Title
A SOFTWARE PACKAGE FOR CALCULATING AXISYMMETRIC MENISCI

Author
Concus, P.

Publication Date
1978-12-01

To be submitted for publication

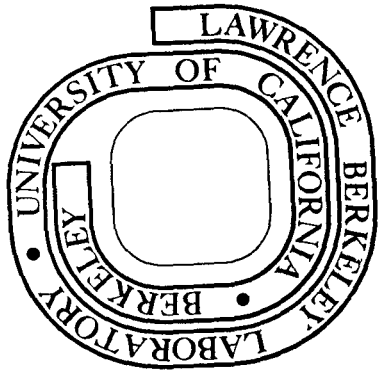
LBL-8700 c.2
Preprint

A SOFTWARE PACKAGE FOR CALCULATING AXISYMMETRIC MENISCI

P. Concus and V. Pereyra

December 1978

Prepared for the U. S. Department of Energy
under Contract W-7405-ENG-48



DISCLAIMER

LBL-8700

A SOFTWARE PACKAGE FOR CALCULATING AXISYMMETRIC MENISCI

P. Concus[†] and V. Pereyra[††]

December, 1978

[†]Lawrence Berkeley Laboratory, University of California,
Berkeley, California 94720

[††]Escuela de Computación, Universidad Central de Venezuela,
Caracas, Venezuela

ABSTRACT

We consider the problem of determining equilibrium menisci in partly-filled rotationally-symmetric containers oriented with axis of symmetry parallel to the gravitational field. The problem is formulated mathematically as a two-point boundary value problem for a set of first order nonlinear ordinary differential equations, whose solution gives a parametric representation of the free surface. Specific formulations are given for the case of a right circular cylindrical container, with either a flat or spheroidal bottom, and for the case of a spheroidal container. The software package CAPIL, presented in the Appendix, obtains numerical approximations to the free surfaces for these cases.

A SOFTWARE PACKAGE FOR CALCULATING AXISYMMETRIC MENISCI

P. Concus and V. Pereyra

1. Introduction

The problem we consider is that of determining a numerical approximation to an equilibrium free surface (meniscus) of a liquid that partly fills a rotationally symmetric container under the action of surface and gravitational forces. The axis of symmetry is assumed oriented parallel to the direction of gravity, and a meniscus is to be determined given the values of the liquid volume, dimensionless gravity/surface tension parameter (Bond number), and liquid-container contact angle.

If the height of the free surface can be expressed as a single-valued function of radius, then the height satisfies a second-order nonlinear ordinary differential equation subject in general to two-point, possibly nonlinear, boundary conditions. (See, for example, [1,2]† and the references therein.) The equation involves a parameter, which is related to the mean curvature of the free surface at the center of symmetry, whose value in general is unknown. Also, the equation must be solved in an interval whose endpoints may not be known explicitly.

In order to represent solutions that may be multiple-valued, a standard parametric form of the equation consisting of a set of three first-order equations is used [2]. This parametrization employs meridional arc-length as the independent variable. By introducing three new dependent variables, we are able to rewrite the problem as a system of six first-order equations without unknown parameters and for which all the constraints can be expressed

---

†References are listed in the Appendix.

as two-point boundary conditions at explicitly prescribed endpoints.

The basic part of our software package for solving this problem numerically is the subroutine PASVA2 and its auxiliary subroutines U2DCGS, CØEGEN, NEWTØN, SYSLIN, DECØMP, and SØLVE. PASVA2 is a general purpose nonlinear two-point boundary value problem solver for first-order systems [3], a description of which is given in the Appendix.

Our software package CAPIL, which is described in the Appendix, consists of a set of driver subroutines, and other subroutines required by PASVA2, for each of the containers of interest. The supplied driver subroutines permit differing types of user interaction. In the simplest mode of operation the user has only to call an appropriate subroutine corresponding to the desired container shape and to supply a few basic parameters. The program will respond with a table of the computed free surface, or with an error message in case of failure. A small main program is provided for each driver subroutine to read the parameters from data cards.

Other options allow the more sophisticated user to have closer control of the computation by permitting choice of some of the input to PASVA2, such as the number of points in the initial mesh, the desired accuracy, the initial mesh distribution, and the initial approximation to the solution. These options may be useful in difficult situations for which the simplest option is inadequate, or for situations in which a sequence of related calculations are to be carried out. In the latter situation the output of one problem can be used to start the next, often at a savings in computational cost.

2. Mathematical formulation

We consider a rotationally symmetric container partly filled with a liquid that is in equilibrium with respect to gravitational and surface forces. The axis of symmetry is assumed to be oriented parallel to the direction of gravity.

If the height u of the free surface of the liquid (measured positively upward from a reference plane) can be represented by a single valued function of the polar radius r, then it satisfies the second-order ordinary differential equation

$$\frac{1}{r}\frac{d}{dr}\left[\frac{r\,\frac{du}{dr}}{\left[1+\left(\frac{du}{dr}\right)^2\right]^{1/2}}\right] - Bu = 2H \tag{1}$$

subject to appropriate boundary conditions. In (1), all quantities are nondimensional; $B = \rho g \ell^2/\sigma$ is the Bond number, with ρ the difference in densities between the liquid and the gas adjacent to the liquid, g the vertical acceleration field (positive downward), σ the interfacial surface tension, and ℓ the characteristic length with respect to which physical lengths are made nondimensional. (The physical lengths are ℓr and ℓu -- for many problems it may be convenient simply to choose ℓ equal to unity.) The (unknown) parameter H is the (nondimensional) mean curvature of the surface at points where $u = 0$ and is related to the pressure difference across the free surface.

In general, for the problems that we shall consider, the surface cannot be represented by a single-value function $u(r)$. Thus we introduce

a parametric representation in terms of (nondimensional) arclength s along a meridian of the free surface measured from the symmetry axis $r = 0$, or from the intersection of the surface with the container if the liquid does not intersect the axis. Let ψ denote the angle between a tangent to the free surface meridian and the r-axis, measured counterclockwise from the positively directed r-axis to the tangent line,

$$\tan \psi = du/dr$$

(see Fig. 1). Then we obtain, corresponding to (1), the equations

$$\frac{du}{ds} = \sin \psi$$

$$\frac{dr}{ds} = \cos \psi \tag{2}$$

$$\frac{d\psi}{ds} = Bu + 2H - \frac{\sin \psi}{r} \quad .$$

The system (2) is to be solved subject to endpoint conditions on r, u, and ψ and the constraint that the contained liquid have a prescribed volume. These end conditions and volume constraint will determine, uniquely in many cases of interest, the desired solution of (2). For some cases, particularly those for which a small volume is specified, there may be two or more physically meaningful solutions. If B is negative, a solution may be physically unstable with respect to small perturbations.

A further transformation is required to place (2) and the end conditions in a form suitable for PASVA2; PASVA2 does not permit the appearance of unknown parameters, such as H that are not also dependent variables, nor an interval length that is unknown -- in this case S, the value of the

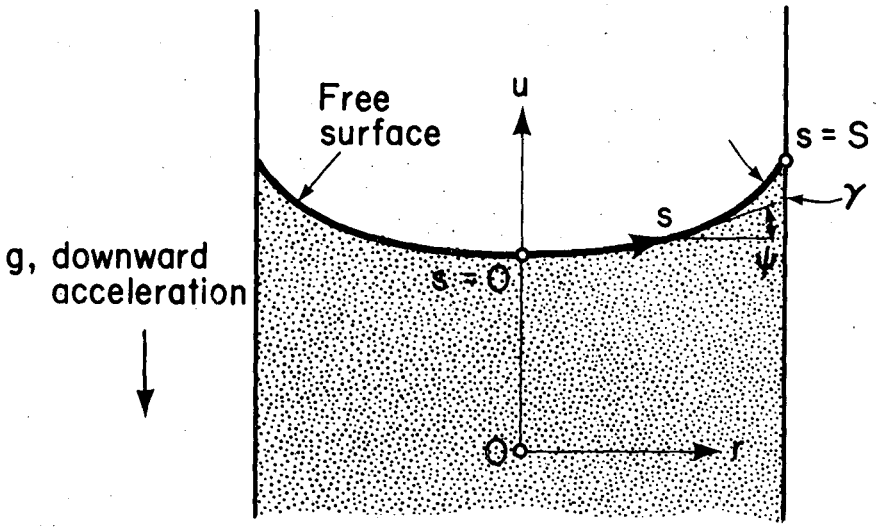


Figure 1

Vertical section of container and equilibrium free surface.

(nondimensional) arclength of the solution at the right-hand endpoint. We append to (2) the equations

$$\frac{dS}{ds} = 0 , \qquad \frac{dH}{ds} = 0 , \tag{3}$$

and make the change of variable

$$t = s/S .$$

Then the system becomes

$$\begin{aligned} \frac{du}{dt} &= S \sin \psi \\ \frac{dr}{dt} &= S \cos \psi \\ \frac{d\psi}{dt} &= S[Bu + 2H - \frac{\sin \psi}{r}] \\ \frac{dS}{dt} &= 0 \\ \frac{dH}{dt} &= 0 , \end{aligned} \tag{4}$$

which is in a form suitable for PASVA2.

If the initial point $t = 0$ is on the symmetry axis $r = 0$, then the regularity condition

$$\lim_{r \to 0} \frac{\sin \psi}{r} = \tfrac{1}{2} Bu + H$$

can be used there to evaluate the right-hand side of (4).

The endpoint conditions at $t = 0$ and $t = 1$ and the volume constraint are discussed in § 3 for each of several containers. In general there result nonlinear, coupled boundary conditions, of the type that PASVA2 has been designed to handle.

To use PASVA2, one must prepare subroutines describing the right-hand side of (4), the boundary conditions, and the Jacobian matrix of partial derivatives, with respect to the dependent variables, of the right-hand side and of the boundary conditions. A driver program is required also, which sets parameters, calculates initial approximations, and prepares the output of the results. Several specific cases, for which we have prepared the necessary auxiliary programs, are discussed in § 3 and in the Appendix.

3. Specific cases

3.1(a) Right circular cylinder - horizontal planar bottom - large liquid volume

For this case we denote the (nondimensional) radius of the cylinder by a and take the bottom to be $u \equiv 0$ (Fig. 2a)†. It is assumed that there is sufficient liquid to cover the bottom entirely.

Let the specified contact angle between the free surface and the cylinder wall (measured interior to the liquid) be γ and the specified (nondimensional) volume of liquid be V. Then the appropriate end-point conditions on (4) are

---

†Figures 2, 3, and 4 can be found in the Appendix.

(5) $$r(0) = \psi(0) = 0 ,$$

(6) $$r(1) = a , \quad \psi(1) = \pi/2 - \gamma ,$$

and that the liquid volume be V. The volume condition could be expressed in this case as a single boundary condition at $t = 1$. Instead, we enforce the volume constraint in the same manner as will be convenient for the cases of containers with curved bottoms, by introducing an auxiliary differential equation for the volume along with two end-point conditions. Let v denote the (nondimensional) volume of liquid, as a function of t, that is contained between the free surface and the horizontal planes passing through u(0) and u(t). Then v(t) satisfies

(7) $$\frac{dv}{dt} = \pi S(a^2 - r^2) \sin \psi$$

with

(8) $$v(0) = 0$$

and

(9) $$v(1) + \pi a^2 u(0) = V .$$

The problem to be solved is thus (4) and (7) in the interval $0 < t < 1$ with end-point conditions (5), (6), (8), and (9).

3.1(b) Right circular cylinder - horizontal planar bottom - small liquid volume

This case is the same as § 3.1(a) except that there is not sufficient liquid to cover the bottom entirely. The liquid is assumed to form an annular ring intersecting the bottom and walls as shown in Fig. 2b. At the initial point the free surface meets the container bottom with the specified contact angle γ. Other cases, such as one for which the liquid forms a sessile drop intersecting the symmetry axis and the container bottom, but not the walls, can be treated similarly but are not considered here.

We express the volume in terms of vertical rather than horizontal slices. This form is not essential here, but will be convenient for the small volume case for containers with curved bottoms, to be discussed later. Let v be the volume of liquid contained between the free surface, the container bottom $u = 0$, and the vertical plane $r = r(t)$. Then for the configuration shown in Fig. 2b v satisfies

$$\frac{dv}{dt} = 2\pi S r u \cos\psi \tag{7'}$$

with (8) and

$$v(1) = V . \tag{9'}$$

The other boundary conditions are

$$u(0) = 0 , \quad \psi(0) = \gamma \tag{5'}$$

and (6). The problem to be solved is thus (4) and (7') in the interval $0 < t < 1$ with end-point conditions (5'), (6), (8), and (9').

3.2(a) Right circular cylinder - spheroidal bottom - large liquid volume

We consider next the case of a right circular cylindrical container with a bottom that is a portion of the upper surface of a spheroid (ellipsoid of revolution). A vertical section of the configuration for which there is sufficient liquid to cover the bottom entirely is depicted in Fig. 3a. The (nondimensional) horizontal and vertical semi-axes of the ellipse forming the bottom are α and β, respectively, with a meridian $u_B(r)$ of the bottom given by

$$u_B(r) = u_C + \beta\left(1 - \frac{r^2}{\alpha^2}\right)^{\frac{1}{2}} .$$

The center of the ellipse is at $(0,u_C)$,

$$u_C = -\beta\left(1 - \frac{a^2}{\alpha^2}\right)^{\frac{1}{2}} ,$$

corresponding to the condition that the bottom passes through $(a,0)$.

The equations describing this case are the same as those for the flat-bottomed cylinder § 3.1(a) except for the right-hand end condition on the volume. One has, in place of (9'),

$$v(1) + \pi a^2 u(0) = V + \pi\left[a^2 u_C + \frac{2\alpha^2}{3\beta^2}\left(u_C^3 + \beta^3\right)\right] . \tag{9''}$$

The problem to be solved is thus (4) and (7) with end-point conditions (5), (6), (8), and (9").

3.2(b) Right circular cylinder - spheroidal bottom - small liquid volume

This case is the same as § 3.2(a) except that there is not sufficient liquid to cover the bottom entirely. The specific configuration considered is the one depicted in Fig. 3b, for which the liquid is assumed to form an annular ring intersecting the bottom and walls. At the initial point the free surface meets the container bottom with the specified contact angle γ, a condition expressed by

$$u(0) - u_B[r(0)] = 0 ,$$

(5")
$$\psi(0) - \tan^{-1}\left[\frac{-\beta^2 r(0)}{\alpha^2\{u_B[r(0)] - u_C\}}\right] = \gamma .$$

As for the flat bottom - small volume case, let v denote the volume of liquid contained between the free surface, the container bottom, and the vertical plane $r = r(t)$. Then

(7")
$$\frac{dv}{dt} = 2\pi S r[u - u_B(r)]\cos\psi .$$

The problem to be solved is thus (4) and (7") with end-point conditions (5"), (6), (8), and (9').

3.3(a) Spheroidal container - large liquid volume

For this case the container is a spheroid (ellipsoid of revolution) with vertical axis of symmetry, and the liquid covers the center $(r = 0)$ of the tank bottom, as depicted in Fig. 4a. The horizontal and vertical

(nondimensional) semi-axes are α and β, respectively, and the origin is chosen to be the tank bottom, so that a point (r_T, u_T) on the tank satisfies

$$\frac{(u_T - \beta)^2}{\beta^2} + \frac{r_T^2}{\alpha^2} = 1 .$$

The equations describing this case are the same as for the cylinder, but with the volume equation given by

$$\frac{dv}{dt} = \pi S \left[\frac{\alpha^2}{\beta^2} u(2\beta - u) - r^2 \right] \sin \psi , \tag{7'''}$$

where $v(t)$ is the volume of liquid between the planes $u = u(0)$ and $u = u(t)$. The right-hand end-point conditions are

$$\begin{cases} \tan^{-1}\left[-\dfrac{\beta^2}{\alpha^2} \dfrac{r(1)}{u(1)-\beta} \right] - \psi(1) = \gamma , & \text{for } u(1) \neq \beta \\ \dfrac{\pi}{2} - \psi(1) = \gamma , & \text{for } u(1) = \beta \end{cases} \tag{6'}$$

$$\left[\frac{u(1) - \beta}{\beta} \right]^2 + \left[\frac{r(1)}{\alpha} \right]^2 = 1 ,$$

where the value of the arc tangent between 0 and π is to be used, and

$$v(1) + \frac{\pi}{3} \left[\frac{\alpha u(0)}{\beta} \right]^2 [3\beta - u(0)] = V . \tag{9'''}$$

The problem to be solved is thus (4) and (7''') in the interval $0 < t < 1$ with end-point conditions (5), (6'), (8), and (9''').

3.3(b) Spheroidal container - small liquid volume

This case is the same as § 3.3(a) except that the liquid does not cover the center of the container, either on the top or the bottom, as depicted in Fig. 4b. Note that for the spheroidal container the "large" and "small" volume cases do not necessarily refer to the actual amount of liquid present, but to the position of the liquid in the container. Obviously, a large amount of contained liquid would tend to correspond to case (a), but for some conditions solutions can exist both for cases (a) and (b), and possibly even for other configurations as well.

For the small-volume case depicted in Fig. 4b we express the liquid volume in terms of horizontal slices in the same manner as for the large volume case § 3.3(a). The condition that the free surface meet the container bottom with contact angle γ at the initial point is given by

$$(5''') \qquad \begin{cases} \tan^{-1}\left[-\dfrac{\beta^2}{\alpha^2} \dfrac{r(0)}{u(0)-\beta} \right] - \psi(0) = -\gamma , & \text{for } u(0) \neq \beta \\ \dfrac{\pi}{2} - \psi(0) = -\gamma , & \text{for } u(0) = \beta \end{cases}$$

$$\left[\frac{u(0)-\beta}{\beta} \right]^2 + \left[\frac{r(0)}{\alpha} \right]^2 = 1 ,$$

where the arc tangent takes on values between 0 and π.

Thus the problem to be solved is (4) and (7''') with end-point conditions (5'''), (6'), (8), and (9').

4. Computer program

The software package CAPIL for obtaining numerical solutions of the cases discussed in § 3 is described in a self-contained manner in the Appendix. We remark that even though Figs. 1-4 depict the case of wetting liquids $0 \leqslant \gamma \leqslant \frac{\pi}{2}$, the formulas and the computer programs generally are valid for the non-wetting case $\frac{\pi}{2} \leqslant \gamma \leqslant \pi$ as well.

5. Acknowledgments

We wish to thank the LBL applied mathematics group, essentially all of whom, particularly N. Albright and R. Kung, at one time or another helped substantially in the task of getting CAPIL into functioning form. P. Symons, program manager of the NASA-supported portion of this work, was helpful with his encouragement and suggestions of some of the utilitarian features that the program should possess.

This work was supported in part by the NASA Lewis Research Center under contract 67705-C and by the Engineering, Mathematical, and Geosciences Division of the Department of Energy under contract W-7405-ENG-48.

APPENDIX

USER'S GUIDE TO CAPIL
A SOFTWARE PACKAGE FOR CALCULATING AXISYMMETRIC MENISCI

Date: December 1978

Language: ANSI FORTRAN

Precision: Single (double available also)

ABSTRACT

CAPIL is a software package for calculating equilibrium capillary free surfaces (menisci) in partly-filled, rotationally-symmetric vertical containers. At its core is the general finite-difference program PASVA2 for solving nonlinear first order systems of ordinary differential equations subject to two-point boundary conditions.

CAPIL contains drivers and other auxiliary subroutines required by PASVA2 for certain selected container configurations. Presently, subroutines are included for (i) the case of a right circular cylindrical container, without top, and either a flat or spheroidal bottom (subroutines CYLIND and CYLCUR, respectively), and (ii) the case of a spheroidal container (subroutine ELLIPS).

TABLE OF CONTENTS

CYLIND: RIGHT CIRCULAR CYLINDRICAL CONTAINER, HORIZONTAL PLANAR BOTTOM

The container is a right circular cylinder of RADIUS a, no top, and a horizontal planar bottom. The user must specify the RADIUS, the contact ANGLE γ between the liquid and the wall, the BØND number B, and the VØLUME of liquid V. On output, a table of the coordinates (r,u) describing the curve that generates the rotationally symmetric free surface is given. Also the value of arclength s along the curve and the angle ψ (in degrees) that the curve makes with the r-axis are given at each tabular point. The value of H, a parameter related to the mean curvature of the free surface (see § 2), is given also. The origin is assumed to be at the bottom of the container, the axis of symmetry being taken as the u-axis (see Fig. 2).

The Bond number is given by $B = \rho g \ell^2/\sigma$, where ρ is the difference in densities between the liquid and the gas adjacent to the liquid, g is the vertical acceleration field (positive downward), and σ is the interfacial surface tension. The quantity ℓ is the reference characteristic length chosen by the user for calculating the Bond number. If ℓ is chosen to be different from unity, then all other lengths must be made consistently nondimensional with respect to ℓ as well. That is, the physical radius of the cylinder is ℓa, the physical coordinates ℓr and ℓu, the physical arclength ℓs, the physical volume $\ell^3 V$, and the physical mean-curvature parameter H/ℓ. All quantities such as a, r, u, s, V, and H input to and output from CYLIND are assumed to be nondimensional.

We distinguish two cases: large volume of liquid and small volume

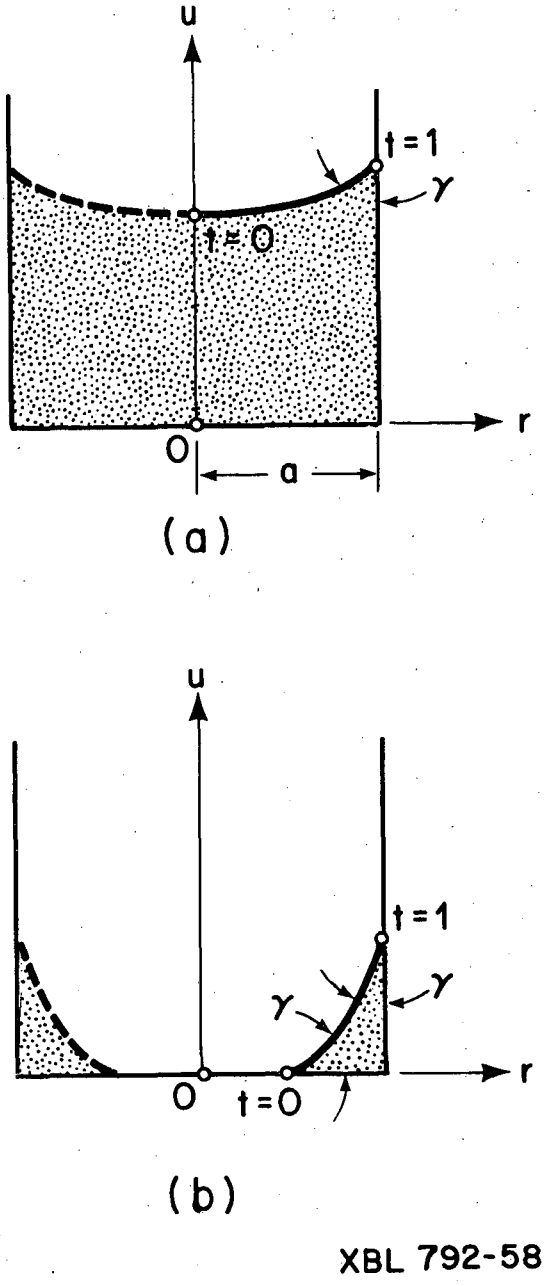


Figure 2

Right circular cylinder - horizontal planar bottom. (a) large volume case. (b) small volume case. The solid portion of the free-surface meridian depicts the computed arc (r(t),u(t)); the dashed portion is the rotationally symmetric image.

of liquid, the important factor being whether or not the liquid covers the bottom completely. The large liquid volume case is formulated mathematically in § 3.1(a) and depicted in Fig. 2a. The small liquid volume case is formulated in § 3.1(b) and depicted in Fig. 2b. Note that the small liquid volume case assumes that the central portion of the bottom is the portion not covered by liquid. Other possible configurations are not included in the program.

The simplest way for the user to obtain an approximation to the parametric representation $[r(t),u(t)]$ of a free surface meridian, with parameter $t = s/S$, where S is the total arclength of the curve, on a mesh

$$0 = t_1 < t_2 < \ldots < t_N = 1 \tag{10}$$

is by using the program MNCLND contained in the package. MNCLND is a small main program that reads input data from cards and calls CYLIND. The input data cards are the following:

Data Card 1 (FORMAT 3E8.0): PAR(1),PAR(3),PAR(4)

Data Card 2 (FORMAT 2I4,5E8.0): II,N,BØND,VØLUME,ANGLE
RADIUS,TØL .

PAR is a control parameter array of dimension 6 used by CYLIND. First we discuss the easy-to-use, automatic option.

PAR(1) = 1.-- Easy-to-use option

If PAR(1) = 1., then PAR(3), PAR(4), N, and TØL need not be specified by the user; they are set internally to default values by CYLIND. The following must be specified by the user.

II

Indicates the volume option to be attempted first.

= 1 Large liquid volume case (Fig. 2a) is attempted first.

= 2 Small liquid volume case (Fig. 2b) is attempted first.

If the program fails to obtain a solution for the indicated volume option, it will switch automatically to the other option, print a warning message, and try again. If the large volume option is specified and a solution is obtained but lies in part outside the container, then the small volume option will be tried also.

= 999 A flag to indicate that no more problems are to be solved and that the program should terminate by means of a STØP command.

BØND

The Bond number, positive, zero, or negative. For certain negative values of the Bond number, the solution and/or the numerical iteration procedure may be unstable.

VØLUME

The prescribed liquid volume V (nondimensional, consistent with BØND).

ANGLE

The contact angle γ in degrees, $0. \leqslant$ ANGLE $\leqslant 180.$.

RADIUS

The cylinder radius a (nondimensional, consistent with BØND).

If it is desired to solve more than one problem for a given Data Card 1, then the user need supply only the desired number of Data Cards 2 following the single Data Card 1, followed finally by an "II = 999" card to terminate the program.

For the input cards

columns	1-8	9-16	17-24	
quantity	PAR(1)			
value	1.0	blank	blank	Data Card 1

columns	1-4	5-8	9-16	17-24	25-32	33-40	41-48	
quantity	II		BØND	VØLUME	ANGLE	RADIUS		
value	1	blank	1.	25.	0.	2.	blank	Data Card 2

columns	1-4		
quantity	II		
value	999	blank	Data Card 2 (termination card)

the following output from CYLIND was obtained on a CDC 7600.

```
**********************************************************
*                                                        *
*   MENISCUS COMPUTATION FOR A RIGHT CIRCULAR CYLINDER   *
*   WITH A FLAT BOTTOM.                                  *
*                                                        *
*   CYLINDER RADIUS          =   .200000E+01             *
*                                                        *
*   (LARGE) LIQUID VOLUME    =   .250000E+02             *
*                                                        *
*   CONTACT ANGLE            =  0.      DEGREES          *
*                                                        *
*   BOND NUMBER              =   .100000E+01             *
*                                                        *
**********************************************************
```

S	R	U	PSI
0.	0.	.157679E+01	0.
.259632E+00	.259368E+00	.158672E+01	4.4032
.519264E+00	.517167E+00	.161696E+01	9.0315
.778896E+00	.771486E+00	.166882E+01	14.1190
.103853E+01	.101974E+01	.174448E+01	19.9239
.129816E+01	.125819E+01	.184685E+01	26.7409
.155779E+01	.148127E+01	.197929E+01	34.9115
.181742E+01	.168078E+01	.214499E+01	44.8345
.207706E+01	.184493E+01	.234558E+01	56.9712
.233669E+01	.195775E+01	.257868E+01	71.8408
.259632E+01	.200000E+01	.283364E+01	90.0000
MAXIMUM ABSOLUTE ERRORS	.89E-04	.27E-03	.33E-04

H = -.494774E+00 ERROR ≤ .58E-04

Note that estimates of the maximum absolute errors in r, u, and ψ are given and also an estimate of the error in H. The column marked s gives the values of arclength $s_i = St_i$, where S is the total arc-length of the computed curve.

In the following example, the large-volume option is specified, but too small a value of V is deliberately given as input. This is a non-wetting case with contact angle 180°.

For the input cards

columns	1-8	9-16	17-24	
quantity	PAR(1)			
value	1.0	blank	blank	Data Card 1

columns	1-4	5-8	9-16	17-24	25-32	33-40	41-48	
quantity	II		BØND	VØLUME	ANGLE	RADIUS		
value	1	blank	1.	3.	180.	2.	blank	Data Card 2

+ termination card

the following output was obtained.

```
**********************************************************
*                                                        *
*   MENISCUS COMPUTATION FOR A RIGHT CIRCULAR CYLINDER   *
*   WITH A FLAT BOTTOM.                                  *
*                                                        *
*   CYLINDER RADIUS          =   .200000E+01             *
*                                                        *
*   (LARGE) LIQUID VOLUME    =   .300000E+01             *
*                                                        *
*   CONTACT ANGLE           =180.000   DEGREES           *
*                                                        *
*   BOND NUMBER              =   .100000E+01             *
*                                                        *
**********************************************************
```

S	R	U	PSI
0.	0.	.651378E+00	0.
.259632E+00	.259368E+00	.641446E+00	-4.4032
.519264E+00	.517167E+00	.611213E+00	-9.0315
.778896E+00	.771486E+00	.559351E+00	-14.1190
.103853E+01	.101974E+01	.483692E+00	-19.9239
.129816E+01	.125819E+01	.381324E+00	-26.7409
.155779E+01	.148127E+01	.248878E+00	-34.9115
.181742E+01	.168078E+01	.831775E-01	-44.8345
.207706E+01	.184493E+01	-.117415E+00	-56.9712
.233669E+01	.195775E+01	-.350512E+00	-71.8408
.259632E+01	.200000E+01	-.605470E+00	-90.0000
MAXIMUM ABSOLUTE ERRORS	.89E-04	.27E-03	.33E-04

H = -.619311E+00 ERROR ≤ .58E-04

**** PART OF THE ABOVE SOLUTION SURFACE
LIES OUTSIDE THE CONTAINER.
SMALL VOLUME OPTION WILL BE TRIED****

S	R	U	PSI
0.	.160900E+01	0.	180.0000
.141996E+00	.147262E+01	.356938E-01	150.2321
.283992E+00	.137483E+01	.135712E+00	118.3168
.425988E+00	.134583E+01	.272391E+00	85.6693
.567984E+00	.139389E+01	.403856E+00	54.3930
.709979E+00	.150149E+01	.494143E+00	25.9540
.851975E+00	.163870E+01	.525907E+00	.4398
.993971E+00	.177698E+01	.498098E+00	-22.9592
.113597E+01	.189377E+01	.418863E+00	-45.2560
.127796E+01	.197209E+01	.301452E+00	-67.3554
.141996E+01	.200000E+01	.163138E+00	-90.0000
MAXIMUM ABSOLUTE ERRORS	.46E-03	.46E-03	.41E-03

H = -.175286E+01 ERROR ≤ .89E-03

Note that although a solution was obtained for II = 1, it was found to lie partly below the container bottom. Thus the II = 2 option was tried automatically, and a satisfactory solution was found.

Next the option is discussed of giving the user control over some of the parameters (input and output) of CYLIND and PASVA2.

PAR(1) = 2. -- User control option

If PAR(1) = 2., then the user must specify values for all of the variables on the input cards. The remaining variables are

N

On input - the number of points in the initial mesh. $N \leq 50$. On output from CYLIND - the number of points in the final mesh. Since mesh refinement may occur within PASVA2, this number may be larger than the number of initial points. The default value of N for PAR(1) = 1. is N = 11, unless $|B| \geq 24$, in which case N = min(40,NB), where NB is the largest integer in $|6B|^{\frac{1}{2}}$.

TØL

The desired error tolerance. The program attempts to compute solutions with maximum absolute error less than TØL at all grid points. The default value of TØL for PAR(1) = 1. is TØL = .005.

PAR(3)

Controls output of intermediate results. These may be useful in troublesome cases.

= 0. Intermediate results are not printed.

= 1. Intermediate results are printed.

PAR(4)

Controls whether default values or user-supplied initial values should be used for the initial mesh $\{t_i\}$ and initial approximations to u, r, ψ, S, H, and v.

= 0. Default values are provided automatically by CYLIND. These are that $\{t_i\}$ are the N uniformly spaced points on [0,1]. For II = 1 the initial approximation is

$$u(t_i) \equiv V/(\pi a^2) , \qquad r(t_i) = a\,t_i ,$$

$$\psi(t_i) = (\frac{\pi}{2} - \gamma)t_i ; \qquad S(t_i) \equiv a ,$$

$$H(t_i) \equiv -\frac{1}{2} Bu(0) , \qquad v(t_i) = \pi a^2 t_i^2 u(0) ;$$

for II = 2 it is

$$u(t_i) = \bar{u}\left(2\,\frac{r(t_i)}{a} - 1\right) , \qquad r(t_i) = \frac{1}{2}\,a(1+t_i) ,$$

$$\psi(t_i) = \gamma + (\frac{\pi}{2} - 2\gamma)\,t_i , \qquad S(t_i) \equiv (\bar{u}^2 + \frac{1}{4}\,a^2)^{\frac{1}{2}} ,$$

$$H(t_i) \equiv 0 , \qquad v(t_i) = 2\pi\bar{u}\left[r(t_i)^2\left(\frac{2}{3}\,\frac{r(t_i)}{a} - \frac{1}{2}\right) + \frac{a^2}{24}\right],$$

where $\bar{u} = \dfrac{12V}{5\pi a^2}$. Note that the initial approximation to ψ must be given in radians, as internal to the program ψ is stored in radians; it is converted to degrees for output only. Similarly, ANGLE is converted in CYLIND from degrees to radians before being stored as γ.

= 1. The user supplies initial values for the mesh $\{t_i\}$ and an initial approximation for the unknown function values u_i, r_i, ψ_i, S_i, H_i, v_i, i = 1,...,N. The initial mesh must satisfy (10) and be stored in the array X(•); the initial unknown function values are stored consecutively in the one-dimensional array Y(•) in the order

```
**************************************************************
*                                                            *
*   MENISCUS COMPUTATION FOR A RIGHT CIRCULAR CYLINDER       *
*   WITH A FLAT BOTTOM.                                      *
*                                                            *
*   CYLINDER RADIUS          =   .200000E+01                 *
*                                                            *
*   (SMALL) LIQUID VOLUME    =   .250000E+02                 *
*                                                            *
*   CONTACT ANGLE            =  0.      DEGREES              *
*                                                            *
*   BOND NUMBER              =   .100000E+01                 *
*                                                            *
**************************************************************

  NEWTON DOES NOT CONVERGE
 FOR MORE DETAILED OUTPUT USE  PAR(3) = 1.
TRY CONTINUATION, BETTER INITIAL VALUES, AND/OR  A FINER MESH


  ---- ERROR 3 ----

   NEWTON DIVERGED

****FAILURE-TRYING LARGE VOLUME OPTION****
```

S	R	U	PSI
0.	0.	.157663E+01	0.
.259649E+00	.259394E+00	.158656E+01	4.4046
.519298E+00	.517205E+00	.161681E+01	9.0326
.778947E+00	.771533E+00	.166868E+01	14.1198
.103860E+01	.101980E+01	.174435E+01	19.9249
.129824E+01	.125824E+01	.184672E+01	26.7423
.155789E+01	.148132E+01	.197917E+01	34.9132
.181754E+01	.168081E+01	.214485E+01	44.8363
.207719E+01	.184497E+01	.234540E+01	56.9724
.233684E+01	.195784E+01	.257842E+01	71.8404
.259649E+01	.200000E+01	.283354E+01	90.0000
MAXIMUM ABSOLUTE ERRORS	.27E-04	.33E-04	.66E-05

H = -.494737E+00 ERROR ≤ .81E-05

As the output indicates, the small-volume option failed to converge, after which an automatic switching to the correct, large-volume option took place. The smaller error than for the earlier example is achieved for this case without addition of extra points. (Recall that TØL is the maximum error over all the variables, not just the ones that are output.)

When making a sequence of calculations in which the parameters are varied through a range of values, the user can save considerable computer time in many cases by using the output of one problem to start the next. This savings is generally greatest if the problems are solved in the order of increasing degree of likely mesh nonuniformity.

In the next example we solve the same problem as above, but now for three successive values of the Bond number BØND = 5., 10., 20.. In order to permit the output of one problem -- solution and mesh values -- to be the input for the next, the driver program MNCLND was altered by replacing the CALL CYLIND (•) statement with an appropriate DØ loop. The program used was the same as that shown in the listing except that the

```
CALL CYLIND (•)
```

statement was replaced with

```
      DØ 30 I = 1,3
      CALL CYLIND (•)
      BØND = BØND * 2.
      PAR(4) = 1.
30    CØNTINUE
```

The data cards were

columns	1-8	9-16	17-24				
quantity	PAR(1)	PAR(3)	PAR(4)				Data Card 1
value	2.0	0.	0.				

columns	1-4	5-8	9-16	17-24	25-32	33-40	41-45	
quantity	II	N	BØND	VØLUME	ANGLE	RADIUS	TØL	Data Card 2
value	1	5	5.	3.	30.	1.	1.E-2	

\+ Termination Card

In the first call to CYLIND PAR(4) = 0., and the default values are used. In successive calls PAR(4) = 1., and the final values of N, X, Y from the previous case are used as initial values. The output from CYLIND is shown below.

```
**********************************************************
*                                                        *
*   MENISCUS COMPUTATION FOR A RIGHT CIRCULAR CYLINDER   *
*   WITH A FLAT BOTTOM.                                  *
*                                                        *
*   CYLINDER RADIUS          =   .100000E+01             *
*                                                        *
*   (LARGE) LIQUID VOLUME    =   .300000E+01             *
*                                                        *
*   CONTACT ANGLE            = 30.000   DEGREES          *
*                                                        *
*   BOND NUMBER              =   .500000E+01             *
*                                                        *
**********************************************************
```

S	R	U	PSI
0.	0.	.794934E+00	0.
.140363E+00	.140255E+00	.799554E+00	3.7939
.280726E+00	.279874E+00	.813727E+00	7.8715
.421089E+00	.418011E+00	.838432E+00	12.5367
.561452E+00	.553390E+00	.875321E+00	18.1395
.701815E+00	.683931E+00	.926707E+00	25.1027
.842178E+00	.806184E+00	.995436E+00	33.9476
.982541E+00	.914501E+00	.108441E+01	45.3200
.112290E+01	.100000E+01	.119515E+01	60.0000
MAXIMUM ABSOLUTE ERRORS	.23E-04	.43E-04	.22E-04

H = -.152116E+01 ERROR ≤ .53E-04

```
**********************************************************
*                                                        *
*   MENISCUS COMPUTATION FOR A RIGHT CIRCULAR CYLINDER   *
*   WITH A FLAT BOTTOM.                                  *
*                                                        *
*   CYLINDER RADIUS          =   .100000E+01             *
*                                                        *
*   (LARGE) LIQUID VOLUME    =   .300000E+01             *
*                                                        *
*   CONTACT ANGLE            = 30.000   DEGREES          *
*                                                        *
*   BOND NUMBER              =   .100000E+02             *
*                                                        *
**********************************************************
```

S	R	U	PSI
0.	0.	.838081E+00	0.
.136490E+00	.136449E+00	.840747E+00	2.2601
.272979E+00	.272674E+00	.849099E+00	4.8409
.409469E+00	.408299E+00	.864309E+00	8.1132
.545959E+00	.542604E+00	.888493E+00	12.5540
.682448E+00	.674089E+00	.924948E+00	18.8224
.818938E+00	.799595E+00	.978367E+00	27.8540
.955428E+00	.912564E+00	.105462E+01	40.9789
.109192E+01	.100000E+01	.115836E+01	60.0000
MAXIMUM ABSOLUTE ERRORS	.93E-04	.12E-03	.34E-04

H = -.390794E+01 ERROR ≤ .28E-03

```
**********************************************************
*                                                        *
*   MENISCUS COMPUTATION FOR A RIGHT CIRCULAR CYLINDER   *
*   WITH A FLAT BOTTOM.                                  *
*                                                        *
*   CYLINDER RADIUS          =   .100000E+01             *
*                                                        *
*   (LARGE) LIQUID VOLUME    =   .300000E+01             *
*                                                        *
*   CONTACT ANGLE            = 30.000   DEGREES          *
*                                                        *
*   BOND NUMBER              =   .200000E+02             *
*                                                        *
**********************************************************
```

S	R	U	PSI
0.	0.	.880242E+00	0.
.133142E+00	.133133E+00	.881355E+00	.9711
.266284E+00	.266224E+00	.884963E+00	2.2060
.399426E+00	.399173E+00	.892063E+00	4.0527
.532568E+00	.531714E+00	.904625E+00	7.0407
.665710E+00	.663087E+00	.926167E+00	12.0503
.798852E+00	.791097E+00	.962671E+00	20.5817
.931994E+00	.909375E+00	.102361E+01	35.2029
.106514E+01	.100000E+01	.111899E+01	60.0000
MAXIMUM ABSOLUTE ERRORS	.28E-03	.22E-03	.13E-03

H = -.868077E+01 ERROR ≤ .75E-03

For this example the starting mesh with $N = 5$ points was incremented to one with $N = 9$ points by CYLIND for the first problem. That mesh was then adequate for the other two problems.

The user will likely alter MNCLND to suit his particular requirements. He should take care to have in any alteration the correct DIMENSIØN, CØMMØN, and CALL CYLIND statements, as provided in the program.

The solution $\{u(s),r(s)\}$ is available to the user through CØMMØN upon exit from CYLIND for further processing, such as plotting.

CYLCUR: RIGHT CIRCULAR CYLINDRICAL CONTAINER, SPHEROIDAL BOTTOM

This case is the same as the one for CYLIND, except that the right circular cylindrical container has a bottom that is spheroidal. A vertical section of the container depicting the portion of the ellipse $u_B(r)$ forming the bottom is shown in Fig. 3,

$$u_B(r) = u_C + \beta\left(1 - \frac{r^2}{\alpha^2}\right)^{\frac{1}{2}} .$$

The user must specify the (nondimensional) horizontal semi-axis α and vertical semi-axis β of the ellipse, in addition to those quantities required by CYLIND. On output the coordinates of the center $(0,u_C)$ of the ellipse are given, as a check for the user. The value of u_C,

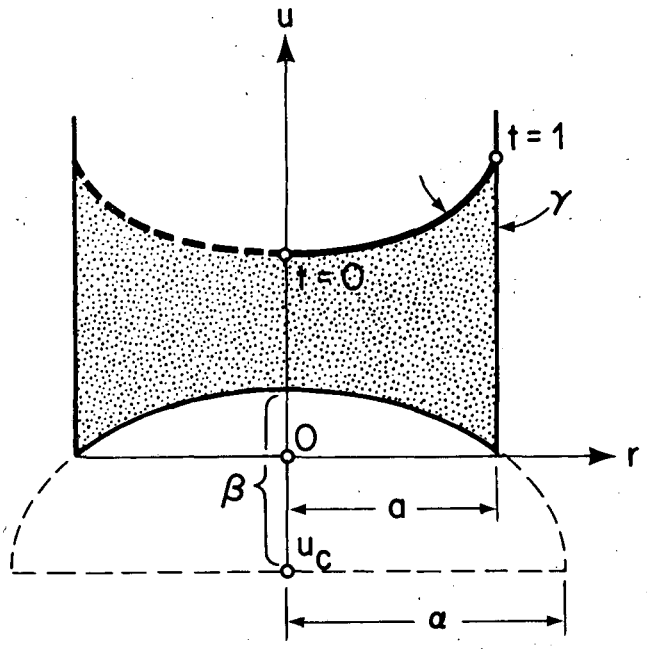
$$u_C = -\beta\left(1 - \frac{a^2}{\alpha^2}\right)^{\frac{1}{2}} ,$$

is such that the ellipse passes through the point $(a,0)$ (Fig. 3).

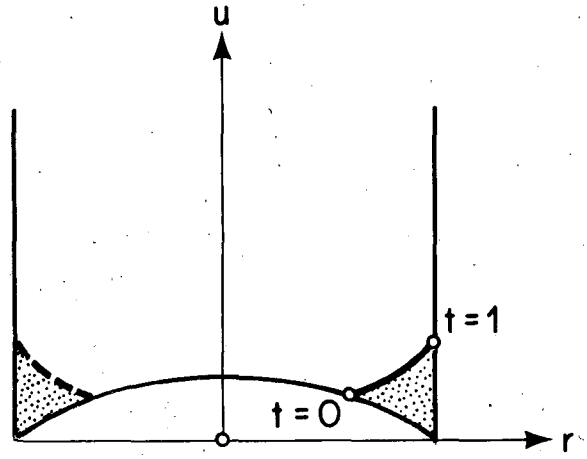
The driver MNCLCR is used in the same manner as MNCLND. Data Card 1 is identical for the two drivers, and the expanded Data Card 2 for MNCLCR is

Data Card 2 (FØRMAT 2I4,7E8.0): II,N,BØND,VØLUME,ANGLE, RADIUS,TØL,HØRZ,VERT.

The additional quantities are



(a)



(b)

XBL 792-582

Figure 3

Right circular cylinder - spheroidal bottom. (a) large volume case. (b) small volume case. The solid portion of the free-surface meridian depicts the computed arc (r(t),u(t)); the dashed portion is the rotationally symmetric image.

HØRZ

The horizontal semi-axis α of the bottom (nondimensional, consistent with BØND). HØRZ must be $\geq$ RADIUS.

VERT

The vertical semi-axis β of the bottom (nondimensional, consistent with BØND).

For the input cards

columns	1-8	9-16	17-24
quantity	PAR(1)		
value	1.0	blank	blank

Data Card 1

columns	1-4	5-8	9-16	17-24	25-32	33-40	41-48	49-56	57-64
quantity	II		BØND	VØLUME	ANGLE	RADIUS		HØRZ	VERT
value	1	blank	5.	3.	0.	1.	blank	1.5	3.

Data Card 2

\+ termination card

the following output from CYLCUR resulted.

MENISCUS COMPUTATION FOR A RIGHT CIRCULAR CYLINDER
WITH A SPHEROIDAL BOTTOM.

CYLINDER RADIUS = .100000E+01

ELLIPSE VERTICAL SEMI-AXIS = .300000E+01

HORIZONTAL SEMI-AXIS = .150000E+01

CENTER AT (0 , -.2236E+01)

(LARGE) LIQUID VOLUME = .300000E+01

CONTACT ANGLE = 0. DEGREES

BOND NUMBER = .500000E+01

S	R	U	PSI
0.	0.	.116547E+01	0.
.127285E+00	.127184E+00	.116974E+01	3.8652
.254571E+00	.253774E+00	.118279E+01	7.9679
.381856E+00	.379013E+00	.120535E+01	12.5587
.509141E+00	.501828E+00	.123863E+01	17.9194
.636426E+00	.620570E+00	.128431E+01	24.3807
.763712E+00	.732653E+00	.134445E+01	32.3380
.890997E+00	.834038E+00	.142118E+01	42.2682
.101828E+01	.918606E+00	.151602E+01	54.7406
.114557E+01	.977600E+00	.162841E+01	70.4136
.127285E+01	.100000E+01	.175300E+01	90.0000
MAXIMUM ABSOLUTE ERRORS	.50E-04	.14E-03	.47E-04

H = -.238884E+01 ERROR ≤ .12E-03

PAR(4)

Controls whether default values or user-supplied initial values should be used for the initial mesh $\{t_i\}$ and initial approximations to u, r, ψ, S, H, and v.

= 0. Default values are provided automatically by CYLIND. These are that $\{t_i\}$ are the N uniformly spaced points on $[0,1]$. For II = 1 the initial approximation is

$$u(t_i) \equiv V/(\pi a^2) , \qquad r(t_i) = a\, t_i ,$$

$$\psi(t_i) = (\frac{\pi}{2} - \gamma) t_i , \qquad S(t_i) \equiv a ,$$

$$H(t_i) \equiv -\frac{1}{2} Bu(0) , \qquad v(t_i) = \pi a^2 t_i^2 u(0) ;$$

for II = 2 it is

$$u(t_i) = \bar{u}\left(2\,\frac{r(t_i)}{a} - 1\right) , \qquad r(t_i) = \frac{1}{2} a(1 + t_i) ,$$

$$\psi(t_i) = \gamma + (\frac{\pi}{2} - 2\gamma)\, t_i , \qquad S(t_i) \equiv (\bar{u}^2 + \frac{1}{4} a^2)^{\frac{1}{2}} ,$$

$$H(t_i) \equiv 0 , \qquad v(t_i) = 2\pi\bar{u}\left[r(t_i)^2\left(\frac{2}{3}\,\frac{r(t_i)}{a} - \frac{1}{2}\right) + \frac{a^2}{24}\right] ,$$

where $\bar{u} = \dfrac{12V}{5\pi a^2}$. Note that the initial approximation to ψ must be given in radians, as internal to the program ψ is stored in radians; it is converted to degrees for output only. Similarly, ANGLE is converted in CYLIND from degrees to radians before being stored as γ.

= 1. The user supplies initial values for the mesh $\{t_i\}$ and an initial approximation for the unknown function values u_i, r_i, ψ_i, S_i, H_i, v_i, $i = 1,\ldots,N$. The initial mesh must satisfy (10) and be stored in the array $X(\cdot)$; the initial unknown function values are stored consecutively in the one-dimensional array $Y(\cdot)$ in the order

u_1	r_1	ψ_1	S_1	H_1	v_1	u_2	r_2	ψ_2	...
↓	↓	↓	↓	↓	↓	↓	↓	↓	
Y(1)	Y(2)	Y(3)	Y(4)	Y(5)	Y(6)	Y(7)	Y(8)	Y(9)	... .

The user must add the necessary FØRTRAN statements to the main program MNCLND to input the desired initial values, taking note that ANGLE is in degrees and ψ_i in radians.

The user should take precautions that the initial values provided for ψ lie in the appropriate quadrants. For the large volume case, ψ generally increases with t from the value of zero at the axis if the free-surface meridian curves upward, and decreases with t if the meridian curves downward. For the small volume case the situation may be more complex, particularly for nonwetting contact angles. For the example shown in Fig. 2b, ψ increases with t from a value between 0 and $\pi/2$ at $t = 0$ to a value between 0 and $\pi/2$ at $t = 1$. Whereas for the nonwetting, small-volume example calculated above, ψ decreases with t from a value of π at $t = 0$ to a value of $-\pi/2$ at $t = 1$.

Examples of the user-control option follow. For the first example we set PAR(3) = PAR(4) = 0. so that there is no intermediate printing and default values are used for the initial approximation. For TØL the value 1.0×10^{-3} is specified, which is smaller than the default value 5.0×10^{-3} used in the first examples.

columns	1-8	9-16	17-24	
quantity	PAR(1)	PAR(3)	PAR(4)	
value	2.0	0.	0.	Data Card 1

columns	1-4	5-8	9-16	17-24	25-32	33-40	41-48	
quantity	II	N	BØND	VØLUME	ANGLE	RADIUS	TØL	
value	2	11	1.	25.	0.	2.	1.E-3	Data Card 2

\+ Termination Data Card 2

The following output was obtained from CYLIND (and PASVA2). Here, the wrong value is assigned deliberately to II so that the small volume option will be tried first.

```
**************************************************************
*                                                            *
*   MENISCUS COMPUTATION FOR A RIGHT CIRCULAR CYLINDER       *
*   WITH A FLAT BOTTOM.                                      *
*                                                            *
*   CYLINDER RADIUS          =   .200000E+01                 *
*                                                            *
*   (SMALL) LIQUID VOLUME    =   .250000E+02                 *
*                                                            *
*   CONTACT ANGLE            =  0.      DEGREES              *
*                                                            *
*   BOND NUMBER              =   .100000E+01                 *
*                                                            *
**************************************************************

  NEWTON DOES NOT CONVERGE
 FOR MORE DETAILED OUTPUT USE  PAR(3) = 1.
TRY CONTINUATION, BETTER INITIAL VALUES, AND/OR  A FINER MESH


  ---- ERROR 3 ----

   NEWTON DIVERGED

****FAILURE-TRYING LARGE VOLUME OPTION****
```

S	R	U	PSI
0.	0.	.157663E+01	0.
.259649E+00	.259394E+00	.158656E+01	4.4046
.519298E+00	.517205E+00	.161681E+01	9.0326
.778947E+00	.771533E+00	.166868E+01	14.1198
.103860E+01	.101980E+01	.174435E+01	19.9249
.129824E+01	.125824E+01	.184672E+01	26.7423
.155789E+01	.148132E+01	.197917E+01	34.9132
.181754E+01	.168081E+01	.214485E+01	44.8363
.207719E+01	.184497E+01	.234540E+01	56.9724
.233684E+01	.195784E+01	.257842E+01	71.8404
.259649E+01	.200000E+01	.283354E+01	90.0000
MAXIMUM ABSOLUTE ERRORS	.27E-04	.33E-04	.66E-05

H = -.494737E+00 ERROR ≤ .81E-05

As the output indicates, the small-volume option failed to converge, after which an automatic switching to the correct, large-volume option took place. The smaller error than for the earlier example is achieved for this case without addition of extra points. (Recall that TØL is the maximum error over all the variables, not just the ones that are output.)

When making a sequence of calculations in which the parameters are varied through a range of values, the user can save considerable computer time in many cases by using the output of one problem to start the next. This savings is generally greatest if the problems are solved in the order of increasing degree of likely mesh nonuniformity.

In the next example we solve the same problem as above, but now for three successive values of the Bond number BØND = 5., 10., 20.. In order to permit the output of one problem -- solution and mesh values -- to be the input for the next, the driver program MNCLND was altered by replacing the CALL CYLIND (•) statement with an appropriate DØ loop. The program used was the same as that shown in the listing except that the

```
CALL CYLIND (•)
```

statement was replaced with

```
      DØ 30 I = 1,3
      CALL CYLIND (•)
      BØND = BØND * 2.
      PAR(4) = 1.
30    CØNTINUE
```

The data cards were

columns	1-8	9-16	17-24				
quantity	PAR(1)	PAR(3)	PAR(4)				Data Card 1
value	2.0	0.	0.				

columns	1-4	5-8	9-16	17-24	25-32	33-40	41-45	
quantity	II	N	BØND	VØLUME	ANGLE	RADIUS	TØL	Data Card 2
value	1	5	5.	3.	30.	1.	1.E-2	

\+ Termination Card

In the first call to CYLIND PAR(4) = 0., and the default values are used. In successive calls PAR(4) = 1., and the final values of N, X, Y from the previous case are used as initial values. The output from CYLIND is shown below.

```
**********************************************************
*                                                        *
*   MENISCUS COMPUTATION FOR A RIGHT CIRCULAR CYLINDER   *
*   WITH A FLAT BOTTOM.                                  *
*                                                        *
*   CYLINDER RADIUS          =   .100000E+01             *
*                                                        *
*   (LARGE) LIQUID VOLUME    =   .300000E+01             *
*                                                        *
*   CONTACT ANGLE            = 30.000   DEGREES          *
*                                                        *
*   BOND NUMBER              =   .500000E+01             *
*                                                        *
**********************************************************
```

S	R	U	PSI
0.	0.	.794934E+00	0.
.140363E+00	.140255E+00	.799554E+00	3.7939
.280726E+00	.279874E+00	.813727E+00	7.8715
.421089E+00	.418011E+00	.838432E+00	12.5367
.561452E+00	.553390E+00	.875321E+00	18.1395
.701815E+00	.683931E+00	.926707E+00	25.1027
.842178E+00	.806184E+00	.995436E+00	33.9476
.982541E+00	.914501E+00	.108441E+01	45.3200
.112290E+01	.100000E+01	.119515E+01	60.0000
MAXIMUM ABSOLUTE ERRORS	.23E-04	.43E-04	.22E-04

H = -.152116E+01 ERROR ≤ .53E-04

```
**********************************************************
*                                                        *
*   MENISCUS COMPUTATION FOR A RIGHT CIRCULAR CYLINDER   *
*   WITH A FLAT BOTTOM.                                  *
*                                                        *
*   CYLINDER RADIUS          =   .100000E+01             *
*                                                        *
*   (LARGE) LIQUID VOLUME    =   .300000E+01             *
*                                                        *
*   CONTACT ANGLE            = 30.000   DEGREES          *
*                                                        *
*   BOND NUMBER              =   .100000E+02             *
*                                                        *
**********************************************************
```

S	R	U	PSI
0.	0.	.838081E+00	0.
.136490E+00	.136449E+00	.840747E+00	2.2601
.272979E+00	.272674E+00	.849099E+00	4.8409
.409469E+00	.408299E+00	.864309E+00	8.1132
.545959E+00	.542604E+00	.888493E+00	12.5540
.682448E+00	.674089E+00	.924948E+00	18.8224
.818938E+00	.799595E+00	.978367E+00	27.8540
.955428E+00	.912564E+00	.105462E+01	40.9789
.109192E+01	.100000E+01	.115836E+01	60.0000
MAXIMUM ABSOLUTE ERRORS	.93E-04	.12E-03	.34E-04

H = -.390794E+01 ERROR ≤ .28E-03

```
**********************************************************
*                                                        *
*   MENISCUS COMPUTATION FOR A RIGHT CIRCULAR CYLINDER   *
*   WITH A FLAT BOTTOM.                                  *
*                                                        *
*   CYLINDER RADIUS          =   .100000E+01             *
*                                                        *
*   (LARGE) LIQUID VOLUME    =   .300000E+01             *
*                                                        *
*   CONTACT ANGLE            = 30.000   DEGREES          *
*                                                        *
*   BOND NUMBER              =   .200000E+02             *
*                                                        *
**********************************************************
```

S	R	U	PSI
0.	0.	.880242E+00	0.
.133142E+00	.133133E+00	.881355E+00	.9711
.266284E+00	.266224E+00	.884963E+00	2.2060
.399426E+00	.399173E+00	.892063E+00	4.0527
.532568E+00	.531714E+00	.904625E+00	7.0407
.665710E+00	.663087E+00	.926167E+00	12.0503
.798852E+00	.791097E+00	.962671E+00	20.5817
.931994E+00	.909375E+00	.102361E+01	35.2029
.106514E+01	.100000E+01	.111899E+01	60.0000
MAXIMUM ABSOLUTE ERRORS	.28E-03	.22E-03	.13E-03

H = -.868077E+01 ERROR ≤ .75E-03

For this example the starting mesh with $N = 5$ points was incremented to one with $N = 9$ points by CYLIND for the first problem. That mesh was then adequate for the other two problems.

The user will likely alter MNCLND to suit his particular requirements. He should take care to have in any alteration the correct DIMENSIØN, CØMMØN, and CALL CYLIND statements, as provided in the program.

The solution $\{u(s),r(s)\}$ is available to the user through CØMMØN upon exit from CYLIND for further processing, such as plotting.

CYLCUR: RIGHT CIRCULAR CYLINDRICAL CONTAINER, SPHEROIDAL BOTTOM

This case is the same as the one for CYLIND, except that the right circular cylindrical container has a bottom that is spheroidal. A vertical section of the container depicting the portion of the ellipse $u_B(r)$ forming the bottom is shown in Fig. 3,

$$u_B(r) = u_C + \beta\left(1 - \frac{r^2}{\alpha^2}\right)^{\frac{1}{2}} .$$

The user must specify the (nondimensional) horizontal semi-axis α and vertical semi-axis β of the ellipse, in addition to those quantities required by CYLIND. On output the coordinates of the center $(0,u_C)$ of the ellipse are given, as a check for the user. The value of u_C,

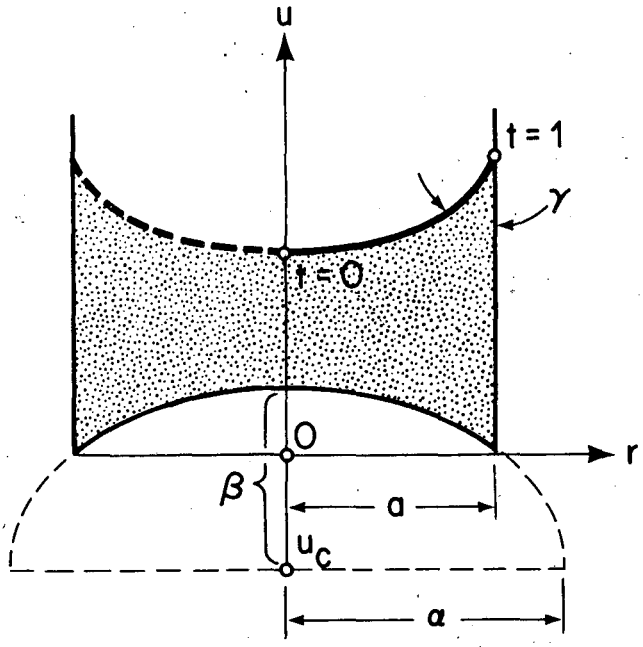
$$u_C = -\beta\left(1 - \frac{a^2}{\alpha^2}\right)^{\frac{1}{2}} ,$$

is such that the ellipse passes through the point $(a,0)$ (Fig. 3).

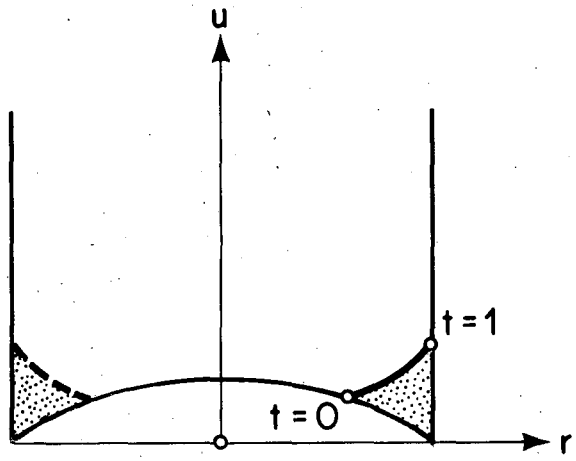
The driver MNCLCR is used in the same manner as MNCLND. Data Card 1 is identical for the two drivers, and the expanded Data Card 2 for MNCLCR is

Data Card 2 (FØRMAT 2I4,7E8.0): II,N,BØND,VØLUME,ANGLE, RADIUS,TØL,HØRZ,VERT.

The additional quantities are



(a)



(b)

XBL 792-582

Figure 3

Right circular cylinder - spheroidal bottom. (a) large volume case. (b) small volume case. The solid portion of the free-surface meridian depicts the computed arc (r(t),u(t)); the dashed portion is the rotationally symmetric image.

HØRZ

The horizontal semi-axis α of the bottom (nondimensional, consistent with BØND). HØRZ must be $\geq$ RADIUS.

VERT

The vertical semi-axis β of the bottom (nondimensional, consistent with BØND).

For the input cards

columns	1-8	9-16	17-24
quantity	PAR(1)		
value	1.0	blank	blank

Data Card 1

columns	1-4	5-8	9-16	17-24	25-32	33-40	41-48	49-56	57-64
quantity	II		BØND	VØLUME	ANGLE	RADIUS		HØRZ	VERT
value	1	blank	5.	3.	0.	1.	blank	1.5	3.

Data Card 2

\+ termination card

the following output from CYLCUR resulted.

```
**********************************************************
*                                                        *
*   MENISCUS COMPUTATION FOR A RIGHT CIRCULAR CYLINDER   *
*   WITH A SPHEROIDAL BOTTOM.                            *
*                                                        *
*   CYLINDER RADIUS          =   .100000E+01             *
*                                                        *
*   ELLIPSE VERTICAL SEMI-AXIS  =   .300000E+01          *
*                                                        *
*   HORIZONTAL SEMI-AXIS        =   .150000E+01          *
*                                                        *
*   CENTER AT (0 , -.2236E+01)                           *
*                                                        *
*   (LARGE) LIQUID VOLUME    =   .300000E+01             *
*                                                        *
*   CONTACT ANGLE            =  0.      DEGREES         *
*                                                        *
*   BOND NUMBER              =   .500000E+01             *
*                                                        *
**********************************************************
```

S	R	U	PSI
0.	0.	.116547E+01	0.
.127285E+00	.127184E+00	.116974E+01	3.8652
.254571E+00	.253774E+00	.118279E+01	7.9679
.381856E+00	.379013E+00	.120535E+01	12.5587
.509141E+00	.501828E+00	.123863E+01	17.9194
.636426E+00	.620570E+00	.128431E+01	24.3807
.763712E+00	.732653E+00	.134445E+01	32.3380
.890997E+00	.834038E+00	.142118E+01	42.2682
.101828E+01	.918606E+00	.151602E+01	54.7406
.114557E+01	.977600E+00	.162841E+01	70.4136
.127285E+01	.100000E+01	.175300E+01	90.0000
MAXIMUM ABSOLUTE ERRORS	.50E-04	.14E-03	.47E-04

H = -.238884E+01 ERROR ≤ .12E-03

The default values for the initial approximation provided in CYLCUR when PAR(4) = 0. are

for II = 1 ,

$$u(t_i) \equiv \bar{u} = u_C + \frac{1}{a^2}\left[\frac{V}{\pi} + \frac{2\alpha^2}{3\beta^2}\,(\beta^3 + u_C^3)\right] , \qquad r(t_i) = at_i ,$$

$$\psi(t_i) = \left(\frac{\pi}{2} - \gamma\right) t_i , \qquad S(t_i) \equiv a ,$$

$$H(t_i) = -\frac{1}{2} B\bar{u} , \qquad v(t_i) = Vt_i^2 ;$$

for II = 2 ,

$$u(t_i) \equiv \bar{u} = u_C + \beta\left[1 - \left(\frac{a}{2\alpha}\right)^2\right]^{\frac{1}{2}} , \qquad r(t_i) = \frac{a}{2}\,(1 + t_i) ,$$

$$\psi(t_i) = \gamma + \left(\frac{\pi}{2} - 2\gamma\right) t_i , \qquad S(t_i) \equiv \frac{1}{2}\,a ,$$

$$H(t_i) \equiv 0 , \qquad v(t_i) = \frac{1}{2}\,\pi\bar{u}\left(r(t_i)^2 - \frac{1}{4}a^2\right) .$$

In other respects the operation of MNCLCR and CYLCUR is the same as that of MNCLND and CYLIND. See the description of CYLIND for a discussion of the usage options.

ELLIPS: SPHEROIDAL CONTAINER

For this case, the container is a spheroid (ellipsoid of revolution) with axis of symmetry oriented vertically. A vertical section is depicted in Fig. 4. The (nondimensional) horizontal and vertical semi-axes are α and β, respectively, and the origin is at the center of the tank bottom.

Note that here "large" and "small" volume of liquid do not necessarily refer to the actual amount of liquid, but rather to the position of the liquid in the container. Obviously, a large amount of contained liquid will tend to correspond to case (a), but for some conditions solutions can exist both for cases (a) and (b), as well as for other configurations.

The usage of ELLIPS and of its driver MNELPS is essentially the same as for the programs for the cylindrical containers. Data Card 1 is the same as previously and here Data Card 2 is

Data Card 2 (FØRMAT 2I4,7E8.0): II,N,BØND,VØLUME,ANGLE, RADIUS,TØL,ALFA,BETA.

II, N, BØND, VØLUME, ANGLE, and TØL are as defined previously.

RADIUS

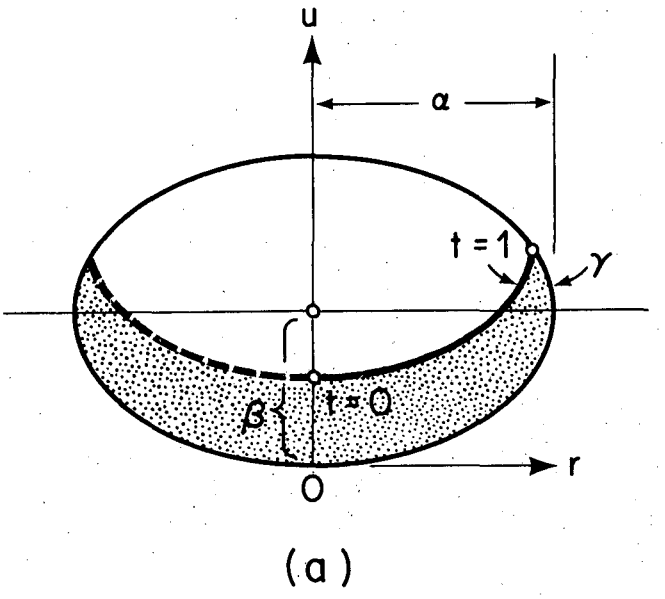
This quantity is not used by MNELPS and may be left blank. It is present for the sake of uniformity of format of data cards 2.

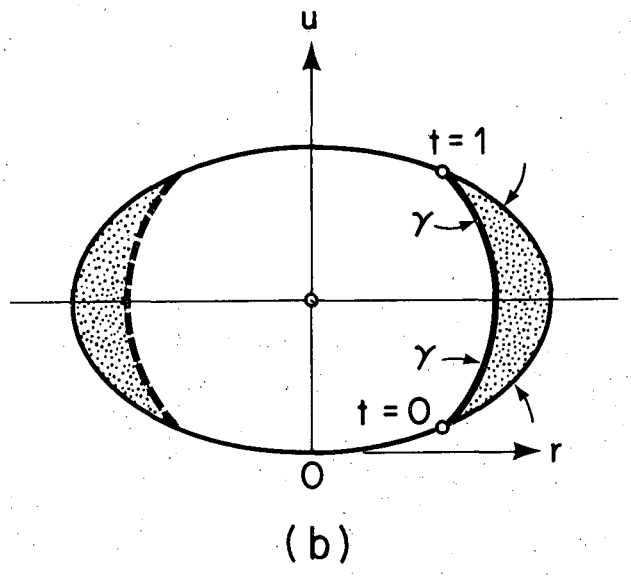
ALFA

The horizontal semi-axis α of the container (nondimensional, consistent with BØND).

BETA

The vertical semi-axis β of the container (nondimensional, consistent with BØND).





XBL 792-583

Figure 4

Spheroidal container. (a) large volume case. (b) small volume case. The solid portion of the free-surface meridian depicts the computed arc (r(t),u(t)); the dashed portion is the rotationally symmetric image.

For the input cards

columns	1-8	9-16	17-24
quantity	PAR(1)		
value	1.0	blank	blank

Data Card 1

columns	1-4	5-8	9-16	17-24	25-32	33-40	41-48	49-56	57-64
quantity	II		BØND	VØLUME	ANGLE			ALFA	BETA
value	1	blank	5.	2.	60.	blank	blank	2.	1.

Data Card 2

\+ termination card

the following output from ELLIPS resulted.

```
**********************************************************
*                                                        *
*   MENISCUS COMPUTATION FOR A SPHEROIDAL                *
*   CONTAINER WITH                                       *
*                                                        *
*   HORIZONTAL SEMI-AXIS (ALFA) =   .200000E+01          *
*   VERTICAL  SEMI-AXIS  (BETA) =   .100000E+01          *
*                                                        *
*   (LARGE) LIQUID VOLUME  =   .200000E+01               *
*                                                        *
*   CONTACT ANGLE          = 60.000   DEGREES            *
*                                                        *
*   BOND NUMBER            =   .500000E+01               *
*                                                        *
**********************************************************
```

S	R	U	PSI
0.	0.	.555314E+00	0.
.146756E+00	.146747E+00	.553937E+00	-1.0814
.293512E+00	.293439E+00	.549698E+00	-2.2513
.440268E+00	.440003E+00	.542253E+00	-3.6054
.587023E+00	.586322E+00	.530994E+00	-5.2564
.733779E+00	.732198E+00	.515002E+00	-7.3438
.880535E+00	.877281E+00	.492972E+00	-10.0466
.102729E+01	.102095E+01	.463117E+00	-13.5994
.117405E+01	.116210E+01	.423062E+00	-18.3127
.132080E+01	.129878E+01	.369764E+00	-24.5983
.146756E+01	.142749E+01	.299597E+00	-33.0000
MAXIMUM ABSOLUTE ERRORS	.11E-03	.52E-04	.66E-04

H = -.151522E+01 ERROR ≤ .33E-04

The default values for the initial approximation provided in ELLIPS when PAR(4) = 0. are

for II = 1 ,

$$u(t_i) \equiv \bar{u} = \frac{V}{\pi\alpha^2} \quad , \qquad r(t_i) = \alpha t_i \left[1 - \left(\frac{\bar{u}}{\beta} - 1\right)^2\right]^{\frac{1}{2}} ,$$

$$\psi(t_i) \equiv 0 \quad , \qquad S(t_i) \equiv \alpha \left[1 - \left(\frac{\bar{u}}{\beta} - 1\right)^2\right]^{\frac{1}{2}} ,$$

$$H(t_i) \equiv -\frac{B\bar{u}}{2} \quad , \qquad V(t_i) = Vt_i^2 \quad ;$$

for II = 2 ,

$$u(t_i) = \beta\left[1 + \frac{2t_i - 1}{\alpha}\left(\alpha^2 - \bar{r}^2\right)^{\frac{1}{2}}\right], \qquad r(t_i) \equiv \bar{r} = \left(\alpha^2 - \frac{V}{2\pi\beta}\right)^{\frac{1}{2}} ,$$

$$\psi(t_i) \equiv \pi/2 \quad , \qquad S(t_i) \equiv \frac{2\beta}{\alpha}\left(\alpha^2 - \bar{r}^2\right)^{\frac{1}{2}} ,$$

$$H(t_i) \equiv \frac{1}{2}\left(\frac{1}{\bar{r}} - B\beta\right) \quad , \qquad V(t_i) = Vt_i \quad .$$

See the description of CYLIND for the details of program usage.

PASVA2

The programs described in the previous sections all call PASVA2 to solve numerically the appropriate system of first-order nonlinear ordinary differential equations subject to two-point boundary conditions. For most problems of interest these driver programs are adequate for obtaining a solution. However, the user may wish to call PASVA2 with his own driver program for certain problems, so as to obtain maximum flexibility and to employ all the features built into this general-purpose solver. A description of PASVA2 follows.

PASVA3, a revised version of PASVA2, is also available [9]. This version is documented somewhat more completely and has a simpler format for communication with the user-supplied subroutines. In its operation, however, it is essentially the same as PASVA2. Copies of the PASVA3 program and writeup can be obtained from the Lawrence Berkeley Laboratory Computer Center.

I. IDENTIFICATION

A. NAME : PASVA2 -- Two point boundary problem solver for nonlinear first order systems

B. AUTHORS : M. Lentini and V. Pereyra

C. DATE : September 1977

D. LANGUAGE : ANSI FORTRAN

E. PRECISION : Single (double available also)

II. ABSTRACT

PASVA2 is an adaptive (variable order, variable step) finite difference solver for nonlinear first order systems of ordinary differential equations subject to nonlinear two point boundary conditions.

This program should adequately solve problems with mild boundary layers, spikes, and in general, problems with solutions varying sharply in some regions. It will also solve smooth and linear problems efficiently and accurately.

This is an updated version of program PASVAR, which is described in detail in reference [3].

III. USAGE

A. Consider the two point boundary value problem:

$$
\begin{aligned}
& y' = f(x, y), && x \in [a, b], \\
& g_1(y(a)) && = 0 \quad \text{p initial conditions}, \\
(1) \qquad & g_2(y(a), y(b)) && = 0 \quad \text{r coupled conditions}, \\
& g_3(y(b)) && = 0 \quad \text{m-(p+r) final conditions},
\end{aligned}
$$

where $y, f, g = (g_1, g_2, g_3)^T$ are m-vectors. We assume that

$$2 \leq m, \quad o \leq p < m, \quad o < p+r \leq m,$$

and that the problem has an isolated solution $y^*(x)$.

To obtain an approximation to $y^*(x)$ on a mesh $\pi^\circ = \{x_i\}_{i=1,\ldots,N}$, $a = x_1 < x_2 < \ldots < x_N = b$, with a maximum absolute error less than TOL (on all components and mesh points).

CALL PASVA2 (M, N, P, R, A, B, TOL, X, Y, ABT, FF,
JACOB, PAR, JERROR)

where(*)

(I) M (Integer) is the number of equations,

(I-ϕ) N (Integer) is on input the number of points in the initial mesh π° and on output the number of points in the final mesh π (which will contain π°),

(I) P (Integer) is the number of initial conditions,

(I) R (Integer) is the number of coupled boundary conditions,

(I) A (Real) is the left end of the interval,

(I) B (Real) is the right end of the interval,

(I) TOL (Real) is the desired tolerance,

(I-ϕ) X(325) (Real array) is a N-vector containing the initial mesh on input and the final mesh on output.

(I-ϕ) Y(650) (Real array) is a N x M vector containing on input the initial guess for the solution vector on the mesh X, and the final computed solution on output. The approximations to the vector valued function $y^*(x)$ at each grid point are stored consecutively as:

(*) (I) stands for input parameter, and (ϕ) for output parameter.

$$y_1^*(x_1),\ y_2^*(x_1),\ \ldots,\ y_m^*(x_1),\ y^*(x_2),\ \ldots,\ y_m^*(x_N).$$

(ϕ) ABT (10) (Real array) is an M-vector containing the components of the estimated maximum absolute error

$$ABT(I) = \max_{\Upsilon = 1,\ldots,N} \left| Y_{(\Upsilon-1)*M+I} - Y_I^*(x_\Upsilon) \right|$$

on the final mesh π.

(I) FF is a user provided subroutine with the heading

SUBROUTINE FF(X, Y, N, F, ALPHA)

where X, Y, N are as above and F is a M x N vector which should be filled with the values of the right hand side vector function f(x, y) computed at each of the grid points $X = (x_1, \ldots, x_N)$, and ordered in the same fashion as Y. The vector ALPHA (M) should be filled with $-g(y(a), y(b)) \equiv -[g_1, g_2, g_3]$.

(I) JACOB is a user provided subroutine with the heading SUBROUTINE JACϕB (X, Y, C, N, A1, B1) where X, Y, N are as above and C is a two dimensional real array of dimensions (650, 10). C should be filled with the values of the Jacobian matrix of f(x, y) with respect to y, evaluated at all the points of the mesh X. These values should be stored in the following way:

$$C = \begin{bmatrix} \frac{\partial f}{\partial y}(x_1,\ y^*(x_1)) \\ \hline \vdots \\ \hline \frac{\partial f}{\partial y}(x_N,\ y^*(x_N)) \end{bmatrix}$$

where the i, k element of the M x M submatrix $\frac{\partial f}{\partial y}(x_j,\ y_j)$ is

$\frac{\partial f_i}{\partial y_k}(x_j, y(x_j))$. The 10 x 10 matrices A1, B1 should be filled with the Jacobians of the boundary conditions in the following format

$$A_1 = \begin{matrix} p\{ \\ r\{ \end{matrix} \begin{bmatrix} g_1 y(a) \\ g_2 y(a) \\ 0 \end{bmatrix} \qquad B_1 = \begin{bmatrix} 0 \\ g_2 y(b) \\ g_3 y(b) \end{bmatrix} .$$

FF and JACOB must be in an EXTERNAL statement in the calling program.

(I) PAR(6) (Real array) is an array which allows the user to choose certain options. If PAR(1) is set to 0, then the default options are activated and the remaining elements of PAR are disregarded.

PAR (1) = 1. indicates that non default options will be active and the user must provide values for all the components of PAR.

0.< PAR(2) < 1: a continuation option with step PAR(2) is activated. The user must then have in his subroutines FF and JACOB a continuation parameter called EPSNU and passed via the statement

COMMON/C1/EPSNU

EPSNU will be changed from 0. to 1. in steps of PAR(2). This is recommended in cases of divergence of Newton's method due to poor initial conditions (see [4]).

Default value: PAR(2) = 0. (no continuation).

PAR(3) = 1. Intermediary results are printed out.

Default value: PAR(3) = 0. (no printed output with the exception of error messages).

PAR(4) = 1. Initial mesh X and trajectory Y are input by the user.

Default value: PAR(4) = 0. (The program provides an initial uniform mesh X with N points, and sets Y $\equiv$ 0).

PAR(5) = 1. This value should only be used if the problem is linear, and it will result in a more efficient performance.

Default value: PAR(5) = 0. (which is the value for nonlinear problems).

PAR(6) - Not used by PASVA2.

(ϕ) JERROR The value of this integer parameter upon output indicates one of a number of termination conditions.

JERROR = 0	Tolerance was achieved.
JERROR = 1	Data error. Check values of M, N, P, R.
JERROR = 2	The program attempted to use a grid with more than 650/M points.
JERROR = 3	Newton diverged.
JERROR = 4	Newton iteration reached round off level.

B. RESTRICTIONS

The program uses the labeled Common blocks NEWT, C1, and the auxiliary subroutines: U2DCGS, COEGEN, SYSLIN, DECOMP, SOLVE. All these are reserved names.

With the present dimensioning a maximum of MMAX = 10 equations can be solved. The maximum number of grid points NMAX depends upon the actual size M of the system being solved: M x NMAX $\leq$ 650.

Instructions on how to change these dimension statements are included as comments in the program.

C, ACCURACY

Normally, the program will end when max ABT(I) $\leq$ TOL. Otherwise,

$$I = 1, \ldots, M$$

an error message will be printed (regardless of the value of PAR (3)).

Since the program controls maximum absolute accuracy over all the components, large changes in scale will imply that some components may have more significant digits than others. In any case, TOL should not be smaller than the computer word allows, and thus, it should satisfy approximately

$$10^{-16} * \max_{x \in [a,b]} | y_j^*(x) | < TOL, \quad j = 1, \ldots, M,$$

for double precision on IBM machines.

IV. PROBLEM DESCRIPTION

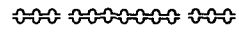
The main published source describing the problem and techniques used in this program are [3,8,9]. See also [4,5,6].

The main difference with the program described in [1] are the following:

(a) A more stable linear equation solver is used. An LU decomposition procedure which preserves the sparse structure of the matrix that appears when Newton's method is applied has been implemented. (see [7] for a description). With this change we have been able to enlarge the class of problems that can be solved with our technique. (b) The Newton solver has been made considerably more efficient by observing that after solving successfully the first system of nonlinear equations we have a fairly accurate Jacobian matrix. This fact coupled with the LU decomposition of (a) is used to implement a very fast and economical modified Newton iteration in which the Jacobians are kept fixed. Therefore, each subsequent iteration only requires the back solving of a linear system in LU form.

(c) Also using the LU decomposition it is now possible to process linear

problems in a more economical way than before, since now the matrix of the system will remain constant throughout the process (unless the mesh is changed, in which case we have to restart).



RESTRICTIONS

The package CAPIL, including PASVA2, uses the following labeled CØMMØN blocks

NEWT, C1, CYLIN, F1, ELIPSE ,

and the SUBRØUTINEs

PASVA2, U2DCGS, CØEGEN, SYSLIN, DECØMP, SØLVE,
CYLIND, CYLCUR, ELLIPS, FF1, FF2, JACØB1, JACØB2,
FCUR1, FCUR2, JCUR1, JCUR2, IMPR .

All these are reserved names. Each of CYLIND, CYLCUR, and ELLIPS contains its own versions of several of the above SUBRØUTINEs; thus only one of these programs should be loaded and run at a time.

Input is read from logical unit 5, and output is written on logical unit 6. During execution several of the subprograms may write output.

ACCURACY

Normally the program will terminate with maximum absolute error over all components $\leq$ TØL. In the default option PAR(1) = 1., TØL is set to 0.005 and N = min[40,max(11,NB)], where NB is the largest integer in $|6.*\text{BØND}|^{\frac{1}{2}}$. If this tolerance fails to be met an error message will be printed.

TØL should not be smaller than a value permitted by the computer word length; a good rule of thumb is to take

$$100 \times \varepsilon \times \max(V,RADIUS) < TØL \ ,$$

where ε is the relative precision of floating point arithmetic ($\varepsilon \approx 10^{-8}$ for single precision on the UNIVAC 1100 Series).

STORAGE

The compiled program for each container, including PASVA2, occupies approximately 17,500 (decimal) locations in memory on the CDC 7600, including CØMMØN blocks and data storage arrays. Approximately 14,250 (decimal) of the locations are occupied by PASVA2. Each program, including comments, consists of approximately 1,800 Fortran statements, of which approximately 1,300 are for PASVA2.

REFERENCES

[1] P. Concus, Static menisci in a vertical right circular cylinder, J. Fluid Mech. 34 (1968), pp. 481-495.

[2] P. Concus and R. Finn, The shape of a pendent liquid drop, Philos. Trans. Roy. Soc. Lond. A (1979) (to appear).

[3] M. Lentini and V. Pereyra, An adaptive finite difference solver for nonlinear two-point boundary problems with mild boundary layers, SIAM J. Num. Anal. 14 (1977), pp. 91-111.

[4] M. Lentini and V. Pereyra, A variable order finite difference method for nonlinear multipoint boundary value problems, Math. Comp. 28 (1974), pp. 981-1003.

[5] M. Lentini and V. Pereyra, Boundary problem solvers for first order systems based on deferred corrections, in Numerical Solutions of Boundary Value Problems for Ordinary Differential Equations, Academic Press, New York (1975).

[6] V. Pereyra, High order finite difference solution of differential equations, Stanford University Comp. Sc. Report STAN-CS-73-348 (1973).

[7] H. B. Keller, Accurate difference methods for nonlinear two-point boundary value problem, SIAM J. Num. Anal. 11 (1974), pp. 305-320.

[8] V. Pereyra, W. H. K. Lee, and H. B. Keller, Solving two point seismic ray tracing problems in a heterogeneous medium, Part 1: A general adaptive finite difference method (to appear in Bull. A. Seism. Soc.).

[9] V. Pereyra, PASVA3: An adaptive finite difference FORTRAN program for first order, nonlinear, ordinary boundary problems (to appear in the Proc. Working Conf. on Codes for Boundary Value Problems in ODE's, Houston, Texas, May 1978; Lecture Notes in Math., Springer-Verlag, Berlin).

PROGRAM LISTINGS

Listings of the programs in CAPIL, including PASVA2, follow. These programs are the (single-precision) versions that were run on the LBL CDC 7600 computer. The versions run on other computers may differ slightly. For example, on the UNIVAC 1100 series, the program cards (first cards) of MNCLND, MNCLCR, MNELPS would have a different format, appropriate to the operating system, and the quantity EPSMAC (see the second page of the listing of PASVA2), which is approximately ten times the relative precision of the machine floating point arithmetic, would have a value of approximately 10^{-7}.

These programs are developmental in nature, and listings of them are included here primarily for reference purposes, not as text to be read. Most of the crucial items are noted in the comment cards, but otherwise no attempt was made to prepare within the programs complete, self-contained documentation.

```
      PROGRAM MNCLND(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
C     MAIN PROGRAM FOR COMPUTING THE MENISCUS
C     IN A RIGHT.CIRCULAR CYLINDER WITH
C     FLAT BOTTOM.
C                              --------SINGLE PRECISION-------
C     ************************************
C
      DIMENSION PAR(6),S(50),U(50),R(50),X(50),Y(350)
      COMMON/CYLIN/VV,GAM,RADIUS,S,U,R,X,Y,TOL,N
C
      READ (5,10) PAR(1),PAR(3),PAR(4)
   10 FORMAT (3E8.0)
C
    1 CONTINUE
C
         READ (5,20) II,N,BOND,VOLUME,ANGLE,RADIUS,TOL
   20    FORMAT (2I4,7E8.0)
C
         IF (II .EQ. 999) STOP
         CALL CYLIND(BOND,VOLUME,ANGLE,RADIUS,II,PAR)
      GO TO 1
      END
```

```
      SUBROUTINE CYLIND(BOND,V,GAMMA,RADIUS,II,PAR)
C
C     RIGHT CIRCULAR CYLINDER WITH A FLAT BOTTOM
C     II=1   LARGE LIQUID VOLUME
C     II=2   SMALL LIQUID VOLUME
C     GAMMA = CONTACT ANGLE IN DEGREES
C     BOND = BOND NUMBER.
C     PAR = PARAMETER ARRAY OF LENGTH 6
C
C     PAR(1) .NE. 2.  CYLIND WILL SET
C     N = MIN(40,MAX(11,SQRT(6.*ABS(BOND))))
C     X = UNIFORM MESH IN <0,1] WITH N
C     POINTS (COUNTING THE END POINTS).
C     Y = APPROPRIATE FIRST APPROXIMATION
C     ON  X.
C     NO INTERMEDIARY OUTPUT WILL BE PRINTED.
C     NO OTHER ELEMENTS OF  PAR  NEED BE
C     FILLED.
C
C     PAR(1) = 2.     THE USER MUST GIVE VALUES
C     FOR  N, PAR(3), PAR(4), AND  TOL (SEE
C     BELOW).
C
C     N .LE. 50 NUMBER OF POINTS IN INITIAL MESH.
C     UPON OUTPUT IT WILL CONTAIN THE NUMBER
C     OF POINTS IN FINAL MESH.
C
C     X = INITIAL MESH WITH N POINTS IN <0,1].
C     X(1) MUST BE = 0.E0 AND  X(N) = 1.E0.
C     THE MESH NEED NOT BE UNIFORM.
C
C     Y = INITIAL APPROXIMATION TO THE SIX
C     VARIABLES  U,R,PSI,S,H,V  ON MESH X.
C
C     PAR(3) = 0.  NO INTERMEDIARY OUTPUT IS
C     PRODUCED.
C     PAR(3) = 1.  INTERMEDIARY OUTPUT IS
C     PRINTED.
C
C     PAR(4) = 1.  THE USER GIVES VALUES FOR X,Y.
C     PAR(4) = 0.  CYLIND ASSIGNS VALUES TO X,Y.
C
C     IF  PAR(1) = 2.  THE CALLING PROGRAM
C     SHOULD CONTAIN THE STATEMENT
C
C     COMMON/CYLIN/VV,GAM,RADIUS,S(50),U(50),RR(50),
C          X(50),Y(350),TOL,N
C
C     TOL =  MAXIMUM DESIRED ABSOLUTE ERROR OVER ALL
C     COMPONENTS OF THE SOLUTION AND ALL MESH POINTS.
C     TOL  SHOULD BE LARGER THAN THE MACHINE PRECISION.
C
C     THE FINAL VALUES OF THE MENISCUS CURVE  U(S) , R(S)
C     ARE AVAILABLE UPON EXIT THROUGH COMMON /CYLIN/
C     FOR FURTHER PROCESSING (I.E. PLOTTING).
C
C
C     ******************************************
C
      EXTERNAL FF1,JACOB1
      EXTERNAL FF2,JACOB2
```

```
      COMMON /F1/ BO
      COMMON /CYLIN/ VV,GAM,RAD,S,U,RR,X,Y,TOL,N
      INTEGER P,R
      DIMENSION ALPHA(7),A1(7, 7),B1(7, 7),X(50),
     2  Y(350),ABT(7),PAR(6)
     3          ,S(50),U(50),RR(50)
      IF(II .NE. 1 .AND. II .NE. 2) II = 1
      WRITE(6,61)
      WRITE(6,60)
      WRITE(6,70)
      WRITE(6,50)
      WRITE(6,70)
      WRITE(6,51) RADIUS
      WRITE(6,70)
      IF (II .NE. 1) GO TO 3
      WRITE(6,52)V
      GO TO 4
3     WRITE(6,53)V
4     WRITE(6,70)
      WRITE(6,54)GAMMA
      WRITE(6,70)
      WRITE(6,56)BOND
      WRITE(6,70)
      WRITE(6,60)
      IF(V .GT. 0.E0 .AND. RADIUS .GT. 0.E0 .AND. GAMMA .GE. 0.E0)
     2          GO TO 10
      WRITE(6,98)
      RETURN
10    BO=BOND
      ICON=0
      M=6
      R=1
      P=3
      A=0.E0
      B=1.E0
      PIH=1.5707963267948966E0
      GAMMA=PIH*GAMMA/90.E0
      VV=V
      GAM=GAMMA
      RAD=RADIUS
      IF(PAR(1) .NE. 1.E0 .AND. PAR(1) .NE. 2.E0)PAR(1)=1.E0
      NO=N
      NB=SQRT(6.*ABS(BOND))
      PAR(2)=0.E0
      PAR(5)=0.E0
      IF(PAR(1) .EQ. 2.E0) GO TO 30
      PAR(1)=1.E0
      PAR(3)=0.E0
      TOL=.5E-2
      RAD=RADIUS
30    IF(II .EQ. 2) GO TO 300
100      IF(ICON .EQ. 3) GO TO 2000
      ICON=ICON+1
      IF(ICON .EQ. 3) WRITE(6,95)
      IF(PAR(1) .EQ. 2.E0) GO TO 32
      N=MIN0(40,MAX0(11,NB))
      GO TO 34
32    N=NO
      IF(PAR(4) .EQ. 1.E0 .AND. ICON .EQ. 1) GOTO 36
34    H=1.E0/(N-1)
      PAR(4)=1.E0
      DO 20 I=1,N
```

```
      X(I)=(I-1)*H
      KI=(I-1)*6
      Y(KI+1)=V/(2.E0*PIH*RADIUS**2)
      Y(KI+2)=X(I)*RADIUS
      Y(KI+3)=(PIH-GAMMA)*X(I)
      Y(KI+4)=RADIUS
      Y(KI+5)=-BOND*Y(1)*.5E0
      Y(KI+6)=X(I)**2*V
   20 CONTINUE
   36 CALL PASVA2(M,N,P,R,A,B,ALPHA,A1,B1,TOL,X,Y,
     2          ABT,FF1,JACOB1,PAR,JERROR)
C
C     THE NEXT BLOCK OF STATEMENTS CHECKS WHETHER THE
C     COMPUTED SURFACE LIES WITHIN THE CONTAINER.
C
      INSIDE=1
      UTOL=-TOL
      RTOL=RADIUS+TOL
      DO 37 INDEX=1,N
         KI=(INDEX-1)*6
         IF(Y(KI+1) .LT. UTOL .OR. Y(KI+2) .GT. RTOL) INSIDE=0
   37 CONTINUE
C
      IF(INSIDE .EQ. 1 .AND. JERROR .NE. 3) GO TO 500
  300 IF(ICON .EQ. 3) GO TO 2000
      ICON = ICON+2
      IF(ICON .EQ.2) GO TO 41
      IF(JERROR.NE.3) GO TO 40
      WRITE(6,96)
      GO TO 41
   40 CALL IMPR(X,Y,N,ABT,JERROR)
      WRITE(6,90)
   41 IF(PAR(1) .EQ. 2.E0) GO TO 42
      N=MINO(40,MAXO(11,NB))
      GO TO 44
   42 N=NO
      IF(PAR(4) .EQ. 1.E0 .AND. ICON .EQ. 2)GO TO 46
   44 H=1.E0/(N-1)
      UBAR=6.E0*V/(5.E0*PIH*RADIUS**2)
      DO 200 I=1,N
      X(I)=(I-1)*H
      KI=6*(I-1)
      Y(KI+2)=.5E0*RADIUS*(X(I)+1.E0)
      Y(KI+1)=UBAR*(2.E0*(Y(KI+2)-RADIUS)/RADIUS+1.E0)
      Y(KI+3)=(PIH-2.E0*GAMMA)*X(I)+GAMMA
      Y(KI+4)=SQRT(UBAR**2+.25E0*RADIUS**2)
      Y(KI+5)=0.E0
      Y(KI+6)=UBAR*(Y(KI+2)**2*(2.E0*Y(KI+2)/
     2          (3.E0*RADIUS)-.5E0)+RADIUS**2/24.E0)*4.*PIH
  200 CONTINUE
      PAR(4)=1.EC
   46 CALL PASVA2(M,N,P,R,A,B,ALPHA,A1,B1,TOL,X,Y,ABT,
     2       FF2,JACOB2,PAR,JERROR)
      IF(JERROR .EQ. 3) GO TO 100
  500 CALL IMPR(X,Y,N,ABT,JERROR)
      GAMMA=90.E0*GAMMA/PIH
   50 FORMAT(1H ,45H*    MENISCUS COMPUTATION FOR A RIGHT CIRCULAR ,
     2   14H CYLINDER    *,/,30H *    WITH A FLAT BOTTOM.       ,29X,1H*)
   51 FORMAT(29H *    CYLINDER RADIUS          =,E14.6,16X,1H*)
   52 FORMAT(1H ,28H*    (LARGE) LIQUID VOLUME  =,E14.6,16X,1H* )
   53 FORMAT(1H ,28H*    (SMALL) LIQUID VOLUME  =,E14.6,16X,1H* )
   54 FORMAT(1H ,17H*    CONTACT ANGLE,10X,1H=,F7.3,10H    DEGREES,13X,
```

```
     2      1H*)
    56     FORMAT(1H ,15H*   BOND NUMBER,12X,1H=,E14.6,16X,1H*)
    60     FORMAT(1H ,59(1H*))
    61     FORMAT(1H1)
    70     FORMAT(1H ,1H*,57X,1H*)
           DO 1000 I=1,N
           S(I)=X(I)*Y(4)
           KI=(I-1)*6
           U(I)=Y(KI+1)
  1000     RR(I)=Y(KI+2)
           RETURN
  2000     WRITE(6,80)
           RETURN
    80     FORMAT(31H ****FAILURE-CHECK YOUR DATA***/)
    90     FORMAT(1H ,42H**** PART OF THE ABOVE SOLUTION SURFACE   /
     2            38H      LIES OUTSIDE THE CONTAINER.           /
     3            38H SMALL VOLUME OPTION WILL BE TRIED****//)
    98     FORMAT(17H ***DATA ERROR***//30H EITHER VOLUME,RADIUS OR GAMMA,
     2            12H IS NEGATIVE)
    95     FORMAT(43H ****FAILURE-TRYING LARGE VOLUME OPTION****//)
    96     FORMAT(43H ****FAILURE-TRYING SMALL VOLUME OPTION****//)
           END
C
C      ------------------------------------------------
C
           SUBROUTINE FF1(X,Y,N,FF,ALPHA)
           DIMENSION X(1),Y(1),FF(1)
     2     ,ALPHA(1)
           COMMON/CYLIN/V,GAMMA,RAD
           COMMON /F1/ B
           PI=3.141592653589792E0
           DO 10 I =1,N
           KI=(I-1)*6
           SENOX=SIN(Y(KI+3))
           COSX=COS(Y(KI+3))
           S=Y(KI+4)
           FF(KI+1)=S*SENOX
           FF(KI+2)=S*COSX
           IF(I .EQ. 1 .OR. ABS(Y(KI+2)) .LT. 1.E-10) GO TO 5
           FF(KI+3)=S*(B*Y(KI+1)+2.E0*Y(KI+5)-SENOX/Y(KI+2))
           GO TO 6
    5      FF(KI+3)=S*(.5E0*B*Y(KI+1)+Y(KI+5))
    6       FF(KI+4)=0.E0
           FF(KI+5)=0.E0
           FF(KI+6)=PI*S*SENOX*(RAD**2-Y(KI+2)**2)
   10      CONTINUE
           KI=6*(N-1)
           ALPHA(1)=-Y(2)
           ALPHA(2)=-Y(3)
           ALPHA(3)=-Y(6)
           ALPHA(4)=V-Y(KI+6)-PI*Y(1)*RAD**2
           ALPHA(5)=.5E0*PI-GAMMA-Y(KI+3)
           ALPHA(6)=RAD-Y(KI+2)
           RETURN
           END
C
C      ------------------------------------------------
C
           SUBROUTINE JACOB1(X,Y,C,N,A1,B1)
           DIMENSION Y(1),X(1),C(350,7),
     2     A1(7,7),B1(7,7)
           COMMON /F1/ B
```

```
      COMMON/CYLIN/ V,GAMMA,RAD
      TPI=6.2831853071795 84E0
      PI=TPI*.5E0
      DO 60 I=1,6
      DO 60 J=1,6
      A1(I,J)=0.E0
   60 B1(I,J)=0.E0
      A1(1,2)=1.E0
      A1(2,3)=1.E0
      A1(3,6)=1.E0
      A1(4,1)=PI*RAD**2
      B1(4,6)=1.E0
      B1(5,3)=1.E0
      B1(6,2)=1.E0
      N6=6*N
      DO 50 I=1,N6
      DO 50 J=1,6
   50 C(I,J)=0.E0
      DO 10 I=1,N
            KI=(I-1)*6
            SINX=SIN(Y(KI+3))
            COSX=COS(Y(KI+3))
            S=Y(KI+4)
            C(KI+1,3)=COSX*S
            C(KI+1,4)=SINX
            C(KI+2,3)=-S*SINX
            C(KI+2,4)=COSX
            C(KI+3,1)=S*B
            IF(I .EQ. 1 .OR. ABS(Y(KI+2)) .LT. 1.E-10) GO TO 5
            C(KI+3,2)=S*SINX/(Y(KI+2)**2)
            C(KI+3,3)=-S*COSX/Y(KI+2)
            C(KI+3,4)=B*Y(KI+1)+2.E0*Y(KI+5)-SINX/Y(KI+2)
            C(KI+3,5)=2.E0*S
            GO TO 6
    5       C(KI+3,4)=.5E0*B*Y(KI+1)+Y(KI+5)
      C(KI+3,1)=.5E0*S*B
            C(KI+3,5)=S
    6       C(KI+6,2)=-TPI*S*SINX*Y(KI+2)
            C(KI+6,3)=PI*S*COSX*(RAD**2-Y(KI+2)**2)
            C(KI+6,4)=PI*SINX*(RAD**2-Y(KI+2)**2)
   10       CONTINUE
      RETURN
      END
C
C     ---------------------------------------------
C
      SUBROUTINE IMPR(X,Y,N,ABT,JERROR)
      DIMENSION X(1),Y(1),ABT(1)
      IF (JERROR .GT. 0) WRITE(6,10)
      WRITE(6,70)
      WRITE(6,20)
      DO 5 I=1,N
      II=(I-1)*6
      S=X(I)*Y(4)
      TE=90.E0*Y(II+3)/1.5707963267948 96E0
    5 WRITE(6,30) S,Y(II+2),Y(II+1),TE
      WRITE(6,40) ABT(2),ABT(1),ABT(3)
      WRITE(6,50) Y(5),ABT(5)
      WRITE(6,70)
      RETURN
   10 FORMAT(1H /44H   THERE HAS BEEN AN ERROR WARNING IN THIS  /
     2  39H  COMPUTATION. CHECK ANSWERS CAREFULLY./)
```

```
   20    FORMAT(1H //,9X,1HS,14X,1HR,13X,1HU,12X,3HPSI//)
   30    FORMAT(1H ,3E15.6,F13.4)
   40    FORMAT(1H ,/4X,7HMAXIMUM/4X,8HABSOLUTE,3X,3E15.2/4X,
        2          6HERRORS/)
   50    FORMAT(1H /7X,
        2    9H    H    = ,E15.6,9X,9H ERROR ≤ ,E10.2///)
   70           FORMAT(1H ,//59(1H-))
         END
C
C        ------------------------------------------
C
         SUBROUTINE FF2(X,Y,N,FF,ALPHA)
         DIMENSION X(1),Y(1),FF(1)
        2 ,ALPHA(1)
         COMMON/CYLIN/V,GAMMA,RAD
         COMMON /F1/ B
         PI=3.14159265358979 2E0
         DO 10 I =1,N
         KI=(I-1)*6
         SENOX=SIN(Y(KI+3))
         COSX=COS(Y(KI+3))
         S=Y(KI+4)
         FF(KI+1)=S*SENOX
         FF(KI+2)=S*COSX
         FF(KI+3)=S*(B*Y(KI+1)+2.E0*Y(KI+5)-SENOX/Y(KI+2))
    6     FF(KI+4)=0.E0
         FF(KI+5)=0.E0
         FF(KI+6)=2.E0*PI*S*COSX*Y(KI+2)*Y(KI+1)
   10    CONTINUE
         KI=6*(N-1)
         ALPHA(1)=-Y(1)
         ALPHA(2)=GAMMA-Y(3)
         ALPHA(3)=-Y(6)
         ALPHA(4)=V-Y(KI+6)
         ALPHA(5)=.5E0*PI-GAMMA-Y(KI+3)
         ALPHA(6)=RAD-Y(KI+2)
         RETURN
         END
C
C        ------------------------------------------
C
         SUBROUTINE JACOB2(X,Y,C,N,A1,B1)
         DIMENSION Y(1),X(1),C(350,7),
        2 A1(7,7),B1(7,7)
         COMMON /F1/ B
         TPI=6.283185307179584E0
         PI=TPI*.5E0
         DO 60 I=1,6
         DO 60 J=1,6
         A1(I,J)=0.E0
   60    B1(I,J)=0.E0
         A1(1,1)=1.E0
         A1(2,3)=1.E0
         A1(3,6)=1.E0
         B1(4,6)=1.E0
         B1(5,3)=1.E0
         B1(6,2)=1.E0
         N6=6*N
         DO 50 I=1,N6
         DO 50 J=1,6
   50    C(I,J)=0.E0
         DO 10 I=1,N
```

```
            KI=(I-1)*6
            SINX=SIN(Y(KI+3))
            COSX=COS(Y(KI+3))
            S=Y(KI+4)
            C(KI+1,3)=COSX*S
            C(KI+1,4)=SINX
            C(KI+2,3)=-S*SINX
            C(KI+2,4)=COSX
            C(KI+3,1)=S*B
            C(KI+3,2)=S*SINX/(Y(KI+2)**2)
            C(KI+3,3)=-S*COSX/Y(KI+2)
            C(KI+3,4)=B*Y(KI+1)+2.E0*Y(KI+5)-SINX/Y(KI+2)
            C(KI+3,5)=2.E0*S
      C(KI+6,1)=TPI*COSX*S*Y(KI+2)
    6       C(KI+6,2)=TPI*S*COSX*Y(KI+1)
            C(KI+6,3)=-TPI*S*SINX*Y(KI+2)*Y(KI+1)
            C(KI+6,4)=TPI*COSX*Y(KI+2)*Y(KI+1)
   10       CONTINUE
      RETURN
      END
```

```
      PROGRAM MNCLCR(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
C     MAIN PROGRAM FOR COMPUTING THE MENISCUS
C     IN A RIGHT CIRCULAR CYLINDER WITH
C     SPHEROIDAL BOTTOM.
C                              ---------SINGLE PRECISION--------
C     ************************************
C
      DIMENSION PAR(6),S(50),U(50),R(50),X(50),Y(350)
      COMMON/CYLIN/VV,GAM,RADIUS,VERT,HORZ,S,U,R,X,Y,TOL,N
C
      READ (5,10) PAR(1),PAR(3),PAR(4)
   10 FORMAT (3E8.0)
C
    1 CONTINUE
C
         READ (5,20) II,N,BOND,VOLUME,ANGLE,RADIUS,TOL,HORZ,VERT
   20    FORMAT (2I4,7E8.0)
C
         IF (II .EQ. 999) STOP
         CALL CYLCUR(BOND,VOLUME,ANGLE,RADIUS,II,PAR,VERT,HORZ)
      GO TO 1
      END
```

```
      SUBROUTINE CYLCUR(BOND,V,GAMMA,RADIUS,II,PAR,VERT,HORZ)
C
C     RIGHT CIRCULAR CYLINDER WITH AN ELLIPSOID OF REVOLUTION BOTTOM.
C     II=1 - LARGE LIQUID VOLUME
C     II=2 - SMALL LIQUID VOLUME
C     GAMMA = CONTACT ANGLE IN DEGREES
C     BOND = BOND NUMBER.
C     VERT = VERTICAL SEMI-AXIS OF ELLIPSE.
C     HORZ = HORIZONTAL  ...   ...   ...
C     HORZ  SHOULD BE .GE.  RADIUS OF CYLINDER.
C     PAR = PARAMETER ARRAY OF LENGTH 6.
C
C              ************
C
C     PAR(1) .NE. 2. CYLIND WILL SET -
C     N = MIN(40,MAX(11,SQRT(6.*ABS(BOND))))
C     X = UNIFORM MESH IN <0,1> WITH N
C     POINTS (COUNTING THE END POINTS).
C     Y = APPROPRIATE FIRST APPROXIMATION
C     ON  X.
C     NO INTERMEDIARY OUTPUT WILL BE PRINTED.
C     NO OTHER ELEMENTS OF  PAR  NEED BE
C     FILLED.
C
C     PAR(1) = 2.    THE USER MUST GIVE VALUES
C     FOR  N, PAR(3), PAR(4), AND  TOL (SEE
C     BELOW).
C
C     N .LE. 50 NUMBER OF POINTS IN INITIAL MESH.
C     UPON OUTPUT IT WILL CONTAIN THE NUMBER
C     OF POINTS IN FINAL MESH.
C
C     X = INITIAL MESH WITH N POINTS IN <0,1>.
C     X(1) MUST BE = 0.E0 AND  X(N) = 1.E0.
C     THE MESH NEED NOT BE UNIFORM.
C
C     Y = INITIAL APPROXIMATION TO THE SIX
C     VARIABLES  U,R,PSI,S,H,V  ON MESH X.
C
C     PAR(3) = 0. NO INTERMEDIARY OUTPUT IS
C     PRODUCED.
C     PAR(3) = 1.  INTERMEDIARY OUTPUT IS
C     PRINTED.
C
C     PAR(4) = 1. THE USER GIVES VALUES FOR X,Y.
C     PAR(4) = 0. CYLCUR ASSIGNS VALUES TO X,Y.
C
C     IF  PAR(1) = 2. THE CALLING PROGRAM
C     SHOULD CONTAIN THE STATEMENT -
C
C     COMMON/CYLIN/VV,GAM,RADIUS,VERT,HORZ,S(50),U(50),RR(50),
C          X(50),Y(350),TOL,N
C
C     TOL = MAXIMUM DESIRED ABSOLUTE ERROR OVER ALL
C     COMPONENTS OF THE SOLUTION AND ALL MESH POINTS.
C     TOL  SHOULD BE LARGER THAN THE MACHINE PRECISION.
C
C     THE FINAL VALUES OF THE MENISCUS CURVE  U(S) , R(S)
C     ARE AVAILABLE UPON EXIT THROUGH COMMON /CYLIN/
C     FOR FURTHER PROCESSING (I.E. PLOTTING).
C
```

```
C
C       ******************************************
C
        EXTERNAL FCUR1,JCUR1
        EXTERNAL FCUR2,JCUR2
        COMMON /F1/ BO
        COMMON /CYLIN/ VV,GAM,RAD,VER,HOR,S,U,RR,X,Y,TOL,N
        INTEGER P,R
        DIMENSION ALPHA(7),A1(7, 7),B1(7, 7),X(50),
     2  Y(350),ABT(7),PAR(6)
     3  ,S(50),U(50),RR(50)
        IF(II .NE. 1 .AND. II .NE. 2) II = 1
        WRITE(6,61)
        WRITE(6,60)
        WRITE(6,70)
        WRITE(6,50)
        WRITE(6,70)
        WRITE(6,51) RADIUS
        WRITE(6,70)
        WRITE(6,210) VERT
        WRITE(6,70)
        WRITE(6,220) HORZ
        E=VERT/HORZ
        UC=-E*SQRT(HORZ**2-RAD**2)
        WRITE(6,70)
        WRITE(6,230)UC
        WRITE(6,70)
        IF (II .NE. 1) GO TO 3
        WRITE(6,52)V
        GO TO 4
  3     WRITE(6,53)V
  4     WRITE(6,70)
        WRITE(6,54)GAMMA
        WRITE(6,70)
        WRITE(6,56)BOND
        WRITE(6,70)
        WRITE(6,60)
        IF(V .GT. 0.E0 .AND. RADIUS .GT. 0.E0 .AND. GAMMA .GE. 0.E0
     2          .AND. HORZ .GE. RADIUS) GO TO 10
        WRITE(6,98)
        RETURN
 10     BO=BOND
        ICON=0
        M=6
        R=1
        P=3
        A=0.E0
        B=1.E0
        PIH=1.570796326794896E0
        PI=2.E0*PIH
        GAMMA=PIH*GAMMA/90.E0
        VV=V
        GAM=GAMMA
        VER=VERT
        HOR=HORZ
        RAD=RADIUS
        IF(PAR(1) .NE. 1.E0 .AND. PAR(1) .NE. 2.E0)PAR(1)=1.E0
        NO=N
        NB=SQRT(6.*ABS(BOND))
        PAR(2)=0.E0
        PAR(5)=0.E0
        IF(PAR(1) .EQ. 2.E0) GO TO 30
```

```
      PAR(1)=1.E0
      PAR(3)=0.E0
      TOL=.5E-2
      RAD=RADIUS
30    IF(II .EQ. 2) GO TO 300
100      IF(ICON .EQ. 3) GO TO 2000
      ICON=ICON+1
      IF(ICON .EQ. 3) WRITE(6,95)
      IF(PAR(1) .EQ. 2.E0) GO TO 32
      N=MIN0(40,MAX0(11,NB))
      GO TO 34
32    N=NO
      IF(PAR(4) .EQ. 1.E0 .AND. ICON .EQ. 1) GO TO 36
34    H=1.E0/FLOAT(N-1)
      PAR(4) = 1.E0
      UBO = (V/PI+HORZ**2*(2.*UC**3/(3.E0*VERT**2)+2.E0*VERT/3.E0))/
     2         RADIUS**2 + UC
      DO 20 I=1,N
      X(I)=FLOAT(I-1)*H
      KI=(I-1)*6
      Y(KI+1)=UBO
      Y(KI+2)=X(I)*RADIUS
      Y(KI+3)=(PIH-GAMMA)*X(I)
      Y(KI+4)=RADIUS
      Y(KI+5)=-BOND*UBO*.5E0
      Y(KI+6)=X(I)**2*V
20    CONTINUE
36    CALL PASVA2(M,N,P,R,A,B,ALPHA,A1,B1,TOL,X,Y,
     2          ABT,FCUR1,JCUR1,PAR,JERROR)
C
C     THE NEXT BLOCK OF STATEMENTS CHECK WHETHER THE COMPUTED
C     SURFACE LIES WITHIN THE CONTAINER.
C
      INSIDE=1
      DO 37 INDEX=1,N
         KI=(INDEX-1)*6
         UTOL=((ABS(Y(KI+1)-UC)+TOL)/VERT)**2
         RTOL=((Y(KI+2)+TOL)/HORZ)**2
         IF((UTOL+RTOL) .LT. 1.) INSIDE=0
         IF(Y(KI+2) .GT. (RADIUS+TOL)) INSIDE=0
         ZTOL=-TOL
         IF(Y(KI+1) .LT. ZTOL) INSIDE=0
37    CONTINUE
C
      IF(INSIDE .EQ. 1 .AND. JERROR .NE. 3) GO TO 500
300   IF(ICON .EQ. 3) GO TO 2000
      ICON = ICON+2
      IF(ICON .EQ.2) GO TO 41
      IF(JERROR.NE.3) GO TO 40
      WRITE(6,96)
      GO TO 41
40    CALL IMPR(X,Y,N,ABT,JERROR)
      WRITE(6,90)
41    IF(PAR(1) .EQ. 2.E0) GO TO 42
      N=MIN0(40,MAX0(11,NB))
      GO TO 44
42    N=NO
      IF(PAR(4) .EQ. 1.E0 .AND. ICON .EQ. 2) GO TO 46
44    H=1.E0/FLOAT(N-1)
      UBO=UC+SQRT(ABS(VERT**2-(E*RADIUS*.5)**2))
      DO 200 I=1,N
      X(I)=FLOAT(I-1)*H
```

```
      KI=6*(I-1)
      Y(KI+1)=UBO
      Y(KI+2)=RADIUS*(X(I)+1.E0)*.5E0
      Y(KI+3)=(PIH-2.E0*GAMMA)*X(I)+GAMMA
      Y(KI+4)=.5E0*RADIUS
      Y(KI+5)=0.E0
      Y(KI+6)=PIH*(Y(KI+2)**2-RADIUS**2*.25E0)*UBO
  200 CONTINUE
      PAR(4)=1.E0
   46 CALL PASVA2(M,N,P,R,A,B,ALPHA,A1,B1,TOL,X,Y,ABT,
     2      FCUR2,JCUR2,PAR,JERROR)
      IF(JERROR .EQ. 3) GO TO 100
  500 CALL IMPR(X,Y,N,ABT,JERROR)
   50 FORMAT(1H ,45H*    MENISCUS COMPUTATION FOR A RIGHT CIRCULAR ,
     2   14H CYLINDER     *,/,30H *    WITH A SPHEROIDAL BOTTOM.,29X,1H*)
   51 FORMAT(29H *    CYLINDER RADIUS          =,E14.6,16X,1H*)
   52 FORMAT(1H ,28H*    (LARGE) LIQUID VOLUME  =,E14.6,16X,1H* )
   53 FORMAT(1H ,28H*    (SMALL) LIQUID VOLUME  =,E14.6,16X,1H* )
   54 FORMAT(1H ,17H*    CONTACT ANGLE,10X,1H=,F7.3,10H   DEGREES,13X,
     2      1H*)
   56 FORMAT(1H ,15H*    BOND NUMBER,12X,1H=,E14.6,16X,1H*)
   61 FORMAT(1H1)
   60 FORMAT(1H ,59(1H*))
   70 FORMAT(1H ,1H*,57X,1H*)
  210 FORMAT(1H ,33H*    ELLIPSE VERTICAL SEMI-AXIS  =,E14.6,11X,1H*)
  220 FORMAT(1H ,33H*    HORIZONTAL SEMI-AXIS        =,E14.6,11X,1H*)
  230 FORMAT(1H ,18H*    CENTER AT (0 ,
     2          E11.4,1H),28X,1H*)
      DO 1000 I=1,N
      S(I)=X(I)*Y(4)
      KI=(I-1)*6
      U(I)=Y(KI+1)
 1000 RR(I)=Y(KI+2)
      RETURN
 2000 IF(Y(1) .LT. 0.E0) WRITE(6,90)Y(1)
      WRITE(6,80)
      RETURN
   80 FORMAT(31H ****FAILURE-CHECK YOUR DATA***/)
   90 FORMAT(1H ,42H**** PART OF THE ABOVE SOLUTION SURFACE    /
     2          38H      LIES OUTSIDE THE CONTAINER.          /
     3          38H SMALL VOLUME OPTION WILL BE TRIED****//)
   98 FORMAT(17H ***DATA ERROR***//30H EITHER VOLUME,RADIUS OR GAMMA,
     2          12H IS NEGATIVE)
   95 FORMAT(43H ****FAILURE-TRYING LARGE VOLUME OPTION****//)
   96 FORMAT(43H ****FAILURE-TRYING SMALL VOLUME OPTION****//)
      END
C
C-----------------------------------------------
C
      SUBROUTINE FCUR1(X,Y,N,FF,ALPHA)
      DIMENSION X(50),Y(350),FF(350)
     2 ,ALPHA(7)
      COMMON/CYLIN/V,GAMMA,RAD,VERT,HORZ
      COMMON /F1/ B
      PI=3.14159265358979E0
      DO 10 I =1,N
      KI=(I-1)*6
      SENOX=SIN(Y(KI+3))
      COSX=COS(Y(KI+3))
      S=Y(KI+4)
      FF(KI+1)=S*SENOX
      FF(KI+2)=S*COSX
```

```
      IF(I .EQ. 1 .OR. ABS(Y(KI+2)) .LT. 1.E-10) GO TO 5
      FF(KI+3)=S*(B*Y(KI+1)+2.E0*Y(KI+5)-SENOX/Y(KI+2))
      GO TO 6
5     FF(KI+3)=S*(.5E0*B*Y(KI+1)+Y(KI+5))
6      FF(KI+4)=0.E0
      FF(KI+5)=0.E0
      FF(KI+6)=PI*S*SENOX*(RAD**2-Y(KI+2)**2)
10    CONTINUE
      E2=(VERT/HORZ)**2
      UC=-VERT/HORZ * SQRT(ABS(HORZ**2-RAD**2))
      KI=6*(N-1)
      ALPHA(1)=-Y(2)
      ALPHA(2)=-Y(3)
      ALPHA(3)=-Y(6)
      ALPHA(4)=V-Y(KI+6)-PI*Y(1)*RAD**2+PI*UC*RAD**2+(VERT**3+UC**3)
     2         *2.E0*PI/(3.E0*E2)
      ALPHA(5)=.5E0*PI-GAMMA-Y(KI+3)
      ALPHA(6)=RAD-Y(KI+2)
      RETURN
      END
C
C-----------------------------------------------------
C
      SUBROUTINE JCUR1(X,Y,C,N,A1,B1)
      DIMENSION Y(350),X(50),C(350,7),
     2  A1(7,7),B1(7,7)
      COMMON /F1/ B
      COMMON/CYLIN/ V,GAMMA,RAD
      TPI=6.2831853071795 84E0
      PI=TPI*.5E0
      DO 60 I=1,6
      DO 60 J=1,6
      A1(I,J)=0.E0
60    B1(I,J)=0.E0
      A1(1,2)=1.E0
      A1(2,3)=1.E0
      A1(3,6)=1.E0
      A1(4,1)=PI*RAD**2
      B1(4,6)=1.E0
      B1(5,3)=1.E0
      B1(6,2)=1.E0
      N6=6*N
      DO 50 I=1,N6
      DO 50 J=1,6
50    C(I,J)=0.E0
      DO 10 I=1,N
            KI=(I-1)*6
            SINX=SIN(Y(KI+3))
            COSX=COS(Y(KI+3))
            S=Y(KI+4)
            C(KI+1,3)=COSX*S
            C(KI+1,4)=SINX
            C(KI+2,3)=-S*SINX
            C(KI+2,4)=COSX
            C(KI+3,1)=S*B
            IF(I .EQ. 1 .OR. ABS(Y(KI+2)) .LT. 1.E-10) GO TO 5
            C(KI+3,2)=S*SINX/(Y(KI+2)**2)
            C(KI+3,3)=-S*COSX/Y(KI+2)
            C(KI+3,4)=B*Y(KI+1)+2.E0*Y(KI+5)-SINX/Y(KI+2)
            C(KI+3,5)=2.E0*S
            GO TO 6
5           C(KI+3,4)=.5E0*B*Y(KI+1)+Y(KI+5)
```

```
      C(KI+3,1)=.5E0*S*B
                C(KI+3,5)=S
     6          C(KI+6,2)=-TPI*S*SINX*Y(KI+2)
                C(KI+6,3)=PI*S*COSX*(RAD**2-Y(KI+2)**2)
                C(KI+6,4)=PI*SINX*(RAD**2-Y(KI+2)**2)
    10          CONTINUE
      RETURN
      END
C
C-----------------------------------------------
C
      SUBROUTINE IMPR(X,Y,N,ABT,JERROR)
      DIMENSION X(50),Y(350),ABT(7)
      IF (JERROR .GT. 0) WRITE(6,10)
      WRITE(6,70)
      WRITE(6,20)
      DO 5 I=1,N
      II=(I-1)*6
      S=X(I)*Y(4)
      TE=90.E0*Y(II+3)/1.5707963267948966E0
    5 WRITE(6,30) S,Y(II+2),Y(II+1),TE
      WRITE(6,40) ABT(2),ABT(1),ABT(3)
      WRITE(6,50) Y(5),ABT(5)
      WRITE(6,70)
      RETURN
   10 FORMAT(1H /44H   THERE HAS BEEN AN ERROR WARNING IN THIS  /
     2  39H  COMPUTATION. CHECK ANSWERS CAREFULLY./)
   20 FORMAT(1H //,9X,1HS,14X,1HR,13X,1HU,12X,3HPSI//)
   30 FORMAT(1H ,3E15.6,F13.4)
   40 FORMAT(1H ,/4X,7HMAXIMUM/4X,8HABSOLUTE,3X,3E15.2/4X,
     2       6HERRORS/)
   50 FORMAT(1H /7X,
     2   9H    H    = ,E15.6,9X,9H ERROR ≤ ,E10.2///)
   70       FORMAT(1H ,//59(1H-))
      END
C
C-----------------------------------------------
C
      SUBROUTINE FCUR2(X,Y,N,FF,ALPHA)
      DIMENSION X(50),Y(350),FF(350)
     2   ,ALPHA(7)
      COMMON/CYLIN/V,GAMMA,RAD,VERT,HORZ
      COMMON /F1/ B
      UBOT(R)=UC+SQRT(ABS(VERT**2-R**2*E2))
      E=VERT/HORZ
      E2=E**2
      UC=-E*SQRT(ABS(HORZ**2-RAD**2))
      PI=3.14159265358979 2E0
      DO 10 I =1,N
      KI=(I-1)*6
      SENOX=SIN(Y(KI+3))
      COSX=COS(Y(KI+3))
      S=Y(KI+4)
      FF(KI+1)=S*SENOX
      FF(KI+2)=S*COSX
      FF(KI+3)=S*(8*Y(KI+1)+2.E0*Y(KI+5)-SENOX/Y(KI+2))
     6  FF(KI+4)=0.E0
      FF(KI+5)=0.E0
      FF(KI+6)=2.E0*PI*S*COSX*Y(KI+2)*(Y(KI+1)-UBOT(Y(KI+2)))
   10 CONTINUE
      KI=6*(N-1)
      TE=UBOT(Y(2))
```

```
      ALPHA(1)=TE-Y(1)
      TE=TE-UC
      ALPHA(2)=GAMMA-Y(3)+ATAN(-E2*Y(2)/TE)
      ALPHA(3)=-Y(6)
      ALPHA(4)=V-Y(KI+6)
      ALPHA(5)=.5E0*PI-GAMMA-Y(KI+3)
      ALPHA(6)=RAD-Y(KI+2)
      RETURN
      END
C
C-----------------------------------------------------
C
      SUBROUTINE JCUR2(X,Y,C,N,A1,B1)
      DIMENSION Y(350),X(50),C(350,7)
     2  ,A1(7,7),B1(7,7)
      COMMON /F1/ B
      COMMON/CYLIN/ V,GAMMA,RAD,VERT,HORZ
      UBOT(R)=UC+SQRT(ABS(VERT**2-R**2*E2))
      E=VERT/HORZ
      E2=E**2
      UC=-E*SQRT(ABS(HORZ**2-RAD**2))
      TE=UBOT(Y(2))-UC
      TPI=6.283185307179584E0
      PI=TPI*.5E0
      DO 60 I=1,6
      DO 60 J=1,6
      A1(I,J)=0.E0
   60 B1(I,J)=0.E0
      A1(1,1)=1.E0
      A1(1,2)=E2*Y(2)/TE
      EYSQ = (E2*Y(2))**2
      A1(2,2)=(E2*TE**2 + EYSQ)/(TE*(TE**2 + EYSQ))
      A1(2,3)=1.E0
      A1(3,6)=1.E0
      B1(4,6)=1.E0
      B1(5,3)=1.E0
      B1(6,2)=1.E0
      N6=6*N
      DO 50 I=1,N6
      DO 50 J=1,6
   50 C(I,J)=0.E0
      DO 10 I=1,N
            KI=(I-1)*6
            SINX=SIN(Y(KI+3))
            COSX=COS(Y(KI+3))
            S=Y(KI+4)
            C(KI+1,3)=COSX*S
            C(KI+1,4)=SINX
            C(KI+2,3)=-S*SINX
            C(KI+2,4)=COSX
            C(KI+3,1)=S*B
            C(KI+3,2)=S*SINX/(Y(KI+2)**2)
            C(KI+3,3)=-S*COSX/Y(KI+2)
            C(KI+3,4)=B*Y(KI+1)+2.E0*Y(KI+5)-SINX/Y(KI+2)
            C(KI+3,5)=2.E0*S
            C(KI+6,1)=TPI*COSX*S*Y(KI+2)
            UBOTT = UBOT(Y(KI+2))
      TE=AMAX1(1.E-10,UBOTT-UC)
    6       C(KI+6,2)=TPI*S*COSX*((Y(KI+1)-UBOTT)
    2       +Y(KI+2)*E2*Y(KI+2)/TE)
            C(KI+6,3)=-TPI*S*SINX*Y(KI+2)*(Y(KI+1)-UBOTT)
            C(KI+6,4)=TPI*COSX*Y(KI+2)*(Y(KI+1)-UBOTT)
```

```
10              CONTINUE
      RETURN
      END
```

```
      PROGRAM MNELPS(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
C     MAIN PROGRAM FOR COMPUTING THE MENISCUS
C     IN A SPHEROIDAL CONTAINER.
C                              ---------SINGLE PRECISION-------
C     ************************************
C
      DIMENSION PAR(6),S(50),U(50),R(50),X(50),Y(350)
      COMMON/ELIPSE/VOLUME,ANGLE,ALFA,BETA,S,U,R,X,Y
C
      READ (5,10) PAR(1),PAR(3),PAR(4)
   10 FORMAT (3E8.0)
C
    1 CONTINUE
C
         READ (5,20) II,N,BOND,VOLUME,ANGLE,RADIUS,TOL,ALFA,BETA
   20    FORMAT (2I4,7E8.0)
C
         IF (II .EQ. 999) STOP
         CALL ELLIPS(BOND,VOLUME,ANGLE,ALFA,BETA,II,PAR,N,TOL)
      GO TO 1
      END
```

```
      SUBROUTINE ELLIPS(BOND,V,GAMMA,ALFA,BETA,II,PAR,N,TOL)
C
C     SPHEROIDAL CONTAINER
C'    II=1   LARGE LIQUID VOLUME
C     II=2   SMALL LIQUID VOLUME
C     GAMMA = CONTACT ANGLE IN DEGREES
C     BOND = BOND NUMBER.
C     ALFA = HORIZONTAL SEMI-AXIS OF ELLIPTICAL
C     CROSS-SECTION.
C     BETA = VERTICAL SEMI-AXIS.
C     PAR = PARAMETER ARRAY OF LENGTH 6
C
C     PAR(1) .NE. 2.  ELLIPS WILL SET
C     N = MIN(40,MAX(11,SQRT(6.*ABS(BOND))))
C     X = UNIFORM MESH IN  <0,1] WITH N
C     POINTS (COUNTING THE END POINTS).
C     Y = APPROPRIATE FIRST APPROXIMATION
C     ON  X.
C     NO INTERMEDIARY OUTPUT WILL BE PRINTED.
C     NO OTHER ELEMENTS OF  PAR  NEED BE
C     FILLED.
C
C     PAR(1) = 2.     THE USER MUST GIVE VALUES
C     FOR  N, PAR(3), PAR(4), AND  TOL (SEE
C     BELOW).
C
C     N .LE. 50 NUMBER OF POINTS IN INITIAL MESH.
C     UPON OUTPUT IT WILL CONTAIN THE NUMBER
C     OF POINTS IN FINAL MESH.
C
C     X = INITIAL MESH WITH N POINTS IN  <0,1].
C     X(1) MUST BE = 0.E0 AND  X(N) = 1.E0.
C     THE MESH NEED NOT BE UNIFORM.
C
C     Y = INITIAL APPROXIMATION TO THE SIX
C     VARIABLES  U,R,PSI,S,H,V  ON MESH X.
C
C     PAR(3) = 0.  NO INTERMEDIARY OUTPUT IS
C     PRODUCED.
C     PAR(3) = 1.  INTERMEDIARY OUTPUT IS
C     PRINTED.
C
C     PAR(4) = 1.  THE USER GIVES VALUES FOR X,Y.
C     PAR(4) = 0.  ELLIPS ASSIGNS VALUES TO X,Y.
C     IF PAR(1) = 2.
C       THE CALLING PROGRAM
C     SHOULD CONTAIN THE  STATEMENTS
C
C     COMMON /ELIPSE/ VOLUME,ANGLE,ALFA,BETA,S,U,R,X,Y
C     DIMENSION S(50),U(50),R(50),X(50),Y(350),PAR(6)
C
C     TOL = MAXIMUM DESIRED ABSOLUTE ERROR OVER ALL
C     COMPONENTS OF THE SOLUTION AND ALL MESH POINTS.
C     TOL  SHOULD BE LARGER THAN THE MACHINE PRECISION.
C
C
C     THE FINAL VALUES OF THE MENISCUS CURVE   U(S) , R(S)
C     ARE AVAILABLE UPON EXIT THROUGH COMMON /ELIPSE/
C     FOR FURTHER PROCESSING (I.E. PLOTTING).
C
C     ******************************************
```

```
C
      EXTERNAL FF1,JACOB1
      EXTERNAL FF2,JACOB2
      COMMON /F1/ BO
      COMMON /ELIPSE/ VV,GAM,ALF,BET,S,U,RR,X,Y
      INTEGER P,R
      DIMENSION ALPHA(7),A1(7, 7),B1(7, 7),X(50),
     2  Y(350),ABT(7),PAR(6),
     3         S(50),U(50),RR(50)
      IF(II .NE. 1 .AND. II .NE. 2) II = 1
      PIH=1.5707963267948960
      WRITE(6,61)
      WRITE(6,60)
      WRITE(6,70)
      WRITE(6,50)
      WRITE(6,70)
      WRITE(6,51) ALFA,BETA
      WRITE(6,70)
      IF (II .NE. 1) GO TO 3
      WRITE(6,52)V
      GO TO 4
   3  WRITE(6,53)V
   4  WRITE(6,70)
      WRITE(6,54)GAMMA
      WRITE(6,70)
      WRITE(6,56)BOND
      WRITE(6,70)
      WRITE(6,60)
      VOLM=8.E0*PIH*ALFA**2*BETA/3.E0
      IF(V .GT. VOLM)GO TO 5
      IF(V .GT. 0.E0 .AND. ALFA .GT. 0.E0
     2         .AND. BETA .GT. 0.E0 .AND. GAMMA .GE. 0.E0)
     2         GO TO 10
      WRITE(6,98)
      RETURN
   5    WRITE(6,80)
      WRITE(6,99) VOLM
      RETURN
  10  BO=BOND
      ALF=ALFA
      BET=BETA
      ICON=0
      M=6
      A=0.E0
      B=1.E0
      GAMMA=PIH*GAMMA/90.E0
      VV=V
      GAM=GAMMA
      IF(PAR(1) .NE. 1.E0 .AND. PAR(1) .NE. 2.E0)PAR(1)=1.E0
      NO=N
      NB=SQRT(6.*ABS(BOND))
      PAR(2)=0.E0
      PAR(5)=0.E0
      IF(PAR(1) .EQ. 2.E0) GO TO 30
      PAR(1)=1.E0
      PAR(3)=0.E0
      TOL=.5E-2
  30  IF(II .EQ. 2) GO TO 300
 100    IF(ICON .EQ. 3) GO TO 2000
      ICON=ICON+1
      IF(ICON .EQ. 3) WRITE(6,95)
      IF(PAR(1) .EQ. 2.E0) GO TO 32
```

```
      N=MINO(40,MAXO(11,NB))
      GO TO 34
   32 N=NO
      IF(PAR(4) .EQ. 1.E0 .AND. ICON .EQ. 1) GOTO 36
   34 H=1.E0/(N-1)
      PAR(4)=1.E0
      TE1=V/(2.EC*PIH*ALFA**2)
      TE2=-BOND*TE1*.5E0
      R=1
      P=3
      TE10=ALFA*SQRT(1.E0-(TE1/BETA-1.E0)**2)
      DO 20 I=1,N
      X(I)=(I-1)*H
      KI=(I-1)*6
      Y(KI+1)=TE1
      Y(KI+2)=X(I)*TE10
      Y(KI+3)=0.E0
      Y(KI+4)=TE10
      Y(KI+5)=TE2
      Y(KI+6)=V*X(I)**2
   20 CONTINUE
   36 CALL PASVA2(M,N,P,R,A,B,ALPHA,A1,B1,TOL,X,Y,
     2          ABT,FF1,JACOB1,PAR,JERROR)
C
C     THE NEXT BLOCK OF STATEMENTS CHECK WHETHER THE COMPUTED
C     SURFACE LIES WITHIN THE CONTAINER.
C
      INSIDE=1
      DO 37 INDEX=1,N
         KI=(INDEX-1)*6
         UTOL=((ABS(Y(KI+1)-BETA)-TOL)/BETA)**2
         RTOL=((Y(KI+2)-TOL)/ALFA)**2
         IF((UTOL+RTOL) .GT. 1) INSIDE=0
   37 CONTINUE
C
      IF(INSIDE .EQ. 1 .AND. JERROR .NE. 3) GO TO 500
  300 IF(ICON .EQ. 3) GO TO 2000
      ICON = ICON+2
      IF(ICON .EC.2) GO TO 41
      IF(JERROR.NE.3) GO TO 40
      WRITE(6,96)
      GO TO 41
   40 CALL IMPR(X,Y,N,ABT,JERROR)
      WRITE(6,90)
   41 IF(PAR(1) .EQ. 2.EC) GO TO 42
      N=MINO(40,MAXO(11,NB))
      GO TO 44
   42 N=NO
      IF(PAR(4) .EQ. 1.E0 .AND. ICON .EQ. 2)GO TO 46
   44 H=1.E0/(N-1)
      TE1=SQRT(V/(4.E0*PIH*BETA))
      TE2=2.E0*BETA*TE1/ALFA
      TE3=SQRT(ALFA**2-V/(4.E0*PIH*BETA))
      R=0
      P=3
      DO 200 I=1,N
      X(I)=(I-1)*H
      KI=6*(I-1)
      Y(KI+1)=BETA*(1.EC+TE1*(2.E0*X(I)-1)/ALFA)
      Y(KI+2)=TE3
      Y(KI+3)=PIH
      Y(KI+4)=TE2
```

```
      Y(KI+5)=.5E0*(1.E0/SQRT(ALFA**2-V/(4.E0*PIH*BETA))-BETA*BOND)
      Y(KI+6)=V*X(I)
  200 CONTINUE
      PAR(4)=1.E0
   46 CALL PASVA2(M,N,P,R,A,B,ALPHA,A1,B1,TOL,X,Y,ABT,
     2      FF2,JACOB2,PAR,JERROR)
      IF(JERROR .EQ. 3) GO TO 100
  500 CALL IMPR(X,Y,N,ABT,JERROR)
      GAMMA=90.E0*GAMMA/PIH
   50 FORMAT(45H *   MENISCUS COMPUTATION FOR A SPHEROIDAL   ,14X,1H*/
     2 20H *   CONTAINER WITH ,39X,1H*)
   51 FORMAT(34H *   HORIZONTAL SEMI-AXIS (ALFA) = ,E14.6,11X,1H*/,
     2  34H *   VERTICAL  SEMI-AXIS  (BETA) = ,E14.6,11X,1H*)
   52 FORMAT(29H *   (LARGE) LIQUID VOLUME  =,E14.6,16X,1H*)
   53 FORMAT(29H *   (SMALL) LIQUID VOLUME  =,E14.6,16X,1H*)
   54 FORMAT(18H *   CONTACT ANGLE,10X,1H=,F7.3,10H   DEGREES,13X,1H*)
   56 FORMAT(16H *   BOND NUMBER,12X,1H=,E14.6,16X,1H*)
   61 FORMAT(1H1)
   60 FORMAT(1H ,59(1H*))
   70 FORMAT(2H *,57X,1H*)
      DO 1000 I=1,N
      S(I)=X(I)*Y(4)
      KI=(I-1)*6
      U(I)=Y(KI+1)
 1000 RR(I)=Y(KI+2)
      RETURN
 2000 WRITE(6,80)
      RETURN
   80 FORMAT(32H ****FAILURE-CHECK YOUR DATA***/)
   90 FORMAT(43H **** PART OF THE ABOVE SOLUTION SURFACE   /
     2     6X,32HLIES OUTSIDE THE CONTAINER.           /
     3         38H SMALL VOLUME OPTION WILL BE TRIED****//)
   98 FORMAT(17H ***DATA ERROR***//33H EITHER VOLUME,ALFA,BETA OR GAMMA,
     2         12H IS NEGATIVE)
   95 FORMAT(43H ****FAILURE-TRYING LARGE VOLUME OPTION****//)
   96 FORMAT(43H ****FAILURE-TRYING SMALL VOLUME OPTION****//)
   99     FORMAT(42H ****MAXIMUM VOLUME FOR THIS CONTAINER IS,E12.4/)
      END
C
C-----------------------------------------------
C
      SUBROUTINE FF1(X,Y,N,FF,ALPHA)
      DIMENSION X(50),Y(350),FF(1),
     2          ALPHA(7)
      COMMON/ELIPSE/V,GAMMA,ALFA,BETA
      COMMON /F1/ B
      PI=3.14159265358979 2E0
      DO 10 I =1,N
      KI=(I-1)*6
      SENOX=SIN(Y(KI+3))
      COSX=COS(Y(KI+3))
      S=Y(KI+4)
      FF(KI+1)=S*SENOX
      FF(KI+2)=S*COSX
      IF(I .EQ. 1 .OR. ABS(Y(KI+2)) .LT. 1.E-10) GO TO 5
      FF(KI+3)=S*(B*Y(KI+1)+2.E0*Y(KI+5)-SENOX/Y(KI+2))
      GO TO 6
   5  FF(KI+3)=S*(.5E0*B*Y(KI+1)+Y(KI+5))
   6   FF(KI+4)=0.E0
      FF(KI+5)=0.E0
      FF(KI+6)=PI*S*SENOX*((ALFA/BETA)**2*Y(KI+1)*
     2       (2.E0*BETA-Y(KI+1))-Y(KI+2)**2)
```

```
 10     CONTINUE
        KI=6*(N-1)
        ALPHA(1)=-Y(2)
        ALPHA(2)=-Y(3)
        ALPHA(3)=-Y(6)
        ALPHA(4)=V-Y(KI+6)-PI*(ALFA*Y(1)/BETA)**2*(BETA-Y(1)/3.E0)
        IF(ABS(Y(KI+1) - BETA) .LE. 1.E-10) GO TO 20
        TE=ATAN(-(BETA/ALFA)**2*Y(KI+2)/(Y(KI+1)-BETA))
        IF(TE .LT. 0.E0) TE=TE+PI
        ALPHA(5)=GAMMA+Y(KI+3)-TE
        GO TO 30
 20     ALPHA(5)=GAMMA+Y(KI+3)-.5E0*PI
 30     CONTINUE
        ALPHA(6)=1.E0-((Y(KI+1)-BETA)/BETA)**2-
     2        (Y(KI+2)/ALFA)**2
        RETURN
        END
C
C-----------------------------------------------------
C
        SUBROUTINE JACOB1(X,Y,C,N,A1,B1)
        DIMENSION Y(350),X(50),C(350,7),
     2          A1(7,7),B1(7,7)
        COMMON /F1/ B
        COMMON/ELIPSE/ V,GAMMA,ALFA,BETA
        TPI=6.2831853071795 84E0
        PI=TPI*.5E0
        DO 60 I=1,6
        DO 60 J=1,6
        A1(I,J)=0.E0
 60     B1(I,J)=0.E0
        A1(1,2)=1.E0
        A1(2,3)=1.E0
        A1(3,6)=1.E0
        A1(4,1)=PI*(ALFA/BETA)**2*Y(1)*(2.E0*BETA-Y(1))
                        KI=(N-1)*6
        B1(4,6)=1.E0
        B1(6,1)=2.E0*(Y(KI+1)-BETA)/(BETA**2)
        IF(ABS(Y(KI+1) - BETA) .LE. 1.E-10) GO TO 40
        TE1=(ALFA*BETA)**2/(ALFA**4*(Y(KI+1)-BETA)**2+
     2      BETA**4*Y(KI+2)**2)
        B1(5,1)=Y(KI+2)*TE1
        B1(5,2)=-TE1*(Y(KI+1)-BETA)
 40     B1(5,3)=-1.E0
        B1(6,2)=2.E0*Y(KI+2)/(ALFA**2)
        N6=6*N
        DO 50 I=1,N6
        DO 50 J=1,6
 50     C(I,J)=0.E0
        DO 10 I=1,N
                KI=(I-1)*6
                SINX=SIN(Y(KI+3))
                COSX=COS(Y(KI+3))
                S=Y(KI+4)
                C(KI+1,3)=COSX*S
                C(KI+1,4)=SINX
                C(KI+2,3)=-S*SINX
                C(KI+2,4)=COSX
                IF(I .EQ. 1 .OR. ABS(Y(KI+2)) .LT. 1.E-10) GO TO 5
                C(KI+3,1)=S*B
                C(KI+3,2)=S*SINX/(Y(KI+2)**2)
                C(KI+3,3)=-S*COSX/Y(KI+2)
```

```
                C(KI+3,4)=B*Y(KI+1)+2.E0*Y(KI+5)-SINX/Y(KI+2)
                C(KI+3,5)=2.E0*S
                GO TO 6
  5             C(KI+3,4)=.5E0*B*Y(KI+1)+Y(KI+5)
      C(KI+3,1)=.5E0*S*B
                C(KI+3,5)=S
   6            C(KI+6,2)=-TPI*S*SINX*Y(KI+2)
      TE1=(ALFA/BETA)**2*Y(KI+1)*(2.E0*BETA-Y(KI+1))
                C(KI+6,3)=PI*S*COSX*(TE1-Y(KI+2)**2)
                C(KI+6,4)=PI*SINX*(TE1-Y(KI+2)**2)
      C(KI+6,1)=TPI*S*SINX*(ALFA/BETA)**2*(BETA-Y(KI+1))
  10            CONTINUE
      RETURN
      END
C
C-----------------------------------------------
C
      SUBROUTINE IMPR(X,Y,N,ABT,JERROR)
      DIMENSION X(50),Y(350),ABT(7)
      IF (JERROR .GT. 0) WRITE(6,10)
      WRITE(6,70)
      WRITE(6,20)
      DO 5 I=1,N
      II=(I-1)*6
      S=X(I)*Y(4)
      TE=90.E0*Y(II+3)/1.5707963267948 96E0
  5   WRITE(6,30) S,Y(II+2),Y(II+1),TE
      WRITE(6,40) ABT(2),ABT(1),ABT(3)
      WRITE(6,50) Y(5),ABT(5)
      WRITE(6,70)
      RETURN
  10  FORMAT(43H  THERE HAS BEEN AN ERROR WARNING IN THIS  /
     2  38H COMPUTATION. CHECK ANSWERS CAREFULLY./)
  20  FORMAT(1H //,9X,1HS,14X,1HR,13X,1HU,12X,3HPSI/)
  30  FORMAT(1H ,3E15.6,F13.4)
  40  FORMAT(1H ,/4X,7HMAXIMUM/4X,8HABSOLUTE,3X,3E15.2/4X,
     2       6HERRORS/)
  50  FORMAT(1H /7X,
     2    9H   H   = ,E15.6,9X,9H ERROR ≤ ,E10.2///)
  70        FORMAT(1H ,//59(1H-))
      END
C
C-----------------------------------------------
C
      SUBROUTINE FF2(X,Y,N,FF,ALPHA)
      DIMENSION X(50),Y(350),FF(1)
     2         ,ALPHA(7)
      COMMON/ELIPSE/V,GAMMA,ALFA,BETA
      COMMON /F1/ B
      PI=3.1415926535897 92E0
      DO 10 I =1,N
      KI=(I-1)*6
      SENOX=SIN(Y(KI+3))
      COSX=COS(Y(KI+3))
      S=Y(KI+4)
      FF(KI+1)=S*SENOX
      FF(KI+2)=S*COSX
      FF(KI+3)=S*(B*Y(KI+1)+2.E0*Y(KI+5)-SENOX/Y(KI+2))
  6    FF(KI+4)=0.E0
      FF(KI+5)=0.E0
      FF(KI+6)=PI*S*(SENOX)*((ALFA/BETA)**2*Y(KI+1)*
     2     (2.E0*BETA-Y(KI+1))-Y(KI+2)**2)
```

```
 10     CONTINUE
        KI=6*(N-1)
        ALPHA(2)=1.E0-(Y(1)/BETA-1.E0)**2-(Y(2)/ALFA)**2
        ALPHA(3)=-Y(6)
        IF(ABS(Y(1) - BETA) .LE. 1.E-10) GO TO 20
        TE=ATAN(-(BETA/ALFA)**2*Y(2)/(Y(1)-BETA))
        IF (TE .LT. 0.E0) TE=TE+PI
        ALPHA(1)=-GAMMA+Y(3)-TE
        GO TO 30
 20     ALPHA(1)=-GAMMA+Y(3)-.5E0*PI
 30     IF(ABS(Y(KI+1) - BETA) .LE. 1.E-10) GO TO 40
        TE=ATAN(-(BETA/ALFA)**2*Y(KI+2)/(Y(KI+1)-BETA))
        IF(TE .LT. 0.E0) TE=TE+PI
        ALPHA(4)=GAMMA+Y(KI+3)-TE
        GO TO 50
 40     ALPHA(4)=GAMMA+Y(KI+3)-.5E0*PI
 50         ALPHA(5)=V-Y(KI+6)
        ALPHA(6)=1.E0-(Y(KI+1)/BETA-1.E0)**2-(Y(KI+2)/ALFA)**2
        RETURN
        END
C
C-----------------------------------------------------
C
        SUBROUTINE JACOB2(X,Y,C,N,A1,B1)
        DIMENSION Y(350),X(50),C(350,7),
     2          A1(7,7),B1(7,7)
        COMMON /F1/ B
        COMMON/ELIPSE/ V,GAMMA,ALFA,BETA
        TPI=6.2831853071795 84E0
        PI=TPI*.5E0
        DO 60 I=1,6
        DO 60 J=1,6
        A1(I,J)=0.E0
 60     B1(I,J)=0.E0
        R0=1.E0/(ALFA**4*(Y(1)-BETA)**2+BETA**4*Y(2)**2)
        KI=(N-1)*6
        R1=1.E0/(ALFA**4*(Y(KI+1)-BETA)**2+BETA**4*Y(KI+2)**2)
        IF(ABS(Y(1) - BETA) .LE. 1.E-10)GO TO 20
        TE0=(ALFA*BETA)**2*R0
        A1(1,1)=Y(2)*TE0
        A1(1,2)=-TE0*(Y(1)-BETA)
 20     A1(1,3)=-1.E0
        A1(2,1)=2.E0*(Y(1)-BETA)/(BETA**2)
        A1(2,2)=2.E0*Y(2)/(ALFA**2)
        IF(ABS(Y(KI+1) -BETA) .LE. 1.E-10) GO TO 30
        TE1=(ALFA*BETA)**2*R1
        B1(4,1)=Y(KI+2)*TE1
        B1(4,2)=-TE1*(Y(KI+1)-BETA)
 30     B1(4,3)=-1.E0
        A1(3,6)=1.E0
        B1(5,6)=1.E0
        B1(6,1)=2.E0*(Y(KI+1)-BETA)/(BETA**2)
        B1(6,2)=2.E0*Y(KI+2)/(ALFA**2)
        N6=6*N
        DO 50 I=1,N6
        DO 50 J=1,6
 50     C(I,J)=0.E0
        DO 10 I=1,N
                KI=(I-1)*6
                SINX=SIN(Y(KI+3))
                COSX=COS(Y(KI+3))
                S=Y(KI+4)
```

```
                C(KI+1,3)=COSX*S
                C(KI+1,4)=SINX
                C(KI+2,3)=-S*SINX
                C(KI+2,4)=COSX
                C(KI+3,1)=S*B
                C(KI+3,2)=S*SINX/(Y(KI+2)**2)
                C(KI+3,3)=-S*COSX/Y(KI+2)
                C(KI+3,4)=B*Y(KI+1)+2.E0*Y(KI+5)-SINX/Y(KI+2)
                C(KI+3,5)=2.E0*S
                C(KI+6,1)=TPI*SINX*S*(ALFA/BETA)**2*
     2          (BETA-Y(KI+1))
  6             C(KI+6,2)=-TPI*S*SINX*Y(KI+2)
          TE1=(ALFA/BETA)**2*Y(KI+1)*(2.E0*BETA-Y(KI+1))
                C(KI+6,3)=S*PI*COSX*(TE1-Y(KI+2)**2)
                C(KI+6,4)=PI*SINX*(TE1-Y(KI+2)**2)
 10             CONTINUE
          RETURN
          END
```

```
      SUBROUTINE PASVA2(M, N, P, R, A, B, ALPHA, A1, B1, TOL, X, Y, ABT,
     2 FF, JACOB, PAR,JERROR)
C
C*************************************************************
C                                                            *
C     PURPOSE =  THIS IS A VARIABLE ORDER, VARIABLE          *
C  STEP SOLVER FOR TWO-POINT BOUNDARY VALUE FIRST            *
C  ORDER SMOOTH NONLINEAR SYSTEMS OF THE FORM                *
C                                                            *
C   (1)     DY = F(X,Y)    ,    A1 . Y(A) + B1 . Y(B) = ALPHA *
C                                                            *
C  (HERE A1 AND B1 ARE MATRICES.)                            *
C  IT ATTEMPTS TO PRODUCE A DISCRETE SOLUTION WITH MAXIMUM   *
C  ABSOLUTE ERROR LESS THAN  TOL  ON A GRID CONTAINING THE   *
C  ONE INPUT BY THE USER.                                    *
C                                                            *
C                                                            *
C  IN CALLING PROGRAM THE PARAMETER ARRAY X MUST BE          *
C  DIMENSIONED AS X( 650/M )                                 *
C                                                            *
      DIMENSION X(50)
C                                                            *
C  DIMENSIONS INVOLVED IN THE FOLLOWING PARAMETER ARRAYS     *
C  ARE MMAX = 10 AND NMAX*MMAX = 650, AND THEY MUST BE       *
C  DIMENSIONED ACCORDINGLY IN THE CALLING PROGRAM            *
C                                                            *
      DIMENSION ALPHA(7), A1(7, 7), B1(7, 7), Y(350),
     X ABT(7)
C                                                            *
C  FOLLOWING ARRAYS ARE WORKING AREAS.                       *
C                                                            *
      DIMENSION U(7), EJ(50), A2(350, 7), C2(350, 7),
     2    DEL(7, 350), UU(350), RES(350)
      INTEGER IC(50, 7), IR(50, 7), IQJ(50)
C                                                            *
C  WORKING AREAS WITH SIZES RELATED TO MAX. NUMBER OF        *
C  DEFERRED CORRECTIONS = 20 , WHICH SHOULD BE ADEQUATE      *
C  FOR ALL PURPOSES.                                         *
C                                                            *
      DIMENSION AA(50), BB(50), C(50)
C                                                            *
C                                                            *
C                        AUTHORS                             *
C                                                            *
C  M. LENTINI  AND  V. PEREYRA  -   SEPTEMBER 1977.              *
C                                                            *
C           *** REFERENCE ***                                *
C                                                            *
C  M. LENTINI AND V. PEREYRA , [AN ADAPTIVE FINITE DIFFE-    *
C  RENCE SOLVER FOR NONLINEAR TWO-POINT BOUNDARY PROBLEMS    *
C  WITH MILD BOUNDARY LAYERS].  SIAM J. NUMER. ANAL. 14      *
C  (1977),PP.91-111.                                         *
C                                                            *
C*************************************************************
C
      COMMON /NEWT/ F(350), HX(50), SK(350), TEMP(7), GRADF(350), NU,
     X  CASI
      COMMON /C1/ EPSNU
      EXTERNAL JACOB, FF
      LOGICAL SING, LIN, CASI
      DIMENSION PAR(6)
```

```
      INTEGER P, R
C
      CASI = .FALSE.
      SING = .FALSE.
      MMAX = 10
      NMAX = 50
      IF (M .GT. 1 .AND. M .LE. MMAX .AND. N .GT. 3 .AND. N .LE. NMAX
     2    .AND. P .GE. 0 .AND. P .LT. M .AND. R .GE. 0 .AND.
     3     P+R .LE. M) GO TO 10
      WRITE(6,960)
      RETURN
C
C                                  INITIALIZATION
C
   10 MPN = M*N
C
C*****************************************************************
C     THIS CONSTANT IS MACHINE DEPENDENT=                        *
C     EPSMAC  IS APPROXIMATELY 10* RELATIVE PRECISION OF         *
C     FLOATING POINT ARITHMETIC                                  *
      EPSMAC = 1.E-13
C*****************************************************************
C
      EPSMAC = AMAX1(EPSMAC, 1.E-4*TOL)
      IF (PAR(1) .GT. 0.) GO TO 17
      DELEPS = 0.
      IPRINT = 0
      VAIN = 0.
      LIN = .FALSE.
      GO TO 24
   17 DELEPS = PAR(2)
      IPRINT = PAR(3) + 0.5E0
      VAIN = PAR(4)
      LIN = PAR(5) .NE. 0.
C
   24 N1 = N - 1
      EPSNU = 0.E0
      IF (DELEPS .EQ. 0.  ) EPSNU = 1.
      RABS = 1.
      JERROR = 0
      EPBAR = 1.E10
      ICON = 3
      NOLD = 1
      NMA = NMAX
      NTOP = MAX1(.9*FLOAT(NMAX), FLOAT(NMAX - 16))
      BMA = 0.07
   29 BB(1) = 1.
      DO 30 I = 2, 50
   30    BB(I) = 0.
      DO 31 JJ = 1, 20
         NU2 = 2*JJ + 1
         AA(NU2) = -FLOAT(JJ)/(2.E0 **(2*JJ - 1)*FLOAT(NU2))
   31    AA(NU2 + 1) = 0.E0
      IF (VAIN.GT.0.  ) GO TO 60
C
C     FIRST  APPROXIMATION  FOR  Y  AND  X
C
      X(1) = A
      X(N) = B
      H = (B - A)/FLOAT(N1)
      DO 35 I = 2, N1
   35    X(I) = A + FLOAT(I - 1)*H
```

```
      DO 50 I = 1, MPN
   50    Y(I) = 0.
   60   H=0.E0
      DO 62 I = 1, N1
         HX(I) = X(I + 1) - X(I)
         IF (HX(I) .GT. H) H = HX(I)
   62    CONTINUE
      HCUA = H**2
      IF (IPRINT .NE. 0) WRITE(6, 19)   TOL, H
C
C                              ....  MAIN BODY OF PASVA2 ......
C
      NU = 0
      EPS = AMAX1(EPSMAC, H**2*.1E0)
C
C                         ....      ENTER  AFTER  MESH  CHANGE    .......
C
  120 ERROLD = 1.0E20
      KMAX = MAX0(15, (N - 2)/2)
      MPN = M*N
      N1 = N - 1
      MPNM = M*N1
      IPM = MPNM + P + 1
      C3 = 0.8
      DO 150 I = 1, MPN
  150    SK(I) = 0.
      IF (NU .EQ. 0) GO TO 210
C
C     AFTER  MESH  CHANGE  WE HAVE TO INITIALIZE  SK
C               IF  NU .GT. 0
C
      CALL FF(X, Y, N, F,ALPHA)
      CALL U2DCGS(NU, 2, 2, N1, M, AA, X, F, RES, IERROR)
      DO 200 I = 1, N1
         KI = (I - 1)*M
         DO 200 J = 1, M
            KIJ = KI + J
  200       SK(KIJ) = HX(I)*RES(KIJ)
      IF (NU .LT. KMAX) GO TO 210
      NU = NU - 1
      GO TO 460
C
C                  NU IS TOO LARGE  GO TO REFINE THE MESH ...
C
C             ***  NEWTON  ITERATION  ***
C
  210 IF (EPSNU .GE. 1.) GO TO 230
      EPS1 = EPS
  220 EPS = AMAX1(RABS, EPS)
  230 CALL NEWTON(M, N, P, R, ALPHA, A1, B1, X, Y, LIN, A2, C2, DEL, FF,
     2 JACOB, JERRCR, IPRINT, EPS, IR, IC, UU, RES)
      IF (JERROR .NE. 3) GO TO 240
      WRITE(6,975)
      RETURN
  240 IF (EPSNU .GE. 1.) GO TO 250
      EPSNU = AMIN1(EPSNU + DELEPS, 1.E0)
      IF (IPRINT .GT. 0) WRITE(6, 5)   EPSNU, EPS
      IF (EPSNU .LT. 1.) GO TO 220
      EPS = EPS1
      EPSNU = 1.
      GO TO 230
C
```

```
C                    CORRECTION AND ERROR CONTROL STARTS
C
  250 CALL U2DCGS(NU + 1, 2, 2, N1, M, AA, X, F, RES, IERROR)
      IF (IERROR .EQ. 1) GO TO 460
      DO 260 I = 1, N1
          II = (I - 1)*M
          DO 260 J = 1, M
             KI = II + J
             AUXI = RES(KI)*HX(I)
             RES(KI) = SK(KI) - AUXI
  260        SK(KI) = AUXI
      IF (ICON .LE. 12) GO TO 530
  270 IF (P .EQ. 0) GO TO 290
      DO 280 I = 1, P
  280    UU(I) = 0.
  290 DO 300 I = IPM, MPN
  300    UU(I) = 0.
      DO 310 I = 1, MPNM
         IP = I+P
  310    UU(IP) = RES(I)
      CALL SYSLIN(M, N, P, R, X, Y, JACOB, A1, B1, A2, C2, DEL, .TRUE.,
     2 SING, IR, IC, UU, UU, LIN)
C
C              ESTIMATE FOR MAX. ABSOLUTE ERROR (BY COMPONENTS)
C
      ICON = 15
      ERRNEW = 0.
      DO 320 J = 1, M
  320    ABT(J) = 0.
      DO 330 I = 1, N
         KK = (I - 1)*M
         DO 330 J = 1, M
            KKJ = KK+J
            U1 = ABS(UU(KKJ))
            IF (U1 .GT. ABT(J)) ABT(J) = U1
  330       CONTINUE
      DO 340 J = 1, M
         IF (ABT(J) .GT. ERRNEW) ERRNEW = ABT(J)
  340    CONTINUE
      K = NU + 1
      IF (IPRINT .EQ. 0) GO TO 350
      WRITE(6,15)
      WRITE(6,13)  (ABT(J), J = 1, M)
      WRITE(6,12)  ERRNEW, NU
      WRITE(6,15)
  350 IF (ERRNEW .LE. TOL) RETURN
C
C            ....           PRECISION  ACHIEVED      ..........
C
      IF (ERRNEW .LE. .1*ERROLD) GO TO 360
      IF (ERRNEW .GT. C3*ERROLD) GO TO 460
      C3 = 0.5E0*C3
C
C      EITHER KEEP CORRECTING ...
C
  360 ERROLD = ERRNEW
      EPS = AMAX1(EPSMAC, 1.E-3*ERROLD)
      NU = NU + 1
      GO TO 230
C
C                  OR REFINE THE MESH, UNLESS JERROR = 4 ...
C
```

```
  460 IF (JERROR .NE. 4) GO TO 470
      WRITE(6,965)
      RETURN
  470 IF (N .LE. NTOP) GO TO 480
C
C             ....          TOO MANY GRID POINTS          ......
C
                  JERROR=2
      WRITE(6,970)
      RETURN
  480 EPS = AMAX1(EPSMAC, 1.E-3*ERROLD)
      IF (ERROLD .LE. 1.E0) GO TO 490
      EPBAR = AMIN1(1.0E0, 1.E-2*ERROLD)
      GO TO 500
  490 EPBAR = .01*ERROLD
  500 ICON = 0
      NOLD = N
      BMA = 1.
      IF (NU .LT. 1) GO TO 530
      DO 510 I = 1, N1
         II = (I - 1)*M
         DO 510 J = 1, M
            KI = II + J
  510       SK(KI) = RES(KI) + SK(KI)
      NU = NU - 1
      IF (NU .EQ. 0) GO TO 530
      CALL U2DCGS(NU, 2, 2, N1, M, AA, X, F, RES, IERROR)
      DO 520 I = 1, N1
         II = (I - 1)*M
         DO 520 J = 1, M
            KI = II + J
  520       RES(KI) = RES(KI)*HX(I) - SK(KI)
C
C                     ***** MESH VARIATION *****
C                     EQUIDISTRIBUTION OF THE  L2  NORM OF THE ERROR
C                     FOR THE O(H**(2*NU+2))  METHOD.
C
  530 ICON = ICON + 1
      IF (IPRINT .NE. 0) WRITE(6, 21)
      ALG = 1.5E0
      SIG02 = 1.  /FLOAT(2*NU + 2)
      TEM = 0.
      UUN = 0.
      DO 550 I = 1, N1
         TE = 0.
         KI = (I - 1)*M
         DO 540 J = 1, M
            KI1 = KI + J
            Z1 = ABS(Y(KI1))
            Z2 = ABS(RES(KI1))
            IF (Z1 .GT. TEM) TEM = Z1
  540       IF (Z2 .GT. TE) TE = Z2
         EJ(I) = (TE/HX(I))**SIG02
  550    UUN = UUN + EJ(I)
C
      IF (ICON .GT. 1 .AND. NOLD .GT. 1) GO TO 560
      EPBAR = AMAX1(AMIN1(EPBAR, TEM *BMA), TOL)
      E = EPBAR**SIG02
      IF (IPRINT .NE. 0) WRITE(6, 27)   E, UUN, TEM, EPBAR
  560 IQ = 0
      N2 = N - 2
      I = 0
```

```
  570 I = I + 1
      IQJ(I) = EJ(I)/E-0.33
      IQ = IQ + IQJ(I)
      IF (I .LE. N2) GO TO 570
      FIN = .04*FLOAT(N)
      IFIN = INT(FIN)
      NMA = MINO(NMAX - N, 70)
      IF (IPRINT .NE. 0) WRITE(6, 23)    IQ, NMA
      IF (IQ .LE. IFIN .OR. NMA .LE. 0) GO TO 720
      IF (IQ .LE. NMA ) GO TO 580
C
C                 WE ATTEMPT TO DIMINISH THE NUMBER OF POINTS TO BE
C                 INTRODUCED
C
      IF (ALG .LT. 1.09) GO TO 720
      ALG = ALG - .1E0
      XN = N + IQ
      XTE = N + NMA
      E = E*XN/AMIN1(ALG*FLOAT(N), XTE)
      IF (IPRINT .EQ. 0) GO TO 560
      WRITE(6,26)   E, ALG
      GO TO 560
C
C                 CONSTRUCT  NEW  MESH  ***
C
  580 J = 2
      DO 590 I = 1, MPN
  590    UU(I) = Y(I)
      SK(1) = A
      NPU = 2*NU + 5
      DO 690 I = 1, N1
         KII1 = I*M
         IFI = J + IQJ(I)
         HI = HX(I)/(FLOAT(IQJ(I) + 1))
         DO 680 L = J, IFI
            SK(L) = X(I) + HI*FLOAT(L- J + 1)
            KII = (L - 1)*M
            IF (IQJ(I) .GT. 0) GO TO 610
            DO 600 K = 1, M
               KIIK = KII+K
               KII1K = KII1+K
  600          Y(KIIK) = UU(KII1K)
            GO TO 680
  610       IF (I .LE. NU + 2) GO TO 630
            IF (I .GT. N - (NU + 3)) GO TO 640
            NP = NU + 2
 620     I86=I
            CALL COEGEN (I86, NPU, NP, C, BB, X, SK(L))
C*******************************
            GO TO 650
  630       NP = I
            GO TO 620
  640       NP = I - N + NPU
            GO TO 620
  650       DO 670 K = 1, M
               YKI = 0.E0
               DO 660 JI = 1, NPU
                  INPJI1 = (I - NP + JI - 1)*M + K
  660             YKI = YKI + C(JI)*UU(INPJI1)
               KI = KII + K
  670          Y(KI) = YKI
  680       CONTINUE
```

```
          J = IFI + 1
  690     CONTINUE
C
          CASI = .FALSE.
          N = N + IQ
          N1 = N - 1
          KI = M*N1
          DO 700 I = 1, M
             IKI = I + KI
             IMPNM = I + MPNM
  700        Y(IKI) = UU(IMPNM)
          H = 0.
          DO 710 I = 2, N1
             I1 = I - 1
             HX(I1) = SK(I) - SK(I1)
             IF (HX(I1) .GT. H) H = HX(I1)
             KI = I1*M + 1
  710        X(I) = SK(I)
C
          X(N) = B
          HX(N1) = X(N) - X(N1)
          IF (HX(N1) .GT. H) H = HX(N1)
          HCUA = H**2
          IF (ABS(EPBAR - TEM*BMA) .LT. 1.E-5) EPBAR = 1.E10
          IF (ICON .GE. 5) ICON = 15
          GO TO 120
  720 IF (NOLD .EQ. 1 .AND. ALG .LT. 1.5E0  .AND. 2*N - 1
         2 .LE. NMAX) GO TO 740
          IF (N .GT. NOLD) GO TO 270
          IF (ALG .LT. 1.45) GO TO 730
          ALG = 1.4
          XN = N + IQ
          XXN = NMAX
          E = E*XN/AMIN1(ALG*FLOAT(N), XXN)
          IF (IPRINT .EQ. 0) GO TO 560
          WRITE(6,26)   E, ALG
          GO TO 560
  730 IF (NU .GT. 0) GO TO 760
C
C                    BISECTION *****
C
          NTEM = 2*N - 1
          IF (NTEM .LE. NMAX) GO TO 740
C
C               ... TOO MANY GRID POINTS  ...
C
          WRITE(6,970)
            JERROR=2
          RETURN
  740 DO 750 I = 1, N1
  750        IQJ(I) = 1
          IQ = N1
          ICON = 0
          GO TO 580
  760 NU = NU - 1
          ICON = 0
C
C         SINCE WE ARE GOING BACK WE MUST RECOMPUTE THE
C         ESTIMATE OF THE ERROR.
C
          MPNM = M*N1
          DO 770 I = 1, MPNM
```

```
  770     SK(I) = RES(I) + SK(I)
          IF (NU .GT. 0) GO TO 790
          DO 780 I = 1, N1
             KI = (I - 1)*M
             DO 780 J = 1, M
                KIJ = KI + J
  780        RES(KIJ) = - SK(KIJ)
          GO TO 530
  790 CALL U2DCGS(NU, 2, 2, N1, M, AA, X, F, RES, IERROR)
          DO 800 I = 1, N1
             HI = HX(I)
             KI = (I - 1)*M
             DO 800 J = 1, M
                KIJ = KI + J
  800        RES(KIJ) = HI*RES(KIJ) - SK(KIJ)
          GO TO 530
C                      **** END OF MESH VARIATION ****
C
C
C
     5 FORMAT(1HO,13H EPSNU - EPS ,2E15.7/)
    12 FORMAT(  18H  ESTIMATED ERROR ,E12.3,15H IN CORRECTION ,
       2    I3)
    13 FORMAT(30H ESTIMATED ERROR BY COMPONENTS/1H ,10E12.3)
    15 FORMAT(1HO,39H----------------------------------------/)
    16 FORMAT(1H1,                    21H NUMBER OF EQUATIONS ,
       5  I2/22H NUMBER OF MESH POINTS,I4/1H ,
       515HLEFT END POINT ,E10.2,2X,16H RIGHT END POINT,
       5  E10.2//20H BOUNDARY CONDITIONS/9H  ALPHA  ,10(E10.2,2X)//)
    18 FORMAT(1HO,10(E10.2,2X))
    19 FORMAT(15H TOLERANCE , H ,2E12.2)
    21 FORMAT(1HO,28H  -----  STEP  CHANGE  -----/)
    23 FORMAT(14H   NEW POINTS ,I3,7H  NMA= ,I3)
    26 FORMAT(1HO,13H LEVEL - ALG ,2E15.5/)
    27 FORMAT(1H ,42HN LEVEL - LOCAL ERROR - MAX. SOL. - EPBAR
       5   ,4E15.5)
   960 FORMAT(1HO,20H  ---- ERROR  1 ----//24H    EITHER  M,N,P OR K
       2     16HARE OUT OF RANGE/)
   965 FORMAT(1HO,20H  ---- ERROR  4 ----//20H    NEWTON ITERATION
       1     9H REACHED ,
       2     15HROUNDOFF LEVEL./37H    IF PRECISION HAS NOT BEEN REACHED
       3     ,12H THAT MEANS /1H ,
       4  56HTHAT TOL IS TOO SMALL FOR THIS PROBLEM AND COMPUTER WORD
       5  10H LENGTH   ,/)
   970 FORMAT(1HO,19H  ---- ERROR 2 ----//26H    THE PROGRAM ATTEMPTED
       2     42HTO USE A GRID WITH MORE THAN  NMAX POINTS./)
   975 FORMAT(1HO,19H  ---- ERROR 3 ----//19H    NEWTON DIVERGED/)
          END
          SUBROUTINE U2DCGS(K, P, Q, N, M, A, X, Y, S, IERROR)
          INTEGER P, Q
          DIMENSION A(50), Y(350), S(350), X(50), C(50)
C
C(((...   ERROR  EXIT   ......
C
          IF (K .GT. (N + 1 - Q)/P .OR. P .LT. 1 .OR. K .LT. 1)
         2  GO TO 100
          IF (Q .EQ. 0) GO TO 10
          DO 20 I = 1, Q
    20    A(I) = 0.
    10 KK1 = Q + P*K
          KK = KK1 - 1
          KMID = KK1/2
          IERROR = 0
```

```
      KMID1 = KMID-1
C
C     UNSYMMETRIC APPROXIMATION  LEFT BOUNDARY
C
      IF (KMID1 .LT. 1) GO TO 25
      DO 5 I = 1, KMID1
        I98=I
         CALL COEGEN(I98, KK1, I98, C, A, X, .5E0  *(X(I + 1) + X(I )))
         IM = (I - 1)*M
         DO 5 L = 1, M
            ACUM = 0.
            DO 4 J = 1, KK1
               JML = (J - 1)*M + L
    4          ACUM = ACUM + C(J)*Y(JML)
            IML = IM + L
    5       S(IML) = ACUM
C
C                  CENTER RANGE
C
   25 NF = N + 1 - KK1 + KMID
      DO 40 I = KMID, NF
        I98=I
         CALL COEGEN(I98, KK1, KMID, C, A, X, .5E0  *(X(I + 1) + X(I)))
         I1 = I - 1
         II = I1 - KMID
         IT = I1*M
         DO 40 L = 1, M
            ACUM = 0.
            DO 38 J = 1, KK1
               IIJML = (II + J)*M + L
   38          ACUM = ACUM + C(J)*Y(IIJML)
            ITL = IT + L
   40       S(ITL) = ACUM
C
C     RIGHT  BOUNDARY
C
      KMIDP1 = KMID+1
      II = N - KK
      III = II - 1
      DO 50 I = KMIDP1, KK
         IAD = II + I
        I98=I
         CALL COEGEN(IAD, KK1, I98, C, A, X, .5E0  *(X(IAD+1) + X(IAD)))
         IT = (III + I)*M
         DO 50 L = 1, M
            ACUM = 0.
            DO 48 J = 1, KK1
               III1JML = (III + J)*M + L
   48          ACUM = ACUM + C(J)*Y(III1JML)
            ITL = IT + L
   50       S(ITL) = ACUM
C
C(((...   REGULAR  EXIT  ......
C
      RETURN
  100 IERROR = 1
      RETURN
      END
      SUBROUTINE COEGEN(IO, N, NP, C, BB, X, XBAR)
      DIMENSION C(50), BB(50), ALF(50), X(50)
C
C        THIS IS A SLIGHTLY MODIFIED VERSION IN FORTRAN IV
```

```
C         OF THE ALGOL PROCEDURE  PVAND , P. 901 OF
C         [SOLUTION OF VANDERMONDE SYSTEMS OF EQUATIONS[ BY
C         A. BJORCK ^ V. PEREYRA. MATH. COMP. 24, PP.893-903
C         (1970),WHERE A COMPLETE DESCRIPTION OF THE METHOD
C         USED CAN BE FOUND.
C
C     ..................................................................
C
      DO 1 I = 1, N
    1    C(I) = BB(I)
      DO 11 I = 1, N
         HI = X(IO + 1) - X(IO)
         IONPI = IO -NP +I
   11    ALF(I) = (X(IONPI) - XBAR)/HI
      NN = N - 1
      N1 = N + 1
      DO 6 I = 1, NN
         LL = N - I
         DO 6 J = 1, LL
            K = N1 - J
    6       C(K) = C(K) - ALF(I)*C(K - 1)
      DO 8 I = 1, NN
         K = N - I
         KM1 = K + 1
         DO 8 J = KM1, N
            JK = J - K
            C(J) = C(J)/(ALF(J) - ALF(JK))
            JM1 = J - 1
    8       C(JM1) = C(JM1) - C(J)
C
      RETURN
      END
      SUBROUTINE NEWTON(M, N, P, R, ALPHA, A1, B1, X, Y, LIN, A2, C2,
     2 DEL, FF, JACOB, JERROR, IPRINT, EPS, IR, IC, UU, RES)
      COMMON /NEWT/ F(350), HX(50), SK(350), TEMP(7), GRADF(350), NU,
     2 CASI
      COMMON /C1/ EPSNU
      DIMENSION ALPHA(7), A1(7, 7), B1(7, 7), X(50),
     2 Y(350), ABT(7), A2(350, 7), C2(350, 7), DEL(7, 350),
     3 UU(350), RES(350)
      LOGICAL STEP, SING, CASI, LIN
        INTEGER IR(50, 7),IC(50, 7),P,R,P1
      EXTERNAL JACOB
C
C       ERROR EXIT
C       JERROR = 3    NON  CONVERGENCE
C
C....>>>            NEWTON ITERATION STARTS       .......
C
      DT = 1.
      MP = M - P
      P1 = P + 1
      T = 1.
      TE = 1.
      TMIN = 2.  **(-8)
      STEP = .FALSE.
      SCPR = 0.
      MPN = M*N
      DO 400  I = 1, MPN
  400    UU(I) = 0.E0
      MPNM = M*(N - 1)
      INWT = 0
```

```
      ROL = 1.E20
      REOLD = 1.0E20
      IF (IPRINT .EQ. 0) GO TO 520
      WRITE(6,14) N
      WRITE(6,5)          NU, EPS
C
C                  RESIDUAL  COMPUTATION
C
  520 RABS = 0.
         CALL FF (X, Y, N, F,ALPHA)
      IF (P .EQ. 0) GO TO 800
      DO 700 I = 1, P
         RES(I) =ALPHA(I)
  700    RABS = RABS + RES(I)**2
C
 800       DO 900 I = 2, N
         I1 = I - 1
         K1 = I1*M
         DO 900 J = 1, M
            K1J = K1 + J
            K1JM = K1J - M
            K1JMP = K1J - MP
            RES(K1JMP) = - Y(K1J) + Y(K1JM) + .5E0  *HX(I1)
     *      *(F(K1J)+F(K1JM))+SK(K1JM)
  900       RABS = RABS + RES(K1JMP)**2
C
      DO 902 J = P1, M
         MPNMJ = MPNM + J
         RES(MPNMJ) =ALPHA(J)
  902    RABS = RABS +ALPHA(J)**2
      RABS1 = SQRT(RABS)
      IF (IPRINT .EQ. 0) GO TO 903
      WRITE(6,8)          INWT, RABS1
  903 IF (INWT .EQ. 0 .AND. EPSNU .EQ. 1.) GO TO 950
C
C                  CHECK FOR CONVERGENCE
C
      IF (RABS1 .GT. EPS) GO TO 910
      IF (EPSNU .LT. 1.) RETURN
C                  JACOBIAN IS KEPT CONSTANT UNTIL NEXT MESH CHANGE.
C                  STEP AND ANGLE CONTROL ARE SHORTCIRCUITED.
      CASI = .TRUE.
      RETURN
C
C
C.....                        NEWTON EXIT
C                  CONVERGENCE  OR  TOO  MANY  ITERATIONS ...
C
C
C                  NEWTON TEST IN ORDER TO AVOID CYCLING
C
  910 IF ((REOLD - RABS .GE. .5E0  *T*SCPR .AND. INWT .LT. 10) .OR.
     2    (NU .GT. 0 .AND. INWT .EQ. 1)) GO TO 950
      IF (INWT .EQ. 10) GO TO 940
C
C                  STEP CONTROL STARTS ****
C
      IF (.NOT. (CASI .OR. LIN)) GO TO 912
      IF (RABS1 .LE. 100.*EPS) GO TO 941
      GO TO 945
  912 IF (STEP) GO TO 917
      STEP = .TRUE.
```

```
      IF (TE .LE. 1.) GO TO 915
      TN = TE
      GO TO 916
  915 TN = 100.*TE
      IF (TN .GT. 1.) GO TO 917
  916 TMIN = TN*2.  **(-8)
      GO TO 918
  917 TN = .5E0  *T
      IF (TN .LT. TMIN .OR. RABS .GT. ROL) GO TO 940
      ROL = RABS
  918 DT = TN - T
      T = TN
      DO 920 I = 1, MPN
  920    Y(I) = Y(I) + DT*UU(I)
      IF (IPRINT .EQ. 0) GO TO 520
      WRITE(6,4)          T, TMIN
      GO TO 520
C
  940 ROL1 = SQRT(ROL)
      IF (ROL1 .GE. 100.*EPS) GO TO 945
      RABS1 = ROL1
  941 DO 942 I = 1, MPN
  942    Y(I) = Y(I) - DT*UU(I)
      WRITE(6,15) NU,RABS1,EPS
C                 NEWTON DID NOT QUITE REACH THE TOLERANCE.
C                 FURTHER MESH REFINEMENTS ARE NOT ALLOWED.
      JERROR = 4
      IF (EPSNU .LT. 1.) RETURN
C                 JACOBIAN IS KEPT CONSTANT UNTIL NEXT MESH CHANGE.
C                 STEP AND ANGLE CONTROL ARE SHORTCIRCUITED.
      CASI = .TRUE.
      RETURN
C*****
C
C     WE ASSUME DIVERGENCE AND RETURN
C
C(((...    ERROR  EXIT  3    ........
C
  945 JERROR = 3
      WRITE(6,16)
      RETURN
  950 CALL SYSLIN(M, N, P, R, X, Y, JACOB, A1, B1, A2, C2, DEL, CASI,
     2  SING, IR, IC, UU, RES, LIN)
      IF (SING) WRITE(6, 13)
      REOLD = RABS
      SCPR = 1.E-10
      IF (CASI .OR. LIN) GO TO 1100
      TMIN = 2.  **(-8)
      GNOR = 0.
      T = 1.
      STEP = .FALSE.
      SCPR = 0.
      PNOR = 0.
      DO 960 I = 1, MPN
         PNOR = PNOR + UU(I)**2
         GNOR = GNOR + GRADF(I)**2
  960    SCPR = SCPR + GRADF(I)*UU(I)
      IF (IPRINT .EQ. 0) GO TO 970
      WRITE(6,6)          PNOR, SCPR, GNOR
  970 IF (PNOR .EQ. 0.) GO TO 940
C
C      WE CHECK IF THE DIRECTION  UU  IS OF DESCENT (AS IT SHOULD)
```

```
C       AND ALSO IF THE IDENTITY  ≤GRADF,UU>=RABS  IS APPROXIMATELY
C       VERIFIED. IF EITHER ONE OF THESE CHECKS FAIL, WE TAKE  GRADF  AS
C       OUR NEXT SEARCH DIRECTION, SINCE THE ABOVE INDICATES THAT
C       UU  IS AN UNRELIABLE DIRECTION DUE TO ILL-CONDITIONING IN
C       THE JACOBIAN.
C
      IF (SCPR) 1040, 1040, 1030
 1030 IF (ABS(SCPR - RABS) .LE. .1  *RABS) GO TO 1045
 1040 SCPR = - 1.
      TE = 1.
      GO TO 1100
 1045 TE = SCPR/PNOR
C
C                  APPROXIMATE SOLUTION IS CORRECTED                    *
C
 1100 DO 1300 I = 1, MPN
         IF (SCPR .LE. 0.  ) UU(I) = GRADF(I)
 1300    Y(I) = Y(I) + UU(I)
      IF (SCPR .LE. 0.  ) SCPR = GNOR
      T = 1.
      INWT = INWT + 1
      GO TO 520
C
C-----------------------------------------------------------------
C
    4 FORMAT(1H0,23H STEP CONTROL = T - DT ,2E15.7/)
    5 FORMAT(12H CORRECTION ,I4,
     2  36H RESIDUAL FOR NEWTON SHOULD BE .LE. ,E14.7)
    6 FORMAT(1H0,42H  NORM. CORRECT. - SCAL.PROD. - NORM.GRAD.
     2  /     4E12.4/)
    8 FORMAT(18H NEWTON ITERATION ,I2,
     2  21H  MAX. ABS. RESIDUAL ,E14.7)
   13 FORMAT(1H0,23H  JACOBIAN IS SINGULAR )
   14 FORMAT(1H0,I5,21H POINTS ON THIS GRID ,E15.7)
   15 FORMAT(1H0,27H  WARNING ---- CORRECTION ,I4/14H  RESIDUAL IN ,
     2  32HNEWTON COULD ONLY BE REDUCED TO ,E12.2,19H  INSTEAD OF BELOW
     3  ,E12.2,/,44H  FURTHER MESH REFINEMENTS ARE NOT ALLOWED. /,
     4  52H  IF UPON TERMINATION  TOL  HAS NOT BEEN REACHED THE/
     5  41H  PROBLEM REQUIRES A LONGER COMPUTER WORD/)
   16 FORMAT(1H0,27H  NEWTON DOES NOT CONVERGE /20H  FOR MORE DETAILED
     2  23HOUTPUT USE  PAR(3) = 1./ 34H TRY CONTINUATION, BETTER INITIAL
     3  28HVALUES, AND/OR  A FINER MESH/)
      END
      SUBROUTINE SYSLIN(M, N, P, R, X, Y, JACOB, A1, B1, A, C, DEL,
     2 CASI, SING, IR, IC, UU, RES, LIN)
C
      COMMON /NEWT/ F(350), HX(50), SK(350), TEMP(7), GRADF(350), NU
      DIMENSION X(1), Y(1), A1(7, 7), B1(7, 7),
     1 UU(350), RES(350), A(350, 7), C(350, 7), DEL(7, 350)
      INTEGER IR(50, 7), IC(50, 7), ICA(7), P, P1, PM, R, R1
      LOGICAL SING, CASI, LIN
      EXTERNAL JACOB
C
C       CASI = .TRUE.  NO DECOMPOSITION
C
      IF (CASI) GO TO 1000
C     ***   THAT IS OK   ***
      P1 = P - 1
      N1 = N - 1
      M1 = M - P
      MMP = M1 - 1
      PM = P + 1
```

```
      R1 = R - 1
C
C                          CONSTRUCTION OF A,C,DEL
C
      CALL JACOB(X, Y, C, N,A1,B1)
      IF (P .EQ. 0) GO TO 10
      DO 5 I = 1, P
         DO 5 J = 1, M
    5       A(I, J) = A1(I, J)
   10 DO 40 II = 1, N1
         H1 = .5E0*HX(II)
         I1 = (II - 1)*M + 1
         I2 = I1 + MMP
         IT = 0
         DO 20 I = I1, I2
            IP = I + P
            IT = IT + 1
            DO 20 J = 1, M
               A(IP, J) = - H1*C(I, J)
               IF (IT .EQ. J) A(IP, J) = A(IP, J) - 1.
   20          CONTINUE
         I3 = I2 + M
         I4 = I1 + P1
         IT = 0
      IF(P .EQ. 0) GO TO 35
         DO 30 I = I1, I4
            IT = IT + 1
            DO 30 J = 1, M
               IM = I + M
               I3IT = I3 + IT
               I2IT = I2 + IT
               A(IM, J) = - H1*C(I3IT, J)
               C(I, J) = - H1*C(I2IT, J)
               IF (IT + M1 .NE. J) GO TO 30
               A(IM, J) = A(IM, J) + 1.
               C(I, J) = C(I, J) - 1.
   30          CONTINUE
   35       IT = 0
         DO 40 I = I1, I2
            IP = I + P
            IT = IT + 1
            IM = I + M
            DO 40 J = 1, M
               C(IP, J) = - H1*C(IM, J)
               IF (IT .EQ. J) C(IP, J) = C(IP, J) + 1.
   40          CONTINUE
C
      I1 = N1*M + PM
      I2 = N*M
      IT = P
      DO 50 I = I1, I2
         IT = IT + 1
         DO 50 J = 1, M
   50       A(I, J) = B1(IT, J)
      IF (R .EQ. 0) GO TO 500
      DO 100 I = 1, R
         DO 100 J = 1, M
            IPLUSP = I + P
  100       DEL(I, J) = A1(IPLUSP, J)
  500 IF (LIN) GO TO 700
C     COMPUTATION  OF  GRADIENT
      IT = 0
```

```
      DO 540 I = 1, N
          I1 = (I - 1)*M
          I2 = I1 - M1
          I3 = I*M
          DO 540 J = 1, M
              IT = IT + 1
              TE = 0.
              DO 510 JJ = 1, M
                  I1J = I1 + JJ
  510             TE = TE + A(I1J, J)*RES(I1J)
              IF (I .EQ. N) GO TO 525
              DO 520 JJ = 1, P
                  I1JJ = I1 + JJ
                  I3JJ = I3 + JJ
  520             TE = TE + C(I1JJ, J)*RES(I3JJ)
  525         IF (I .EQ. 1) GO TO 539
              DO 530 JJ = 1, M1
                  I2J = I2 + JJ
  530             TE = TE + C(I2J, J)*RES(I2J)
  539         I1J = I1 + J
  540         GRADF(I1J) = TE
C
      IF (R .EQ. 0) GO TO 700
      I3P = N1*M + P
      DO 560 J = 1, M
         TE = 0.
         DO 550 I = 1, R
            I3PI = I3P + I
  550       TE = TE + DEL(I, J)*RES(I3PI)
  560    GRADF(J) = GRADF(J) + TE
  700 CALL DECOMP(A, C, DEL, M, N, P, R, IR, IC, SING, ICA)
      IF (SING) GO TO 2000
 1000 CALL SOLVE(A, C, DEL, RES, M, N, P, R, IR, IC, UU)
 2000 RETURN
      END
      SUBROUTINE DECOMP(A, C, DEL, M, N, P, R, IR, IC, SING, ICA)
      LOGICAL SING
      DIMENSION A(350, 7), C(350, 7), AUX(14, 7),
     X  DEL(7, 350)
      INTEGER IR(50, 7), IC(50, 7), ICA(7), P, P1, R
      SING = .FALSE.
      P1 = P + 1
      MP = M + P
      N1 = N - 1
C
C                       MAIN LOOP
C
      DO 5000 II = 1, N1
         IX = (II - 1)*M
         IXM = IX + M
         IXI = IX - M
         DO 7 I = 1, M
            IC(II, I) = I
            IR(II, I) = I
            DO 7 J = 1, M
               IXADDI = IX + I
    7          AUX(I, J) = A(IXADDI, J)
      IF (P) 200,200,5
    5  DO 8 I = 1, P
         IADDM = I + M
         IXADDI = IX + I
         DO 8 J = 1, M
```

```
    8          AUX(IADDM, J) = C(IXADDI, J)
C
C       REDUCTION OF FIRST P ROWS OF A1(II)
C
          DO 100 I = 1, P
             TE = 0.
             DO 10 J = I, M
                Y = ABS(AUX(I, J))
                IF (Y .LE. TE) GO TO 10
                TE = Y
                IPIV = J
   10           CONTINUE
             IF (TE .EQ. 0.) GO TO 9999
C
C       COLUMN INTERCHANGES
C
             IF (IPIV .EQ. I) GO TO 22
             ITE = IC(II, I)
             IC(II, I) = IC(II, IPIV)
             IC(II, IPIV) = ITE
             DO 20 IS = 1, MP
                TE = AUX(IS, I)
                AUX(IS, I) = AUX(IS, IPIV)
   20           AUX(IS, IPIV) = TE
             IF (II .EQ. 1) GO TO 22
             I1 = IXI + P1
             I2 = IXI + M
             DO 21 IS = I1, I2
                TE = C(IS, I)
                C(IS, I) = C(IS, IPIV)
                C(IS, IPIV) = TE
   21 CONTINUE
C
C       FACTORIZATION
C
   22        AK = 1./AUX(I, I)
             I1 = I + 1
             DO 25 IS = I1, MP
                XMU = AUX(IS, I)*AK
                AUX(IS, I) = XMU
                DO 25 JS = I1, M
   25              AUX(IS, JS) = AUX(IS, JS) - XMU*AUX(I, JS)
  100        CONTINUE
C
  200     DO 300 J = P1, M
             TE = 0.
             DO 210 I = J, MP
                Y = ABS(AUX(I, J))
                IF (Y .LE. TE) GO TO 210
                TE = Y
                IPP = I
  210           CONTINUE
             IF (TE .EQ. 0.) GO TO 9999
C
C       ROW INTERCHANGES
C
             IF (IPP .EQ. J) GO TO 260
             IPIV = IPP - P
             JP = J - P
             ITE = IR(II, JP)
             IR(II, JP) = IR(II, IPIV)
             IR(II, IPIV) = ITE
```

```
            DO 220 JS = 1, M
               TE = AUX(J, JS)
               AUX(J, JS) = AUX(IPP, JS)
  220          AUX(IPP, JS) = TE
            II1 = IX + J
            II2 = IX + IPP
            IF (IPP .GT. M) GO TO 240
            DO 230 JS = 1, M
               TE = C(II1, JS)
               C(II1, JS) = C(II2, JS)
               C(II2, JS) = TE
  230 CONTINUE
            GO TO 260
  240       DO 250 JS = 1, M
               TE = C(II1, JS)
               C(II1, JS) = A(II2, JS)
  250          A(II2, JS) = TE
C
C      FACTORIZATION
C
  260       IF (J .EQ. M) GO TO 300
            AK = 1./AUX(J, J)
            J1 = J + 1
            DO 280 IS = J1, MP
               XMU = AUX(IS, J)*AK
               AUX(IS, J) = XMU
               DO 280 JS = J1, M
  280             AUX(IS, JS) = AUX(IS, JS) - XMU*AUX(J, JS)
  300       CONTINUE
C
C      RESTORING B1(II+1) AND AL(II)
C
       IF( P .EQ. 0) GO TO 353
          I1 = M - P + 1
          DO 350 IS = I1, M
             IP = IR(II, IS)
             IF (IP .EQ. IS) GO TO 350
             ISP = IS + P + IXI
             IPP = IP + P + IX
             DO 310 JS = 1, M
                C(ISP, JS) = A(IPP, JS)
  310 CONTINUE
  350       CONTINUE
 353        DO 355 I = 1, M
            IXADDI = IX + I
            DO 355 J = 1, M
  355          A(IXADDI, J) = AUX(I, J)
C
         IS = IX + 1
         ISP = IX + P
         DO 360 JS = 1, M
  360       ICA(JS) = IC(II, JS)
         JS = 1
  365    IF (JS .GE. M) GO TO 408
         IP = ICA(JS)
         IF (IP .NE. JS) GO TO 370
         JS = JS + 1
         GO TO 365
C
  370    INI = JS
         ICA(INI) = INI
C
```

```
C       COLUMN INTERCHANGES FOR B1(II+1)
C
      IF(P .EQ. 0) GO TO 400
  373    DO 375 I1 = IS, ISP
            TE = C(I1, INI)
            C(I1, INI) = C(I1, IP)
            C(I1, IP) = TE
  375 CONTINUE
  400       IF (R .EQ. 0) GO TO 407
C
C       COLUMN INTERCHANGES OF D(II)
C
            IPP = IX + IP
            JX = IX + INI
            DO 406 ISSSSS = 1, R
               TE = DEL(ISSSSS, IPP)
               DEL(ISSSSS, IPP) = DEL(ISSSSS, JX)
  406          DEL(ISSSSS, JX) = TE
  407       INI = IP
            IP = ICA(IP)
            ICA(INI) = INI
            IF (IP .NE. JS) GO TO 373
            JS = JS + 1
            GO TO 365
C
  408       IF (R .EQ. 0) GO TO 429
C
C       SOLUTION OF DE(II)*AL(II)=-+DELTA(II)+
C
            I1 = IX + 1
            DO 422 IS = 1, R
               DO 416 JS = I1, IXM
                  TE = DEL(IS, JS)
                  JSX = JS - IX
                  IF (JS .EQ. I1) GO TO 414
                  J1 = JS - 1
                  DO 412 KS = I1, J1
  412                TE = TE - DEL(IS, KS)*A(KS, JSX)
  414             DEL(IS, JS) = TE/A(JS, JSX)
  416             CONTINUE
               M1 = M - 1
               DO 420 JS = 1, M1
                  J = IXM - JS
                  TE = DEL(IS, J)
                  J1 = J + 1
                  DO 418 KS = J1, IXM
                  JIX = J - IX
  418                TE = TE - DEL(IS, KS)*A(KS, JIX)
  420             DEL(IS, J) = TE
  422          CONTINUE
C
C       COMPUTATION OF DE(II+1)=-DE(II)*GA(II)
C
            IXP = IX + P1
            DO 424 IS = 1, R
               DO 424 JS = 1, M
                  TE = 0.
                  DO 423 KS = IXP, IXM
  423                TE = TE - DEL(IS, KS)*C(KS, JS)
                  IXMJS = IXM + JS
  424             DEL(IS, IXMJS) = TE
C
```

```
C       SOLUTION OF BE(II+1)*AL(II)=B(II+1)
C
  429     I1 = IX + 1
        IF(P .EQ. 0) GO TO 5000
          I2 = IX + P
          DO 600 IS = I1, I2
            DO 450 JS = 1, M
              IXJS = IX + JS
              TE = C(IS, JS)
              IF (JS .EQ. 1) GO TO 440
              J1 = JS - 1
              DO 430 KS = 1, J1
                IXKS = IX + KS
  430           TE = TE - C(IS, KS)*A(IXKS, JS)
  440         C(IS, JS) = TE/A(IXJS, JS)
  450         CONTINUE
            M1 = M - 1
            DO 480 JS = 1, M1
              J = M - JS
              TE = C(IS, J)
              J1 = J + 1
              DO 460 KS = J1, M
                IXKS = IX + KS
  460           TE = TE - C(IS, KS)*A(IXKS, J)
              C(IS, J) = TE
  480 CONTINUE
C
C       COMPUTATION OF AL(II+1)=A(II+1)-BE(II+1)*GA(II)
C
            ISM = IS + M
            DO 500 JS = 1, M
              TE = A(ISM, JS)
              DO 490 KS = P1, M
                IXKS = IX + KS
  490           TE = TE - C(IS, KS)*C(IXKS, JS)
  500         A(ISM, JS) = TE
  600       CONTINUE
 5000     CONTINUE
C
C       COMPLETE COMPUTATION OF ALFA(N)
C
        I2 = N1*M
        I1 = I2 + P
        IF (R .EQ. 0) GO TO 6000
        DO 5100 IS = 1, R
          I1S = I1 + IS
          DO 5100 JS = 1, M
            I2JS = I2 + JS
 5100       A(I1S, JS) = A(I1S, JS) + DEL(IS, I2JS)
C
C       L U DECOMPOSITION OF ALFA(N) (COLUMN PIVOTING)
C
 6000 I4 = I1 - M + 1
      I1 = I2 + 1
      I3 = N*M - 1
      I8 = I3 + 1
      DO 6050 I = 1, M
 6050   IC(N, I) = I
      DO 7000 I = I1, I3
        I0 = I - I2
        TE = 0.
        DO 6100 J = I0 , M
```

```
            Y = ABS(A(I, J))
            IF (Y .LE. TE) GO TO 6100
            TE = Y
            IPIV = J
 6100       CONTINUE
         IF (TE .EQ. 0.) GO TO 9999
C
C       COLUMN INTERCHANGES
C
         IF (IPIV .EQ.   I0) GO TO 6500
         ITE = IC(N, I0)
         IC(N, I0) = IC(N, IPIV)
         IC(N, IPIV) = ITE
         DO 6200 IS = I1, I8
            TE = A(IS, I0)
            A(IS, I0) = A(IS, IPIV)
 6200       A(IS, IPIV) = TE
         DO 6300 IS = I4, I2
            TE = C(IS, I0)
            C(IS, I0) = C(IS, IPIV)
 6300       C(IS, IPIV) = TE
C
C       FACTORIZATION
C
 6500    AK = 1./A(I, I0)
         I01 = I0 + 1
         I7 = I + 1
         DO 6600 IS = I7, I8
            XMU = A(IS, I0)*AK
            A(IS, I0) = XMU
            DO 6600 JS = I01, M
 6600          A(IS, JS) = A(IS, JS) - XMU*A(I, JS)
 7000    CONTINUE
C
      RETURN
 9999 SING = .TRUE.
      RETURN
      END
      SUBROUTINE SOLVE(A, C, DEL, Y, M, N, P, R, IR, IC, U)
      DIMENSION A(350, 7), C(350, 7), DEL(7, 350), X(350),
     2   U(350), Y(350)
      INTEGER IR(50, 7), IC(50, 7), P, R, P1, P2
        M1=M-1
        N2=N+1
        N1=N-1
        P1=P-1
        P2=P+1
      IF(P .EQ. 0) GO TO 12
      DO 10 I = 1, P
   10    X(I) = Y(I)
   12   MN = M*N1
      DO 15 I = 1, M
         MNADDI = MN + I
   15    X(MNADDI) = Y(MNADDI)
      DO 50 I = 1, N1
         IX = (I - 1)*M + P
         DO 50 J = 1, M
            IXP = IX + IR(I, J)
            IXJ = IX + J
   50       X(IXJ) = Y(IXP)
C
C                          SOLVE  L * Y = X
```

```
C
        IF(P .EQ. 0) GO TO 110
      DO 100 I = 2, N
         I1 = M*(I - 2) + 1
         I2 = I1 + P1
         I11 = I1 - 1
      IF(P .EQ. 0) GO TO 110
         DO 100 J = I1, I2
            JM = J + M
            TE = X(JM)
            DO 80 K = 1, M
               I11K = I11 + K
   80          TE = TE - C(J, K)*X(I11K)
  100       X(JM) = TE
C
  110    N1M = N1*M
      IXN = N1M + P
      IF (R .EQ. 0) GO TO 135
      DO 130 II = 1, R
         I1 = IXN + II
         TE = X(I1)
         DO 120 J = 1, N1M
  120       TE = TE - DEL(II, J)*X(J)
  130    X(I1) = TE
C
C                          SOLVE  U * Z = Y
C
  135 DO 200 I = 1, N
         II = N2 - I
         I0 = (II - 1)*M
         I1 = I0 + 2
         I2 = II*M
         IF (I .EQ. 1) GO TO 148
         DO 145 J = P2, M
            I0J = I0 + J
            TE = X(I0J)
            DO 140 K = 1, M
               I2K = I2 + K
  140          TE = TE - C(I0J, K)*X(I2K)
  145       X(I0J) = TE
  148    DO 160 J = I1, I2
            K1 = J - I1 + 1
            TE = X(J)
            DO 150 K = 1, K1
               I0K = I0 + K
  150          TE = TE - A(J, K)*X(I0K)
  160       X(J) = TE
         X(I2) = X(I2)/A(I2, M)
         DO 200 J = 1, M1
            J1 = I2 - J
            TE = X(J1)
            K1 = M - J + 1
            DO 180 K = K1, M
               I0K = I0 + K
  180          TE = TE - A(J1, K)*X(I0K)
            J1I0 = J1 - I0
  200       X(J1) = TE/A(J1, J1I0)
C
C                          INTERCHANGES IN  X
C
      DO 250 I = 1, N
         IX = (I - 1)*M
```

```
          DO 250 J = 1, M
             IXP = IX + IC(I, J)
             IXJ = IX + J
  250        U(IXP) = X(IXJ)
C
      RETURN
      END
```

TECHNICAL INFORMATION DEPARTMENT
LAWRENCE BERKELEY LABORATORY
UNIVERSITY OF CALIFORNIA
BERKELEY, CALIFORNIA 94720