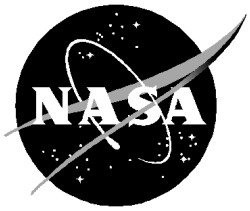
NASA/TM—98-207890



# An Approximate Axisymmetric Viscous Shock Layer Aeroheating Method for Three-Dimensional Bodies

*Irina G. Brykina*

*Carl D. Scott*

May 1998

# The NASA STI Program Office . . . in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA Scientific and Technical Information (STI) Program Office plays a key part in helping NASA maintain this important role.

The NASA STI Program Office is operated by Langley Research Center, the lead center for NASA's scientific and technical information. The NASA STI Program Office provides access to the NASA STI Database, the largest collection of aeronautical and space science STI in the world. The Program Office is also NASA's institutional mechanism for disseminating the results of its research and development activities. These results are published by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA's counterpart of peer-reviewed formal professional papers but has less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or cosponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and mission, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services that complement the STI Program Office's diverse offerings include creating custom thesauri, building customized databases, organizing and publishing research results . . . even providing videos.

For more information about the NASA STI Program Office, see the following:

- Access the NASA STI Program Home Page at ***http://www.sti.nasa.gov***

- E-mail your question via the Internet to help@sti.nasa.gov

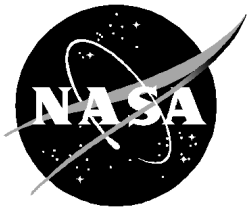
- Fax your question to the NASA Access Help Desk at (301) 621-0134

- Telephone the NASA Access Help Desk at (301) 621-0390

- Write to:

NASA Access Help Desk
NASA Center for AeroSpace Information
800 Elkridge Landing Road
Linthicum Heights, MD 21090-2934

NASA/TM—98-207890



# An Approximate Axisymmetric Viscous Shock Layer Aeroheating Method for Three-Dimensional Bodies

*Irina G. Brykina*
*Moscow State University*

*Carl D. Scott*
*NASA Johnson Space Center*

National Aeronautics and
Space Administration

**Lyndon B. Johnson Space Center**
Houston, Texas 77058-4406

May 1998

Available from:

NASA Center for AeroSpace Information
7121 Standard
Hanover, MD 21076-1320

National Technical Information Service
5285 Port Royal Road
Springfield, VA 22161

# Contents



**Tables**



**Figures**

# Acronyms

| | |
|---|---|
| 2D | two-dimensional |
| 3D | three-dimensional |
| EAB | equivalent axisymmetric body |
| VSL | viscous shock layer |

# Abstract

A technique is implemented for computing hypersonic aeroheating, shear stress, and other flow properties on the windward side of a three-dimensional (3D) blunt body. The technique uses a 2D/axisymmetric flow solver modified by scale factors for a corresponding equivalent axisymmetric body. Examples are given in which a 2D solver is used to calculate the flow at selected meridional planes on elliptic paraboloids in reentry flight. The report describes the equations and the codes used to convert the body surface parameters into input used to scale the 2D viscous shock layer equations in the axisymmetric viscous shock layer code. Very good agreement is obtained with solutions to finite rate chemistry 3D thin viscous shock layer equations for a finite rate catalytic body.


# Introduction

To assess the design of reentry spacecraft with respect to aeroheating, a number of techniques are available, depending on the actual geometry and the needed accuracy. For highest fidelity, solutions to the full Navier-Stokes equations would be required, but this is very time-consuming both in human labor as well as computer time. It is not practical to use the Navier-Stokes solutions for quick assessments of proposed vehicle shapes because of the amount of time it takes to define an adequate flow field grid and to run the solution until convergence. The purpose of this work is to implement a rapid approximate technique for predicting aeroheating on the windward side of three-dimensional (3D) vehicles at angle of attack. The technique involves modifying 2D flow equations for an axisymmetric body at zero angle of attack using scaling parameters in the equations that accounts for 3D effects in the flow. The method was developed originally by Brykina, et al.[1,2] and the factors only depend on body geometry and Reynolds number.

In preliminary stages of design, an approximate geometry and correlations of the heat flux on such geometries may be sufficient. Such tools as Detra, Kemp and Riddell[3], or Fay and Riddell[4] on the stagnation point of spheres may be adequate, or similarly there are relations for swept cylinders. Likewise, flat plate correlations may be used on certain wing shapes or flaps. The MINIVER or LANMIN[5] code can be used for performing approximate heating calculations for simple shapes. The INCHES code by Zoby and Simmonds[6] is an axisymmetric technique that relies on the Maslen technique for shock shape. A somewhat higher fidelity engineering code that also takes into account variable entropy edge conditions is the AEROHEAT code.[7] It uses axisymmetric analog technique for 3D boundary layers. There have been some adequate attempts to use axisymmetric codes for simple shapes such as the viscous shock layer (VSL) method [8,9] and even apply them to complex shapes at angle of attack, by approximating the shape as an hyperboloid of revolution.[10,11] Higher-order approximations have been developed for 3D shapes such as the 3D VSL method and 3D parabolized Navier-Stokes. One method for

avoiding using full Navier-Stokes is to solve the 3D Euler (inviscid) flow equations for the shock layer flow around a body, then use the results as boundary layer edge conditions for an axisymmetric boundary layer solution.[12,13] Even this quasi-3D technique is very labor- and computer-intensive. The technique developed in this report is one of intermediate complexity, in which 2D axisymmetric methods may be used with modifications to approximate the flow field and heat flux on a 3D geometry. This technique yields very good approximate heating that includes surface catalytic effects for chemically nonequilibrium flows. Likewise, one also obtains other flow properties such as species concentrations near the surface. Although the method can be used to scale any set of 2D viscous flow equations, it is applied here to the chemically nonequilibrium 2D VSL equations, specifically, the Miner and Lewis code[14] as modified to include finite wall catalysis boundary conditions.[15]

This report presents scale factors used to modify the VSL equations written in body-oriented coordinates. The method requires computing scaling parameters based on body surface derivatives. The code CONVERT computes these scale factors and other geometry parameters needed in the VSL3D code*. Paraboloids of elliptical cross section are used here to compare with previous work and to illustrate how the method is used. The paraboloid geometry is defined by an auxiliary program, PARG. For each meridional plane and angle of attack of interest, it computes the coordinates and surface derivatives that the code CONVERT uses. To validate this approximate technique we compare its results with 3D results. The Miner and Lewis code solves the 2D VSL equations in body-surface and body-normal coordinates ($S,\eta$), whereas the original 3D body is defined in Cartesian coordinates. Therefore, a third code, INTPX, is used as a post-processor. It uses the output coordinates from the VSL3D code to interpolate the original body coordinates to obtain corresponding locations in the original coordinate system. This is useful for locating the flow properties at locations on the original 3D body.

Results are obtained for several cases for paraboloids of elliptic cross section and compared with 3D VSL solutions.

# Analysis

## Body Coordinates and Input Parameters for VSL3D Code

The modified 2D VSL equations are solved in a streamwise direction along the surface of an axisymmetric body that is equivalent to specific meridional planes $\varphi$ of the 3D body. There is an equivalent axisymmetric body (EAB) for each desired meridional plan and angle of attack. The meridional plane is defined by a plane through the stagnation point, parallel to the wind velocity. The windward symmetry plane represents $\varphi=0$ and the leeward symmetry plane corresponds to

* In this report we denote the axisymmetric viscous shock layer code as VSL and the one modified to handle the 3D EAB as VSL3D.

φ=180°. The input parameters required by the VSL3D code are the axial distance from the stagnation point $z_n$, the distance from the axis to the body $r_n$, the surface distance s, the local body curvature in the $r_n$,$z_n$-plane, κ, and the angle θ, which is the local body tangent with respect to the velocity vector $V_{inf}$. See Figure 1 for a pictorial definition of these parameters. In addition, the modified VSL code requires the scale factor $H/H_s$, which is the ratio of mean curvature of the 3D body to that of the EAB. The code CONVERT computes these parameters from coordinates and surface derivatives for each meridional plane of the 3D body defined in a Cartesian system of coordinates by the equation z=f(x,y). The pitch plane (plane of symmetry) is defined by y=0. The geometrical stagnation point is the location on the windward pitch plane at which the local surface of the body is normal to the velocity vector.

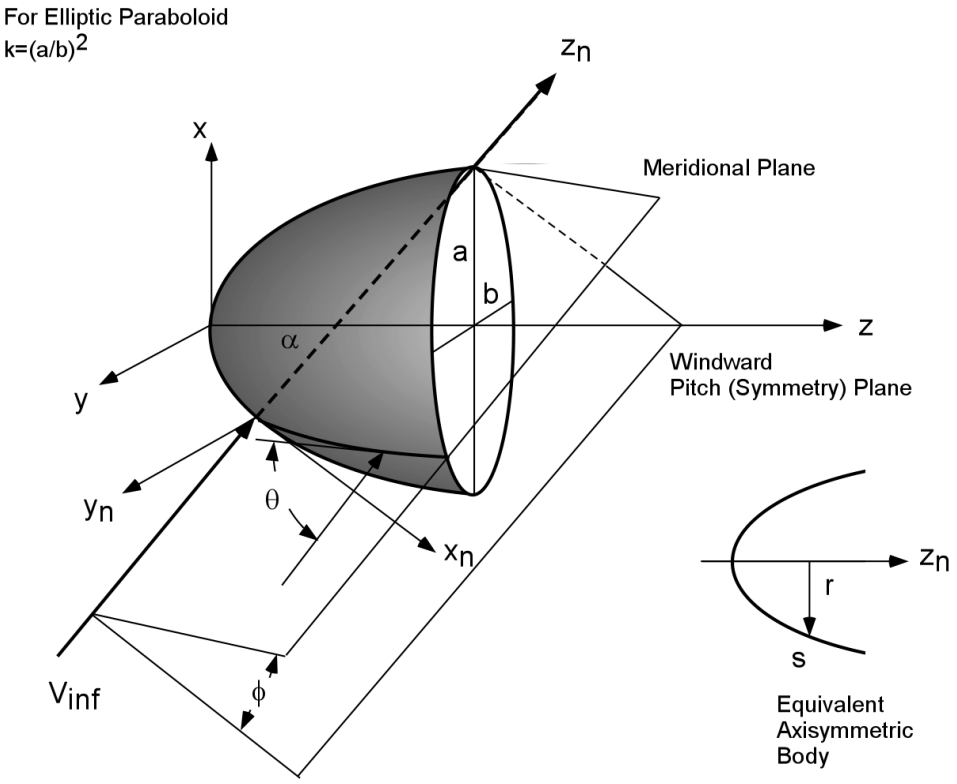



**Figure 1** Coordinate systems for 3D and equivalent axisymmetric bodies.

The mean curvature of the EAB is given by

$$H_s = \frac{\kappa + r^{-1}\cos\theta}{2} \tag{1}$$

and the mean curvature of the 3D body at a point on the surface is

$$H = \frac{f_{xx}(1+f_y^2) + f_{yy}(1+f_x^2) - 2f_{xy}f_x f_y}{2(1+f_x^2+f_y^2)^{3/2}}. \tag{2}$$

These expressions are obtained using differential geometry and their specific forms depend on the coordinate system used. The scale factor $H/H_s$ is applied to the characteristic length or to the Reynolds number that appears in the nondimensional form of the flow equations. It is derived in Ref. 1.

Surface coordinates and surface derivatives in the Cartesian frame are inputs to the code CONVERT, along with the corresponding angle of attack and the meridional angle. To calculate these parameters, it was necessary to deal with several special cases where certain factors are singular or ill conditioned because of the local geometry. It is also necessary to define these quantities at the stagnation because it is a singular point. The following analysis gives the different forms of the equations used in the program CONVERT, which computes these quantities and the equivalent body radius r and streamwise distance s that the VSL code uses.

At the stagnation point $z_n = s = r_n = 0$ and $\theta = \pi/2$. At points away from the stagnation point, we have the following equations for the axial distance $z_n$, and the other transformed coordinates $x_n$ and $y_n$:

$$x_n = (z - z_0)\sin\alpha + (x - x_0)\cos\alpha \tag{3}$$

$$z_n = (z - z_0)\cos\alpha - (x - x_0)\sin\alpha \tag{4}$$

$$y_n = y \tag{5}$$

where $z_0$ and $x_0$ are coordinates of the stagnation point in the original Cartesian coordinate system. The EAB origin is the stagnation point ($x_o$, 0, $z_o$), and its axis $z_n$ is directed along the free stream velocity. The coordinates of the stagnation point are defined by the equations

$$f_{xo} = \tan\alpha$$

$$z_o = f(x_o, 0).$$

The equation of the meridional plane passing through the axis $z_n$ is

$$y = x_n \tan\varphi = \left[(z - z_o)\sin\alpha + (x - x_o)\cos\alpha\right]\tan\varphi . \tag{6}$$

The body angle $\theta$ is given by

$$\sin\theta = \frac{f_x \sin\alpha + \cos\alpha}{\sqrt{1 + f_x^2 + f_y^2}} . \tag{7}$$

To find the coordinates $r_n$ and s we must integrate the body radius and the arc length along the surface of the 3D body in a meridional plane. These quantities are given by

$$r_n = \int_0^{z_n} \frac{dr_n}{dz_n} dz_n \tag{8}$$

and

$$s = \int_0^{Z_n} \frac{ds}{dz_n} dz_n \tag{9}$$

where

$$\frac{dr_n}{dz_n} = \tan\theta \tag{10}$$

$$\frac{ds}{dz_n} = \cos^{-1}\theta \tag{11}$$

Since these integrands are singular at $z_n = 0$, we perform an integration in the direction different from $z_n$ from $z_n = 0$ to some small distance away, $z_{nl}$. These integrals $r_n(z_n)$ and $s(z_n)$ are then performed as

$$r_n = \int_0^{x_{nl}} \frac{dr_n}{dx} dx + \int_{z_{nl}}^{Z_n} \frac{dr_n}{dz_n} dz_n \tag{12}$$

for $\varphi \neq 90°$, or

$$r_n = \int_0^{y_{nl}} \frac{dr_n}{dy} dy + \int_{z_{nl}}^{Z_n} \frac{dr_n}{dz_n} dz_n \tag{13}$$

if $\varphi = 90°$.

$$s = \int_0^{x_{nl}} \frac{ds}{dx} dx + \int_{z_{nl}}^{Z_n} \frac{ds}{dz_n} dz_n \tag{14}$$

if $\varphi \neq 90°$, or

$$s = \int_0^{y_{nl}} \frac{ds}{dy} dx + \int_{z_{nl}}^{Z_n} \frac{ds}{dz_n} dz_n \tag{15}$$

if $\varphi = 90°$. The point at which we make the crossover is a parameter that we have taken arbitrarily to be $z_{nl} = 0.1$. These expressions are integrated using Simpson's rule.

For the region away from the stagnation point where $z_n > z_{nl}$ we integrate along the axial coordinate, where we use the relations (10) and (11). In this zone we integrate with respect to $z_n$ from $z_{nl}$ to $Z_n$.

The surface arc length ds and radial distance dr are determined from the following equations where we integrate in the x or y direction. When $\varphi$ is very near $\pi/2$ the differential dr is given by

$$\frac{dr_n}{dy} = \frac{dr_n}{dz_n} f_y \cos\alpha = \frac{(f_x \sin\alpha + \cos\alpha) f_y \cos\alpha}{\sqrt{(f_x \cos\alpha - \sin\alpha)^2 + f_y^2)}} = \tan\theta f_y \cos\alpha \tag{16}$$

and ds is given by

$$\frac{ds}{dy} = \frac{ds}{dz_n} f_y \cos\alpha = f_y \cos\alpha \sqrt{\frac{1 + f_x^2 + f_y^2}{(f_x \cos\alpha - \sin\alpha)^2 + f_y^2)}} = f_y \frac{\cos\alpha}{\cos\theta}. \tag{17}$$

When $\varphi \neq \pi/2$ and for points for which $z \leq z_{nl}$

$$\frac{dr_n}{dx} = \frac{dr_n}{dz_n}[(f_x + df_y)\cos\alpha - \sin\alpha] = \frac{(f_x \sin\alpha + \cos\alpha)[(f_x + df_y)\cos\alpha - \sin\alpha]}{\sqrt{(f_x \cos\alpha - \sin\alpha)^2 + f_y^2}} \tag{18}$$

$$= \tan\theta[(f_x + df_y)\cos\alpha - \sin\alpha]$$

and

$$\frac{ds}{dx} = \frac{ds}{dz_n}[(f_x + df_y)\cos\alpha - \sin\alpha] = [(f_x + df_y)\cos\alpha - \sin\alpha]\sqrt{\frac{(1 + f_x^2 + f_y^2)}{(f_x \cos\alpha - \sin\alpha)^2 + f_y^2}} \tag{19}$$

$$= \left[\left(f_x + df_y\right)\cos\alpha - \sin\alpha\right] / \cos\theta$$

If $\varphi = \pi/2$, then at the stagnation point $[dr_n/dy]_0 = [ds/dy]_0 = 1$.

If $\varphi \neq \pi/2$, then

$$\left[\frac{dr_n}{dx}\right]_0 = \left[\frac{ds}{dx}\right]_0 = \frac{f_{xx0} + d_0^2 f_{yy0}}{\sqrt{f_{xx0}^2 \cos^2\alpha + d_0^2 f_{yy0}^2}}. \tag{20}$$

We can obtain the curvature in terms of the surface derivatives of the 3D body and angle of attack $\alpha$. The body in a Cartesian coordinate frame is defined as $z=f(x,y)$. The body curvature is defined from

$$\kappa = -\frac{d\theta}{ds} = -\frac{d(\sin\theta)}{dz_n}. \tag{21}$$

When $\varphi \neq \pi/2$, we have

$$\kappa = \frac{(f_{xx} + df_{xy})\left[f_x \cos\alpha - \sin\alpha\left(f_y^2 + 1\right)\right] + (f_{xy} + df_{yy})(f_x f_y \sin\alpha + f_y \cos\alpha)}{(1 + f_x^2 + f_y^2)^{3/2}\left[(f_x + df_y)\cos\alpha - \sin\alpha\right]} \tag{22}$$

where the full derivative along the meridional plane is

$$d = \frac{dy}{dx} = \frac{\cos\alpha + f_x \sin\alpha}{1 - f_y \sin\alpha \tan\varphi} \tan\varphi. \tag{24}$$

When $\varphi = \pi/2$ equation (22) reduces to

$$\kappa = \frac{f_{xy}\left[f_x \cos\alpha - \sin\alpha\left(f_y^2 + 1\right)\right] + f_{yy} f_y (f_x \sin\alpha + \cos\alpha)}{f_y \cos\alpha (1 + f_x^2 + f_y^2)^{3/2}}. \tag{25}$$

The ratio of mean curvatures for the 3D body to an EAB is given by

$$H / H_s = \frac{f_{xx}(1+f_y^2)+f_{yy}(1+f_x^2)-2f_{xy}f_xf_y}{(\kappa + r_n^{-1}\cos\theta)(1+f_x^2+f_y^2)^{3/2}} \tag{26}$$

where $r_n$ is the distance from the axis of the EAB.

The angle of attack is given in the input as $\alpha$; the meridional angle is $\varphi$. Two forms of equation for the body curvature $\kappa$ are used, depending on whether the meridional angle $\varphi = \pi/2$. If $\varphi \neq \pi/2$, we use the form

$$\kappa_0 = \frac{(f_{xx0}+d_0 f_{xy0})^2\cos\alpha + d_0(f_{xy0}+d_0 f_{yy0})f_{yy0}(f_{x0}\sin\alpha+\cos\alpha)}{(1+f_{x0}^2)^{3/2}\cos\alpha(f_{xx0}+2d_0 f_{xy0}+d_0{}^2 f_{yy0})} \tag{27}$$

where

$$d_0 = \frac{\cos\alpha + f_{x0}\sin\alpha}{1-f_{y0}\sin\alpha\tan\varphi}\tan\varphi\,. \tag{28}$$

If $\varphi$ is about $\pi/2$, then (22) reduces to the form

$$\kappa_0 = \frac{f_{xy0}^2 \quad \cos\alpha + f_{yy0}{}^2(f_{x0}\sin\alpha+\cos\alpha)}{f_{yy0}\cos\alpha(1+f_{x0}^2)^{3/2}}\,. \tag{29}$$

The subscript $_0$ denotes the stagnation point.

The ratio of average curvatures at the stagnation point is

$$(H/H_s)_0 = \frac{f_{xx0}(1+f_{yo}^2)+f_{yy0}(1+f_{x0}^2)}{2\kappa_o(1+f_{x0}^2)^{3/2}} = \frac{(f_{xxo}\cos^2\alpha + f_{yyo})\cos\alpha}{2\kappa_o} \tag{30}$$

After calculating these variables, they are then normalized with respect to the stagnation point curvature:

$$z_n = z_n{}^* \kappa_0{}^* \tag{31}$$

$$r_n = r_n{}^* \kappa_0{}^* \tag{32}$$

$$s = s^* \kappa_0{}^* \tag{33}$$

$$H/H_s = (H/H_s)^* \kappa_0{}^* \tag{34}$$

$$\kappa = \kappa^* / \kappa_0{}^* \tag{35}$$

where the superscript * denotes the dimensional or unscaled variables. The scaled variables, along with the body angle $\theta$, are output from the CONVERT code to be used as input in the VSL3D code. They represent the axisymmetric body that is equivalent to the actual 3D body

along a meridional plane. Appendix A gives a listing of the computer code CONVERT used to generate input for the VSL3D code.

## Modifications to the VSL Code

The VSL equations for an axisymmetric body are given in Ref. 8, but they must be modified slightly for correct application of the EAB. From similarity relations[2] it follows that, to obtain heat flux, shear stress, and species concentrations on an actual 3D body, we have to solve 2D equations for the EAB with variable Reynolds number, $Re^* = (H_s/H)Re$. The VSL equations are thus modified by the transformation using the 3D body parameters. The only places in the VSL code that require modification are in the calculation of Re or the Reynolds number parameter $\varepsilon^2 = 1/\mathrm{Re}$ and the parameter WREF$\propto R_o$. These parameters, associated with length scaling, are multiplied by $H/H_s$ at each point along the surface of the EAB.

Since the VSL3D code has additional inputs due to its new features, a sample input file is included in Appendix B. Appendix C contains the UNIX run stream for a series of cases. The output from the VSL3D code is found in Fortran unit 23, a sample of which is found in Appendix D. The output from the interpolation code is found in Fortran unit 50, a sample of which is found in Appendix E. Note that the nomenclature on output from the VSL3D code is different from that found in this paper. Table 1 defines the correspondence between the two systems of nomenclature.

**Table 1 Symbol Correspondence**

| Symbols used in this report | Symbols in VSL3D and INTPX code output |
|---|---|
| $r_n$ | RS, rs(I) |
| $z_n$ | XB, xb(I) |
| s | S, s(I) |
| q | qw |
| $\kappa$ | ck |
| $H/H_s$ | HHS |
| $\tau$ | tauw |
| $\varepsilon$ | eps |

## Application to an Elliptic Paraboloid

To test the coding and the accuracy of the methodology, we have implemented a code that computes the shape parameters for a paraboloid having an elliptic cross section. This can be

expressed in Cartesian coordinates as $z = \frac{1}{2}(x^2 + ky^2)$, where $\sqrt{k}$ =a/b is the ratio of principal axes (x-to-y) of the elliptic cross section (or ratio of principal curvatures at the apex of the paraboloid). An auxiliary code (PARG) was used to calculate the shape and surface derivatives along planes through the wind axis of the paraboloid at angle of attack. The output of PARG is then used as the input of CONVERT to generate the geometry data file for VSL3D. PARG generates lines of data, given k, the angle of attack $\alpha$, and the desired meridional plane $\varphi$. The windward symmetry plane corresponds to $\varphi = 0$ and the leeward symmetry plane to $\varphi = 180°$. The location of the stagnation point is simply the point where the normal to the surface is parallel to the wind axis, or where $dz/dx = x_0 = \tan\alpha$ and $y_0=0$. Then $z_0 = \frac{1}{2}x_0^2 = \frac{1}{2}\tan^2\alpha$. An array of coordinates along a meridional plane on the surface of the paraboloid is set up in equal steps in x unless $\varphi = 90°$. If $\varphi = 90°$, then equal steps in y are taken and all $x = x_0$. When the body surface angle becomes parallel to the free stream flow (in the code when $\cos\varphi < 0.01$) the line of points is terminated because we can only calculate the flow where the surface is presented to the wind. When $\varphi \neq 0$ or 180° and if $\alpha \neq 0$ we have the expression

$$y = \frac{1 - \sqrt{1 - k\tan^2\varphi\sin^2\alpha[x^2 + 2x\cot\alpha - (2 + \tan^2\alpha)]}}{k\sin\alpha\tan\varphi} \tag{36}$$

or if $\alpha = 0$ then

$$y = x\tan\varphi. \tag{37}$$

If $\varphi$=0 or 180°, then y=0.

We then find the array of z values from

$$z = f(x,y) = \frac{1}{2}(x^2 + ky^2). \tag{38}$$

The surface derivatives are determined analytically and we get the expressions $f_x = x$, $f_y = ky$, $f_{xx} = 1$, $f_{xy} = 0$, and $f_{yy} = k$. A listing of the code that generates the paraboloid geometry is given in Appendix F.

## Results

To test the code we have applied it to several flight cases for elliptic paraboloids. Table 2 describes the flight conditions and Table 3 gives the geometrical parameters used in the calculations. The surface catalytic recombination coefficients are taken from Ref. 16; and the wall temperature is calculated in the code under the assumption of radiation equilibrium.

**Table 2 Flight Conditions for Test Case for Elliptic Paraboloids**

| Altitude, km | Velocity, km/s | Density, $kg/m^3$ | Temperature, K |
|---|---|---|---|
| 70 | 7.25 | $5.91x10^{-5}$ | 200 |

**Table 3 Geometry Configuration for Test Cases**

| Radius at Stag. Point in x-z plane, m | Angle of Attack, $\alpha$ | Cross Section Axis Ratio, k | Meridional Plane, $\varphi$ |
|---|---|---|---|
| 0.5 | 0 | 0.25 | 0 |
| 0.5 | 0 | 0.25 | 45 |
| 0.5 | 0 | 0.25 | 63.4 |
| 0.5 | 0 | 0.25 | 76 |
| 0.5 | 0 | 0.25 | 90 |
| 0.7 | 15 | 0.4 | 0 |
| 0.7 | 15 | 0.4 | 180 |
| 0.7 | 30 | 0.4 | 0 |
| 0.7 | 30 | 0.4 | 180 |
| 0.7 | 45 | 0.4 | 0 |
| 0.7 | 45 | 0.4 | 180 |
| 0.5 | 0 | 0.4 | 0 |
| 0.5 | 0 | 0.4 | 45 |
| 0.5 | 0 | 0.4 | 90 |

The first comparison is a flight case for an elliptic paraboloid at angle of attack of zero. This elliptic paraboloid has a ratio of principal axes of 0.5, or k=0.25. The method was applied to five meridional planes $\varphi$ where the PARG code was used to compute the geometry and surface derivatives for each $\varphi$. The code CONVERT transformed these 3D parameters (coordinates and surface derivatives) into EAB coordinates parameters $z_n$, $r_n$, s, $\kappa$, $\theta$, and $H/H_s$. These parameters were input into the VSL3D code (the modified axisymmetric VSL code8) and solutions were found. Then the streamwise values were interpolated to transform these locations back to the 3D body coordinates using a code given in Appendix G. The results for the first comparison are shown in Figure 2, where the solid curves are from the axisymmetric analog solution described

here and the open circles are solutions of the 3D VSL equations from Ref. 17, solved by an implicit finite difference scheme[18] of fourth-order accuracy in normal coordinates and second-order accuracy in longitudinal coordinates as implemented by Shcherbak[1] for 3D viscous flows. The agreement is quite good, within about 7% or better.

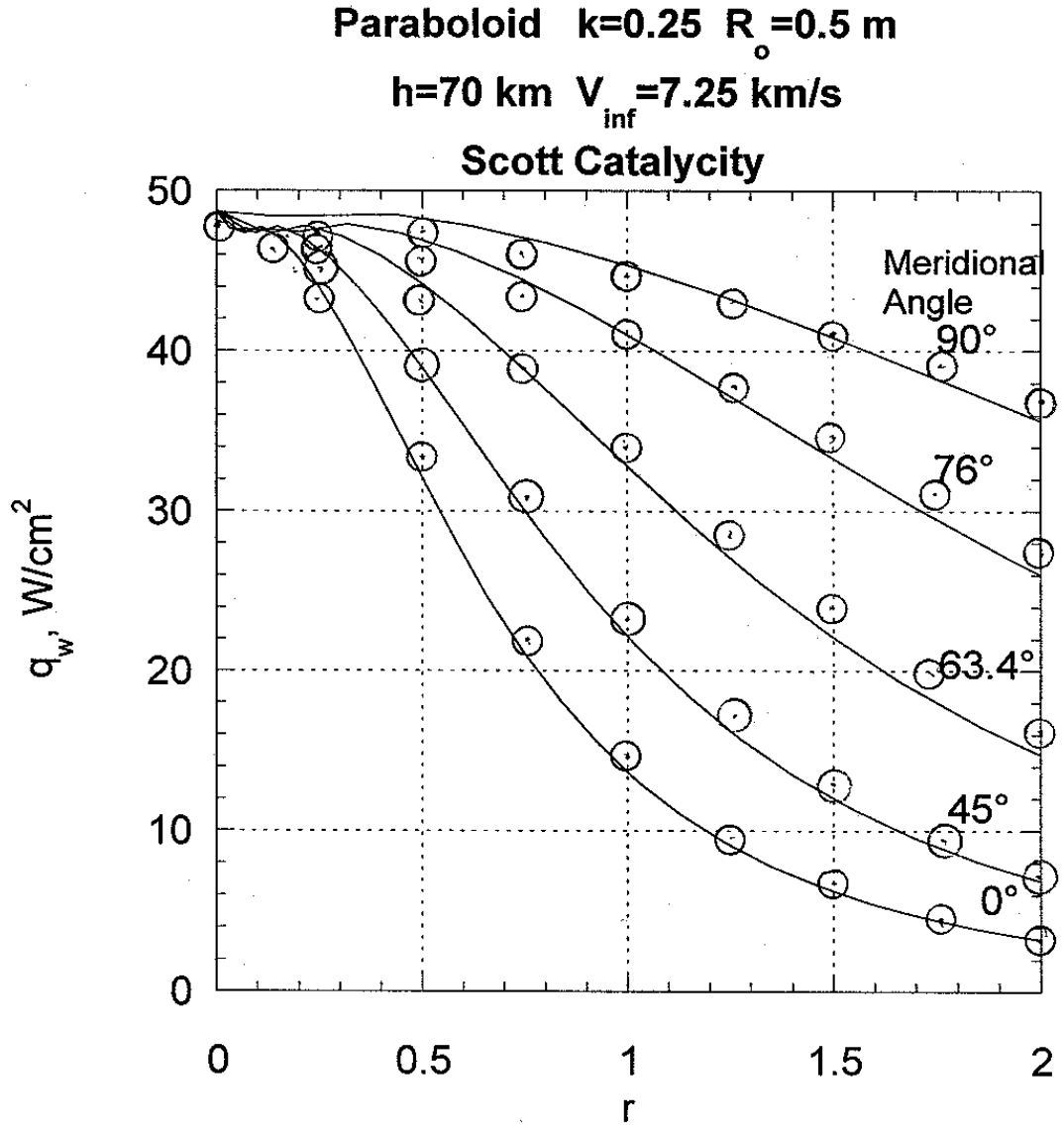



**Figure 2** Heat flux distribution at various meridional angles on an elliptic paraboloid: k=0.25, $R_o$=0.5 m, h=70 km, $V_\infty$=7.25 km/s, catalytic rates from Ref. 15, and radiative equilibrium wall. Open circles are 3D viscous shock layer calculations from Ref. 17.

The second comparison is for the same flight case but with a body of larger radius at the stagnation point ($R_o$=0.7 m) with elliptic cross section having k=0.4. The angle of attack was varied, and the results for the pitch plane (plane of symmetry) are given in Figure 3. These results are compared with 3D VSL solutions from Ref. 2. Here the agreement is good, but tends to diminish on the windward side far away from the stagnation point. The shear stress for this case is given in Figure 4 and is compared with 3D solutions from Ref. 2.

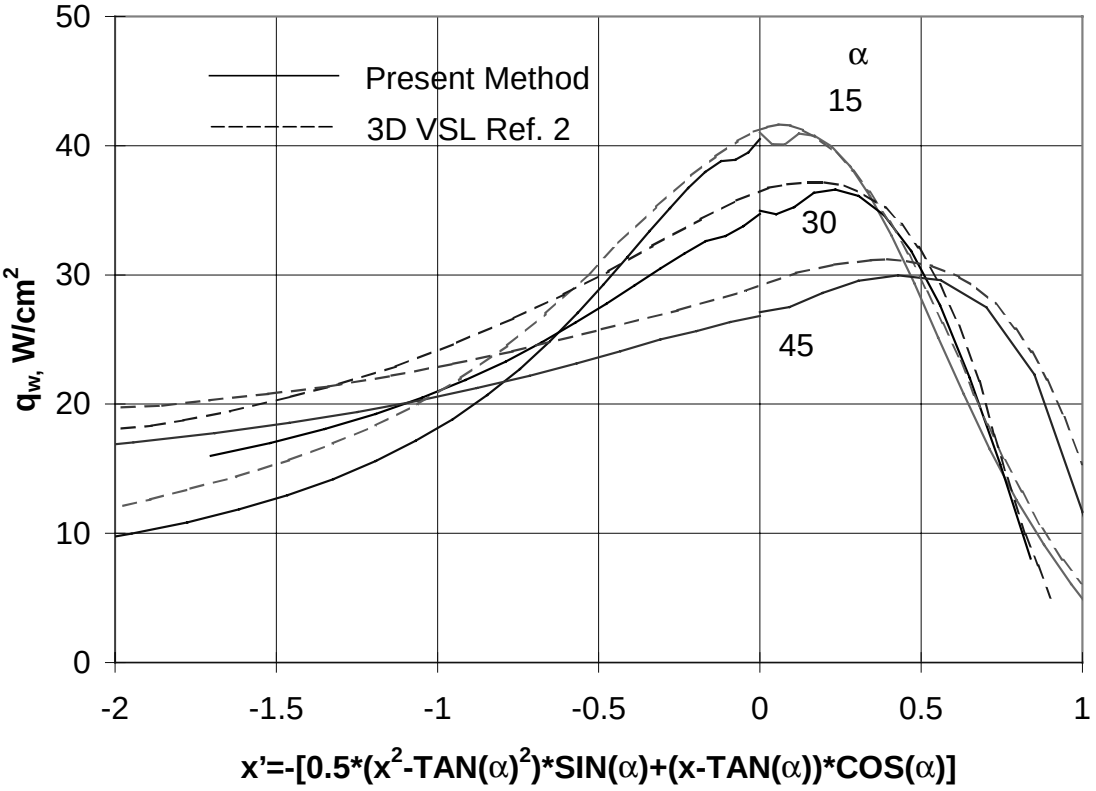



**Figure 3** Heat flux distribution on symmetry plane of elliptic paraboloid at various angles of attack with: $R_o$=0.7 m, k=0.4, h=70 km, $V_\infty$=7.25 km/s, catalytic rates from Ref. 15, and radiative equilibrium wall. Dashed lines are 3D viscous shock layer calculations from Ref. 2.

As example of surface atom mass fractions, we show results for an elliptic paraboloid having a principal axis ratio of 0.707, or k=0.5 at an altitude of 70 km, velocity of 7.25 km/s, and nose radius in the pitch plane of symmetry of 0.7 m. Fig. 5 shows the mass fractions of oxygen and nitrogen in three meridional planes. Other flow field properties are available for these equivalent axisymmetric bodies calculated by the VSL3D code.*

* The source code for the modified Miner and Lewis viscous shock layer code (VSL3D) is available from the author at ES3, NASA Johnson Space Center, Houston, TX 77058.

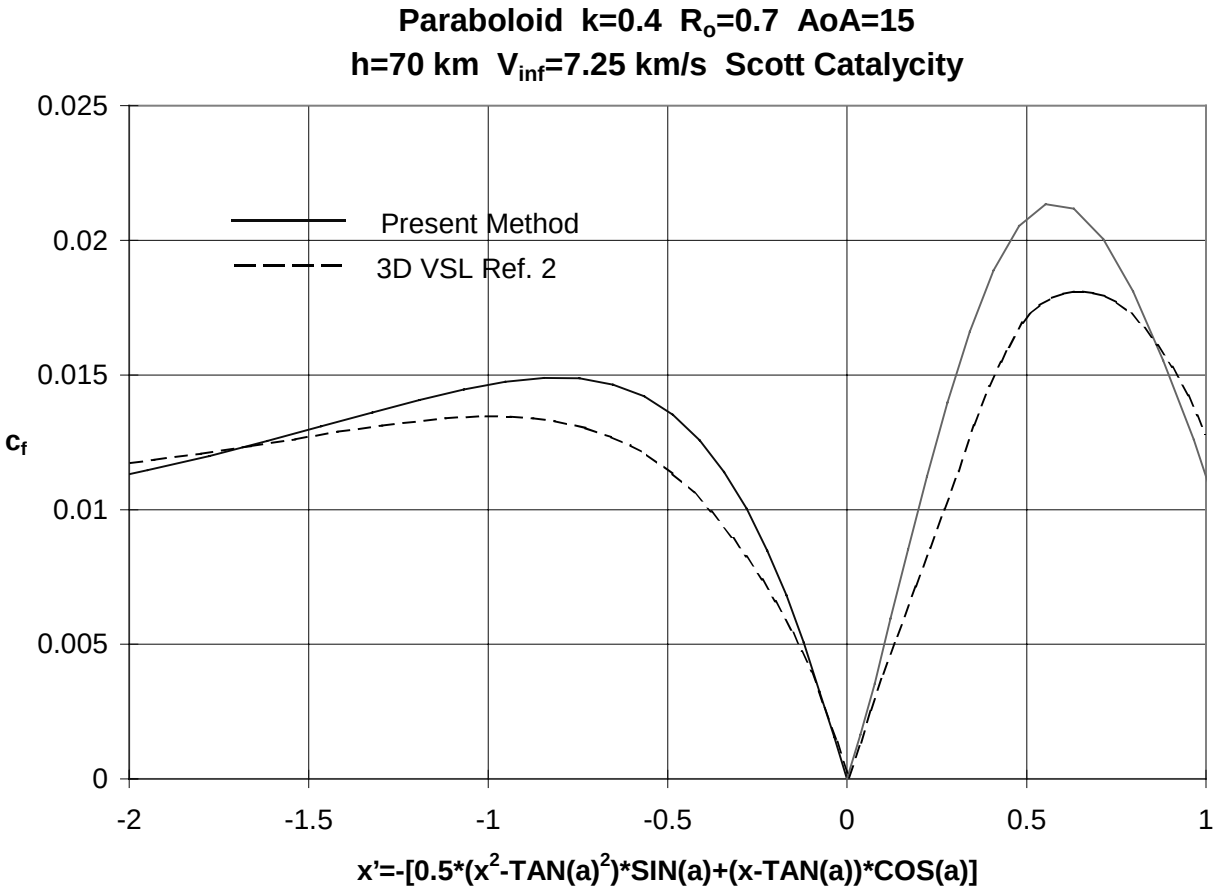



**Figure 4** Shear stress distribution on elliptic paraboloid at an angle of attack of 15°: $R_o$=0.5 m, k=0.4, h=70 km, $V_\infty$=7.25 km/s, catalytic rates from Ref. 15, and radiative equilibrium wall. Dashed lines are 3D viscous shock layer calculations from Ref. 2.

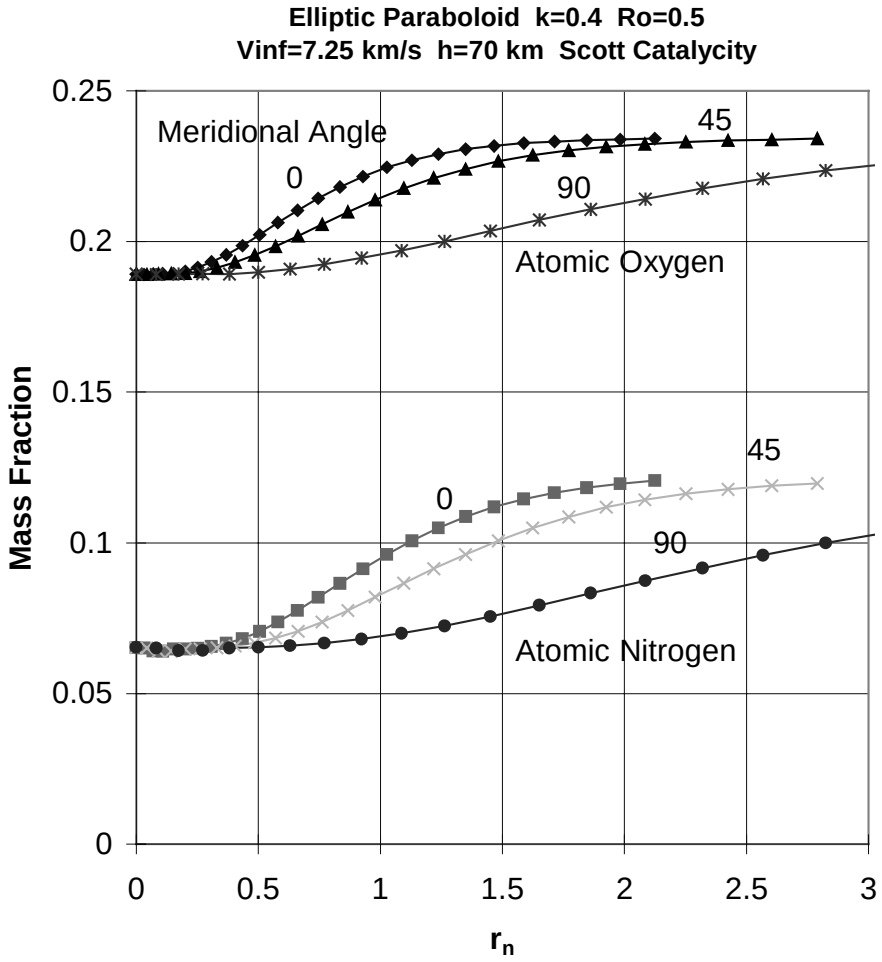



**Figure 5** Mass fraction distributions of N and O on meridional planes of an elliptic paraboloid: $R_o$=0.5 m, k=0.4, h=70 km, $V_\infty$=7.25 km/s, catalytic rates from Ref. 15, and radiative equilibrium.

# Conclusions

An axisymmetric analog technique has been developed and codes to implement it have been described that allows one to calculate heat flux distributions on 3D bodies or 2D bodies at an angle of attack (thus in a 3D flow field). The technique follows that of Brykina, et al.,[1] and results from it have been compared with 3D VSL calculations of heat flux on elliptic paraboloids. The technique involved modifying the Miner and Lewis[8,15] nonequilibrium, seven-species, 2D VSL code. The results indicate that there is very good agreement between the EAB results and the 3D calculations.

The method also can be implemented in any axisymmetric flow code to simulate a 3D body or flow by modifying the equations with the scale factor $H/H_s$ and using the EAB coordinates. This method can be applied to any blunt geometry that can be described by coordinates and surface derivatives. Further applications will be the subject of a future publication.

[1] Brykina, I. G., Rusakov, V. V., and Sherbak, V. G., "A Method of Determining the Heat Fluxes and Skin Friction in Three-Dimensional Problems of Hypersonic Flow Using Two-Dimensional Solutions," *Dokl. Akad. Nauk SSSR*, Vol. 316, 1991, pp. 62-66.

[2] Brykina, I. G., "The similarity method for solving three-dimensional super- and hypersonic viscous flows over blunted bodies," *Proceedings of the Second European Symposium on Aerothermodynamics for Space Vehicles*, ESTEC, Noordwijk, The Netherlands, 21-25 November 1994. (ESA SP-367, Feb. 1995, pp. 109-114).

[3] Detra, R. W., Kemp, N. N., and Riddell, F. R., "Addendum to 'Heat Transfer to Satellite Vehicles Reentering the Atmosphere,'" *Jet Propulsion*, Vol. 27, No. 12, Dec. 1957, pp. 1256-1257.

[4] Fay, J. A. and Riddell, F. R., "Theory of Stagnation-Point Heat Transfer in Dissociated Air," *Journal of Aerospace Sciences*, Vol. 25, No. 2, Feb. 1958, pp. 73-85.

[5]Engle, C. D. and Praharaj, S. C., "MINIVER Upgrade for the AVID System, Vol. I: LANMIN User's Manual," NASA CR-172212, August 1982.

[6] Zoby, E. V. and Simmonds, A. L., "Engineering Flowfield Method with Angle-of-Attack Applications," *Journal of Spacecraft and Rockets*, Vol. 22, July-August 1985, pp. 398-404.

[7] DeJarnette, F. R. and Hamilton, H. H., II, "Aerodynamic Heating on 3-D Bodies Including the Effects of Entropy Layer Swallowing," *Journal of Spacecraft and Rockets*, Vol. 12, January 1975, pp. 5-12.

[8] Miner, E. W. and Lewis, C. H., "Hypersonic Ionizing Air Viscous Shock-Layer Flows Over Nonanalytical Blunt Bodies," NASA CR-2550, May 1975.

[9] Moss, J. N., "Stagnation and Downstream Viscous Shock Layer Solutions with Radiation and Coupled Ablation Injection," AIAA Paper 74-73, June 1974.

[10] Shinn, J. L., Moss, J. N., and Simmonds, A. L., "Viscous Shock Layer Heating Analysis for the Shuttle Windward Plane with Surface Finite catalytic Recombination Rates," AIAA Paper 82-0842, June 1982.

[11] Scott, C. D., "Nonequilibrium and Catalysis on Shuttle Heat Transfer," *Journal of Spacecraft and Rockets*, Vol. 22, No. 5, Sept.-Oct. 1985, pp. 489-499.

[12] Goodrich, W. D., Li, C.-P., Houston, C. K., Chin, P. B., and Olmedo, L., "Numerical computations of Orbiter Flowfields and Laminar Heating Rates," *Journal of Spacecraft and Rockets*, Vol. 14, May 1977, pp. 257-264.

[13] Rakich, J. V. and Lanfranco, M. J., "Numerical Computation of Space Shuttle Laminar Heating and Surface Streamlines," *Journal of Spacecraft and Rockets*, Vol. 14, May 1977, pp. 265-272.

[14] Miner, E. W. and Lewis, C. H., "Computer User's Guide for a Chemically Reacting Viscous Shock Layer Code," NASA CR-2551, May 1975.

[15] Scott, C. D., "Effects of Nonequilibrium and Wall Catalysis on Shuttle Heat Transfer," *Journal of Spacecraft and Rockets*, Vol. 22, No. 5, Sept.-Oct., 1985, pp. 489-499.

[16] Scott, C. D., "Catalytic Recombination of Nitrogen and Oxygen on High-Temperature Reusable Surface Insulation," in *Aerothermodynamics and Planetary Entry*, A. L. Crosbie, Ed., AIAA, 1981.

[17] Brykina, I. G., Rusakov, V. V., and Shcherbak, V. G., "The Similarity Relations for Calculation of Chemically Nonequilibrium Hypersonic Flows Over Blunted Bodies," Report No. 3971, Institute of Mechanics, Moscow State University, October, 1990.

[18] Petukhov, I. V., "The Numerical Calculation of Two-dimensional Flows in Boundary Layer," in *Numerical Methods for Solving differential Equations and Quadrature Formulas*, Moscow, Nauka, 1964, pp. 304-325.

## Appendix A - Listing of the program CONVERT that computes equivalent axisymmetric coordinates and scaling parameter $H/H_s$ from input surface geometry data

```
c  Program Convert
c
      parameter (nn=1000)
      dimension z(nn),fx(nn),fy(nn),fxx(nn),fyy(nn),fxy(nn)
      dimension r(nn),s(nn),ck(nn),th(nn),hhs(nn)
      common/body/ xm(nn),ym(nn),x(nn),y(nn),fs(nn),fr(nn),zn(nn),n
      external fsx,frx,frz,fsz,fsy,fry ,fsxm,frxm
c
      character*8 na,me
      real*4 k,lzn
c  Input angle of attack alpha, degrees
c  Input angle fi and number of points n for meridional plane
c  Input file for geometry along meridional plane
c  Input small interval lzn near stagnation point for integral on x
c     There should be several grid points within the interval lzn.
c     Integrate wrt x near stagnation point and wrt z elsewhere.
      alpha=0.
      fi=0.
      read(31,*)k,alpha,fi
      write(6,*)k,alpha,fi
      fid=fi
      j=1
 1     read(31,*,end=999,err=888) x(j),y(j),z(j),fx(j),fy(j),
     *  fxx(j),fxy(j),fyy(j)
      xm(j)=-x(j)
      ym(j)=-y(j)
      j=j+1
      go to 1
 888  write(6,*)'  In cgeomn:  Error reading unit 31.'
 999  continue
      n=j-1
      write(33,32)( x(j),y(j),z(j),fx(j),fy(j),fxx(j),
     *  fxy(j),fyy(j),j=1,5)
      write(6,*)' no. of points read',n
32      format(8g16.6)
      lzn=0.1
c  End of input
c
      alpha=alpha*1.5707963/90
      fi=fi*1.5707963/90
      if(abs(fi-1.5708).lt.0.001) go to 26
      a=tan(fi)
 26   ca=cos(alpha)
      sa=sin(alpha)
c
c  Stagnation point
      write(29,*)'zn  ck  th'
      zn(1)=0.
      s(1)=0.
      r(1)=0.
      th(1)=1.5707963
      if(abs(fi-1.5708).lt.0.001) then
        ck(1)=(fxy(1)**2*ca+fyy(1)**2*(fx(1)*sa+ca))/
     *   (fyy(1)*ca*(1.+fx(1)**2)**1.5)
      else
        da=(ca+fx(1)*sa)/(1.-a*fy(1)*sa)
c234567890123456789012345678901234567890123456789012345678901234567890123456789012
```

```
         ck(1)=((fxx(1)+a*da*fxy(1))*(fxx(1)+da*a*fxy(1))*ca+a*da*
     *    (fxy(1) +a*da*fyy(1))*fyy(1)*(fx(1)*sa+ca))
     * /((1.+fx(1)**2)**1.5*ca*(fxx(1)+2.*a*da*fxy(1)+da**2*
     *  a**2*fyy(1)))
      endif
      hhs(1)=0.5*(fxx(1)+fyy(1)*(1.+fx(1)**2))/
     * ((1.+fx(1)**2)**1.5*ck(1))
      write(29,*)zn(1),ck(1),th(1)*90./1.5707963
c
c  Meridional plane
c  Compute z-distance (zn), curvature (ck) and angle (th):
c
      do 30 j=2,n
         zn(j)=(z(j)-z(1))*ca-(x(j)-x(1))*sa
      if(abs(fi-1.5708).lt.0.001) then
         ck(j)=(fxy(j)*(fx(j)*ca-fy(j)**2*sa-sa)+fyy(j)*fy(j)*
     * (fx(j)*sa+ca))/(fy(j)*ca*(1+fx(j)**2+fy(j)**2)**1.5)
      else
        da=(ca+fx(j)*sa)/(1.-a*fy(j)*sa)
         ck(j)=((fxx(j)+a*da*fxy(j))*(fx(j)*ca-fy(j)**2*sa-sa)+
     *   (fxy(j)+a*da*fyy(j))*(fx(j)*fy(j)*sa+fy(j)*ca))/
     *   ((1+fx(j)**2+fy(j)**2)**1.5*((fx(j)+a*da*fy(j))*ca-sa))
      endif
         th(j)=asin((fx(j)*sa+ca)/SQRT(1+fx(j)**2+fy(j)**2))
      write(29,*)zn(j),ck(j),th(j)*90./1.5707963
  30  continue

c
c  Compute body radius (r) and surface distance (s):
      j=1
      do while(zn(j).lt.lzn)
         j=j+1
      end do
      jj=j
c
      if(abs(fi-1.5708).lt.0.001) then
      fr(1)=1.
      fs(1)=1.
      do 37 j=2,jj+1
         fr(j)=(sa*fx(j)+ca)/sqrt((fx(j)*ca-sa)**2+fy(j)**2) *fy(j)*ca
         fs(j)=sqrt((1+fx(j)**2+fy(j)**2)/((fx(j)*ca-sa)**2+fy(j)**2))*
     *      fy(j)*ca
c234567890123456789012345678901234567890123456789012345678901234567890123456789012
  37  continue
      do 39 j=2,jj
         call simp(fsy,y(j-1),y(j),sy)
         call simp(fry,y(j-1),y(j),ry)
         s(j)=sy+s(j-1)
         r(j)=ry+r(j-1)
  39  continue
      else
        if(a.lt.-.00000001)then
        d=a/ca
        fr(jj)=(fxx(1)+d*fxy(1)+d*(fxy(1)+d*fyy(1)))/
     *   sqrt((fxx(1)+d*fxy(1))**2*ca**2+(fxy(1)+d*fyy(1))**2)
        fs(jj)=fr(jj)
        xm(jj)=x(1)
        do 41 j=2,jj+1
         i=jj-j+1
        da=(ca+fx(j)*sa)/(1.-a*fy(j)*sa)
         fr(i)=(sa*fx(j)+ca)/sqrt((fx(j)*ca-sa)**2+fy(j)**2)
```

```
     *   *((fx(j)+a*da*fy(j))*ca-sa)
         fs(i)=sqrt((1+fx(j)**2+fy(j)**2)/((fx(j)*ca-sa)**2+fy(j)**2))*
     *   ((fx(j)+a*da*fy(j))*ca-sa)
         fr(i)=-fr(i)
         fs(i)=-fs(i)
         xm(i)=x(j)
  41    continue
        do 51 i=2,jj
         call simp(fsxm,xm(i-1),xm(i),ss)
         call simp(frxm,xm(i-1),xm(i),rr)
         s(i)=ss+s(i-1)
         r(i)=rr+r(i-1)
  51   continue
        else
        d=a/ca
        fr(1)=(fxx(1)+d*fxy(1)+d*(fxy(1)+d*fyy(1)))/
     *   sqrt((fxx(1)+d*fxy(1))**2*ca**2+(fxy(1)+d*fyy(1))**2)
        fs(1)=fr(1)
        do 40 j=2,jj+1
        da=(ca+fx(j)*sa)/(1.-a*fy(j)*sa)
         fr(j)=(sa*fx(j)+ca)/sqrt((fx(j)*ca-sa)**2+fy(j)**2)
     *   *((fx(j)+a*da*fy(j))*ca-sa)
         fs(j)=sqrt((1+fx(j)**2+fy(j)**2)/((fx(j)*ca-sa)**2+fy(j)**2))*
     *   ((fx(j)+a*da*fy(j))*ca-sa)
  40    continue
  34  format(4g16.6)
        do 50 j=2,jj
         call simp(fsx,x(j-1),x(j),ss)
         call simp(frx,x(j-1),x(j),rr)
         s(j)=ss+s(j-1)
         r(j)=rr+r(j-1)
  50   continue
      endif
        endif
c
      Do 60 j=jj-1,n
         fr(j)=tan(th(j))
         fs(j)=1/cos(th(j))
  60  continue
      write(32,*)'x(j),zn(j),fr(j),fs(j)'
      write(32,34)(x(j),zn(j),fr(j),fs(j),j=1,n)
      do 70 j=jj+1,n
         call simp(fsz,zn(jj),zn(j),sss)
         call simp(frz,zn(jj),zn(j),rrr)
         s(j)=s(jj)+sss
         r(j)=r(jj)+rrr
  70  continue
c
c  Compute a ratio of average curvatures of 3D and axisymm. bodies hhs:
      do 80 j=2,n
         h=0.5*(fxx(j)*(1+fy(j)**2)+fyy(j)*(1+fx(j)**2)-
     *   2*fxy(j)*fx(j)*fy(j))/(1+fx(j)**2+fy(j)**2)**1.5
         hs=0.5*(ck(j)+cos(th(j))/r(j))
         hhs(j)=h/hs
  80  continue
c
c  Do scaling for body with stagnation point radius = 1:
      do 90 j=2,n
         zn(j)=zn(j)*ck(1)
         r(j)=r(j)*ck(1)
         s(j)=s(j)*ck(1)
```

```
         ck(j)=ck(j)/ck(1)
         hhs(j)=hhs(j)*ck(1)
  90  continue
         zn(1)=zn(1)*ck(1)
         r(1)=r(1)*ck(1)
         s(1)=s(1)*ck(1)
         hhs(1)=hhs(1)*ck(1)
         ck(1)=ck(1)/ck(1)
c
c  Create a file for M.-L. code:
      hhs1=1.
c     write(40,*) ' jj alpha, fi, n ',jj,alpha,fid,n
      write(40,88)(zn(j),r(j),s(j),ck(j),th(j)
     *  ,hhs(j),j=1,n)
c    *  ,hhs1,j=1,n)
 88     format(6g16.6)
      stop
        end

        function fsx(xx)
        parameter (nn=1000)
      common/body/ xm(nn),ym(nn),x(nn),y(nn),fs(nn),fr(nn),zn(nn),n
        call intrp3(xx,x,fs,n,fsxx)
        fsx=fsxx
        return
        end
c
        function frx(xx)
        parameter (nn=1000)
      common/body/ xm(nn),ym(nn),x(nn),y(nn),fs(nn),fr(nn),zn(nn),n
        call intrp3(xx,x,fr,n,frxx)
        frx=frxx
        return
        end
c
        function fsxm(xx)
        parameter (nn=1000)
      common/body/ xm(nn),ym(nn),x(nn),y(nn),fs(nn),fr(nn),zn(nn),n
        call intrp3(xx,x,fs,n,fsxxm)
        fsxm=fsxxm
        return
        end
c
        function frxm(xx)
        parameter (nn=1000)
      common/body/ xm(nn),ym(nn),x(nn),y(nn),fs(nn),fr(nn),zn(nn),n
        call intrp3(xx,x,fr,n,frxx)
        frxm=frxx
        return
        end
c
        function fsy(yy)
        parameter (nn=1000)
      common/body/ xm(nn),ym(nn),x(nn),y(nn),fs(nn),fr(nn),zn(nn),n
        call intrp3(yy,y,fs,n,fsyy)
        fsy=fsyy
        return
        end
c
        function fry(yy)
        parameter (nn=1000)
```

```
      common/body/ xm(nn),ym(nn),x(nn),y(nn),fs(nn),fr(nn),zn(nn),n
        call intrp3(yy,y,fr,n,fryy)
        fry=fryy
        return
        end
c
        function fsz(zz)
        parameter (nn=1000)
      common/body/ xm(nn),ym(nn),x(nn),y(nn),fs(nn),fr(nn),zn(nn),n
        call intrp3(zz,zn,fs,n,fszz)
        fsz=fszz
        return
        end
c
        function frz(zz)
        parameter(nn=1000)
      common/body/ xm(nn),ym(nn),x(nn),y(nn),fs(nn),fr(nn),zn(nn),n
        call intrp3(zz,zn,fr,n,frzz)
        frz=frzz
        return
        end
c
      SUBROUTINE INTRP3 (XX,X,Y,NPNTS,YY)
C
C     SUBROUTINE INTRP3 SETS UP THE CALLING ARGUMENT FOR
C     SUBROUTINE INTER3
C
C     SUBROUTINE INTRP3 CALLS SUBROUTINE INTER3.
C
C     SUBROUTINE INTRP3 IS CALLED BY MAIN.
C
C     YY IS THE VALUE RETURNED FROM ARRAY Y
C     WHICH CORRESPONDS TO THE VALUE XX IN ARRAY X
C
      DIMENSION X(NPNTS), Y(NPNTS)
C
      DATA SMALLT / 1.0D-6 /
      FAC=1.0d0+SMALLT
      JC=0
10    JC=JC+1
      IF (XX.GT.X(JC)*FAC) GO TO 10
      IF (JC.LT.2) JC=2
      IF (JC.GT.(NPNTS-1)) JC=NPNTS-1
      CALL INTER3 (XX,X(JC-1),X(JC),X(JC+1),Y(JC-1),Y(JC),Y(JC+1),YY)
      RETURN
      END
      SUBROUTINE INTER3 (X,X1,X2,X3,F1,F2,F3,F)
C
C     SUBROUTINE INTER3 INTERPOLATES FOR THE VALUE F CORRESPONDING TO
C     POINT X USING 3 POINT LAGRANGIAN INTERPOLATION.
C
C     SUBROUTINE INTER3 IS CALLED BY SUBROUTINES INTRP3, HCP, AND HCPA.
C
C     ASSUMES X1 .LE. X .LE. X3.
C
C     WRITE(0,*) ' INTER3: ENTRY'
C
      AN1=(X-X2)*(X-X3)
      AN2=(X-X1)*(X-X3)
      AN3=(X-X1)*(X-X2)
      DN1=(X1-X2)*(X1-X3)
```

```
      DN2=(X2-X1)*(X2-X3)
      DN3=(X3-X1)*(X3-X2)
      CN1=AN1/DN1
      CN2=AN2/DN2
      CN3=AN3/DN3
      F=CN1*F1+CN2*F2+CN3*F3
C     WRITE(0,*) 'INTER3: RETURN'
      RETURN
      END
c integral by simpson's rule
c
        subroutine simp(func,a,b,s)
        parameter (eps=1.e-2,jmax=20)
        ost=-1.e30
        os=-1.e30
        do 11 j=1,jmax
          call trapzd(func,a,b,st,j)
          s=(4.*st-ost)/3.
          if(abs(s-os).lt.eps*abs(os)) return
          os=s
          ost=st
  11    continue
        pause 'too many steps'
        end
c
        subroutine trapzd(func,a,b,s,n)
        external func
        if(n.eq.1)then
          s=0.5*(b-a)*(func(a)+func(b))
          it=1
        else
          tnm=it
          del=(b-a)/tnm
          x=a+0.5*del
          sum=0.
          do 11 j=1,it
            sum=sum+func(x)
            x=x+del
  11    continue
        s=0.5*(s+(b-a)*sum/tnm)
        it=2*it
        endif
        return
        end
```

## Appendix B - Input data file for the modified Miner and Lewis axisymmetric viscous shock layer code, VIS3D

```
Z=70. km  Ro=0.5 m  Brykina  aoa=0 VINF=7.25 KM/SEC   Finite CAT Scott
 $input
 ALT=70.00,     ! For info and labeling only
 BRAD=1.640,    ! Nose radius in feet (in pitch plane)
 DS=.03,        ! Initial streamwise step size
 IEND=200,      ! Maximum number of streamwise steps
 SEND=3.0,      ! Final S value in nose radius
 RINF=1.14e-7, ! Free steam density, slugs/cuft
 TB=2600.,      ! Wall temperature, R
 TINF=360.,     ! Free stream temperature, R
 UINF=23786.,   ! Free stream velocity, ft/s
 XLE=1.,        ! Lewis number
 IGEOM=3,       ! Geometry flag. 3=geometry read in from unit 4
 NAN=1,NDATA=1,NS=6,NSI=6,   !  VSL standard inputs, eg. No. of species
 CAT=4,         ! Catalycity flag.  4 = compiled in value for Scott's RCG values
 DSN=1.,RAT=1.1,DSNN=.004,   !  Streamwise clustering parameters
 THINI=1.,      ! Shock layer option. 1=Thin Shock layer equations  0=Full VSL
 PRNTCI=0.,NTSH=20,XKETA=1.08,KEND=1,KTWAL=0, ! VSL standard input options
 ITSW=1,        ! Flag for Tw.  0=Const. input value.  1=Radiation equilibrium
 $end
           .233                                        .767 ! Free stream mass fractions
 .1        .133      0.        .1        0.        .677 ! Initial wall mass fracs.
```

## Appendix C - Sample output from VSL3D code Fortran Unit 23

| k | i | S | XB | RS | ck | HHS | eps | qw | tauw | cf | C(O) | C(O2) | C(NO) | C(N) | C(NO+) | C(N2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1.98E-02 | 50.5 | 0 | 0.00E+00 | 0.1798 | 4.46E-02 | 1.89E-02 | 1.37E-04 | 9.94E-11 | 0.7565 |
| 1 | 2 | 0.033 | 0.0005 | 0.033 | 1 | 1 | 1.98E-02 | 49.38 | 6.572 | 8.76E-04 | 0.1799 | 4.46E-02 | 1.88E-02 | 1.37E-04 | 9.92E-11 | 0.7565 |
| 1 | 3 | 0.0693 | 0.0024 | 0.0692 | 1 | 1 | 1.98E-02 | 49.38 | 13.97 | 1.86E-03 | 0.1784 | 4.61E-02 | 1.88E-02 | 1.29E-04 | 9.92E-11 | 0.7565 |
| 1 | 4 | 0.1092 | 0.006 | 0.109 | 1 | 1 | 1.98E-02 | 50.13 | 22.82 | 3.04E-03 | 0.1776 | 4.68E-02 | 1.90E-02 | 1.25E-04 | 9.93E-11 | 0.7565 |

## Appendix D - Sample output from INTPX code Fortran Unit 50

ns=26 nb=201 ni=201

| s(i) | xb(i) | rs(i) | x | y | z | r | zn | qw | tau | cf | cO | cO2 | cNO | cN | cNO+ | cN2 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 48.7 | 0 | 0 | 0.192 | 3.69E-02 | 1.06E-02 | 6.62E-02 | 9.83E-11 | 0.6943 |
| 0.033 | 0.0005 | 0.033 | 0 | 0.132 | 0.0022 | 0.132 | 0.0005 | 48.42 | 3.134 | 2.03E-03 | 0.1921 | 3.69E-02 | 1.06E-02 | 6.62E-02 | 9.81E-11 | 0.6943 |
| 0.0693 | 0.0024 | 0.0692 | 0 | 0.277 | 0.0096 | 0.277 | 0.0024 | 48.46 | 6.686 | 4.33E-03 | 0.192 | 3.69E-02 | 1.06E-02 | 6.61E-02 | 9.80E-11 | 0.6944 |
| 0.1092 | 0.0059 | 0.109 | 0 | 0.4359 | 0.0238 | 0.4359 | 0.0059 | 48.51 | 10.53 | 6.82E-03 | 0.1919 | 3.71E-02 | 1.06E-02 | 6.62E-02 | 9.82E-11 | 0.6943 |
| 0.1532 | 0.0116 | 0.1526 | 0 | 0.6104 | 0.0466 | 0.6104 | 0.0116 | 47.87 | 14.55 | 9.42E-03 | 0.192 | 3.69E-02 | 1.06E-02 | 6.64E-02 | 9.81E-11 | 0.6941 |
| 0.2015 | 0.02 | 0.2001 | 0 | 0.8007 | 0.0801 | 0.8007 | 0.02 | 46.81 | 18.67 | 1.21E-02 | 0.1926 | 3.64E-02 | 1.05E-02 | 6.67E-02 | 9.80E-11 | 0.6939 |
| 0.2546 | 0.0317 | 0.252 | 0 | 1.0078 | 0.127 | 1.0078 | 0.0317 | 45.32 | 22.79 | 1.48E-02 | 0.1937 | 3.55E-02 | 1.02E-02 | 6.73E-02 | 9.79E-11 | 0.6933 |
| 0.3131 | 0.0475 | 0.3083 | 0 | 1.2331 | 0.1901 | 1.2331 | 0.0475 | 43.42 | 26.73 | 1.73E-02 | 0.1952 | 3.41E-02 | 9.78E-03 | 6.84E-02 | 9.77E-11 | 0.6925 |
| 0.3774 | 0.0681 | 0.3692 | 0 | 1.4766 | 0.2726 | 1.4766 | 0.0681 | 41.1 | 30.34 | 1.97E-02 | 0.1972 | 3.42E-02 | 9.23E-03 | 7.01E-02 | 9.74E-11 | 0.6911 |
| 0.4481 | 0.0945 | 0.4348 | 0 | 1.7386 | 0.3778 | 1.7386 | 0.0945 | 38.41 | 33.43 | 2.17E-02 | 0.1998 | 3.02E-02 | 8.53E-03 | 7.24E-02 | 9.71E-11 | 0.6891 |

## Appendix E - UNIX run stream for computing a set of cases for a paraboloid.

```
parg
cp fort.31 f31.25-0-0
cgeomn
cp fort.40 fort.4
cp fort.40 f40.25-0-0
vsl3d<in-S1.6tre>ou.25-0-0s1.6tre
cp fort.23 f23.25-0-0s1.6tre
intpx
cp fort.50 f50.25-0-0s1.6tre
parg
cp fort.31 f31.25-0-45
cgeomn
cp fort.40 fort.4
cp fort.40 f40.25-0-45
vsl3d<in-S1.6tre>ou.25-0-45s1.6tre
cp fort.23 f23.25-0-45s1.6tre
intpx
cp fort.50 f50.25-0-45s1.6tre
parg
cp fort.31 f31.25-0-63.4
cgeomn
cp fort.40 fort.4
cp fort.40 f40.25-0-63.4
vsl3d<in-S1.6tre>ou.25-0-63.4s1.6tre
cp fort.23 f23.25-0-63.4s1.6tre
intpx
cp fort.50 f50.25-0-63.4s1.6tre
parg
cp fort.31 f31.25-0-76
cgeomn
cp fort.40 fort.4
cp fort.40 f40.25-0-76
vsl3d<in-S1.6tre>ou.25-0-76s1.6tre
cp fort.23 f23.25-0-76s1.6tre
intpx
cp fort.50 f50.25-0-76s1.6tre
parg
cp fort.31 f31.25-0-90
cgeomn
cp fort.40 fort.4
cp fort.40 f40.25-0-90
vsl3d<in-S1.6tre>ou.25-0-90s1.6tre
cp fort.23 f23.25-0-90s1.6tre
intpx
cp fort.50 f50.25-0-90s1.6tre
```

## Appendix F - Listing of the program PARG that generates geometric data for an elliptic paraboloid

```
C  Program 3D paraboloid geometry along meridional plane
c
      parameter (nn=1000)
      dimension y(nn),z(nn),fx(nn),fy(nn),fxx(nn),fyy(nn),fxy(nn)
      dimension x(nn)
c
      real*4 k
c  Input ratio of main curvatures at a stagnation point k
c  Input angle of attack alpha, degrees
c  Input angle fi, number of points n and step dx for meridional plane
c
      k=0.4
      alpha=30.
      fi=0.
      n=201
      write(6,*)' Input k,alpha,fi'
      read(5,*)k,alpha,fi
      write(6,*)' Input file number'
c     read(5,*)ifile
c     write(ifile,*)k,alpha,fi
      write(31,*)k,alpha,fi
        dx=0.04
      if(fi.gt.90.)dx=-dx
c  End of input
c
      alpha=alpha*1.5707963/90
      fi=fi*1.5707963/90
      if(abs(fi-1.5708).lt.0.001)go to 17
      a=tan(fi)
 17   ca=cos(alpha)
      sa=sin(alpha)
c
      write(6,*)'ca sa',ca,sa
c  Stagnation point
        x(1)=sa/ca
        y(1)=0
        z(1)=0.5*x(1)**2
        fx(1)=x(1)
        fy(1)=0
        fxy(1)=0
        fxx(1)=1
        fyy(1)=k
c
c  Meridional plane
        do 21 j=2,n
           cosfi=(sa*fx(j-1)+ca)/SQRT(1+fx(j-1)**2+fy(j-1)**2)
           if(cosfi.lt.0.01) go to 22
              if(abs(fi-1.5708).lt.0.001)then
                y(j)=y(j-1)+dx
                x(j)=x(1)
                else
                x(j)=x(j-1)+dx
                fx(j)=x(j)
                if(abs(a).lt.0.000001)then
                   y(j)=0.
                else
                 if(alpha.ne.0.)then
                   sss=(1-k*a*a*sa*(sa*x(j)**2
```

```
     *                 +2*ca*x(j)-2*sa-sa**3/ca**2))
c                  write(34,*)j,x(j),sss
                   if(sss.le.0) then
c                  write(6,*)' In parg:  sss is less than 0.'
                   go to 777
                   endif
                  y(j)=(1-SQRT(1-k*a*a*sa*(sa*x(j)**2+
     *            2*ca*x(j)-2*sa-sa**3/ca**2)))/(a*k*sa)
                 else
                  y(j)=a*x(j)
                 endif
               endif
             endif
      z(j)=0.5*(x(j)**2+k*y(j)**2)
      fxx(j)=1
      fy(j)=k*y(j)
      fyy(j)=k
      fxy(j)=0
      jj=j
 777              continue
  21  continue
  22  nfi=jj
c
c     write(ifile,32) (x(j),y(j),z(j),fx(j),fy(j),fxx(j),
      write(31,32) (x(j),y(j),z(j),fx(j),fy(j),fxx(j),
     *  fxy(j),fyy(j),j=1,nfi)
32      format(8g16.6)
        stop
        end
```

## Appendix G - Listing of program INTPX that interpolates geometrical data by streamwise distances calculated in the viscous shock layer code

```
c  Program INTPX
      dimension x(300),y(300),z(300),rn(300),rs(300),xb(300),s(300)
      dimension zn(300),sn(300),qw(300),ta(300),c(300,5),cf(300)
      character*8 a
c  Arrays x,y,z,zn,rn,sn have indices that correspond to the same input
c    body points.
c  Arrays s,xb,rs are output from VSL at the same body point, but not same as
c   for the input above.
c  This code interpolates the input arrays to find values corresponding to the
c   output points.
 1    format(15x,f10.0,2f15.0 )
      ni=0
      do 2 i=1,300
         read(4,3,end=999,err=888)zn(i),rn(i),sn(i),ck,th,hss
         ni=ni+1
 2    continue
 888  write(6,*)' In intpx Error detected in read of unit 4'
 999  continue
      read(31,*)k,alpha,fi
 3    format(8g16.6)
      nb=0
      do 22 i=1,300
         read(31,*,end=997,err=887)x(i),y(i),z(i),fx,fy,fxx,fxy,fyy
         nb=nb+1
 22   continue
 887  write(6,*)' In intpx Error detected in read of unit 31'
 997  continue
      ns=0
      read(23,10)a
 10   format( a8)
      do 4 i=1,300
       read(23,550,end=998,err=886)k,ii,s(i),xb(i),rs(i),ck,hhs,eps
     * ,qw(i),ta(i),cf(i),(c(i,j),j=1,6)
         ns=ns+1
 4    continue
 886  write(6,*)' In intpx Error detected in read of unit 23'
 998  continue
 550  format(2i4,3f10.4,13g12.4)
         write(50,*)' ns=',ns,'  nb=',nb,'   ni=',ni
         write(50,*)'s(i) xb(i) rs(i) x  y  z  r   zn  qw  tau  cf
     * cO  cO2  cNO  cN  cNO+  cN2  '
      if(ni.gt.nb)ni=nb
      do 20 i=1,ns
         call intrp3(s(i),sn,x,ni,xx)
         call intrp3(s(i),sn,y,ni,yy)
         call intrp3(s(i),sn,z,ni,zz)
         call intrp3(s(i),sn,r,ni,rr)
         call intrp3(s(i),sn,zn,ni,zzn)
         rr=sqrt(xx**2+yy**2)
         write(50,60)s(i),xb(i),rs(i),xx,yy,zz,rr,zzn,qw(i),ta(i),
     *   cf(i),  (c(i,j),j=1,6)
 20   continue
 60   format(3f10.4,5f10.4,9g12.4)
      stop
      end
c
      SUBROUTINE INTRP3 (XX,X,Y,NPNTS,YY)
```

```
C
C     SUBROUTINE INTRP3 SETS UP THE CALLING ARGUMENT FOR
C     SUBROUTINE INTER3
C
C     SUBROUTINE INTRP3 CALLS SUBROUTINE INTER3.
C
C     SUBROUTINE INTRP3 IS CALLED BY MAIN.
C
C     YY IS THE VALUE RETURNED FROM ARRAY Y
C     WHICH CORRESPONDS TO THE VALUE XX IN ARRAY X
C
C
      DIMENSION X(NPNTS), Y(NPNTS)
C
      DATA SMALLT / 1.0D-6 /
      FAC=1.0d0+SMALLT
      JC=0
10    JC=JC+1
      IF (XX.GT.X(JC)*FAC) GO TO 10
      IF (JC.LT.2) JC=2
      IF (JC.GT.(NPNTS-1)) JC=NPNTS-1
      CALL INTER3 (XX,X(JC-1),X(JC),X(JC+1),Y(JC-1),Y(JC),Y(JC+1),YY)
      RETURN
      END
      SUBROUTINE INTER3 (X,X1,X2,X3,F1,F2,F3,F)
C
C     SUBROUTINE INTER3 INTERPOLATES FOR THE VALUE F CORRESPONDING TO
C     POINT X USING 3 POINT LAGRANGIAN INTERPOLATION.
C
C     SUBROUTINE INTER3 IS CALLED BY SUBROUTINES INTRP3, HCP, AND HCPA.
C
C     ASSUMES X1 .LE. X .LE. X3.
C
C     WRITE(0,*) ' INTER3: ENTRY'
C
      AN1=(X-X2)*(X-X3)
      AN2=(X-X1)*(X-X3)
      AN3=(X-X1)*(X-X2)
      DN1=(X1-X2)*(X1-X3)
      DN2=(X2-X1)*(X2-X3)
      DN3=(X3-X1)*(X3-X2)
      CN1=AN1/DN1
      CN2=AN2/DN2
      CN3=AN3/DN3
      F=CN1*F1+CN2*F2+CN3*F3
C     WRITE(0,*) 'INTER3: RETURN'
      RETURN
      END
```

## REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

| 1. AGENCY USE ONLY *(Leave Blank)* | 2. REPORT DATE | 3. REPORT TYPE AND DATES COVERED |
|---|---|---|
| | May 1998 | NASA Technical Memorandum |

**4. TITLE AND SUBTITLE**
An Approximate Axisymmetric Viscous Shock Layer Aeroheating Method for Three-Dimensional Bodies

**5. FUNDING NUMBERS**

**6. AUTHOR(S)**
Irina G. Brykina* and Carl D. Scott

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**
Lyndon B. Johnson Space Center
Houston, Texas 77058

**8. PERFORMING ORGANIZATION REPORT NUMBERS**
S-840

**9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**
National Aeronautics and Space Administration
Washington, DC 20546-0001

**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**
TM-1998-207890

**11. SUPPLEMENTARY NOTES**

**12a. DISTRIBUTION/AVAILABILITY STATEMENT**
* Unclassified/Unlimited
Available from the NASA Center for Aerospace Information (CASI)
7121 Standard
Hanover, MD 21076-1320

**12b. DISTRIBUTION CODE**

**13. ABSTRACT** *(Maximum 200 words)*

A technique is implemented for computing hypersonic aeroheating, shear stress, and other flow properties on the windward side of a three-dimensional (3D) blunt body. The technique uses a 2D/axisymmetric flow solver modified by scale factors for a corresponding equivalent axisymmetric body. Examples are given in which a 2D solver is used to calculate the flow at selected meridional planes on elliptic paraboloids in reentry flight. The report describes the equations and the codes used to convert the body surface parameters into input used to scale the 2D viscous shock layer equations in the axisymmetric viscous shock layer code. Very good agreement is obtained with solutions to finite rate chemistry 3D thin viscous shock layer equations for a finite rate catalytic body.

| 14. SUBJECT TERMS | 15. NUMBER OF PAGES | 16. PRICE CODE |
|---|---|---|
| flow measurement, flow, axisymmetric flow, parabolic bodies, shock layers, viscous flow, heat flux | 37 | |

| 17. SECURITY CLASSIFICATION OF REPORT | 18. SECURITY CLASSIFICATION OF THIS PAGE | 19. SECURITY CLASSIFICATION OF ABSTRACT | 20. LIMITATION OF ABSTRACT |
|---|---|---|---|
| Unclassified | Unclassified | Unclassified | Unlimited |

Standard Form 298 (Rev Feb 89) (MS Word Mar 97) NSN 7540-01-280-5500
Prescribed by ANSI Std. 239-18
298-102