE.LT.1) IWDGE=1
     IF(IWDGE.GT.NWEDG) IWDGE=NWEDG
     L=X/BW
     IF(L.GE.NW) L=NW-1
     M=R/BH
     IF(M.GE.NH) M=NH-1
     K=NWEDG*(L*NH+M)+IWDGE
     IF(K.LE.NBX) GO TO 305
     WRITE(6,40)L,M,IWDGE,NW,NH,NWEDG,K,MT,N,X,Y,Z,R,TANG
     IF(DUMP) CALL ABEND(4)
     STOP
 305 J=NB(MT,K)+1
     IF(J.LE.MNB) GO TO 308
     IF(DEBUG(1)) WRITE(6,44) MT,K,MNB,AIME
 308 NB(MT,K)=J
     LB(N)=K
     GO TO 295
 310 CONTINUE
     NBM(MT,1)=0
     DO 320 M=1,NBX
     A=NB(MT,M)
     DB(MT,M)=A*DFA(MT)/PNB(M)
     NBM(MT,M+1)=NBM(MT,M)+NB(MT,M)
     NBN(M)=NBM(MT,M)
 320 CONTINUE
     IF(NM(MT).GT.NMAX) NMAX=NM(MT)
     DO 325 N=1,NG
     Q=LB(N)
     NBN(Q)=NBN(Q)+1
     NA=NBN(Q)
 325 LM(MT,NA)=N
 330 CONTINUE
     IF(SAMP.LT.ITS) GO TO 335
     CALL ACCUM(NMC,NPB,PNB,NB,PAU,PAV,PAW,ER,TMP,TRP,XV,YV,ZV,LM,MSP,
    1NSP,NBM)
     SAMP=0
     IF(TIME.LE.TST) GO TO 335
     CALL AVRGE(PNB,DB,DBA,NB,NBT,XV,YV,ZV,XVA,YVA,ZVA,TMP,TMPA,TRP,TRP
    1A,MSP,NSP)
     NAV=NAV+1
 335 CPA=ELTIME(0)
     CPI=CPC+CPM+CPB+CPA
     CPJ=2.*CPI+5.
 340 CPUTYM=TPIND(0)
```

```
      IF(DEBUG(3)) WRITE(6,33)AIME,CPC,CPM,CPI,CPB,CPA,(NM(I),I=1,3),NMAX
      IF(TIME.EQ.0) GO TO 355
      IF((TIME.GE.TLIM).OR.(CPUTYM.LE.CPJ)) GO TO 345
      IF(PRT.LT.ITP) GO TO 280
      PRT=0                                                             RUN6650
 345  WRITE(6,30)AIME,KAWLS                                             RUN6660
      IF(DEBUG(3))    WRITE(6,31)CPUTYM                                 RUN6670
      WRITE(6,32)  (NM(I),I=1,3)
      WRITE(6,34)  ((NCOL(I,J),J=1,3),I=1,3)
      WRITE(6,35)  (JNT(I),I=1,3)
      IF(LARGE.NE.0) GO TO 360
      WRITE(6,36)NMAX                                                   RUN6740
      IF(.NOT.SAVE) GO TO 355                                           RUN6750
      IF(PRT.NE.0.AND.CPUTYM.GT.CPJ.AND.TIME.LT.TLIM) GO TO 355         RUN6900
      REWIND 9
      WRITE(9)  DENP,U,XREF,TRP,KAWLS,NL,NW,NH,BW,BH,NREG,XLB,XLC,PI,ND,
     2          S,SINANG,COSANG,AKN,AKT,NBX,RM,XR,TIME,DTM,TI,ITS,ITP,TST,
     3          TLIM,RMA,RNU,DIR,XSTART,MNM,MNB,TR,BZC,CN7,DRF,FCP,FNA,
     4          HTF,INM,LLM,NAV,NMAX,NWEDG,PRT,SAMP,AKN1,AKN2,AKT1,AKT2,
     5          BTA,C1,C2,C3,C7,C8,DAM,DFA,FL,DELANG,FDN,HTI,HTR,JNT,KNM,
     6          NB,WTM,C4,VRM,NCOL,CTI,CTE,CNI,CNR,SN,
     7          ST,D1,D2,D3,D4,NRAN,VELR,RMP,RMN,RMF,IFLUX,FLUXIN,
     8          XLIM,COEFF,XCB,XS,YCB,TB,ALPHA,SIGMA,NTS,
     9          UTL,UTT,VTS,HTS,HTSI,ENT,EEM,TMPA,
     A          DBA,NB,NBT,TMP,XV,XVA,YV,YVA,ZV,ZVA,T,DB,FNB,XC,YC,ZC,
     B          PAU,PAV,PAW,PAX,PAY,PAZ,LCOL,LM,
     C          ETA,PHI,CHI,CN,CM,CNG,CMG,CN8,TRP,TRPA,MSP,ANGLE,TP,
     D          UTLI,UTTI,VTSI,ER,EMB,LB,NBM,NBN,VEL,PPV,PCOL,JV
      WRITE(6,50)
 355  CONTINUE                                                          RUN7050
      WRITE(6,25) RMP,RMN,RMF
      DO 356 MT=1,MSP
      FIN1=ENT(MT,1)
      FIN2=ENT(MT,2)
      RF1=IFLUX(MT,1)*DTM/DT
      RF2=IFLUX(MT,2)*DTM/DT
      RNF1=1.-RF1/FIN1
      RNF2=(FIN2-RF2)/FIN1
      RFS1=RNF1*FLUXIN(MT)/RNU(MT)
      RFS2=RNF2*FLUXIN(MT)/RNU(MT)
 356  WRITE(6,26) RMA(MT),FIN1,RF1,RNF1,RFS1,FIN2,RF2,RNF2,RFS2
      IF(TIME.EQ.0) GO TO 280
      IF(TIME.LE.TST) GO TO 350
      CALL PRINT1(DT,COSANG,SINANG,RMA,RNU,DRF,FCP,HTF,FL,HTI,HTR,CTI,
     1CTE,CNI,CNR)
      CALL PRINT2(AKN,XSTART,DT,RNU,RMA,DRF,FCP,HTF,UTLI,UTTI,VTSI,HTSI,
     1DELANG,NWEDG,XS,XCB,YCB,HTS,NTS,UTL,UTT,VTS,LIMIT(3),LIMIT(1),MSP)
      CALL PRINT4(MSP,CHI,RNU,NSP,TRPA,FDN,WTM,DBA,NBT,TMPA,XVA,
     1YVA,ZVA,1,NBT,XC,YC,ZC)
      GO TO 353
 350  CONTINUE                                                          RUN6860
      CALL PRINT4(MSP,CHI,RNU,NSP,TRP,FDN,WTM,DB,NB,TMP,XV,YV,ZV,
     10,NB,XC,YC,ZC)
 353  IF(DEBUG(2)) WRITE(6,1)
      IF((TIME.LT.TLIM).AND.(CPUTYM.GT.CPJ)) GO TO 280
```

ORIGINAL PAGE IS
OF POOR QUALITY


```
      IF(IC.EQ.ICOPY) RETURN
      IC=IC+1                                                           RUN7080
      WRITE(6,2)
      WRITE(6,4)
      WRITE(6,5) (MT,RMA(MT),RNU(MT),PCOL(MT),FLUXIN(MT),(ENT(MT,K),K=1,2
     1),MT=1,MSP)
      WRITE(6,2)                                                        RUN7120
      WRITE(6,3)IC                                                      RUN7130
      CALL PRINTA(NWEDG,TITLE,NAME,XCB,YCB,TB,ALPHA,SIGMA,XLIM,
     1COEFF,LIMIT,MSP)
      CALL PRINTB(FNA,MSP,FNB,NM,XLIM,XC,YC,ZC,NB,NSP)
      SAVE=.FALSE.                                                      RUN7180
      GO TO 345                                                         RUN7190
  360 WRITE(6,38) (DBG1(I,LARGE),I=1,3)                                 RUN7200
      IF((REDO).AND.(TIME.LE.TST)) GO TO 364
      IF(DUMP) CALL ABEND(9)                                            RUN7220
      STOP                                                              RUN7230
  364 CONTINUE
      IF(NEW) GO TO 365
      REWIND 9
      READ(9) DENF,U,XREF,TRF,KAWLS,KL,NW,NH,BW,BH,NREG,XLB,XLC,PI,ND,
     2        S,SINANG,COSANG,AKN,AKT,NBX,RM,XR,TIME,DTM,TI,ITS,ITP,TST,
     3        TLIM,RMA,RNU,DIR,XSTART,MNM,MNB,TB,BZC,CN7,DRF,FCF,FNA,
     4        HTF,INM,LLM,NAV,NMAX,NWEDG,PRT,SAMP,AKN1,AKN2,AKT1,AKT2,
     5        BTA,C1,C2,C3,C7,C8,DAM,DFA,PL,DELANG,FDN,HTI,HTR,JNT,KNM,
     6        NM,WTM,C4,VRM,NCOL,CTI,CTR,CNI,CNR,SN,
     7        ST,D1,D2,D3,D4,NRAN,VELR,EMP,RMN,AMP,IFLUX,FLUXIN,
     8        XLIM,COEFF,XCB,XS,YCB,TB,ALPHA,SIGMA,NTS,
     9        UTL,UTT,VTS,HTS,HTSI,ENT,REM,TMPA,
     A        DBA,NB,NBT,TMP,XV,XVA,YV,YVA,ZV,ZVA,T,DB,FNB,XC,YC,ZC,
     B        PAU,PAV,PAW,PAX,PAY,PAZ,ICOL,LM,
     C        ETA,PHI,CHI,CN,CM,CNG,CMG,CN8,TRP,TRPA,MSP,ANGLE,TF,
     D        UTLI,UTTI,VTSI,ER,RMB,LB,NBM,NBN,VEL,PFV,PCOL,JV
  365 ANM=INM                                                           RUN7260
      INM=9*ANM/10
      DDN=.9*DDN
      DRF=DRF/.9
      FCF=FCF/.9
      HTF=HTF/.9
      DO 370 MM=1,MSP
      FDN(MM)=FDN(MM)*INM/ANM
      LL(MM)=9*LL(MM)/10
      DO 366 KK=1,MSP
  366 CN8(KK,MM)=CN8(KK,MM)*.9
      DO 370 NK=1,2
      ENT(MM,NK)=ENT(MM,NK)*INM/ANM
  370 REM(MM,NK)=0.0
      IF(NEW) GO TO 220
      TST=TIME+TST
      TI=0.
      PRT=ITP
      WRITE(6,2)
      WRITE(6,4)
      WRITE(6,5) (MT,RMA(MT),RNU(MT),PCOL(MT),FLUXIN(MT),(ENT(MT,K),K=1,2
     1),MT=1,MSP)
```

```
      WRITE(6,2)
      IF((LARGE.EQ.2).OR.(LARGE.EQ.3)) GO TO 280
      REDO=.FALSE.
      GO TO 360
      END                                                               RUN7410

      SUBROUTINE DIAG(N,ITEST,NUM)                                      DIAG010
      REAL*8 PARAM(10)/'   NWEDGE','    NREG','      ND','     NPX',
     1 NNM','      NNB','      NBX','      NS','      NJ','     NSP'/
                                                                        DIAG040
C-----------------------------------------------------------------------DIAG050
                                                                        DIAG060
C                   FORMATS                                             DIAG070
                                                                        DIAG080
   32 FORMAT(9X,'ENT,REM,ENTS,REMS,PTH,THETA,DTH')                      DIAG090
   42 FORMAT(///5X,43H ARRAY DIMENSIONS ARE ABOUT TO BE VIOLATED./)     DIAG100
   44 FORMAT(5X,17H MAXIMUM VALUE IS,I5,19H, WHEREAS YOU INPUT,I5,3H  (,
     1A8,1H))                                                           DIAG120
   56 FORMAT(/5X,78H IF YOU DESIRE TO USE THIS VALUE, THE FOLLOWING AREADIAG130
     1YS MUST BE RE-DIMENSIONED./)                                      DIAG140
   62 FORMAT(9X,'HTS,HTSI,NTS,NTSP,UTL,UTT,VTS')                        DIAG150
   64 FORMAT(9X,'XLIM,COEFF'//11X,'NOTE THAT THE XLIM ARRAY MUST BE DIMEDIAG160
     1NSIONED TO 3 MORE THAN THE COEFF ARRAY.')                         DIAG170
   66 FORMAT(9X,'XCB,XS,YCB,TB,ALPHA,SIGMA')                            DIAG180
   68 FORMAT(9X,'DBA,NB,NBT,TMP,TMPA,XV,XVA,YV,YVA,ZV,ZVA,T,DB')        DIAG190
   70 FORMAT(9X,'PAU,PAV,PAW,PAX,PAY,PAZ,LCOL')
   72 FORMAT(8X,3H LM)                                                  DIAG210
   74 FORMAT(//5X,76H IF YOU CHANGE THE ARRAY DIMENSIONS, ALSO CHANGE THDIAG220
     1E 'LIMIT' DATA STATEMENT.)                                        DIAG230
   75 FORMAT(9X,'ALL ARRAYS ASSOCIATED WITH SPECIES')
   76 FORMAT(9X,'PNB,XC,YC,ZC')                                         DIAG240
   78 FORMAT(9X,'PV,NTCV,NTCP,MS,IWS,SL,DELS,TANGN')                    DIAG250
   80 FORMAT(9X,'VEL,PFV')                                              DIAG260
                                                                        DIAG270
C-----------------------------------------------------------------------DIAG280
                                                                        DIAG290
      WRITE(6,42)                                                       DIAG300
      WRITE(6,44)ITEST,NUM,PARAM(N)                                     DIAG310
      WRITE(6,56)                                                       DIAG320
      GO TO (1,2,3,4,5,6,7,8,9,10),N
    1 WRITE(6,62)                                                       DIAG340
      WRITE(6,32)                                                       DIAG350
      GO TO 11
    2 WRITE(6,64)                                                       DIAG370
      GO TO 11
    3 WRITE(6,66)                                                       DIAG390
      WRITE(6,62)                                                       DIAG400
      GO TO 11
    4 WRITE(6,68)                                                       DIAG420
      WRITE(6,76)
      GO TO 11
    5 WRITE(6,70)                                                       DIAG440
      GO TO 11
    6 WRITE(6,72)                                                       DIAG460
      GO TO 11
```

ORIGINAL PAGE IS
OF POOR QUALITY


```
    7 WRITE(6,80)                                                   DIAG480
      GO TO 11
    8 WRITE(6,78)                                                   DIAG500
      GO TO 11
    9 WRITE(6,80)                                                   DIAG520
      GO TO 11
   10 WRITE(6,75)
   11 WRITE(6,74)
      STOP                                                          DIAG540
      END                                                           DIAG550

      SUBROUTINE PRINTA(NWEDG,TITLE,NAME,XCB,YCB,TB,ALPHA,SIGMA,XLIM,
     1COEFF,LIMIT,MSP)
      INTEGER TST,TLIM,TIME                                         PRA0030
      LOGICAL SAVE,NEW                                              PRA0040
      DIMENSION LIMIT(1),TITLE(6),NAME(2),XCB(1),YCB(1),TB(1)
      DIMENSION ALPHA(3,1),SIGMA(3,1),XLIM(1),COEFF(4)
      DIMENSION RNU(3),RMA(3),CHI(3),DIR(3,3),PHI(3,3),ETA(3,3)
      DIMENSION WTM(3),DAM(3,3),VELS(3),XSP(3)
      COMMON /FIRST/NL,NW,NH
      COMMON /SECND/BW,BH,RMP,RMN,RMF
      COMMON /THIRD/PI,NREG,S,SINANG,COSANG,AKN,AKT,AKN1,AKN2,AKT1,AKT2
      COMMON /FORTH/NBX,RM,XR,DUMP,C9,LL(3),LLM
      COMMON /FIFTH/ND,TIME,DTM,TI,ITS,ITP,TST,TLIM,RMA,RNU,DIR     PRA0100
      COMMON /SIXTH/RMB,XSTART,INM,MNM,MNB,NEW,SAVE,PERCNT,NSR,TR
      COMMON/EIGTH/DENP,U,TP,ANGLE,TRF,CHI,PHI,ETA,WTM,DAM,VELR,XREF
      DATA NOT/'NOT '/                                              PRA0120
                                                                    PRA0130
-------------------------------------------------------------------PRA0140
                                                                    PRA0150
                    FORMATS                                         PRA0160
                                                                    PRA0170
    1 FORMAT(16X,40('-'),T74,'I'//9X,'3-D',I2,'-FLUID PROGRAM - ')
    2 FORMAT('+',31X,A4)                                            PRA0190
    3 FORMAT('+',35X,        'A RESTART OF A PREVIOUS RUN',T74,'I'/12X,2PRA0200
     1A4,' - ',6A4,'  - ',I2,' REGIONS',T74,'I',16(/T74,'I'))       PRA0210
    4 FORMAT(7X,'FRONT OF BODY =',E12.4,' XSTART  MAX HEIGHT =',E12.4,'
     1RMB',T74,'I'/7X,'BODY TEMPERATURES =',F12.2,' T FR.STRM.',F12.2,
     2' T ENTR.',F12.2,' T CAVITY'/7X,'X-LIMIT',T37,'BODY COEFFICIENTS',
     3T74,'I')
    6 FORMAT(5F14.6,3X,'I')                                         PRA0240
   10 FORMAT(1X,72('-'))                                            PRA0250
   12 FORMAT(//14X,'PARAMETERS OF SEGMENTS FOR BODY COLLISIONS',T96,'I'/
     18X,'X-COORD.   TEMP.    ALPHA1   ALPHA2   ALPHA3   SIGMA1   SIGMA2
     2   SIGMA3    AREAS',T96,'I')
   14 FORMAT(4X,E12.4,7F9.4,E12.4,T96,'I')
   17 FORMAT(///25X,'ARRAY STORAGE USED'/5X,I6,'      *',10I6,T96,'I')
   18 FORMAT(1H1/17X,'LENGTH OF CELL IN MEAN-FREE-PATHS = ',F12.4,'  BW'
     A,T76,'I'
     1/17X,'HEIGHT OF CELL IN MEAN-FREE-PATHS = ',F12.4,'  BH',T76,'I'
     2/16X,'NUMBER OF L1 CELLS ALONG FLOW AXIS =',I13,'  NW',T76,'I'
     3/17X,'NUMBER OF L1 CELLS IN RADIAL DIR. =',I13,'  NH',T76,'I'
     4/21X,'NUMBER OF LEVELS OF CELL SIZE =',I13,'  NL',T76,'I')
   23 FORMAT(3X,'NUMBER OF AZIMUTHAL WEDGES WITHIN   ',I3,' DEGREES =',IPRA0470
     113,'  NWEDG   I'/)
```

```
 24 FORMAT(16X,'BASIC TIME INTERVAL FOR COLLISIONS =',E13.4,'  DTM
   1 I'/8X,'TIME INTERVAL FOR SAMPLING FLOW FIELD INFO =',E13.4,'  DTS
   2     I'/24X,'TIME INTERVAL FOR PRINTING =',E13.4,'  DTP     I'/9X,
   3'TIME TO STEADY-STATE CONDITIONS (ASSUMED) =',E13.4,'  TST     I'/
   419X,'TIME AT WHICH RUN IS TERMINATED =',E13.4,'  TLIM    I'/)
 25 FORMAT(/5X,'PRESSURE RATIO(INSIDE/ENTRANCE) - EITHER TYPE =',F13.5
   1,'  RMP',T76,'I'/6X,'DENSITY RATIO(INSIDE/ENTRANCE) - EITHER TYPE
   2=',F13.5,'  RMN',T76,'I'/9X,'FLUX RATIO(INSIDE/ENTRANCE) - EITHER
   3TYPE =',F13.5,'  RMF',T76,'I')
 26 FORMAT(5X,'FREE STREAM NUMBER OF MOLECULES - EITHER TYPE =',I13,'
   A INM     I'/9X,'INITIAL NUMBER OF MOLECULES -    MAXIMUM    =',I13,
   1'  LLM      I'/9X,'MAXIMUM NUMBER OF MOLECULES - EITHER TYPE =',I13
   2,'  MNM     I'/1X,'MAX NUMBER OF MOLECULES IN ANY CELL - EITHER TY
   3PE =',I13,'  MNB     I')
 27 FORMAT( /22X,'VELOCITY OF FREE STREAM FLOW =',E13.4,' U',T76,'I'/1
   19X,'SPEED RATIO OF FREE STREAM FLOW =',E13.4,' S',T76,'I'/19X,'MAC
   AH NUMBER OF FREE STREAM FLOW =',E13.4,' M',T76,'I'/19X,'SPECIFIC H
   BEAT RATIO (CALCULATED)=',E13.4,' GAMMA',T76,'I'/         35X,'ANG
   2LE OF ATTACK =',F13.4,' ANGLE    I'/16X,'NUMBER DENSITY OF FREE ST
   3REAM FLOW =',E13.4,' N',T76,'I'/19X,'TEMPERATURE OF FREE STREAM FL
   4OW =',F13.4,' TF',T76,'I'/16X,'MOLE FRACTIONS OF FREE STREAM FLOW
   5=',3E13.4,' RNU I'/16X,'MOLECULAR WEIGHTS OF SPECIES ABOVE =',3F13
   6.4,' RMA I'/18X,'INITIAL NUMBERS OF SPECIES ABOVE =',3I13,' LL  I'
   7)
 28 FORMAT( /10X,'REFERENCE TEMPERATURE FOR MOLECULAR DATA =',F13.4,'
   1TRF',T90,'I'/14X,'CROSS-SECTION',26X,'TEMP EXPONENT',T90,'I'/3(3X,
   23E12.4,3X,3F12.6,T90,'I'/)/ 5X,'CHI/2-1',11X,'ROTATIONAL PARAMETER
   3 PHI',T90,'I'/3(F12.4,5X,3F12.6,T90,'I'/))
 29 FORMAT(/9X,'DATA SAVED ON TAPE 9')
 30 FORMAT(  31X,'REF MOLECULAR SPEED =',E13.4,' VELR',T76,'I'/20X,'SP
   1ECIES FREE STREAM MOLECULAR SPEEDS',T76,'I'/14X,3E16.6,T76,'I'/26X
   2,'REFERENCE MEAN FREE PATH =',E13.4,' XREF',T76,'I'/26X,'SPECIES M
   3EAN FREE PATHS',T76,'I'/14X,3E16.6,T76,'I'/11X,'LONGITUDINAL KNUDS
   4EN NUMBER (FREE STRM.)=',E13.4,' AKN',T76,'I'/13X,'TRANSVERSE KNUD
   5SEN NUMBER (FREE STRM.)=',E13.4,' AKT',T76,'I'/11X,'LONGITUDINAL K
   6NUDSEN NUMBER ( ENTRANCE )=',E13.4,' AKN1',T76,'I'/13X,'TRANSVERSE
   7 KNUDSEN NUMBER ( ENTRANCE )=',E13.4,' AKT1',T76,'I'/11X,'LONGITUD
   8INAL KNUDSEN NUMBER (  CAVITY  )=',E13.4,' AKN2',T76,'I'/13X,'TRAN
   9SVERSE KNUDSEN NUMBER (  CAVITY  )=',E13.4,' AKT2',T76,'I')
C
C-----------------------------------------------------------------------
C
    IARRAY=708+LIMIT(3)*(32+56*LIMIT(1))+20*LIMIT(2)+LIMIT(4)*(120+4*L
   1IMIT(6))+56*LIMIT(5)+LIMIT(8)*(68+96*LIMIT(9))+20*LIMIT(7)+224*LIM
   2IT(1)
    WRITE(6,1) MSP
    IF(NEW) WRITE(6,2) NOT
    WRITE(6,3) NAME,TITLE,NREG
    WRITE(6,4)XSTART,RMB
    DO 100 I=1,NREG
100 WRITE(6,6)XLIM(I+1),(COEFF(J),J=1,4)
    WRITE(6,10)
    WRITE(6,12)
    DO 110 I=1,ND
110 WRITE(6,14) XCB(I),TB(I),(ALPHA(J,I),J=1,3),(SIGMA(J,I),J=1,3),YCB
```

ORIGINAL PAGE IS
OF POOR QUALITY


```
     1(I)
      WRITE(6,10)
      WRITE(6,17)IARRAY,(LIMIT(I),I=1,10)
      WRITE(6,18)BW,BH,NW,NH,NL
      IETAZ=180.
      WRITE(6,23)IETAZ,NWEDG
      DTS=DTM*ITS
      DTP=DTM*ITP
      AST=DTM*TST
      ALIM=DTM*TLIM
      CHT=0.0
      DO 120 J=1,MSP
  120 CHT=CHT+CHI(J)*RNU(J)
      GAMMA=(7.+2.*CHT)/(5.+2.*CHT)
      AM=S*SQRT(2./GAMMA)
      WRITE(6,24)DTM,DTS,DTP,AST,ALIM
      WRITE(6,26) INM,LLM,MNM,MNB
      WRITE(6,25) RMP,RMN,RMF
      WRITE(6,27)U,S,AM,GAMMA,ANGLE,DENF,TF,(RNU(I),I=1,3),(RMA(I),I=1,3
     1),(LL(I),I=1,3)
      WRITE(6,28) TRF,((DIR(I,K),K=1,3),(ETA(I,K),K=1,3),I=1,3),(CHI(I),
     1(PHI(I,K),K=1,3),I=1,3)
      DO 210 I=1,3
      VELS(I)=0.0
  210 XSP(I)=0.0
      DO 220 J=1,MSP
      VELS(J)=VELR/SQRT(WTM(J))
      XT=0.0
      DO 215 M=1,MSP
  215 XT=XT+RNU(M)*DAM(J,M)*SQRT(1.+WTM(J)/WTM(M))
  220 XSP(J)=1.414214*XREF/XT
      WRITE(6,30) VELR,(VELS(I),I=1,3),XREF,(XSP(I),I=1,3),AKN,AKT,
     1AKN1,AKT1,AKN2,AKT2
      IF(SAVE) WRITE(6,29)
      RETURN
      END

      SUBROUTINE PRINTB(FNA,MSP,FNB,NM,XLIM,XC,YC,ZC,NB,N)
      INTEGER*2 NB
      DIMENSION FNB(1),NM(1),XLIM(1),XC(1)
      DIMENSION YC(1),ZC(1),NB(N,1)
      COMMON /FIRST/NL,NW,NH
      COMMON /THIRD/PI,NREG,S,SINANG,COSANG,AKN,AKT
      COMMON /FORTH/NBX
    1 FORMAT(1H1)
    2 FORMAT(2X,'-------------------------------CELL GEOMETRY------------
     1------------'/2X, 'BOX  LEVEL     POSITION OF CENTER     VOLUME
     2INITIAL    POPULATION'/2X,'NUM.',12X,'X',7X,'Y    THETA',12X,'  TOT
     3AL ',' EACH SPECIES ',' CELL#')
    3 FORMAT(1X,I4,I5,3X,2F8.3,F7.1,E12.3,2X,I4,2X,3I5,3X,I4)
    4 FORMAT(2X,'---------------TOTALS--------------',E12.4,8X,3I5)
      WRITE(6,1)
      WRITE(6,2)
      DO 200 I=1,NBX
      X=(XC(I)-XLIM(1))*AKN
```

```
      Y=YC(I)*AKT*2.0
      IX=X*NW+1
      IY=Y*NH+1
      M1=NB(1,I)
      M2=0
      IF(MSP.GE.2) M2=NB(2,I)
      M3=0
      IF(MSP.GE.3) M3=NB(3,I)
      MM=M1+M2+M3
  140 WRITE(6,3) IX,IY,X,Y,ZC(I),FNB(I),MM,M1,M2,M3,I                  PRB0370
  200 CONTINUE
      NM2=0
      IF(MSP.GE.2) NM2=NM(2)
      NM3=0
      IF(MSP.GE.3) NM3=NM(3)
      WRITE(6,4) FNA,NM(1),NM2,NM3                                     PRB0390
      RETURN                                                           PRB0400
      END

      SUBROUTINE CELL(A,B,KW,KH,XO,DELANG,NWEDG,XC,YC,ZC,FNB)
      DIMENSION XC(1),YC(1),ZC(1),FNB(1)                               CELL030
      COMMON /THIRD/PI                                        ---------CELL040
C-----------------------------------------------------------------------CELL050
C                                                                      CELL060
C     THE PURPOSE OF THIS SUBROUTINE IS TO                             CELL070
C        1. COMPUTE THE VOLUME OF EACH CELL (ALL 3 POSSIBLE LEVELS)    CELL080
C           AND STORE THE RESULT IN THE ARRAY CALLED 'FNB'.            CELL090
C        2. COMPUTE THE X, R, AND THETA COORDINATES OF THE CENTER OF   CELL100
C           EACH CELL (ALL 3 POSSIBLE LEVELS) AND STORE THE RESULTS INCELL110
C           ARRAYS CALLED 'XC', 'YC', AND 'ZC'.                        CELL120
C-----------------------------------------------------------------------CELL130
      I=0
      X=XO-0.5*A
      FACTOR=DELANG*PI*B*B*A/180.
      DO 110 K=1,KW
      X=X+A
      Y=-.5*B
      DO 110 L=1,KH
      Y=Y+B
      Z=-.5*DELANG
      DO 110 M=1,NWEDG
      Z=Z+DELANG
      I=I+1
      XC(I)=X
      YC(I)=Y
      ZC(I)=Z
  110 FNB(I)=FACTOR*(2*L-1)                                            CELL360
      RETURN                                                           CELL370
      END

      SUBROUTINE IMPACT(RM,G1,G2,G3,ET,EI,PHI,CHI,ETA,XM,CIM)
      COMMON/THIRD/PI
      IF(PHI.EQ.0.) GO TO 20
```

```
      IF(CHI.EQ.0.) GO TO 20
      DP=PHI*CHI-1.
      DS=PHI*(2.-.5*ETA)-1.
      E=ET+EI
   10 X=RAND(0)
      IF(X.EQ.0.0) GO TO 10
      XT=X**DP*(1.-X)**DS
      IF(XT.GT.XM) GO TO 15
      CIM=CIM+XT
      IF(CIM.LT.XM) GO TO 10
      CIM=CIM-XM
   15 ET=(1.-PHI)*ET+(1.-X)*PHI*E
      EI=(1.-PHI)*EI+X*PHI*E
   20 GP=SQRT(ET/RM)
      EP=2.*PI*RAND(0)
      CSX=2.*RAND(0)-1.
      SSX=SQRT(1.-CSX**2)
      G1=GP*CSX
      G2=GP*SSX*COS(EP)
      G3=GP*SSX*SIN(EP)
      RETURN
      END

      SUBROUTINE GAS(NWEDG,DELANG,ND,BTA,C1,DPA,NM,FNB,DB,NB,NBM,NBN,
     1PAU,PAV,PAW,PAX,PAY,PAZ,XLIM,COEFF,LM,I2,I3,XCB,TB,LARGE,
     2MNM,MNB,DEBUG1,LCOL,IP,ER,CHI,CNG,CMG,I,LB)
      INTEGER*2 LM(I,1),LCOL(I,1),LB(1),NBM(I,1),NBN(1)
      INTEGER*2 NB
      LOGICAL DUMP,DEBUG1                                               GAS0060
      DIMENSION BTA(1),C1(1),DPA(1),NM(1),FNB(1),CHI(1)
      DIMENSION DB(I,1),NB(I,1),PAU(I,1),PAV(I,1),PAW(I,1)
      DIMENSION PAX(I,1),PAY(I,1),PAZ(I,1),ER(I,1),COEFF(4),XLIM(1)
      DIMENSION CNG(1),CMG(1),XCB(1),TB(1)
      COMMON /FIRST/NL,NW,NR
      COMMON /SECND/BW,BH,RMP,RMN,RMF
      COMMON /THIRD/PI,NREG,S,SINANG,COSANG,AKN,AKT
      COMMON /FORTH/NBX,RM,XR,DUMP,C9,LL(3)
-----------------------------------------------------------------------GAS0150
                                                                        GAS0160
       THE PURPOSE OF THIS SUBROUTINE IS TO                             GAS0170
           1. COMPUTE THE INITIAL VELOCITY OF EACH MOLECULE.            GAS0180
              THE VELOCITY ARRAYS ARE 'PAU', 'PAV', AND 'PAW'.          GAS0190
           2. COMPUTE THE INITIAL POSITION OF EACH MOLECULE.            GAS0200
              THE POSITION ARRAYS ARE 'PAX', 'PAY', AND 'PAZ'.          GAS0210
           3. CREATE AN ARRAY WHICH STORES THE CELL POPULATIONS-        GAS0220
              'NB' FOR THE ACTUAL POPULATIONS                           GAS0230
           4. CREATE A CROSS-REFERENCING ARRAY (WHOSE CONSTRUCTION IS   GAS0250
              INDICATED BY A COMMENT CARD) CALLED 'LM'.                 GAS0260
           5. COMPUTE AN ARRAY WHICH STORES THE NUMBER DENSITY IN EACH  GAS0270
              CELL                                                      GAS0280
                                                                        GAS0290
-----------------------------------------------------------------------GAS0300
    2 FORMAT(/' SOMETHING IS WRONG WITH BOX NUMBERING IN GAS'/9I5,5E14.5
     1//)
    3 FORMAT(/' SOMETHING WRONG IN CELL VOLUMES IN  GAS'/5X,5I5,2E14.5)
```

```
    4 FORMAT(' NB(',I2,',',I4,') POPULATION EXCEEDED ',I3,' IN GAS')
      BP=(1.-RMN)
      CP=1.-RMN**2
      DO 180 MT=1,IP
      N=0
  110 N=N+1
      IF(N.GT.LL(MT)) GO TO 180
      IF(N.GT.NNM) GO TO 190
  120 P=.001+.998*RAND(0)
      IF(BP.NE.0.0) P=(1.-SQRT(1.-CP*P))/BP
      X=XLIM(1)+P*XR
      R=RM*SQRT(RAND(0))
      D=PI*RAND(0)
      PAX(MT,N)=X
      PAY(MT,N)=R*COS(D)
      PAZ(MT,N)=R*SIN(D)
      DO 126 J=2,ND
      IF(X.LE.XCB(J)) GO TO 128
  126 CONTINUE
      WRITE(6,3) J,ND,MT,LL(MT),N,X,XCB(J)
      IF(DUMP) CALL ABEND(14)
      STOP
  128 TL=TB(J-1)+(TB(J)-TB(J-1))*(X-XCB(J-1))/(XCB(J)-XCB(J-1))
  130 V=4.*RAND(0)
      VV=V*V
      C1(MT)=C1(MT)+VV*EXP(1.-VV)
      IF(C1(MT).LT.1.) GO TO 130
      C1(MT)=C1(MT)-1.
      A=1.-2.*RAND(0)
      B=SQRT(1.-A*A)
      C=2.*PI*RAND(0)
      V=V/BTA(MT)*SQRT(TL)
      PAU(MT,N)=V*A
      PAV(MT,N)=V*B*COS(C)
      PAW(MT,N)=V*B*SIN(C)
      EX=0.0
      IF(CHI(MT).LE.-1.) GO TO 136
  135 EX=9.*RAND(0)
      IF(EX.EQ.0.0) GO TO 135
      XT=EX**CHI(MT)*EXP(-EX)
      IF(XT.GE.CMG(MT)) GO TO 136
      CNG(MT)=CNG(MT)+XT
      IF(CNG(MT).LT.CMG(MT)) GO TO 135
      CNG(MT)=CNG(MT)-CMG(MT)
  136 ER(MT,N)=EX*TL
      TANG=180.*(1.-D/PI)
      IWDGE=TANG/DELANG+1
      IF(IWDGE.GT.NWEDG) IWDGE=NWEDG
      L=X/BW
      IF(L.GE.NW) L=NW-1
      M=R/BH
      IF(M.GE.NH) M=NH-1
      K=NWEDG*(L*NH+M)+IWDGE
      IF(K.LE.NBX) GO TO 165
      WRITE(6,2)L,M,IWDGE,NW,NH,NWEDG,K,MT,N,X,Y,Z,R,TANG
```

```
      IF(DUMP) CALL ABEND(11)                                           GAS0830
      STOP                                                              GAS0840
  165 J=NB(MT,K)+1                                                      GAS1190
      LCOL(MT,N)=0
      IF(J.LE.MNB) GO TO 166                                            GAS1240
      IF(DEBUG1) WRITE(6,4) MT,K,MNB                                    GAS1250
  166 NB(MT,K)=J                                                        GAS1270
      LB(N)=K
  167 IF(N.LT.LL(MT)) GO TO 110
      NM(MT)=N                                                          GAS1310
      NBM(MT,1)=0
      DO 170 N=1,NBX                                                    GAS1330
      A=NB(MT,N)                                                        GAS1360
      DB(MT,N)=A*DFA(MT)/FNB(N)                                         GAS1370
      NBM(MT,N+1)=NBM(MT,N)+NB(MT,N)
      NBN(N)=NBM(MT,N)
  170 CONTINUE                                                          GAS1380
      NG=NM(MT)
      DO 175 N=1,NG
      NQ=LB(N)
      NBN(NQ)=NBN(NQ)+1
      NA=NBN(NQ)
  175 LM(MT,NA)=N
  180 CONTINUE                                                          GAS1390
      RETURN                                                            GAS1400
  190 LARGE=1                                                           GAS1410
      RETURN                                                            GAS1420
      END                                                               GAS1430

      SUBROUTINE FLOW(NWEDG,MNM,LARGE,BTA,C1,C7,C8,ENT,REM,LCOL,IP,NM,
     1SN,ST,TBI,PAU,PAV,PAW,PAX,PAY,PAZ,ER,CHI,CNG,CMG,I,JV,FCOL,VEL,
     2PPV)
      INTEGER*2 LCOL
      DIMENSION BTA(1),NM(1),SN(1),ST(1)
      DIMENSION C1(1),C7(1),C8(1),CMG(1),FCOL(1),VELK(4)
      DIMENSION PAU(I,1),PAV(I,1),PAW(I,1),PAX(I,1),PAY(I,1),PAZ(I,1)
      DIMENSION ENT(3,1),REM(3,1),LCOL(I,1),ER(I,1),CHI(1),CNG(1)
      DIMENSION VEL(JV,4,1),PPV(JV,4,1)
      COMMON /THIRD/PI                                                  FLO0100
      COMMON /FORTH/NBX,RM,XR                                           FLO0110
C-----------------------------------------------------------------------FLO0120
C      THE PURPOSE OF THIS SUBROUTINE IS TO ADD A NEW BATCH OF MOLECULESFLO0130
C      TO THE SAMPLE THROUGH THE UPSTREAM BOUNDARY.                     FLO0140
C-----------------------------------------------------------------------FLO0150
      DO 370 MT=1,IP
      XGO=0.                                                            FLO0180
      E=1.                                                              FLO0190
      FRAC=FCOL(MT)
      ARG=SN(MT)
      STT=ST(MT)
      TV=1./BTA(MT)
      TR=1.
      DO 180 NT=1,2                                                     FLO0200
      VM=(SQRT(ARG**2+2.)+ARG)/2.
      SM=AMAX1(0.0,VM-4.)
```

```
      SMM=VM+4.-SM
      AM=ENT(MT,NT)+REM(MT,NT)
      M=AM
      REM(MT,NT)=AM-M
      IF(M.EQ.0) GO TO 170                                          FL00340
      DO 160 N=1,M                                                  FL00360
      IF(NM(MT).GE.MNM) GO TO 380
      NM(MT)=NM(MT)+1                                               FL00370
      NMX=NM(MT)                                                    FL00390
      R=RM*SQRT(RAND(0))
      D=PI*RAND(0)
      PAY(MT,NMX)=R*COS(D)                                          FL00420
      PAZ(MT,NMX)=R*SIN(D)                                          FL00430
      LCOL(MT,NMX)=0
      IF(FRAC.EQ.0.0) GO TO 130
      PF=RAND(0)
      IF(PF.GT.FRAC) GO TO 130
      KMX=3
      VELK(4)=0.0
      IF(CHI(MT).GT.-1) KMX=4
      DO 110 K=1,KMX
      P=RAND(0)
      DO 102 J=2,JV
      IF(PPV(J,K,MT).GT.P) GO TO 105
  102 CONTINUE
      VELK(K)=VEL(JV,K,MT)
      GO TO 110
  105 VELK(K)=VEL(J-1,K,MT)+(P-PPV(J-1,K,MT))*(VEL(J,K,MT)-VEL(J-1,K,
     1MT))/(PPV(J,K,MT)-PPV(J-1,K,MT))
  110 CONTINUE
      PAU(MT,NMX)=VELK(1)
      PAV(MT,NMX)=VELK(2)
      PAW(MT,NMX)=VELK(3)
      ER(MT,NMX)=VELK(4)
      GO TO 160
  130 V=SM+RAND(0)*SMM
      C1(MT)=C1(MT)+V*EXP(VM**2-V**2+2.*ARG*(V-VM))/VM
      IF(C1(MT).LT.1.) GO TO 130                                    FL00480
      C1(MT)=C1(MT)-1.                                              FL00490
      PAU(MT,NMX)=E*V*TV
  140 V=8.*RAND(0)-4.                                               FL00510
      C7(MT)=C7(MT)+EXP(-V*V)                                       FL00520
      IF(C7(MT).LT.1.) GO TO 140                                    FL00530
      C7(MT)=C7(MT)-1.                                              FL00540
      PAV(MT,NMX)=STT+V*TV
  150 V=8.*RAND(0)-4.                                               FL00560
      C8(MT)=C8(MT)+EXP(-V*V)                                       FL00570
      IF(C8(MT).LT.1.) GO TO 150                                    FL00580
      C8(MT)=C8(MT)-1.                                              FL00590
      PAW(MT,NMX)=V*TV
      X=0.0
      IF(CHI(MT).LE.-1.) GO TO 156
  155 X=9.*RAND(0)
      IF(X.LE.0.0) GO TO 155
      XT=X**CHI(MT)*EXP(-X)
```

```
      IF(XT.GE.CNG(MT)) GO TO 156
      CNG(MT)=CNG(MT)+XT
      IF(CNG(MT).LT.CMG(MT)) GO TO 155
      CNG(MT)=CNG(MT)-CMG(MT)
  156 ER(MT,NMX)=X*TR                                                   FL00610
  160 PAX(MT,NMX)=XGO                                                   FL00620
  170 CONTINUE
      ARG=0.0                                                           FL00640
      XGO=XR                                                            FL00650
      E=-1.
      FRAC=0.0
      STT=0.0
      TV=SQRT(TBI)/BTA(MT)
      TR=TBI
  180 CONTINUE                                                          FL00660
  370 CONTINUE                                                          FL01140
      RETURN                                                            FL01150
  380 LARGE=2                                                           FL01160
      RETURN                                                            FL01170
      END                                                               FL01180
      SUBROUTINE COLIDE(CN,CM,WTM,DB,DBA,NB,NCOL,LCOL,PAU,PAV,PAW,ER,T,
     1LM,MT,I2,I3,ETA,PHI,CHI,CN8,NP,NBM)
      INTEGER TIME                                                      COL0030
      INTEGER*2 LM(NP,1), LCOL(NP,1)
      INTEGER*2 NBM,NB
      DIMENSION CN(3,3,1),CM(3,3,1),WTM(1),DB(NP,1),DBA(NP,1),NB(NP,1)
      DIMENSION NBM(NP,1),NCOL(3,1),T(NP,NP,1),ETA(3,1),PHI(3,1),CHI(1)
      DIMENSION PAU(NP,1),PAV(NP,1),PAW(NP,1),ER(NP,1),CN8(3,1),WA(2)
      COMMON /FORTH/NBX                                                 COL0080
      COMMON /FIFTH/ND,TIME,DTM                                         COL0090
C-----------------------------------------------------------------------COL0100
C     THE PURPOSE OF THIS SUBROUTINE IS TO ADVANCE THE ELAPSED TIMES INCOL0110
C     CELLS BY AN AMOUNT APPROXIMATELY EQUAL TO THE PRE-SELECTED COLLISCOL0120
C     TIME. THERE ARE FOUR TIMES FOR EACH CELL, SAVED IN AN ARRAY CALLECOL0130
C     'T', CORRESPONDING TO THE FOUR TYPES OF MOLECULAR COLLISIONS WHICCOL0140
C     CAN OCCUR. TO ADVANCE THE VARIOUS TIMES, AN APPROPRIATE NUMBER OFCOL0150
C     THE CORRESPONDING MOLECULAR COLLISIONS IS COMPUTED. THE ACTUAL    COL0160
C     MOLECULES TO COLLIDE ARE SELECTED AT RANDOM, AND THEIR VELOCITY VCOL0170
C     DIRECTIONS AFTER COLLISION ARE SELECTED AT RANDOM.                COL0180
C-----------------------------------------------------------------------COL0190
      AIME=DTM*TIME
      DO 240 MTA=1,MT
      DO 230 MTB=1,MTA
      D = WTM(MTA) + WTM(MTB)
      WA(MTA)=WTM(MTA)/D
      WA(MTB)=WTM(MTB)/D
      RM=WTM(MTA)*WTM(MTB)/D
      CHT=CHI(MTA)+CHI(MTB)+2.0
      PHT=PHI(MTA,MTB)
      ETT=ETA(MTA,MTB)
      DO 220 N=1,NBX
      IF(T(MTA,MTB,N).LT.AIME) GO TO 100
      IF(T(MTB,MTA,N).GE.AIME) GO TO 220
  100 NA=NB(MTA,N)*NB(MTB,N)
```

```
      IF(MTA.EQ.MTB) NA=(NA-NB(MTA,N))/2.
      IF(NA.LT.1) GO TO 220
      KS=0
 120  KC=0
      CPUT=ELTIME(0)
      KS=KS+1
      IF(KS.GT.NA) GO TO 220
 130  KC=KC+1
      IF(KC.GT.NA) GO TO 220
 135  I=NB(MTA,N)*RAND(0)+1+NBM(MTA,N)
      IF(I.GT.NBM(MTA,N+1)) I=NBM(MTA,N+1)
      J=LM(MTA,I)
 140  K=NB(MTB,N)*RAND(0)+1+NBM(MTB,N)
      IF(K.GT.NBM(MTB,N+1)) K=NBM(MTB,N+1)
      IF(MTA.EQ.MTB.AND.I.EQ.K) GO TO 140
      L=LM(MTB,K)
      GM1=WA(MTA)*PAU(MTA,J)+WA(MTB)*PAU(MTB,L)
      GM2=WA(MTA)*PAV(MTA,J)+WA(MTB)*PAV(MTB,L)
      GM3=WA(MTA)*PAW(MTA,J)+WA(MTB)*PAW(MTB,L)
      G1=PAU(MTA,J)-PAU(MTB,L)
      G2=PAV(MTA,J)-PAV(MTB,L)
      G3=PAW(MTA,J)-PAW(MTB,L)
      GS=G1**2+G2**2+G3**2
      IF(GS.LT.1.0E-8) GO TO 130
      ET=RM*GS
      EI=ER(MTA,J)+ER(MTB,L)
      VR=GS**(.5-ETT/2.)
      IF(VR.GE.CM(MTA,MTB,1)) GO TO 160
      CN(MTA,MTB,1)=CN(MTA,MTB,1)+VR
      IF(CN(MTA,MTB,1).LT.CM(MTA,MTB,1)) GO TO 130
      CN(MTA,MTB,1)=CN(MTA,MTB,1)-CM(MTA,MTB,1)
 160  CONTINUE
      CPUT=ELTIME(0)
      CALL IMPACT(RM,G1,G2,G3,ET,EI,PHT,CHT,ETT,CM(MTA,MTB,2),CN(MTA,MTB
     1,2))
 165  CONTINUE
      CPUT=ELTIME(0)
      IF(PHT.EQ.0.) GO TO 175
      X1=0.0
      IF(CHI(MTA).EQ.-1.) GO TO 175
      X1=1.0
      IF(CHI(MTB).EQ.-1.) GO TO 175
 170  X1=RAND(0)
      IF((CHI(MTA).EQ.0.).AND.(CHI(MTB).EQ.0.)) GO TO 175
      XT=X1**CHI(MTA)*(1.-X1)**CHI(MTB)
      IF(XT.GT.CM(MTA,MTB,3)) GO TO 175
      CN(MTA,MTB,3)=CN(MTA,MTB,3)+XT
      IF(CN(MTA,MTB,3).LT.CM(MTA,MTB,3)) GO TO 170
      CN(MTA,MTB,3)=CN(MTA,MTB,3)-CM(MTA,MTB,3)
 175  CONTINUE
      C=DBA(MTA,N)
      D=DBA(MTB,N)
      IF(C.EQ.0.0) C=DB(MTA,N)
      IF(D.EQ.0.0) D=DB(MTB,N)
      IF(T(MTA,MTB,N).GE.AIME) GO TO 180
```

```
      PAU(MTA,J)=GM1+WA(MTB)*G1
      PAV(MTA,J)=GM2+WA(MTB)*G2
      PAW(MTA,J)=GM3+WA(MTB)*G3
      IF(PHT.GT.0.) ER(MTA,J)=X1*EI
      LCOL(MTA,J)=1+LCOL(MTA,J)
      NCOL(MTA,MTB)=NCOL(MTA,MTB)+1                                     COL0810
      T(MTA,MTB,N)=T(MTA,MTB,N)+CN8(MTA,MTB)/NB(MTA,N)/D/VR
      IF(MTA.EQ.MTB) GO TO 190
  180 IF(T(MTB,MTA,N).GE.AIME) GO TO 210
  190 PAU(MTB,L)=GM1-WA(MTA)*G1
      PAV(MTB,L)=GM2-WA(MTA)*G2
      PAW(MTB,L)=GM3-WA(MTA)*G3
      IF(PHT.GT.0.) ER(MTB,L)=(1.-X1)*EI
      LCOL(MTB,L)=1+LCOL(MTB,L)
      NCOL(MTB,MTA)=NCOL(MTB,MTA)+1                                     COL0920
      T(MTB,MTA,N)=T(MTB,MTA,N)+CN8(MTB,MTA)/NB(MTB,N)/C/VR
  210 CONTINUE
      IF(T(MTA,MTB,N).LT.AIME.OR.T(MTB,MTA,N).LT.AIME) GO TO 120
  220 CONTINUE
  230 CONTINUE
  240 CONTINUE
      RETURN
      END

      SUBROUTINE MOVE(KSWCH,AKN,NWEDG,ISTART,I2,I3,I4,I5,DELANG,
     1BTA,C2,C3,DPA,PL,HTI,HTR,JNT,KNM,NM,XCB,XLIM,CTI,CTR,CNI,
     2CNR,ALPHA,SIGMA,COEFF,HTS,HTSI,NTS,UTL,UTT,VTS,PAU,PAV,PAW,PAX,
     3PAY,PAZ,LCOL,TB,IP,ER,CHI,CNG,CMG,I,UTLI,UTTI,VTSI,IFLUX)
      INTEGER*2 LCOL(I,1)
      INTEGER TST,TIME
      LOGICAL DUMP                                                      MOV0070
      REAL LAM,MU,NU                                                    MOV0080
      DIMENSION BTA(1),C2(1),C3(1),PL(1),HTI(1)
      DIMENSION HTR(1),TB(1),XCB(1),ALPHA(3,1),SIGMA(3,1),COEFF(4)
      DIMENSION PAU(I,1),PAV(I,1),PAW(I,1),CTI(3,1),CTR(3,1)
      DIMENSION CNI(3,1),CNR(3,1),DPA(1),JNT(1),XLIM(1),KNM(1),NM(1)
      DIMENSION HTS(3,I2,I3),HTSI(3,I2,I3),NTS(3,I2,I3)
      DIMENSION UTLI(3,I2,I3),UTTI(3,I2,I3),VTSI(3,I2,I3)
      DIMENSION UTL(3,I2,I3),UTT(3,I2,I3),VTS(3,I2,I3)
      DIMENSION PAX(I,1),PAY(I,1),PAZ(I,1)
      DIMENSION ER(I,1),CNG(1),CMG(1),CHI(1),IFLUX(3,2)
      COMMON /THIRD/PI,NREG                                             MOV0170
      COMMON /FORTH/NBX,RM,XR,DUMP                                      MOV0180
      COMMON /FIFTH/ND,TIME,DTM,TI,ITS,ITP,TST
      COMMON /SVNTH/LAM,MU,NU,MT,N,J,XI,YI,ZI,TUSE                      MOV0200
      NAMELIST/CHECK/TIME,X,Y,Z,DX,DY,DZ,TLEFT,RADS,RMS,XR
C-----------------------------------------------------------------------
C      THE PURPOSE OF THIS SUBROUTINE IS TO ADVANCE THE SPATIAL POSITION
C      OF ALL THE MOLECULES BY AN AMOUNT APPROPRIATE TO THEIR CURRENT VE
C      LOCITIES AND THE PRE-SELECTED COLLISION TIME.
C-----------------------------------------------------------------------
      NAREA=NREG+1                                                      MOV0270
      RMS=RM**2
      DO 150 MT=1,IP
      N=KNM(MT)                                                         MOV0290
```

```
   10 N=N+1                                                             MOV0300
      TLEFT=DTS                                                         MOV0310
      IF(KSWCH.EQ.1) TLEFT=TLEFT*RAND(0)                                MOV0320
      IF(N.GT.NM(MT)) GO TO 150                                         MOV0330
   15 LAM=PAU(MT,N)                                                     MOV0340
      MU=PAV(MT,N)                                                      MOV0360
      NU=PAW(MT,N)                                                      MOV0370
      XI=PAX(MT,N)                                                      MOV0380
      YI=PAY(MT,N)                                                      MOV0390
      ZI=PAZ(MT,N)                                                      MOV0400
      DX=TLEFT*LAM                                                      MOV0410
      DY=TLEFT*MU                                                       MOV0420
      DZ=TLEFT*NU                                                       MOV0430
      X=XI+DX                                                           MOV0440
      Y=YI+DY                                                           MOV0450
      Z=ZI+DZ                                                           MOV0460
      RADS=Y**2+Z**2
      TUSE=TLEFT
      KS=0
      IF(RADS.GT.RMS) CALL INTERS(X,Y,Z,RMS,KS)
      IF(X.GT.XLIM(1)) GO TO 18
      IFLUX(MT,1)=IFLUX(MT,1)+1
      GO TO 100
   18 IF(X.LT.XLIM(NAREA)) GO TO 19
      IFLUX(MT,2)=IFLUX(MT,2)+1
      GO TO 100
   19 CONTINUE
      IF(KS.GT.0) CALL DRAG(AKN,NWEDG,XSTART,I2,I3,I4,I5,DELANG,JNT,BTA,
     1C2,C3,DFA,FL,HTI,HTR,TB,XCB,CTI,CTR,CNI,CNR,ALPHA,SIGMA,RM,HTS,
     2HTSI,NTS,UTL,UTT,VTS,LCOL,IP,ER,CHI,CNG,CMG,I,UTLI,UTTI,VTSI)
      IF(Z.GT.0.0) GO TO 20
      Z=-Z
      NU=-NU
   20 CONTINUE
      PAX(MT,N)=X
      PAY(MT,N)=Y
      PAZ(MT,N)=Z
      PAU(MT,N)=LAM
      PAV(MT,N)=MU
      PAW(MT,N)=NU
      IF((X.GT.0.0).AND.(X.LT.XR).AND.((Y**2+Z**2).LT.RMS)) GO TO 25
      WRITE(6,CHECK)
      GO TO 100
   25 CONTINUE
      TLEFT=TLEFT-TUSE
      IF(TLEFT.GT.0.0) GO TO 15
      GO TO 10
  100 NZ=NM(MT)                                                         MOV0820
      PAX(MT,N)=PAX(MT,NZ)                                              MOV0830
      PAY(MT,N)=PAY(MT,NZ)                                              MOV0840
      PAZ(MT,N)=PAZ(MT,NZ)                                              MOV0850
      PAU(MT,N)=PAU(MT,NZ)                                              MOV0860
      PAV(MT,N)=PAV(MT,NZ)                                              MOV0870
      PAW(MT,N)=PAW(MT,NZ)                                              MOV0880
      ER(MT,N)=ER(MT,NZ)
```

```
      LCOL(MT,N)=LCOL(MT,NZ)                                            MOV0900
      N=N-1                                                             MOV0910
      NM(MT)=NM(MT)-1                                                   MOV0920
      GO TO 10                                                          MOV0930
  150 CONTINUE                                                          MOV0940
      RETURN                                                            MOV0950
      END                                                               MOV0960
      SUBROUTINE ACCUM(I2,I3,PNB,NB,PAU,PAV,PAW,ER,TMP,TRP,XV,YV,ZV,LM,
     1IP,I,NBM)
      INTEGER*2 LM(I,1),NBM(I,1)
      INTEGER*2 NB
      DIMENSION PNB(1),NB(I,1),PAU(I,1),PAV(I,1),PAW(I,1),TMP(I,1)
      DIMENSION XV(I,1),YV(I,1),ZV(I,1),ER(I,1),TRP(I,1)
      COMMON /PORTH/NBX                                                 ACUM050
C-----------------------------------------------------------------------ACUM060
C      THE PURPOSE OF THIS SUBROUTINE IS TO ACCUMULATE TEMPERATURES,    ACUM070
C      VELOCITIES, AND DENSITIES IN VARIOUS ARRAYS FOR DETERMINING THE  ACUM080
C      AVERAGE FLOW FIELD PROPERTIES AFTER STEADY-STATE HAS BEEN REACHEDACUM090
C-----------------------------------------------------------------------ACUM100
      DO 180 N=1,NBX                                                    ACUM120
      DO 110 MT=1,IP
      XV(MT,N)=0.0                                                      ACUM160
      YV(MT,N)=0.0                                                      ACUM170
      ZV(MT,N)=0.0                                                      ACUM180
      TMP(MT,N)=0.                                                      ACUM190
      TRP(MT,N)=0.0
      TTX=0.                                                            ACUM200
      TTY=0.                                                            ACUM210
      TTZ=0.                                                            ACUM220
      TTR=0.0
      M=NB(MT,N)                                                        ACUM230
      IF(M.LT.1) GO TO 110                                              ACUM240
      U=0.                                                              ACUM250
      V=0.                                                              ACUM260
      W=0.                                                              ACUM270
      DO 100 L=1,M                                                      ACUM280
      NA=NBM(MT,N)+L
      J=LM(MT,NA)
      PU=PAU(MT,J)                                                      ACUM300
      PV=PAV(MT,J)                                                      ACUM310
      PW=PAW(MT,J)                                                      ACUM320
      U=U+PU
      V=V+PV
      W=W+PW
      TTR=TTR+ER(MT,J)
      TTX=TTX+PU*PU
      TTY=TTY+PV*PV
  100 TTZ=TTZ+PW*PW
      M=NB(MT,N)
      XV(MT,N)=U/M                                                      ACUM390
      YV(MT,N)=V/M                                                      ACUM400
      ZV(MT,N)=W/M                                                      ACUM410
      TMP(MT,N)=(TTX+TTY+TTZ)/M                                         ACUM420
      TRP(MT,N)=TTR/M
```

```
 110 CONTINUE                                                           ACUM430
 180 CONTINUE                                                           ACUM440
     RETURN                                                             ACUM450
     END                                                                ACUM460

     SUBROUTINE AVRGE(FNB,DB,DBA,NB,NBT,XV,YV,ZV,XVA,YVA,ZVA,TMP,TMPA,
    1TRP,TRPA,IP,I)
     INTEGER*2 NB,NBT
     DIMENSION FNB(1),DB(I,1),DBA(I,1),NB(I,1),NBT(I,1),TMP(I,1)
     DIMENSION TMPA(I,1),XV(I,1),XVA(I,1),YV(I,1),YVA(I,1),ZV(I,1)
     DIMENSION ZVA(I,1),TRP(I,1),TRPA(I,1)
     COMMON /FORTH/NBX                                                  AVG0050
-----------------------------------------------------------------------AVG0060
      THE PURPOSE OF THIS SUBROUTINE IS TO COMPUTE THE AVERAGE FLOW     AVG0070
      FIELD PROPERTIES.                                                 AVG0080
-----------------------------------------------------------------------AVG0090
     DO 110 N=1,NBX                                                     AVG0110
     DO 100 MT=1,IP
     A=NBT(MT,N)                                                        AVG0150
     B=NB(MT,N)
     C=A+B                                                              AVG0170
     NBT(MT,N)=C                                                        AVG0180
     IF(C.LT.1.) GO TO 100                                              AVG0190
     DBA(MT,N)=(DBA(MT,N)*A+DB(MT,N)*B)/C                               AVG0200
     XVA(MT,N)=(XVA(MT,N)*A+XV(MT,N)*B)/C                               AVG0210
     YVA(MT,N)=(YVA(MT,N)*A+YV(MT,N)*B)/C                               AVG0220
     ZVA(MT,N)=(ZVA(MT,N)*A+ZV(MT,N)*B)/C                               AVG0230
     TMPA(MT,N)=(TMPA(MT,N)*A+TMP(MT,N)*B)/C                            AVG0240
     TRPA(MT,N)=(TRPA(MT,N)*A+TRP(MT,N)*B)/C
 100 CONTINUE                                                           AVG0250
 110 CONTINUE                                                           AVG0260
     RETURN                                                             AVG0270
     END                                                                AVG0280

     SUBROUTINE DRAG(AKN,NWEDG,ISTART,I2,I3,I4,I5,DELANG,JNT,BTA,
    1C2,C3,DFA,FL,HTI,HTR,TB,XCB,CTI,CTR,CNI,CNR,ALPHA,SIGMA,RM,HTS,
    2HTSI,NTS,UTL,UTT,VTS,LCOL,IP,ER,CHI,CNG,CMG,I,UTLI,UTTI,VTSI)
     INTEGER*2    LCOL                                                  DRG0040
     INTEGER TIME,TST                                                   DRG0050
     REAL LAM,MU,NU,JAY,KAY                                             DRG0060
     DIMENSION BTA(1),C2(1),C3(1),FL(1),HTI(1),CTI(3,1),CTR(3,1)
     DIMENSION HTR(1),TB(1),XCB(1),ALPHA(3,1),SIGMA(3,1)
     DIMENSION HTS(3,I2,I3),HTSI(3,I2,I3),NTS(3,I2,I3)
     DIMENSION UTL(3,I2,I3),UTT(3,I2,I3),VTS(3,I2,I3),LCOL(I,1)
     DIMENSION UTLI(3,I2,I3),UTTI(3,I2,I3),VTSI(3,I2,I3)
     DIMENSION CNI(3,1),CNR(3,1),DFA(1),JNT(1)
     DIMENSION ER(I,1),CNG(1),CMG(1),CHI(1)
     COMMON /THIRD/PI                                                   DRG0140
     COMMON /FIFTH/ND,TIME,DTM,TI,ITS,ITP,TST                           DRG0150
     COMMON /SYNTH/LAM,MU,NU,MT,N,J,XCL,YCL,ZCL                         DRG0160
:----------------------------------------------------------------------    170
      THE PURPOSE OF THIS SUBROUTINE IS TO ACCUMULATE THE DRAG AND HEAT    180
:     TRANSFER INCREMENTS ON THE BODY CONTRIBUTED BY EACH MOLECULE WHICH   190
:     COLLIDES WITH THE BODY. IN ADDITION, EACH MOLECULE WHICH COLLIDES    200
:     WITH THE BODY IS ASSIGNED AN APPROPRIATE NEW VELOCITY (OF REFLECTION) 210
```

```
:      WHICH IS USED TO CONTINUE ITS SPATIAL TRANSLATION (IN SUBROUTINE DRAG)
:-----------------------------------------------------------------------  230
      CALL NORMAL(JAY,KAY,RM)
      JNT(MT)=JNT(MT)+1
      TANG=180.*ATAN2(ZCL,-YCL)/PI
      IWDG=TANG/DELANG+1
      IF(IWDG.GT.NWEDG) IWDG=NWEDG
      D=(LAM*LAM+MU*MU+NU*NU)
      G=ER(MT,N)
      H=G
      DO 100 M=2,ND
      IF(XCL.LT.XCB(M)) GO TO 110
  100 CONTINUE
  110 TBX=TB(M-1)+(TB(M)-TB(M-1))*(XCL-XCB(M-1))/(XCB(M)-XCB(M-1))
      WI=(NU*JAY-MU*KAY)
      VID=MU*JAY+NU*KAY
      UID=LAM
      E=RAND(0)
      IF(E.LT.SIGMA(MT,M)) GO TO 115
      VRD=-VID
      URD=UID
      WR=WI
      GO TO 125
  115 V=4.*RAND(0)
      C2(MT)=C2(MT)+.544331*V*V*V*EXP(1.5-V*V)
      IF(C2(MT).LT.1.) GO TO 115
      C2(MT)=C2(MT)-1.
      IF( NTS(MT,M,IWDG).NE.0) GO TO 117
      ATR=ALPHA(MT,M)*TBX  /SIGMA(MT,M)
      GO TO 118
  117 ATR=ALPHA(MT,M)*TBX  /SIGMA(MT,M)+(1.-ALPHA(MT,M)/SIGMA(MT,M))*HTS
     1I(MT,M,IWDG)/ NTS(MT,M,IWDG)/(3.+CHI(MT))
  118 ABR=SQRT(ATR)
      V=V*ABR/BTA(MT)
      AA=RAND(0)
      A=SQRT(AA)
      B=SQRT(1.-AA)
      C=2.*PI*RAND(0)
      VRD=V*A
      URD=V*B*COS(C)
      WR=V*B*SIN(C)
      IF(CHI(MT).EQ.-1.) GO TO 125
  122 X=9.*RAND(0)
      IF(X.EQ.0.0) GO TO 122
      XTEMP=1.0
      IF(CHI(MT).NE.0.0) XTEMP=X**CHI(MT)
      CNG(MT)=CNG(MT)+XTEMP*EXP(-X)
      IF(CNG(MT).LT.CMG(MT)) GO TO 122
  124 CONTINUE
      CNG(MT)=CNG(MT)-CMG(MT)
      IF(CNG(MT).GE.CMG(MT)) GO TO 124
      ER(MT,N)=X*ATR
      H=ER(MT,N)
  125 UR=URD
      LAM=UR
```

```
      MU=JAY*VRD-KAY*WR
      NU=KAY*VRD+JAY*WR
      IF(TIME.LE.TST) RETURN
      XMZ=(XCL-XSTART)*AKN                                              DRG0730
      YMZ= RM*AKN                                                       DRG0740
      B=(URD*URD+VRD*VRD+WR*WR)                                         DRG0750
      UTI=UID*UID+WI*WI                                                 DRG0760
      UYI=-WI*KAY
      UYR=-WR*KAY
      FL(MT)=FL(MT)+DFA(MT)                                             DRG0790
      HTI(MT)=HTI(MT)+D+G
      HTR(MT)=HTR(MT)-B-H
      CTI(MT,1)=CTI(MT,1)+UID                                           DRG0820
      CTI(MT,2)=CTI(MT,2)+UYI                                           DRG0830
      CTI(MT,3)=CTI(MT,3)+(XMZ*UYI-YMZ*UID)                             DRG0840
      CNI(MT,2)=CNI(MT,2)+VID*JAY                                       DRG0860
      CNI(MT,3)=CNI(MT,3)+XMZ*JAY*VID
      CTR(MT,1)=CTR(MT,1)-URD                                           DRG0880
      CTR(MT,2)=CTR(MT,2)-UYR                                           DRG0890
      CTR(MT,3)=CTR(MT,3)-(XMZ*UYR-YMZ*URD)                             DRG0900
      CNR(MT,2)=CNR(MT,2)-VRD*JAY                                       DRG0920
      CNR(MT,3)=CNR(MT,3)-XMZ*JAY*VRD
      NTS(MT,M,IWDG)=NTS(MT,M,IWDG)+1                                   DRG0940
      UTLI(MT,M,IWDG)=UTLI(MT,M,IWDG)+UID
      UTL(MT,M,IWDG)=UTL(MT,M,IWDG)+(UID-URD)                           DRG0960
      UTTI(MT,M,IWDG)=UTTI(MT,M,IWDG)+ WI
      UTT(MT,M,IWDG)=UTT(MT,M,IWDG)+(WI-WR)                             DRG0970
      VTSI(MT,M,IWDG)=VTSI(MT,M,IWDG)-VID
      VTS(MT,M,IWDG)=VTS(MT,M,IWDG)+(VRD-VID)                           DRG0980
      HTSI(MT,M,IWDG)=HTSI(MT,M,IWDG)+D+G
      HTS(MT,M,IWDG)=HTS(MT,M,IWDG)+D-B+G-H
      RETURN                                                            DRG1170
      END                                                               DRG1180

      SUBROUTINE INTERS(X,Y,Z,RMS,KS)
      REAL LAM,MU,NU
      COMMON /SYNTH/LAM,MU,NU,MT,N,J,XI,YI,ZI,TUSE
    1 FORMAT(//' SOMETHING IS WRONG IN INTERS'/6E15.6/4E15.6)
    2 FORMAT(//' TROUBLE IN INTERS - TYME =',E15.6/6E15.6/4E15.6)
      A=MU**2+NU**2
      B=(YI*MU+ZI*NU)/A
      C=(RMS-YI**2-ZI**2)/A
      IF(C.LT.0.0) C=0.0
      DISCR=B**2+C
      IF(DISCR.GE.0.0) GO TO 10
      WRITE(6,1) XI,YI,ZI,MU,NU,RMS,A,B,C,DISCR
      STOP
   10 TYME=SQRT(DISCR)-B
      IF(C.EQ.0.0) TYME=0.0
      IF((TYME.LE.TUSE).AND.(TYME.GE.0.0)) GO TO 11
      WRITE(6,2) TYME,XI,YI,ZI,MU,NU,RMS,A,B,C,DISCR
      IF(TYME.GT.TUSE)TYME=TUSE
      IF(TYME.LT.0.0) TYME=0.0
   11 KS=1
      TUSE=TYME
```

```
      XI=XI+LAM*TYME
      YI=YI+MU*TYME
      ZI=ZI+NU*TYME
      IF(ZI.GT.0.0) GO TO 20
      ZI=-ZI
      NU=-NU
   20 CONTINUE
      YI=.9999*YI
      ZI=.9999*ZI
      X=XI
      Y=YI
      Z=ZI
      RETURN
      END

      SUBROUTINE NORMAL(JAY,KAY,RM)
      REAL LAM,MU,NU,JAY,KAY                                            NORM020
      COMMON /SVNTH/LAM,MU,NU,MT,N,J,XCL,YCL,ZCL                        NORM040
      JAY=-YCL/RM
      KAY=-ZCL/RM
      RETURN                                                            NORM130
      END                                                               NORM140

      SUBROUTINE PRINT1(DT,COSANG,SINANG,RMA,RNU,DRF,FCF,HTF,FL,HTI,HTR,
     1CTI,CTR,CNI,CNR)
      DIMENSION DD(3),WD(2,5),PP(4,4),QQ(4,4),RR(4,4),SS(4,4),TT(4,4)
      DIMENSION UU(4,4),P1(4,4),Q1(4,4),R1(4,4),PA(4),PB(4),PC(4)
      DIMENSION FL(1),HTI(1),HTR(1),CTI(3,1),CTR(3,1),CNI(3,1),CNR(3,1)
      DIMENSION RMA(1),RNU(1)
      DATA WD/'X-FO','RCE ','Y-FO','RCE ','Z-MO','MENT','DRAG','    ','LPT10070
     1IFT','    '/                                                      PT10080
C-----------------------------------------------------------------------PT10090
C      THE PURPOSE OF THIS SUBROUTINE IS TO PRINT OUT THE GROSS SURFACE PT10100
C      COEFFICIENTS OF THE BODY.                                        PT10110
C-----------------------------------------------------------------------PT10120
C                                                                       PT10130
C                                                                       PT10140
C                                                                       PT10150
C                 FORMATS                                               PT10160
C                                                                       PT10170
C                                                                       PT10180
    1 FORMAT(//1X,50('*'),' GROSS SURFACE COEFFICIENTS ',50('*')/' MOLEC
     1ULAR WEIGHT',12X,F8.3,3(19X,F8.3)/25X,
     2  'INC.     REF.     TOT.         INC.     REF.     TOT.         INC.
     3REF.     TOT.         INC.     REF.     TOT.')
   10 FORMAT(' NUMBER FLUX          ', 4(F9.3,18X))
   12 FORMAT(1X,2A4,2X,'SHEAR       ',4(3F8.3,3X))
   14 FORMAT(11X,'PRESSURE   ',4(3F8.3,3X))
   16 FORMAT(11X,'TOTAL      ',4(3F8.3,3X)/)
   18 FORMAT(' HEAT TRANSFER',7X,4(3F8.3,3X)/)
C                                                                       PT10280
C***********************************************************************PT10290
C                                                                       PT10300
C                                                                       PT10310
      RMR=0.0
```

```
      DO 50 MT=1,3
      DD(MT)=RMA(MT)*RNU(MT)*DRF/DT
   50 RMR=RMR+RMA(MT)*RNU(MT)
      WRITE(6,1)  (RMA(MT),MT=1,3),RMR
      PF=FL(1)*FCF/DT
      QF=FL(2)*FCF/DT
      RF=FL(3)*FCF/DT
      SF=PF+QF+RF
      WRITE(6,10)  PF,QF,RF,SF
      DO 200 I=1,3
      PP(4,I)=0.0
      QQ(4,I)=0.0
      RR(4,I)=0.0
      SS(4,I)=0.0
      TT(4,I)=0.0
      UU(4,I)=0.0
      P1(4,I)=0.0
      Q1(4,I)=0.0
      R1(4,I)=0.0
      DO 150 MT=1,3
      PP(MT,I)=CTI(MT,I)*DD(MT)/RMR
      QQ(MT,I)=CTR(MT,I)*DD(MT)/RMR
      SS(MT,I)=CNI(MT,I)*DD(MT)/RMR
      TT(MT,I)=CNR(MT,I)*DD(MT)/RMR
      P1(MT,I)=PP(MT,I)+SS(MT,I)
      Q1(MT,I)=QQ(MT,I)+TT(MT,I)
      RR(MT,I)=PP(MT,I)+QQ(MT,I)
      UU(MT,I)=SS(MT,I)+TT(MT,I)
      R1(MT,I)=P1(MT,I)+Q1(MT,I)
      PP(4,I)=PP(4,I)+PP(MT,I)
      QQ(4,I)=QQ(4,I)+QQ(MT,I)
      RR(4,I)=RR(4,I)+RR(MT,I)
      SS(4,I)=SS(4,I)+SS(MT,I)
      TT(4,I)=TT(4,I)+TT(MT,I)
      UU(4,I)=UU(4,I)+UU(MT,I)
      P1(4,I)=P1(4,I)+P1(MT,I)
      Q1(4,I)=Q1(4,I)+Q1(MT,I)
      R1(4,I)=R1(4,I)+R1(MT,I)
  150 CONTINUE
      WRITE(6,12) (WD(J,I),J=1,2),(PP(K,I),QQ(K,I),RR(K,I),K=1,4)
      WRITE(6,14) (SS(K,I),TT(K,I),UU(K,I),K=1,4)
      WRITE(6,16) (P1(K,I),Q1(K,I),R1(K,I),K=1,4)
  200 CONTINUE
      AA=COSANG
      BB=SINANG
      DO 300 I=4,5
      DO 250 K=1,4
      PP(K,4)=AA*PP(K,1)+BB*PP(K,2)
      QQ(K,4)=AA*QQ(K,1)+BB*QQ(K,2)
      RR(K,4)=AA*RR(K,1)+BB*RR(K,2)
      SS(K,4)=AA*SS(K,1)+BB*SS(K,2)
      TT(K,4)=AA*TT(K,1)+BB*TT(K,2)
      UU(K,4)=AA*UU(K,1)+BB*UU(K,2)
      P1(K,4)=AA*P1(K,1)+BB*P1(K,2)
      Q1(K,4)=AA*Q1(K,1)+BB*Q1(K,2)
```

```
250 R1(K,4)=AA*R1(K,1)+BB*R1(K,2)                                          PT10760
    WRITE(6,12) (WD(J,I),J=1,2),(PP(K,4),QQ(K,4),RR(K,4),K=1,4)
    WRITE(6,14) (SS(K,4),TT(K,4),UU(K,4),K=1,4)
    WRITE(6,16) (P1(K,4),Q1(K,4),R1(K,4),K=1,4)
    AA=-SINANG                                                             PT10800
    BB=COSANG                                                              PT10810
300 CONTINUE                                                               PT10820
    HD=HTF/DT                                                              PT10830
    PA(4)=0.0
    PB(4)=0.0
    PC(4)=0.0
    DO 400 MT=1,3
    PA(MT)=HTI(MT)*RMA(MT)*RNU(MT)*HD/RMR
    PB(MT)=HTR(MT)*RMA(MT)*RNU(MT)*HD/RMR
    PC(MT)=PA(MT)+PB(MT)
    PA(4)=PA(4)+PA(MT)
    PB(4)=PB(4)+PB(MT)
    PC(4)=PC(4)+PC(MT)
400 CONTINUE
    WRITE(6,18)  (PA(I),PB(I),PC(I),I=1,4)
    RETURN                                                                 PT10950
    END                                                                    PT10960

    SUBROUTINE PRINT2(AKN,XSTART,DT,RNU,RMA,DRF,FCF,HTF,UTLI,UTTI,VTSI
   1,HTSI,DELANG,NWEDG,XS,XCB,YCB,HTS,NTS,UTL,UTT,VTS,I2,I3,IP)
    DIMENSION RMA(1),RNU(1),XS(1),XCB(1),YCB(1)
    DIMENSION HTS(3,I2,I3),NTS(3,I2,I3),UTL(3,I2,I3)
    DIMENSION UTT(3,I2,I3),VTS(3,I2,I3),UTLI(3,I2,I3),UTTI(3,I2,I3)
    DIMENSION VTSI(3,I2,I3),HTSI(3,I2,I3)
    COMMON /FIFTH/ND                                                       PT20060
-----------------------------------------------------------------------PT20070
    THE PURPOSE OF THIS SUBROUTINE IS TO PRINT OUT THE DISTRIBUTION     PT20080
    ON SURFACE OF THE SURFACE COEFFICIENTS                              PT20090
-----------------------------------------------------------------------PT20100
                                                                       PT20110
                                                                       PT20120
                                                                       PT20130
                 FORMATS                                               PT20140
                                                                       PT20150
                                                                       PT20160
  8 FORMAT(//1X,45('*'),' DISTRIBUTION ON SURFACE ',45('*')/71X,'INC.
   1   TOT.      INC.      TOT.      INC.      TOT.'/11X,'SEGMENT GEOMETRY',
   214X,'MOL.      MOLE  SAMP     NUM.     SKIN      SKIN     PRES-     PRES-
   3 HEAT      HEAT'/' NO.   CENTER     DELX     CENTER   DELANG',4X,'WGHT.   F
   4RACT.',10X,
   5          'FLUX    FRCTN    FRCTN    SURF     SURF     TRNSF    TRNSF')
 10 FORMAT(1X,I3,F8.3,F7.3,F9.3,F8.3,1X,2F8.4,I6,7F8.4)
 11 FORMAT(37X,F8.4,'   1.0000',I6,7F8.4)
                                                                       PT20230
***********************************************************************PT20240
                                                                       PT20250
                                                                       PT20260
    RMR=0.0
    DO 50 MT=1,IP
 50 RMR=RMR+RMA(MT)*RNU(MT)
```

```
      WRITE(6,8)
      I=0                                                               PT20280
      DO 110 N=2,ND                                                     PT20320
      DTY=DT*YCB(N)/180.                                                PT20330
      P=XS(N)                                                           PT20340
      Q=2.*((XCB(N)-XSTART)*AKN-XS(N))                                  PT20350
      J=0                                                               PT20380
      R=-.5*DELANG
      PMLT=FCP/(DTY*DELANG)
      QMLT=DRP/(DTY*DELANG)
      SMLT=RTP/(DTY*DELANG)
      DO 100 K=1,NWEDG
      R=R+DELANG
      I=I+1                                                             PT20490
      J=J+1                                                             PT20500
      M3=0
      P3=0.0
      Q3=0.0
      Q4=0.0
      R3=0.0
      R4=0.0
      S3=0.0
      S4=0.0
      DO 90 MT=1,IP
      M1=NTS(MT,N,J)
      M3=M3+M1
      P1=NTS(MT,N,J)*PMLT*RNU(MT)
      P3=P3+P1
      Q1=SQRT(UTLI(MT,N,J)**2+UTTI(MT,N,J)**2)*RNU(MT)*RMA(MT)*QMLT/RMR
      Q2=SQRT( UTL(MT,N,J)**2+ UTT(MT,N,J)**2)*RNU(MT)*RMA(MT)*QMLT/RMR
      Q3=Q3+Q1
      Q4=Q4+Q2
      R1=VTSI(MT,N,J)*RNU(MT)*RMA(MT)*QMLT/RMR
      R2= VTS(MT,N,J)*RNU(MT)*RMA(MT)*QMLT/RMR
      S1=HTSI(MT,N,J)*RNU(MT)*RMA(MT)*SMLT/RMR
      S2= HTS(MT,N,J)*RNU(MT)*RMA(MT)*SMLT/RMR
      R3=R3+R1
      R4=R4+R2
      S3=S3+S1
      S4=S4+S2
   90 WRITE(6,10)I,P,Q,R,DELANG,RMA(MT),RNU(MT),M1,P1,Q1,Q2,R1,R2,S1,S2
      WRITE(6,11)RMR,M3,P3,Q3,Q4,R3,R4,S3,S4
  100 CONTINUE                                                          PT20660
  110 CONTINUE                                                          PT20670
      RETURN                                                            PT20680
      END                                                               PT20690

      SUBROUTINE PRINT4(MSP,CHI,RNU,I,TRP,FDN,WTM,DB,NS,TMP,XV,
     1YV,ZV,KS,NB,XC,YC,ZC)
      INTEGER*2 NB,NS
      DIMENSION FDN(1),RNU(1),CHI(1),WTM(1),TMP(I,1),TRP(I,1)
      DIMENSION DB(I,1),NB(I,1),XV(I,1),YV(I,1),ZV(I,1),DBT(3),NS(I,1)
      DIMENSION XC(1),YC(1),ZC(1)
      COMMON /SECND/BW,BH,RMP,RMN,RMF
      COMMON /FORTH/NBX,RM,XR
```

```
C-----------------------------------------------------------------------PT50060
C     THE PURPOSE OF THIS SUBROUTINE IS TO PRINT OUT THE INSTANTANEOUS PT50070
C     FLOW-FIELD PROPERTIES.                                           PT50080
C-----------------------------------------------------------------------PT50090
                                                                        PT50100
                                                                        PT50110
                                                                        PT50120
C                    FORMATS                                            PT50130
                                                                        PT50140
                                                                        PT50150
    1 FORMAT(//1X,45('*'),' INSTANTANEOUS FLOW FIELD INFORMATION ',45('*PT50160
     1'))                                                               PT50170
    2 FORMAT(/2X,'HORIZONTAL NUMBER=',I3,3X,'VERTICAL NUMBER=',I3,3X,'HO
     1RIZONTAL POSITION=',F8.5,3X,' RADIAL POSITION =',F8.5/2X,'BOX*
     2ANGLE   SAMP DENSITY MACH NO   X VEL.   Y VEL.   Z VEL.   T(KIN)   T(RO
     3T)   TEMP.',14X,'MOLE FRACTIONS')
    3 FORMAT(//1X,46('*'),' ACCUMULATED FLOW FIELD INFORMATION ',46('*')
     1)
    4 FORMAT(1X,I4,E11.3,I6,8F8.3,3X,3E11.3)
                                                                        PT50220
C***********************************************************************PT50230
                                                                        PT50240
                                                                        PT50250
      IF(KS.EQ.0) WRITE(6,1)
      IF(KS.NE.0) WRITE(6,3)
      DO 40 MT=1,3
   40 DBT(MT)=0.0
      FDA=0.
      CHT=0.
      DO 50 MT=1,MSP
      CHT=CHT+CHI(MT)*RNU(MT)
   50 FDA=FDA+FDN(MT)*WTM(MT)
      YCT=0.0
      XCT=0.0
      DO 110 N=1,NBX                                                    PT50290
      IF((XC(N).EQ.XCT).AND.(YC(N).EQ.YCT)) GO TO 52
      XCT=XC(N)
      YCT=YC(N)
      XXT=XCT/XR
      YYT=YCT/RM
      IXC=XCT/BW + 1
      IYC=YCT/BH + 1
      WRITE(6,2) IXC,IYC,XXT,YYT
   52 ZCT=ZC(N)
      NSAMP=0
      DBA=0.
      XVM=0.
      YVM=0.
      ZVM=0.
      TMPM=0.
      TRPM=0.
      E=0.
      F=0.
      DO 100 MT=1,MSP
      NSAMP=NSAMP+NS(MT,N)
```

ORIGINAL PAGE IS
OF POOR QUALITY


```
      XVM=XVM+XV(MT,N)*RNU(MT)*WTM(MT)*NB(MT,N)
      YVM=YVM+YV(MT,N)*RNU(MT)*WTM(MT)*NB(MT,N)
      ZVM=ZVM+ZV(MT,N)*RNU(MT)*WTM(MT)*NB(MT,N)
      DBA=DBA+DB(MT,N)*WTM(MT)
      TMPM=TMPM+WTM(MT)*RNU(MT)*TMP(MT,N)*NB(MT,N)
      TRPM=TRPM+RNU(MT)*NB(MT,N)*TRP(MT,N)
      E=E+WTM(MT)*RNU(MT)*NB(MT,N)
  100 F=F+RNU(MT)*NB(MT,N)
      DBA=DBA/FDA
      IF(E.EQ.0.0) GO TO 55
      XVM=XVM/E
      YVM=YVM/E
      ZVM=ZVM/E
      VS=XVM**2+YVM**2+ZVM**2
      TMPM=TMPM/E-VS
      TRPM=TRPM/F
   55 CONTINUE
      TTM=(TMPM+TRPM)/(2.5+CHT)
      TMPM=TMPM/1.5
      IF(CHT.NE.-1.) TRPM=TRPM/(1.+CHT)
      AMS=SQRT(VS)
      IF(TTM.GT.0.) AMS=SQRT((5.+2.*CHT)*VS/(TTM*(3.5+CHT)))
      CCZ=COS(ZCT/57.29578)
      SCZ=SQRT(1.-CCZ**2)
      RVM=ZVM*SCZ-YVM*CCZ
      TVM=YVM*SCZ+ZVM*CCZ
      DO 60 MT=1,MSP
      DBT(MT)=RNU(MT)*NB(MT,N)
      IF(F.NE.0.) DBT(MT)=DBT(MT)/F
   60 CONTINUE
      WRITE(6,4) N,ZCT,NSAMP,DBA,AMS,XVM,RVM,TVM,TMPM,TRPM,TTM,(DBT(J),
     1J=1,3)
  110 CONTINUE                                                       PT50470
      RETURN                                                         PT50480
      END                                                            PT50490
ENTRY
 CONTRL NAME='INTE','RNAL',TITLE=' PAR','ABOL','A AT',' 95K','M  M','ON. ',
  DEBUG=.F.,.F.,.T.,NEW=.F.,SAVE=.T.,ICOPY=0,REDO=.T. &END
 TIMES DTM=.005,ITS=6,ITP=6,TST=2,TLIM=12 &END
 FLOREF LLM=2000,MNM=5000,MNB=150,MSP=1,MET=0,U=7485.9,ANGLE=28.,RNU=1.,2*0.,
  RMA=28.94,0.,0.,TF=195.51,DENF=2.52E+19 &END
 MOLEC TRP=1000,DIR=3.5E-19,ETA=.104,PHI=0.0,CHI=-1.,ACR=.001 &END
 SHAPES BODY=0.0,1000.,.00235 &END
 SHAPES BODY=.0025,555.,0.0,2*1.0 &END
 SHAPES BODY=.0050,345.,0.0,2*1.0 &END
 SHAPES BODY=.0100,300.,0.0,2*1.0 &END
 SHAPES BODY=.0200,300.,0.0,2*1.0 &END
 SHAPES BODY=.0300,300.,0.0,2*1.0 &END
 SHAPES BODY=.0400,300.,0.0,2*1.0 &END
 SHAPES BODY=.0500,300.,0.0,2*1.0 &END
 SHAPES BODY=.0600,300.,0.0,2*1.0 &END
 SHAPES BODY=.0700,300.,0.0,2*1.0 &END
 SHAPES BODY=.0800,300.,0.0,2*1.0 &END
 SHAPES BODY=.0870,300.,1.0,2*1.0 &END
 GEOM NWEDG=2,NW=20,NH=3, &END
```

```
;INCUPL FLUXIN=2.1429,FCOL=1.0,RMP=0.3,JV=22,KMX=3 &END
;INOUT VARG=0.,1.,2.,3.,4.,5.,6.,7.,8.,9.,10.,11.,12.,13.,14.,15.,16.,17.,18.,
 19.,20.,21.,CURV=0.0,.070,.170,.282,.369,.459,.537,.599,.656,.710,.750,.785,
 .815,.845,.872,.900,.922,.951,.975,.988,.996,1.00, &END
;INOUT VARG=-20.,-19.,-17.,-15.,-13.,-11.,-9.,-7.,-5.,-3.,-1.,1.,3.,5.,7.,9.,
   11.,13.,15.,17.,19.,CURV=4*0.,.003,.013,.036,.084,.149,.250,.406,.611,.762,
   .871,.932,.962,.984,.995,.999,3*1.0, &END
;INOUT VARG=-20.,-19.,-17.,-15.,-13.,-11.,-9.,-7.,-5.,-3.,-1.,1.,3.,5.,7.,9.,
 11.,13.,15.,17.,19.,CURV=4*0.0,.003,.013,.036,.084,.149,.250,.406,.611,.762,
   .871,.932,.962,.984,.995,.999,3*1.0, &END
```

1. Report No. NASA CR-172464	2. Government Accession No.	3. Recipient's Catalog No.

4. Title and Subtitle Investigation of the External Flow Analysis for Density Measurements at High Altitude	5. Report Date November 1984
	6. Performing Organization Code
7. Author(s) George K. Bienkowski	8. Performing Organization Report No.
	10. Work Unit No.
9. Performing Organization Name and Address Princeton University Dept. of Mechanical and Aerospace Engineering Princeton, NJ 08544	11. Contract or Grant No. NSG-1630
	13. Type of Report and Period Covered Contractor Report 7/1/79-8/31/82
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, DC 20546	14. Sponsoring Agency Code 506-63-37-02

15. Supplementary Notes

Langley Technical Monitor: R. C. Blanchard
Final Report

16. Abstract

This report presents the results of analysis performed on the external flow around the shuttle orbiter nose region at the Shuttle Upper Atmosphere Mass Spectrometer (SUMS) inlet orifice. The purpose of the analysis is to quantitaively characterize the flow conditions to facilitate SUMS flight data reduction and subsequent determination of orbiter aerodynamic force coefficients in the hypersonic rarefied flow regime. Experimental determination of aerodynamic force coefficients requires accurate simultaneous measurement of forces (or acceleration) and dynamic pressure along with independent knowlege of density and velocity. SUMS provides independent measurement of dynamic pressure; however, it does so indirectly and requires knowledge of the relationship between measured orifice conditions and the dynamic pressure which can only be determined on the basis of molecule or theory for a winged configuration.

Monte Carlo direct simulation computer codes were developed for both the flow field solution at the orifice and for the internal orifice flow. These codes were used to study issues associated with geometric modeling of the orbiter nose geometry and the modeling of intermolecular collisions including rotational energy exchange and a preliminary analysis of vibrational excitation and dissociation effects. Data obtained from preliminary simulation runs are presented. Conclusions from these results are included along with recommendations for a SUMS flight data reduction procedure and future work to generate the data required for that procedure.

17. Key Words (Suggested by Author(s)) Reentry Aerodynamics Rarefied Aerodynamics Mass Spectrometer	18. Distribution Statement [redacted]

19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 230	22. Price