stichting
mathematisch
centrum

AFDELING NUMERIEKE WISKUNDE NW 70/79 AUGUSTUS
(DEPARTMENT OF NUMERICAL MATHEMATICS)

M. BAKKER & M.S. VAN DEN BERG

A PROGRAM TO SOLVE ROTATING PLASMA PROBLEMS

Preprint

2e boerhaavestraat 49 amsterdam

Printed at the Mathematical Centre, 49, 2e Boerhaavestraat, Amsterdam.

The Mathematical Centre, founded the 11-th of February 1946, is a non-profit institution aiming at the promotion of pure mathematics and its applications. It is sponsored by the Netherlands Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O).

1980 Mathematics subject classification: 65N30

A program to solve rotating plasma problems*)

by

M. Bakker & M.S. van den Berg**)

ABSTRACT

It is shown that the solution of a rotating plasma problem minimizes a suitably chosen functional. This variational problem is solved by means of the Ritz-Galerkin method using piecewise bilinear functions and applying Newton-Côtes-like quadrature. The resulting linear system with sparse non-negative definite matrix is solved by an adaptive conjugate gradient method.

KEY WORDS & PHRASES: *plasmadynamics, plasmacentrifuge, finite element method, elliptic equations*

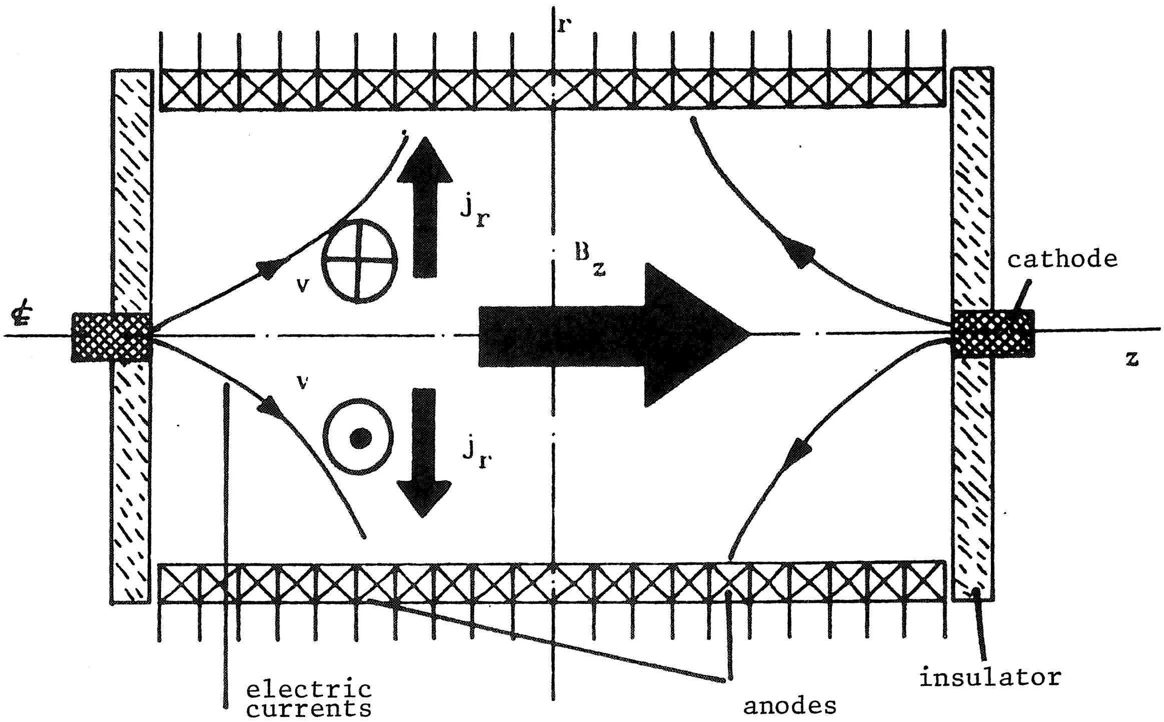
*) This report will be submitted for publication elsewhere.

**) FOM Instituut voor Atoom- en Molecuulfysica, Amsterdam

1. INTRODUCTION

When inside a cylindrical system a radial electric current flows through an electrically conducting viscous medium in the presence of an externally applied axial magnetic field, the resulting Lorentz force will put the medium into an azimuthal motion. Equilibrium is achieved when the Lorentz force balances the opposing viscous force. This principle is applied in case of plasma centrifuges for mass separation of gaseous mixtures.

A calculation model is developed for a system, in which a current flows between two cathodes and a number of anodes. The cathodes are located at the center of the two endplates closing the system and the ringshaped anodes at a prescribed distance on the cylinderwall. The electrode configuration is symmetrical in respect to the symmetry plane of the cylinder (fig.1).



B = magnetic field

j_r = radial component of electric current

v = azimuthal velocity

Figure 1. The physical model of the rotating plasma problems

The result is a current distribution with a radial, axial and in presence of an axial magnetic field also an azimuthal component. The distribution of the coefficient of viscosity and the electrical conductivity of the incompressible medium is assumed to be uniform.

We define

$$v = V_\phi/U_0; \quad \phi = \Phi/(U_0 B_z R); \quad u = \frac{v}{x};$$

$$x = r/R; \quad y = z/L; \tag{1.1}$$

$$\gamma = R/L; \quad Ha = B_z R(\sigma/\eta)^{\frac{1}{2}}; \quad \sigma\;/\sigma_\perp = 1+\beta^2;$$

where

(1.2)

V_ϕ	- azimuthal velocity;
U_0	- characteristic velocity;
Φ	- electric potential;
B_z	- magnetic induction;
R,L	- radius and half length of cylinder;
$\sigma_\perp,\sigma$	- normal and tangential component; of conductivity;
η	- dynamic viscosity;
γ	- inverse aspect ratio;
Ha	- Hartmann constant;
β	- Hall parameter.

For u and ϕ, the following system of partial differential equations can be derived (see VAN DEN BERG [1])

$$\frac{\partial^2 u}{\partial x^2} + \frac{3}{x}\frac{\partial u}{\partial x} + \gamma^2\frac{\partial^2 u}{\partial y^2} - \frac{Ha^2}{1+\beta^2}\left(u - \frac{1}{x}\frac{\partial \phi}{\partial x}\right) = 0; \tag{1.3a}$$

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{1}{x}\frac{\partial \phi}{\partial x} + \gamma^2(1+\beta^2)\frac{\partial^2 \phi}{\partial y^2} - xu_x - 2u = 0; \quad 0 \le x,y \le 1, \tag{1.3b}$$

with boundary conditions (see figure 2)

$$
(1.4)\qquad
\begin{aligned}
&\Gamma_4:\ x = 0: && \frac{\partial u}{\partial x} = \frac{\partial \phi}{\partial x} = 0;\\
&\Gamma_2:\ x = 1: && u = 0;\quad \frac{\partial \phi}{\partial x} = f_1(y);\\
&\Gamma_1:\ y = 0: && \frac{\partial u}{\partial y} = \frac{\partial \phi}{\partial y} = 0;\\
&\Gamma_3:\ y = 1: && u = 0;\quad \frac{\partial \phi}{\partial y} = f_2(x).
\end{aligned}
$$

REMARK. As a matter of fact, VAN DEN BERG [1] derives a system of PDEs for v and ϕ. Since, however, the equation for v is potentially troublesome at $x = 0$, it seems numerically preferable to work with (1.3).

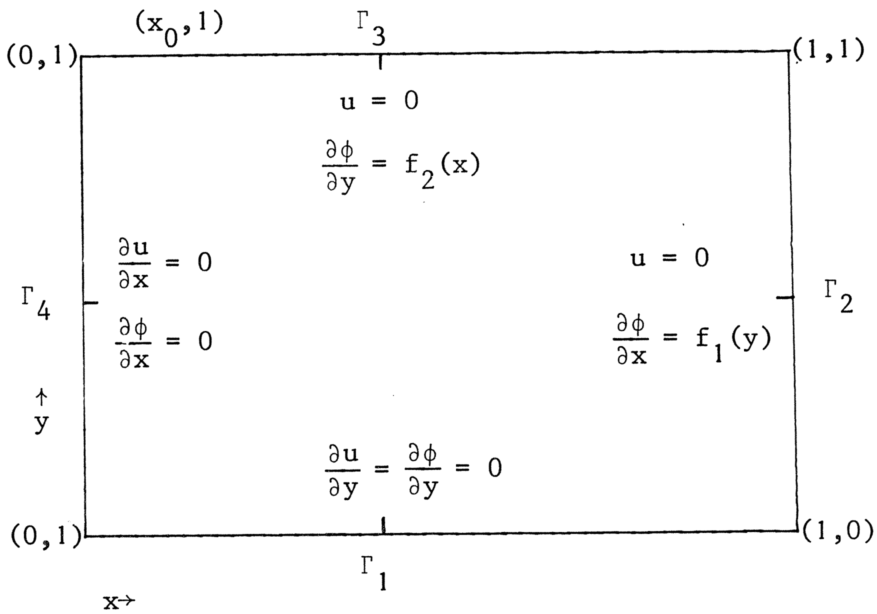


Figure 2. Boundary conditions on Γ

If we multiply both sides of (1.3b) with x and integrate partially over $R = [0,1]\times[0,1]$, it turns out that $f_1(y)$ and $f_2(x)$ are not independent functions but satisfy the integral relation

$$
(1.5)\qquad \int_0^1 f_1(y)dy + \gamma^2(1+\beta^2)\int_0^1 xf_2(x)dx = 0.
$$

Corresponding with the homogenity of the current from the cathodes, $f_2(x)$ is a piecewise constant function, defined by

$$
(1.6) \qquad f_2(x) = \begin{cases} C_2 > 0, & 0 \le x < x_0; \\ 0 \quad , & \text{elsewhere.} \end{cases}
$$

2. VARIATIONAL FORMULATION OF THE PROBLEM

We define the function spaces

$$
(2.1) \qquad \begin{aligned} C^0(R) &= \{f \,|\, f \text{ continuous on } R;\ \frac{\partial f}{\partial x} \text{ and } \frac{\partial f}{\partial y} \\ &\qquad \text{continuous almost everywhere on } R\}; \\ C_0^0(R) &= \{f \,|\, f \in C^0(R);\ f = 0, \text{ on } \Gamma_2 \text{ and } \Gamma_3\}\ . \end{aligned}
$$

and the (linear and bilinear) functionals

$$
(2.2) \qquad \begin{aligned} I(p,q) &= \iint_R x^3[p_x^2+\gamma^2 p_y^2]dxdy \\ &+ \frac{Ha^2}{1+\beta^2} \iint_R x[(xp-q_x)^2 + \gamma^2(1+\beta^2)q_y^2]dxdy; \\ &\qquad\qquad p \in C_0^0(R); \qquad q \in C^0(R); \end{aligned}
$$

$$
(2.3) \qquad b(q) = \frac{Ha^2}{1+\beta^2} \int_\Gamma G(\sigma)q(\sigma)d\sigma, \quad q \in C^0(R);
$$

$$
(2.4) \qquad G(\sigma) = \begin{cases} 0, & \sigma \in \Gamma_1 \cup \Gamma_4; \\ \gamma^2 x f_2(x)(1+\beta^2), & \sigma \in \Gamma_3; \\ f_1(y), & \sigma \in \Gamma_2; \end{cases}
$$

$$
(2.5) \qquad a_1(p,q) = \iint_R x^3(p_x q_x+\gamma^2 p_y q_y + \frac{Ha^2}{1+\beta^2}\, pq)dxdy;
$$

$$
(2.6) \qquad a_2(p,q) = \frac{Ha^2}{1+\beta^2} \iint_R x^2 p q_x dxdy;
$$

$$(2.7) \qquad a_3(p,q) = \gamma^2 \mathrm{Ha}^2 \iint_R x p_y q_y \, dxdy.$$

THEOREM 1. *If* (u,ϕ) *is a solution of* (1.3)-(1.4), *it minimizes*

$$(2.8) \qquad E(p,q) = I(p,q) - 2b(q); \; (p,q) \in C_0^0(R) \times C^0(R);$$

and satisfies the relations

$$(2.9a) \qquad a_1(u,p) - a_2(p,\phi) = 0, \qquad p \in C_0^0(R);$$

$$(2.9b) \qquad -a_2(u,q) + a_3(\phi,q) = b(q), \; q \in C^0(R).$$

Moreover, u *is unique and* ϕ *is unique up to an additive constant.*

PROOF. If we multiply (1.3a) with x^3p and (1.3b) with $[x\mathrm{Ha}^2/(1+\beta^2)]q$, then after partial integration we obtain (2.9). By using (2.9), we also obtain that

$$E(p,q)-E(u,\phi) = I(u-p,\phi-q), \geq 0, \; p \in C_0^0(R); \; q \in C_0^0(R),$$

which proves that $E(u,\phi)$ is a minimum. It is not a strong minimum, for $I(u-p,\phi-q) = 0$, if and only if $u \equiv p$ and $\phi - q \equiv$ constant. □

REMARK. One can make ϕ unique by introducing an additional constraint. For technical reasons, we define $\phi(0,1) = 0$.

3. FINITE ELEMENT SOLUTION

It is a standard result (see e.g. MITCHELL & WAIT [3] or COURANT & HILBERT [2]) that one can approximate (u,ϕ) by minimizing $E(p,q)$ over a finite-dimensional subsapce $S_0 \times S$ of $C_0^0(R) \times C^0(R)$. For S, we select the space of piecewise bilinear functions, given a partition of R in rectangles $[x_{i-1},x_i] \times [y_{j-1},y_j]$ (see e.g. STRANG & FIX [5] or MITCHELL & WAIT [2]). This partition should be given by partitions of the x-interval and the y-interval on the input record (see §7). For S_0, we select the subspace of S satisfying the zero boundary conditions on Γ_2 and Γ_3.

The minimization of E over the finite element space results in the system of linear equations

$$\text{(3.1)} \qquad \begin{pmatrix} A_1 & -A_2 \\ -A_2^T & A_3 \end{pmatrix} \begin{pmatrix} \vec{U} \\ \vec{\Phi} \end{pmatrix} = \begin{pmatrix} 0 \\ \vec{c} \end{pmatrix}$$

with

$$\text{(3.2)} \qquad \begin{aligned} A_1 &= (a_1(B_i^0,B_j^0)); \quad & A_2 &= (a_2(B_i^0,B_j)); \\ A_3 &= (a_3(B_i,B_j)); \quad & \vec{c} &= (b(B_i)), \end{aligned}$$

where $\{B_i^0\}$ and $\{B_i\}$ are the basis functions of S_0 and S, respectively and where a_1, a_2, a_3 and b are given by (2.3)-(2.7). A_1, A_2 and A_3 are 9-diagonal, while A_1 and A_3 are also symmetric; hence the overall matrix is symmetric and 27- diagonal.

Further trimming of the matrix

System (3.1) can be made still sparser if the energy functional E is approximated by a suitable quadrature rule.

On any rectangle $[x_{i-1},x_i]\times[y_{j-1},y_j]$, we define

$$\text{(3.3)} \qquad \int_{x_{i-1}}^{x_i}\int_{y_{j-1}}^{y_j} x^m F(x,y)\,dx\,dy \simeq \omega_1 F(x_{i-1},y_{j-1}) + \omega_2 F(x_{i-1},y_j) + \omega_3 F(x_i,y_{j-1}) + \omega_4 F(x_i,y_j),$$

where the weights $\omega_1,\ldots,\omega_4$ are chosen such that (3.3) is exact, if F is bilinear on $[x_{i-1},x_i]\times[y_{j-1},y_j]$. If we use (3.3) to approximate the integral I defined by (2.2), it is consequently used to approximate the entries of the matrices A_1, A_2 and A_3. The result is a further trimming of these matrices: A_1 and A_3 are reduced to penta-diagonal matrices and A_3 even to a tri-diagonal one. The overall reduction is from 27 to 11 diagonals. The vanishing of so many matrix entries is due to the fact that the corresponding integrands of (3.2) vanish on all the grid-points (x_i,y_j), hence (3.3) yields a zero value. It can be proved that the order of accuracy of the finite element solution is not affected by using (3.3) (see STRANG & FIX [5], ch.IV).

TABLE I. Results of test - example

INTEGRAL VALUE	-.0146
HALL PARAMETER	13.3000
GAMMA	.2500
HARTMANN CONSTANT	23.0000
XO	.0625
JUMPING POINTS OF F1(Y)	.2500 .3125

V

X \ Y	0.0000	.1000	.1500	.2000	.2500	.2800	.3125	.3400	.4200	.5000	.6000	.7000	.8000	.9000	1.000
0.0000	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0
.0200	-.10	-.10	-.11	-.11	-.11	-.11	-.12	-.12	-.13	-.14	-.17	-.21	-.28	-.38	0.0
.0400	-.20	-.21	-.21	-.21	-.22	-.22	-.23	-.24	-.26	-.28	-.34	-.42	-.54	-.71	0.0
.0625	-.31	-.32	-.32	-.33	-.34	-.35	-.35	-.36	-.39	-.43	-.51	-.62	-.79	-1.00	0.0
.0800	-.40	-.40	-.41	-.42	-.43	-.44	-.45	-.46	-.49	-.54	-.63	-.76	-.94	-1.14	0.0
.1000	-.49	-.50	-.50	-.51	-.53	-.54	-.55	-.56	-.60	-.66	-.75	-.89	-1.08	-1.24	0.0
.1500	-.70	-.70	-.71	-.73	-.74	-.75	-.77	-.78	-.83	-.89	-1.00	-1.14	-1.30	-1.37	0.0
.2000	-.87	-.88	-.89	-.90	-.91	-.93	-.94	-.95	-1.00	-1.06	-1.16	-1.28	-1.41	-1.38	0.0
.2500	-1.00	-1.01	-1.02	-1.03	-1.05	-1.06	-1.07	-1.08	-1.12	-1.18	-1.26	-1.36	-1.44	-1.34	0.0
.3750	-1.19	-1.20	-1.20	-1.21	-1.22	-1.23	-1.24	-1.25	-1.27	-1.31	-1.36	-1.42	-1.44	-1.25	0.0
.5000	-1.20	-1.20	-1.21	-1.21	-1.22	-1.22	-1.22	-1.22	-1.24	-1.25	-1.28	-1.31	-1.30	-1.09	0.0
.6250	-1.06	-1.06	-1.07	-1.07	-1.07	-1.07	-1.07	-1.07	-1.07	-1.07	-1.08	-1.09	-1.06	-.87	0.0
.7500	-.81	-.82	-.82	-.82	-.82	-.82	-.81	-.81	-.79	-.79	-.78	-.78	-.75	-.61	0.0
.8750	-.46	-.47	-.47	-.48	-.47	-.47	-.46	-.45	-.44	-.42	-.41	-.41	-.39	-.31	0.0
1.0000	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0

ϕ

X \ Y	0.0000	.1000	.1500	.2000	.2500	.2800	.3125	.3400	.4200	.5000	.6000	.7000	.8000	.9000	1.000
0.0000	-20.55	-20.50	-20.43	-20.34	-20.21	-20.11	-20.00	-19.88	-19.47	-18.90	-17.86	-16.23	-13.52	-8.75	0.0
.0200	-20.57	-20.52	-20.45	-20.36	-20.23	-20.14	-20.02	-19.91	-19.51	-18.95	-17.95	-16.39	-13.83	-9.34	-.9
.4000	-20.61	-20.56	-20.50	-20.40	-20.28	-20.19	-10.08	-19.98	-19.60	-19.07	-18.15	-16.75	-14.52	-10.70	-3.3
.0625	-20.68	-20.63	-20.57	-20.49	-20.38	-20.30	-20.19	-20.10	-19.75	-19.29	-18.49	-17.34	-15.65	-13.09	-8.9
.0800	-20.75	-20.71	-20.66	-20.58	-20.47	-20.40	-20.31	-20.22	-19.90	-19.49	-18.80	-17.86	-16.59	-14.95	-13.3
.1000	-20.86	-20.82	-20.77	-20.70	-20.60	-20.54	-20.45	-20.38	-20.10	-19.75	-19.18	-18.45	-17.57	-16.65	-16.0
.1500	-21.19	-21.16	-21.13	-21.08	-21.01	-20.97	-20.91	-20.86	-20.69	-20.48	-20.17	-19.81	-19.45	-19.15	-19.0
.2000	-21.57	-21.56	-21.53	-21.50	-21.46	-21.44	-21.40	-21.37	-21.27	-21.15	-20.90	-20.82	-20.66	-20.55	-20.5
.2500	-21.98	-21.97	-21.95	-21.93	-21.91	-21.90	-21.88	-21.86	-21.81	-21.74	-21.66	-21.58	-21.52	-21.47	-21.4
.3750	-22.98	-22.98	-22.97	-22.97	-22.96	-22.96	-22.95	-22.95	-22.94	-22.92	-22.90	-22.88	-22.87	-22.86	-22.8
.5000	-23.83	-23.83	-23.83	-23.83	-23.83	-23.82	-23.82	-23.82	-23.82	-23.81	-23.81	-23.80	-23.80	-23.80	-23.8
.6250	-24.54	-24.54	-24.54	-24.54	-24.54	-24.54	-24.54	-24.53	-24.53	-24.53	-24.52	-24.52	-24.52	-24.52	-24.5
.7500	-25.14	-25.14	-25.14	-25.14	-25.13	-25.13	-25.13	-25.13	-25.12	-25.11	-25.10	-25.09	-25.08	-25.08	-25.0
.8750	-25.69	-25.69	-25.68	-25.67	-25.66	-25.65	-25.64	-25.64	-25.61	-25.58	-25.54	-25.52	-25.49	-25.48	-25.4
1.0000	-26.22	-26.25	-26.28	-26.33	-26.31	-26.24	-26.18	-26.12	-26.00	-25.90	-25.80	-25.73	-25.69	-25.66	-25.6

Figure 3. Graph of $\phi(x,y)$

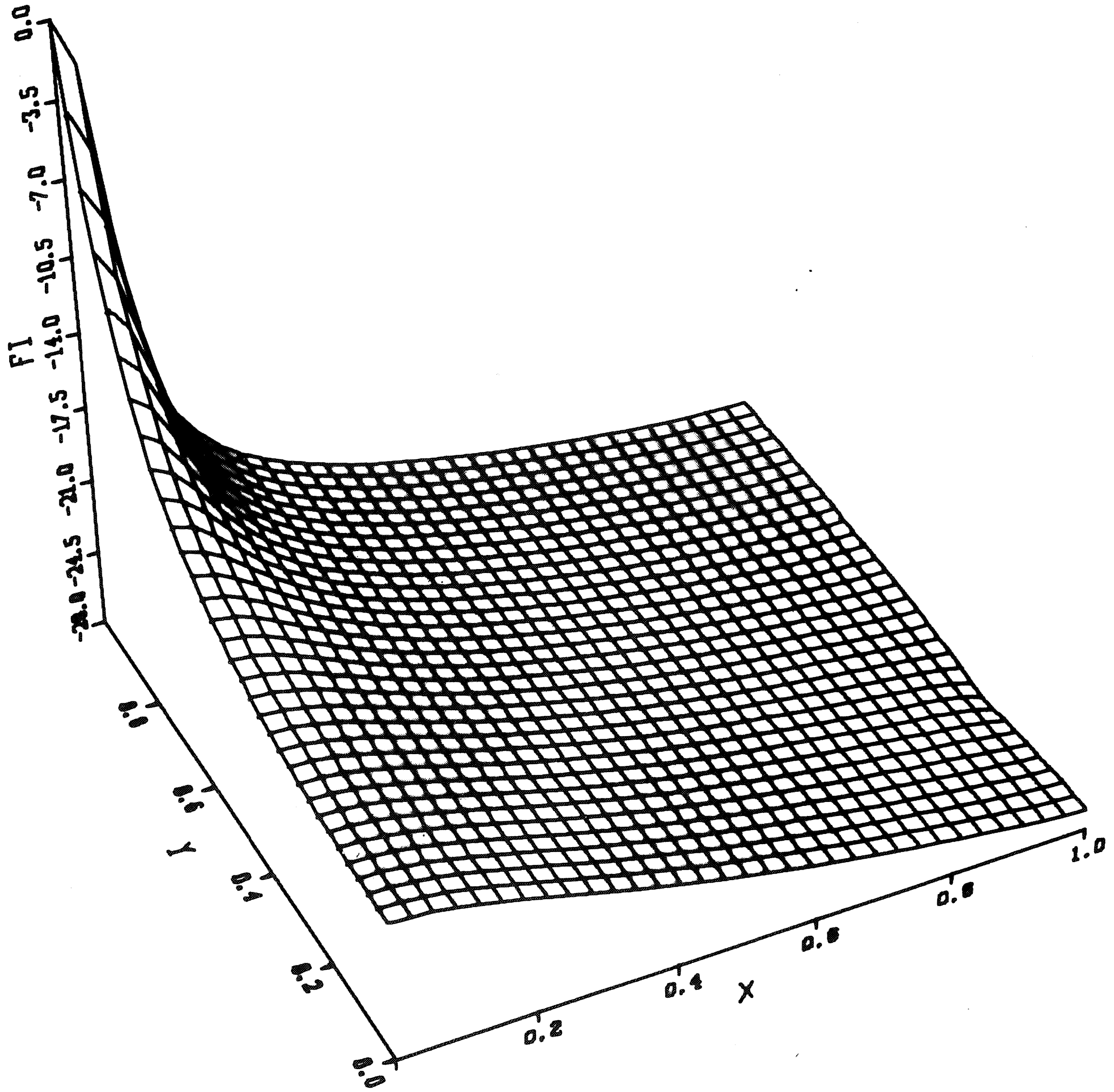
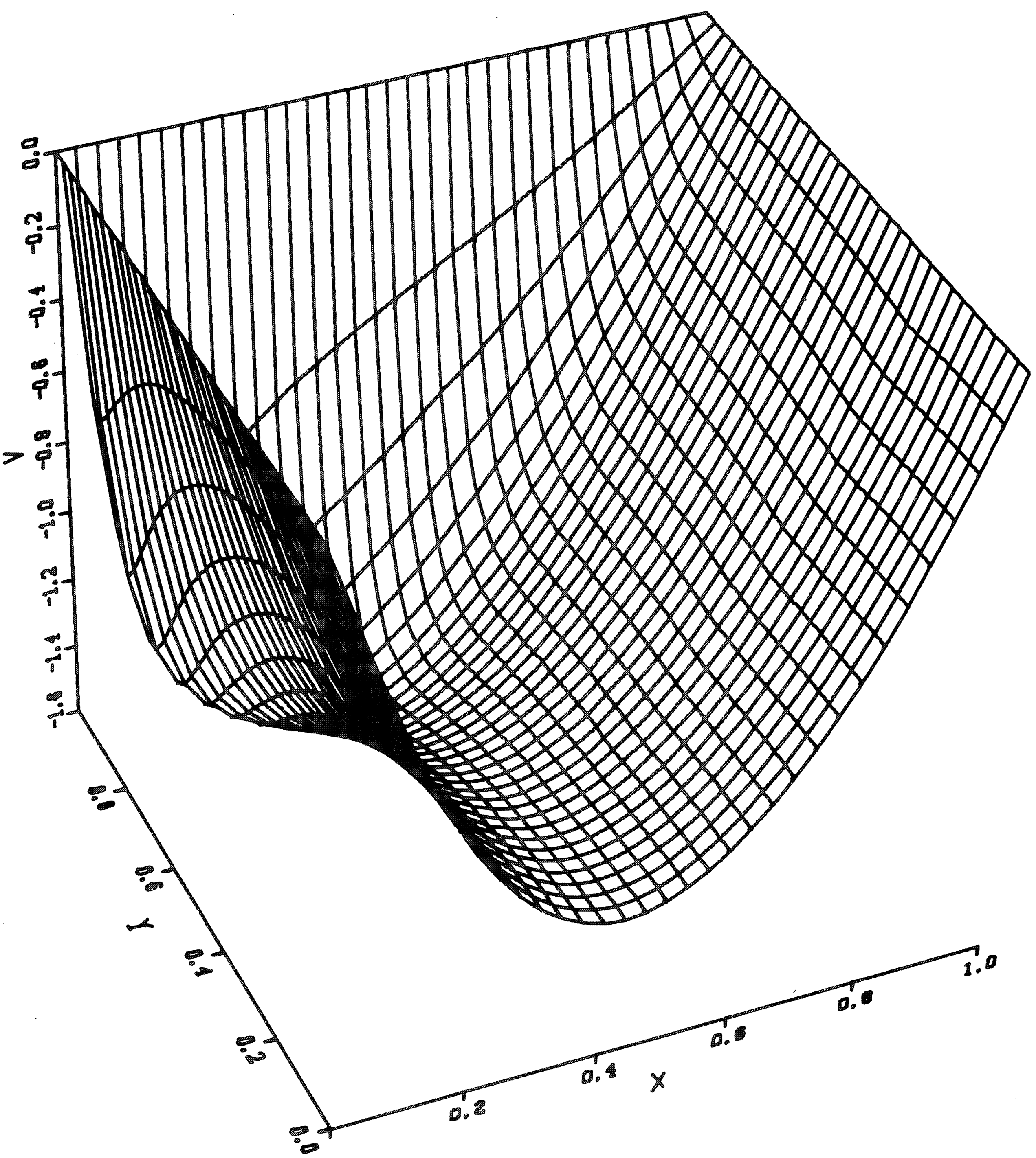


Figure 4. Graph of v(x,y)



Solution of the linear system

The linear system is solved by means of the conjugate gradient method (CG method). This iterative method (see REID [4]) is very suitable for symmetric nonnegative definite systems of sparse structure.

4. TEST-EXAMPLE

In the test-problem we solved (see table I), $f_1(y)$ and $f_2(x)$ are piecewise constant functions defined by

$$f_1(y) = \begin{cases} C_1, & 0.25 \le y \le 0.3125; \\ 0\ , & \text{elsewhere}; \end{cases}$$

$$f_2(x) = \begin{cases} C_2, & 0 \le x \le 0.0625; \\ 0\ , & \text{elsewhere}. \end{cases}$$

From the value of $\int_0^1 f_1(y)dy$, it turns out that C_1 and C_2 have the values -0.23944 and 0.67188, respectively.

For plotting the graphs of v and ϕ (see figs. 3 and 4), the library DISSPLA was used. We see that near the cathodes both v and ϕ have boundary layers. They also have boundary layers near the anode but less significant ones.

5. PROGRAM DESCRIPTION

In this chapter, we give a brief description of the subroutine PLASMA and the other subprograms. For a more detailed description, we refer to the comment lines in the software package.

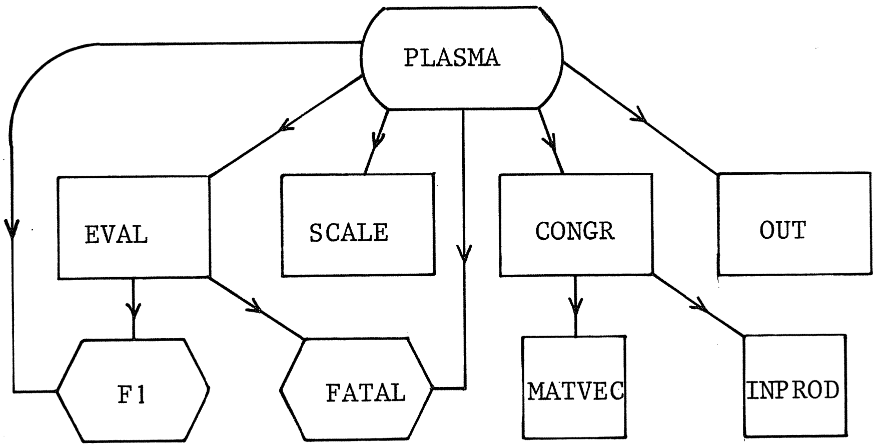


Figure 5. Hierarchical structure of subprograms

PLASMA- This is the driving subroutine which reads the input data, solves the problem and enables the user to manipulate the output data. See also next §.

EVAL- This subroutine computes the matrix and the right hand side of problem (3.6).

SCALE- This subroutine scales system (3.1) to

$$x = Dy \tag{4.1}$$

$$Sy = DADy = Db,$$

where D is a diagonal matrix whose entries are given by

$$d_i = a_{ii}^{-\frac{1}{2}}$$

CONGR- This subroutine iteratively solves (4.1) by means of the CG method.

MATVEC- This subroutine computes the matrix-vector product Sv, given a vector v.

INPROD- This subroutine computes the inner product of two vectors.

FATAL- This subroutine terminates the program if inconsistencies of the input record are discovered, or if the workspace (see §6) is not large enough.

F1- In this subprogram, f_1 is assignated its value. Only the profile is to be given, i.e. the function value, up to a constant factor. This factor is computed from the value of

$$(5.2) \qquad Q = \int_0^1 f_1(y)dy$$

which is to be given on the input record.

OUT- In this subroutine, the user is allowed to manipulate the input and output data, as he desires. A default subroutine is given in the program.

6. WORKSPACE

PLASMA has a work-array of dimension NWORK as formal parameter. NWORK should be at least 31 MN, where M and N are the respective lengths of the x-grid and y-grid.

7. THE INPUT RECORD

The following parameters have to be read in

1)	Q, β, γ, Ha, x_0	in FORMAT (F12.4);
2)	M	in FORMAT (I4);
3)	$x_1,\ldots,x_M$	in FORMAT (F12.4);
4)	N,	in FORMAT (I4);
5)	$y_1,\ldots,y_N$	in FORMAT (F12.4);
6)	ND	in FORMAT (I4);
7)	$yd_1,\ldots,yd_{ND}$	in FORMAT (F12.4);

where Q, β, γ, Ha and x_0 are given by (5.2), (1.2) and (1.8), respectively, and M, N and ND are the number of x-points, the number of y-points and the number of discontinuity points of $f_1(y)$, respectively.

The user has to take care of the following

a) the x-grid and y-grid have to be strictly monotone and, of course, x_1 and y_1 should be zero, while x_M and y_N should be one. If not, the program is terminated and a message of the reason is printed;

b) The array of discontinuity points *yd* should contain at least one point, even if $f_1(y)$ is continuous. In this case, the user is advised to give zero or one. Furthermore, each of the discontinuity points should occur in the y-grid. If not, the program is terminated with a message of the reason;

c) The x-grid should contain x_0, otherwise the program is terminated too;

d) Around the discontinuity points of $f_1(y)$ and $f_2(x)$, the user is advised to refine the grid. This especially holds for x_0, since there the jump of $\partial\phi/\partial n$ is generally larger than near the discontinuity points of $f_1(y)$. This may be illustrated by figure 4. However, he should take care that the rectangles of the partition do not become too "lean", i.e. the ratio of length and width should remain reasonable. For an argumentation, we refer to STRANG & FIX [5].

8. THE COMMON BLOCKS

8.1 *COMMON/PRBLM/*

GAMSQ - γ^2

GAMSQB - $\gamma^2(1+\beta^2)$

HASQ - $Ha^2/(1+\beta^2)$

NXO - grid number of x_0: $X(NXO) = x_0$;

AA - value of $(1+\beta^2) \int_0^1 f_1(y)dy$;

XO - x_0

FAC - scaling factor of $f_1(y)$.

8.2 *COMMON/WKSP/*

Except the three last members, this block contains integer pointers, locating the arrays within the global array *WORK*.

MATRIX - locates the matrix of the system

*WORK (MATRIX + 3*L - 2)* - L-th non-zero entry;

*WORK (MATRIX + 3*L - 1)* - its row number;

*WORK (MATRIX + 3*L)* - its column number; L = 1,...,NEL/3

NX - locates the x-grid

WORK (NX + L) - L-th x-point, L = 1,...,M;

NY - locates the y-grid

WORK (NY + L) - L-th y-point, L = 1,...,N;

NFYD - locates the jumping points of $f_1(y)$

WORK (NFYD + L) - L-th jumping point, L = 1,...,ND;

NFR, NSC, NG, NH - locate auxiliary arrays of dimension 2MN

NU - locates the approximate solution of (1.3)

*WORK (NU + (J-1) * M + I)* - approximation of v in (x(I),y(J));

*WORK (NU + MN + (J-1) * M + I)*- approximation of ϕ in (X(I),y(J));

NEL - three times the number of non-zero entries in the strict upper-diagonal part;

NBOUND - the actual dimension of workspace needed;

NWORK - the dimension of *WORK*.

Acknowledgements. The authors wish to thank professor J. Kistemaker for his assistance and Mr. F. Vitalis for his careful reading of the manuscript.

LITERATURE

[1] M.S. VAN DEN BERG, *Theory on a partially ionized gas centrifuge*, to appear in 1979 at FOM.

[2] R. COURANT & D. HILBERT, *Methods of Mathematical Physics*, vol. I, Interscience, New York, 1953.

[3] A.R. MITCHELL & R. WAIT, *The Finite Element Method in Partial Differential Equations*, Wiley and Sons, Chichester, 1977.

[4] J.K. REID (editor), *Large Sparse Sets of Linear Equations*, p.231-ff., Academic Press, London, 1971.

[5] G. STRANG & G.J. FIX, *An Analysis of the Finite Element Method*, Prentice-Prentice-Hall, inc., Englewood Cliffs, N.J., 1973.

APPENDIX

Below, the sourcetexts of the subprograms, the test-program and its input record are printed.

```
C
      SUBROUTINE PLASMA(WORK,NWORK)
      DIMENSION WORK(NWORK)
C
C-----------------------------------------------------------------------
C     THIS PACKAGE CONSISTS OF THE FOLLOWING PARTS :
C
C         (1) THE (DRIVING) SUBROUTINE PLASMA
C
C         (2) THE SUBROUTINE EVAL
C
C         (3) THE SUBROUTINE SCALE
C
C         (4) THE SUBROUTINE CONGR
C
C         (5) THE SUBROUTINE MATVEC
C
C         (6) THE SUBROUTINE INPROD
C
C         (7) THE SUBROUTINE FATAL
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C     THE PARTS (8) - (11) TOGETHER FORM THE TEST PROGRAM ; THEY ALSO
C     SERVE AS AN ILLUSTRATION OF THE USE OF THE SOFTWARE PACKAGE
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C         (8) THE MAIN PROGRAM ,TO BE WRITTEN BY THE USER
C
C         (9) THE FUNCTION F1 ,TO BE WRITTEN BY THE USER
C
C        (10) THE SUBROUTINE OUT ,TO BE WRITTEN BY THE USER
C
C        (11) AN INPUT RECORD , TO BE PROVIDED BY THE USER
C
C-----------------------------------------------------------------------
```

```
C
C     AT THE BEGINNING OF THE SUBROUTINE PLASMA , THE USER IS INSTRUCTE
C     HOW TO USE THE PROGRAM IN THE FOLLOWING SEVEN SECTIONS :
C
C         (I)  DESCRIPTION OF THE PROBLEM IT SOLVES
C
C        (II)  METHOD OF SOLVING THE BOUNDARY PROBLEM
C
C       (III)  BRIEF DESCRIPTIONS OF THE OTHER SUBPROGRAMS
C
C        (IV)  IMPLEMENTATION OF F1
C
C         (V)  INPUT PARAMETERS
C
C        (VI)  IMPLEMENTATION OF OUT
C
C       (VII)  WORKSPACE
C
C------------------------------------------------------------------
C
C        (I)   DESCRIPTION OF THE PROBLEM
C
C     PLASMA NUMERICALLY SOLVES THE ELLIPTIC BOUNDARY PROBLEM
C
C
C              VXX + VX/X - V/X**2 + GAMMA**2*VYY -
C     (1)
C                  - (HA**2/(1.+BETA**2))*(V - FIX) = 0,
C
C
C               FIXX + FIX/X +
C     (2)
C                   + GAMMA**2*(1 + BETA**2)*FIYY - VX - V/X = 0,
C
C
C          0 < X < 1 ;   0 < Y < 1,
C
C
C     WHERE
C
C
C        V - VELOCITY OF ROTATING PLASMA
C
C        VX = DV/DX  ;  VXX = DVX/DX ; VY = DV/DY  ;  VYY = DVY/DY
C
C        FI - ELECTRIC POTENTIAL WITHIN CYLINDER ;
C
C        FIX = DFI/DX ; FIXX = DFIX/DX ; FIY = DFI/DY ; FIYY = DFIY/DY
C
C
```

```
C      PROBLEM PARAMETERS:
C
C
C         HA     -   HARTMANN CONSTANT;
C
C         GAMMA -   INVERSE ASPECT RATIO, I.E. A SCALING PARAMETER;
C
C         BETA   -   HALL PARAMETER ;
C
C
C
C      THE BOUNDARY CONDTIONS ARE:
C
C          X = 0 : V = 0  ;  FIX = 0;
C
C          Y = 0 : VY = 0 ;  FIY = 0;
C
C          X = 1 : V = 0  ;  FIX = F1(Y);
C
C          Y = 1 : V = 0  ;  FIY = F2(Y);
C
C
C      F2(X) AND F1(Y) SATISFY THE INTEGRAL RELATION
C
C
C         INTEGRAL(0,1,F1(Y)DY)   +
C
C                  +  GAMMA**2*(1 + BETA**2)*INTEGRAL(0,1,X*F2(X)DX)   =  0
C
C
C
C      FOR A MORE DETAILED DESCRIPTION OF THE PROBLEM WE REFER TO
C      THE LONG WRITE-UP .
C
C-----------------------------------------------------------------------
C
C         (II)  METHOD OF SOLVING THE BOUNDARY PROBLEM
C
C      THIS PROBLEM IS SOLVED BY THE FINITE ELEMENT METHOD;
C      THE USER HAS TO PROVIDE GRIDS OF THE X-INTERVAL AND
C      THE Y-INTERVAL IN A WAY TO BE SPECIFIED IN THE CHAPTER
C      ON THE INPUT PARAMETERS ; THE PROGRAM DELIVERS APPROXIMATIONS
C      OF V(X(I),Y(J)) AND FI(X(I),Y(J)) ; SEE SECTION (VI);
C
C      SEE FOR A DESCRIPTION OF THE FINITE ELEMENT METHOD
C      THE COVERING PAPER AND FOR A DESCRIPTION OF THE OUTPUT
C      SECTION (VI) ;
C
C-----------------------------------------------------------------------
C
```

```
C          (III) BRIEF DESCRIPTION OF THE SUBPROGRAMS
C
C
C          SUBROUTINE EVAL
C
C          THIS SUBROUTINE EVALUATES THE MATRIX AND THE RIGHT HAND
C          SIDE OF THE LINEAR SYSTEM TO SOLVED ; THE NON-ZERO ENTRIES
C          OF THE SPARSE MATRIX AND THE COMPONENTS OF THE RIGHT HAND
C          SIDE ARE STORED IN THE WORK-ARRAY WORK ;
C
C          SUBROUTINE SCALE
C
C          THIS SUBROUTINE SCALES THE OTHERWISE ILL-CONDITIONED PROBLEM
C          TO A MORE STABLE PROBLEM
C
C          SUBROUTINE CONGR
C
C          THIS SUBROUTINE ITERATIVELY SOLVES THE SPARSE
C          PROBLEM BY MEANS OF THE CONJUGATE GRADIENT METHOD
C
C          SUBROUTINE MATVEC
C
C          THIS SUBROUTINE COMPUTES THE MATRIX-VECTOR PRODUCT A*P
C          FOR ANY GIVEN INPUT VECTOR P ; THE NON-ZERO ENTRIES OF
C          A PLUS THEIR LOCATIONS ARE STORED IN THE ARRAY WORK ;
C
C          SUBROUTINE FATAL
C
C          THIS SUBROUTINE TERMINATES THE PROGRAM IF AN INCONSISTENCY
C          ONTHE INPUT RECORD IS DISCOVERED , E.G. NON-MONOTONICITY OF
C          THE X-GRID ; AN EXPLAINING MESSAGE IS PRINTED
C
C          MAIN PROGRAM NAMED MCFOM
C
C          IN THE MAIN PROGRAM THE WORK-ARRAY IS DECLARED WTH THE PROPER
C          BOUNDS (SEE SECTION VI) AFTER WHICH THE DRIVING SUBROUTINE
C          PLASMA IS CALLED
C
C          FUNCTION F1(Y)
C
C          THIS FUNCTION IS PART OF THE BOUNDARY PROBLEM AND IS TO BE
C          IMPLEMENTED BY THE USER ; SEE SECTION (IV)
C
C          SUBROUTINE OUT(X,Y,YDISC,V,FI,M,N,,NDISC,AA,BETA,GAMMA,HA,X0)
C
C          THIS SUBROUTINE ENABLES THE USER TO MANIPULATE THE SOLUTION OF
C          THE PROBLEM AS HE DESIRES ; SEE SECTION (VI)
C
C-----------------------------------------------------------------
```

```
C
C         (IV) IMPLEMENTATION OF F1
C
C
C     F2(X) IS POSITIVE CONSTANT ON THE INTERVAL [0,X0], X0 = 0.06,
C     AND ZERO ELSEWHERE ; THE VALUE OF F2(X) IS INDIRECTLY
C     GIVEN BY THE NUMERIC VALUE OF INTEGRAL(0,1,F1(Y)DY);
C          ------
C     OF F1(Y)  , ONLY THE PROFILE IS GIVEN , I.E. THE FUNCTION VALUE
C     UP TO A CONSTANT FACTOR ; F1(Y) IS GIVEN BY MEANS OF A FUNCTION
C     SUBPROGRAM OF THE FORM
C
C        FUNCTION F1(Y)
C        REAL Y
C
C           . ASSIGNATION OF F1
C
C        RETURN
C        END
C
C----------------------------------------------------------------------
C
C        (V)  INPUT PARAMETERS
C
C     ON AN INPUT RECORD , THE FOLLOWING PARAMETERS HAVE TO BE GIVEN ;
C     BETWEEN BRACKETS , THEIR FORMAT IS GIVEN ;
C
C
C
C        AA       -    THE NUMERIC VALUE OF INTEGRAL(0,1,F1(Y)DY)
C                              (F12.4)
C
C        BETA     -    THE HALL PARAMETER
C                              (F12.4)
C
C        GAMMA    -    THE INVERSE ASPECT RATIO
C                              (F12.4)
C
C        HA       -    THE HARTMANN CONSTANT
C                              (F12.4)
C
C        X0       -    THE JUMPING POINT OF F2(X)
C                              (F12.4)
C
C
C     AFTER THESE PARAMETERS , A PARTITION OF THE X-INTERVAL HAS TO BE
C     GIVEN  THIS PARTITION IS OF THE FORM
C
C        0.0 = X(1) < X(2) < ... < X(M) = 1.0;
C
```

```
C     AND MUST AT ANY RATE CONTAIN THE POINTS 0, X0 AND 1 ; OTHERWISE ,
C     THE PROGRAM IS TERMINATED IMMEDIATELY; THE PARTITION HAS TO BE
C     GIVEN IN THE FOLLOWING FORM:
C
C        M       -   THE NUMBER OF X-GRIDPOINTS
C                             (I4)
C
C        X(1)    -   THE FIRST GRID-POINT ; THIS SHOULD BE EQUAL TO ZERO
C                             (F12.4)
C          .
C          .
C          .
C          .
C
C        X(M)    -   THE LAST GRID-POINT ; THIS SHOULD BE EQUAL TO ONE ;
C                             (F12.4)
C
C     AFTER THIS X-GRID , A PARTITION OF THE Y-INTERVAL HAS TO BE
C     READ   THIS PARTITION IS OF THE FORM
C
C        0.0 = Y(1) < Y(2) < ... < Y(N) = 1.0;
C
C     SINCE , HOWEVER , F1(Y) MAY BE DISCONTINUOUS , THE Y-GRID SHOULD
C     CONTAIN ANY DISCONTINUITY POINTS ; TO THIS END , THE USER IS
C     OBLIGED TO GIVE THESE POINTS ON THE INPUT RECORD ; FOR TECHNICAL
C     REASONS , HE SHOULD GIVE AT LEAST ONE JUMPING POINT ;
C     SO IF F1(Y) IS CONTINUOUS , HE SHOULD GIVE 0.0 OR 1.0 AS
C     (DUMMY) DISCONTINUITY POINTS
C     SUMMARILY , HE SHOULD GIVE THE FOLLOWING PARAMETERS :
C
C        N        -   THE NUMBER OF Y - POINTS
C                             (I4)
C
C         Y(1) -      THE FIRST Y-GRID-POINT , Y(1) = 0.0
C                             (F12.4)
C           .
C           .
C           .
C           .
C
C         Y(N) -      THE LAST Y-GRID-POINT , Y(N) = 1.0
C                             (F12.4)
C
C        NDISC -      THE NUMBER OF DISCONTINUITY POINTS OF F1(Y)
C                             (I4)
C
C        YD(1) -      THE FIRST DISCONTINUTY POINT OF F1(Y)
C                             (F12.4)
C           .
C           .
C           .
C
C
C        YD(ND)-      THE LAST DISCONTINUTY POINT OF F1(Y)
C                             (F12.4)
C
C-----------------------------------------------------------------------
```

```
C
C        (VI)   IMPLEMENTATION OF OUT
C
C     THE PROBLEM PARAMETERS AA , BETA , GAMMA , HA , X0  AND
C     THE ARRAYS  X(M)  , Y(N)  , YDISC(NDISC)  , V(M,N) AND FI(M,N)
C     ARE THE FORMAL PARAMETERS OF OUT , WHERE
C
C        X(1)  , X(2)  , ... , X(M) ARE THE GRID-POINTS OF THE X-INTERVAL
C
C        Y(1)  , Y(2)  , ... , Y(N) ARE THE GRID-POINTS OF THE Y-INTERVAL
C
C        YDISC(1)  ,  ... , YDISC(NDISC) ARE THE JUMPING POINTS OF F1(Y)
C
C        V(I,J) IS AN APPROXIMATION OF V(X(I),Y(J)) ,
C                                 I = 1 , ... ,  M ; J = 1 , ...  , N
C
C        FI(I,J) IS AN APPROXIMATION OF FI(X(I),Y(J)) ,
C                                 I = 1 , ... ,  M ; J = 1 , ...  , N
C
C
C     BY MEANS OF THIS SUBROUTINE THE USER CAN MANIPULATE THE
C     INPUT AND OUTPUT PARAMETERS AS HE DESIRES .
C
C     THE SUBROUTINE IS OF THE FORM
C
C        SUBROUTINE OUT(X,Y,YDISC,V,FI,M,N,NDISC,AA,BETA,GAMMA,HA,X0)
C        DIMENSION X(M)  , Y(N)  , YDISC(NDISC)


C        DIMENSION V(M,N)  , FI(M,N)
C
C              .
C              .  MANIPULATION WITH THE PARAMETERS
C              .
C
C        RETURN
C        END
C
C---------------------------------------------------------------------
C
C     (VII) WORKSPACE
C
C     THE ONLY PARAMETER OF PLASMA IS A WORK-ARRAY OF DIMENSION
C     NWORK ; THE USER HAS TO TAKE CARE THAT NWORK SHOULD AT LEAST BE
C     EQUAL TO 31*M*N ; IF NOT , THE PROGRAM MAY BE TERMINATED WITH
C     A MESSAGE OF THE REASON ;
C
C
C
C---------------------------------------------------------------------
C
```

```
C
      COMMON /PRBLM/ GAMSQ, GAMSQB, HASQ, NX0, AA, X0, FAC
      COMMON /WKSP/ MATRIX, NX, NY, NFYD, NFR,
     S NSC, NU, NG, NH, NEL, NBOUND, NWRK
      LOGICAL LOG
      NWRK = NWORK
      DO 10 I = 1,NWRK
10    WORK(I) = 0.0
C
C     NOW, THE INTEGRAL VALUE AA, THE HALL PARAMETER BETA,
C     THE RATIO ASPECT GAMMA, THE HARTMANN CONSTANT AND X0 ARE READ
C
      READ(5,9991) AA, BETA, GAMMA, HA, X0
      GAMSQ = GAMMA**2
      GAMSQB = GAMSQ*(1.+BETA**2)
      HASQ = HA**2/(1.+BETA**2)
      AA = AA*(1. + BETA**2)
      NX = 0
C
C     THE NUMBER OF X-POINTS IS READ
C
      READ(5,9992) M
      NBOUND = M
C
C     THE X-POINTS ARE READ
C
      DO  20 L = 1,M
 20   READ(5,9991) WORK(NX + L)
C
C     THE MONOTONICITY OF THE X-GRID IS CHECKED, AND A CHECK
C     IS MADE IF X(1) = 0.0 AND X(M) = 1.0
C
      DO  30 L = 2,M
 30   IF (WORK(NX + L) .LE. WORK(NX + L -1))
     S CALL FATAL(30HX-GRID NOT STRICTLY MONOTONE     )
      IF (WORK(NX + 1) .NE. 0.)
     S CALL FATAL(30H X(1) UNEQUAL ZERO                )
      IF (WORK(NX + M) .NE. 1.)
     S CALL FATAL(30H X(M) UNEQUAL ONE                 )
C
C     THE NUMBER OF Y-POINTS IS READ AND THE NUMBER OF
C     POINTS WHERE F1(Y) IS DISCONTINUOUS


C
      READ(5,9992) N
C
      NBOUND = NBOUND + N
      NY = M
      NFYD= NY + N
      DO  40 L = 1,N
 40   READ(5,9991) WORK(NY+L)
C
```

```
      READ(5,9992) NDISC
      NBOUND = NBOUND + NDISC
      DO  50 L = 1,NDISC
 50   READ(5,9991) WORK(NFYD+L)
C
C     THE MONOTONICITY OF THE Y-GRID IS CHECKED ;
C     FURTHERMORE , Y(1) AND Y(N) ARE CHECKED TO BE
C     EQUAL TO ZERO AND ONE , RESPECTIVELY.
C
C
      DO  60 L = 2,N
 60   IF (WORK(NY + L) .LE. WORK(NY + L -1))
     S CALL FATAL(30H Y-GRID NON-MONOTONE                  )
      IF (WORK(NY + N) .NE. 1.)
     S CALL FATAL(30H Y(N) UNEQUAL ONE                     )
      IF (WORK(NY + 1) .NE. 0.)
     S CALL FATAL(30H Y(1) UNEQUAL ZERO                    )
CC
C     THE X-GRID IS CHECKED TO CONTAIN X0
C
      NX0 = 0
      DO 5 L = 1,M
5     IF (ABS(X0-WORK(NX + L)) .LT. 1.E-10) NX0 = L
      IF (NX0 .LE. 1)
     S CALL FATAL(30H X-GRID DOES NOT CONTAIN X0          )
C
C     THE Y-GRID IS CHECKED TO CONTAIN THE DISCONTNUITY POINTS OF F1
C
      DO 80 L = 1,NDISC
      YDISC = WORK(L + NFYD)
      LOG = .FALSE.
      DO  70 I = 1,N
 70   LOG = LOG .OR. ABS(YDISC - WORK(NY+I)) .LT. 1.E-6
      IF (LOG) GOTO  80
      WRITE(6,9993) YDISC
      CALL FATAL (20H Y-GRID INCORRECT           )
 80   CONTINUE
C
C
C     THE SCALING FACTOR OF F1(Y) IS COMPUTED
C
      S = 0.
      DO 90 L = 2,N
      S = S + F1((WORK(NY+L)+WORK(NY+L-1))/2.)*(WORK(NY+L)-WORK(NY+L-1)
90    CONTINUE
      FAC = AA/S
C
      MN = M*N
      MN2 = MN*2
      NFR = NBOUND
      NSC = NFR + MN2
      NBOUND = NSC + MN2
      IF (NBOUND.GT.NWRK) CALL FATAL(30H UPPER BOUND OF WORK TOO SMALL)
C
C     NOW , THE SUBROUTINE EVAL IS CALLED ; THIS SUBROUTINE
```

```
C      COMPUTES THE NONZERO ENTRIES OF THE MATRIX A AND THE RIGHT HAND
C      SIDE VECTOR B
C
       CALL EVAL(M,N,MN,MN2,WORK)
C
C      NEXT , THE SYSTEM  A*X = B IS BEING SCALED TO THE SYSTEM
C
C            X = S*Y
C
C            S*A*S*Y = S*B ;
C
C      WITH S A DIAGONOAL MATRIX DEFINED BY
C
C            S(I) = A(I,I)**(-0.5)
C
       CALL SCALE(WORK)
       NU = NBOUND
       DO 100 L = 1,MN2
       WORK(NFR + L) = WORK(NFR + L)*WORK(NSC + L)
       WORK(NU + L) = 0.
100    CONTINUE
C
       NG = NU + MN2
       NH = NG + MN2
       NBOUND = NH + MN2
       IF (NBOUND.GT.NWRK) CALL FATAL(30H UPPER BOUND OF WORK TOO SMALL)
C
       TOL = 1.E-20
C
C      THE SCALED SYSTEM IS BEING ITERATIVELY SOLVED BY MEANS
C      OF THE CONJUGATE GRADIENT METHOD
C
       CALL CONGR(WORK(NU+1),WORK(NFR+1),
      S WORK(NG+1),WORK(NH+1),MN2,TOL,RES,WORK)
C
C      THE SOLUTION IS BEING RESCALED
C
       DO 110 I = 1,MN2
       WORK(NU + I) = WORK(NSC + I)*WORK(NU + I)
110    CONTINUE
       SUBTR = WORK(NU + MN2 - M + 1)
C
C      THE VELOCITY AND THE POTENTIAL ARE NOW COMPUTED DEFINITELY
C
       DO 130 L = 1,M
       XL = WORK(NX + L)
       DO 120 K = 1 , N
       LOCAL = (K - 1)*M + L
       WORK(NU + LOCAL) = WORK(NU + LOCAL )*XL
       WORK(NU + LOCAL + MN) = WORK(MN + NU + LOCAL ) - SUBTR
120    CONTINUE
130    CONTINUE
C
```

```
C      NOW , THE SUBROUTINE OUT IS CALLED , WHICH ENABLES THE USER
C      TO MANIPULATE THE OUTPUT DATA AS DESIRED
C
       CALL OUT(WORK(NX+1),WORK(NY+1),WORK(NFYD+1),WORK(NU+1),
     S WORK(NU+MN+1),M,N,NDISC,AA/(1.+BETA**2),BETA,GAMMA,HA,X0)
C
C
9991   FORMAT(F12.4)
9992   FORMAT(I4)
9993   FORMAT(///,4X,15H JUMPING POINT  ,F10.4,4X,
     S 30H IS NOT CONTAINED IN Y-GRID     ,///)
C
       STOP
       END
C
       SUBROUTINE EVAL(M,N,MN,MN2,WORK)
C
C-----------------------------------------------------------------------
C
C      THIS SUBROUTINE COMPUTES THE NON-ZERO ENTRIES OF THE MATRIX
C      AND THE RIGHT HAND SIDE OF THE LINEAR SYSTEM
C
C             A11*V + A12*FI = B1 ;
C
C             A21*V + A22*FI = B2
C
C
C      WHERE A11 AND A22 ARE SYMMETRIC PENTA-DIAGONAL AND A12 AND A21
C      ARE TRI-DIAGONAL AND A21 IS THE TRANSPOSE OF A12 ; ALSO ,
C      THE LOCATIONS OF THE NON-ZERO ENTRIES ARE COMPUTED;
C
C-----------------------------------------------------------------------
C
       COMMON /PRBLM/ GAMSQ, GAMSQB, HASQ, NX0, AA, X0, FAC
       DIMENSION WORK(1)
       COMMON /WKSP/ MATRIX, NX, NY, NFYD, NFR,
     S NSC, NU, NG, NH, NEL, NBOUND, NWRK
C
C      AT FIRST THE POINTERS DENOTING THE LOCATION OF THE DIAGONALS
C      IN THE ARRAY WORK ARE COMPUTED
C
C
C      A11
C
       NDIA11 = 0
C
C      THE MAIN DIAGONAL OF A11
C
       NC1U11 = NDIA11 + MN
C
C      THE FIRST CODIAGONAL OF A11
```

```
C
      NC2U11 = NC1U11 + MN - 1
C
C     THE SECOND CODIAGONAL OF A11
C
C
C     A12
C
      NC1L12 = NC2U11 + MN - M
C
C     THE LOWER CODIAGONAL OF A12
C
      NDIA12 = NC1L12 + MN - 1
C
C     THE MAIN DIAGONAL OF A12
C
      NC1U12 = NDIA12 + MN
C
C     THE UPPER CODIAGONAL OF A12
C
C
C     A22
C
      NDIA22 = NC1U12 + MN - 1
C
C     THE MAIN DIAGONAL OF A22
C
      NC1U22 = NDIA22 + MN
C     THE FIRST CO-DIAGONAL OF A22
C
      NC2U22 = NC1U22 + MN - 1
C     THE SECOND CO-DIAGONAL OF A22
C
      MATRIX = NBOUND
      NBOUND = NC2U22 + MN - M + NBOUND
      NEL = NBOUND - MATRIX
      WRITE(6,2) MATRIX + 3*NEL
      IF (NBOUND.GT.NWRK) CALL FATAL(30H UPPER BOUND OF WORK TOO SMALL)
C
C
C     RECTANGLE BY RECTANGLE , THE CONTRIBUTIONS OF THE NON-ZERO
C     ENTRIES ARE COMPUTED AND ADDED
C
      XR = 0.
      XR2 = 0.
      XR3 = 0.
      DO 30 I = 2,M
      XL = XR
      XL3 = XR3
      XL2 = XR2
      XR = WORK(NX + I)
      DX = XR - XL
```

```
      XR2 = XR*XR
      XR3 = XR*XR2
      XLXR2 = XL*XR2
      XRXL2 = XR*XL2
      XLXR = XL*XR
      DR = XR - XL
      VXL1 = XL/3. + XR/6.
      VXR1 = XL/6. + XR/3.
      VXM1 = VXL1 + VXR1
      VXL2 = XL2/4.+ XLXR/6. + XR2/12.
      VXR2 = XR2/4.+ XLXR/6. + XL2/12.
      VXM2 = VXL2 + VXR2
      VXL3 = XL3*.2 + XRXL2*.15  + XLXR2*.1 +  XR3*.05
      VXR3 = XR3*.2 + XRXL2*.1  + XLXR2*.15 +  XL3*.05
      VXM3 = VXR3 + VXL3
      YR = 0.
      DO 20 J = 2,N
      YL = YR
      YR = WORK(NY + J)
      DY = YR - YL
      VYL = 0.5
      VYM = 1.
      LOWL = (J - 2)*M + I - 1
      LOWR = LOWL + 1
      LUPL = LOWL + M
      LUPR = LUPL + 1
C
C     LOWL, LOWR, LUPL EN LUPR ARE THE INDICES OF THE LOWER-LEFT,
C     LOWER-RIGHT, UPPER-LEFT AND UPPER-RIGHT CORNER OF THE
C     RECTANGLE [X(I-1),X(I);Y(J-1),Y(J)] , RESPECTIVELY
C
C
C
C
C
C                  (X(I-1),Y(J))                    (X(I),Y(J))
C                        !                               !


C
C                  LUPL  *-------------------------------*  LUPR
C                        !                               !
C                        !                               !
C                        !                               !
C                  LOWL  *-------------------------------*  LOWR
C
C                        !                               !
C                  (X(I-1),Y(J-1))                  (X(I),Y(J-1))
C
C
```

```
C
C     LOWER-LEFT
C
      WORK(MATRIX + LOWL + NDIA11) = WORK(MATRIX + LOWL + NDIA11)
     $ + VXM3*VYL*DY/DX
     $ + VXL3*VYM*DX/DY*GAMSQ
     $ + VXL3*VYL*DX*DY*HASQ
      WORK(MATRIX + LOWL + NDIA12) = WORK(MATRIX + LOWL + NDIA12)
     $ + HASQ*VXL2*VYL*DY
      WORK(MATRIX + LOWL + NDIA22) = WORK(MATRIX + LOWL + NDIA22)
     $ + VXM1*VYL*DY/DX
     $ + VXL1*VYM*DX/DY*GAMSQB
      WORK(MATRIX + LOWL + NC1U11) = WORK(MATRIX + LOWL + NC1U11)
     $ - VXM3*VYL*DY/DX
      WORK(MATRIX + LOWL + NC2U11) = WORK(MATRIX + LOWL + NC2U11)
     $ - VXL3*VYM*DX/DY*GAMSQ
      WORK(MATRIX + LOWL + NC1U12) = WORK(MATRIX + LOWL + NC1U12)
     $ - HASQ*VXL2*VYL*DY
      WORK(MATRIX + LOWL + NC1U22) = WORK(MATRIX + LOWL + NC1U22)
     $ - VXM1*VYL*DY/DX
      WORK(MATRIX + LOWL + NC2U22) = WORK(MATRIX + LOWL + NC2U22)
     $ - VXL1*VYM*DX/DY*GAMSQB
C
C     LOWER-RIGHT
C
       WORK(MATRIX + LOWR + NDIA11) = WORK(MATRIX + LOWR + NDIA11)
     $ + VXM3*VYL*DY/DX
     $ + VXR3*VYM*DX/DY*GAMSQ
     $ + VXR3*VYL*DX*DY*HASQ
       WORK(MATRIX + LOWR + NDIA12) = WORK(MATRIX + LOWR + NDIA12)
     $ - HASQ*VXR2*VYL*DY
      WORK(MATRIX + LOWR + NDIA22) = WORK(MATRIX + LOWR + NDIA22)
     $ + VXM1*VYL*DY/DX
     $ + VXR1*VYM*DX/DY*GAMSQB
       WORK(MATRIX + LOWR + NC2U11) = WORK(MATRIX + LOWR + NC2U11)
     $ - VXR3*VYM*DX/DY*GAMSQ
      WORK(MATRIX + LOWL + NC1L12) = WORK(MATRIX + LOWL + NC1L12)
     $ + HASQ*VXR2*VYL*DY
      WORK(MATRIX + LOWR + NC2U22) = WORK(MATRIX + LOWR + NC2U22)
     $ - VXR1*VYM*DX/DY*GAMSQB
C
C     UPPER-LEFT
C
      WORK(MATRIX + LUPL + NDIA11) = WORK(MATRIX + LUPL + NDIA11)
     $ + VXM3*VYL*DY/DX
     $ + VXL3*VYM*DX/DY*GAMSQ
     $ + VXL3*VYL*DX*DY*HASQ
      WORK(MATRIX + LUPL + NDIA12) = WORK(MATRIX + LUPL + NDIA12)
     $ + HASQ*VXL2*VYL*DY
       WORK(MATRIX + LUPL + NDIA22) = WORK(MATRIX + LUPL + NDIA22)
     $ + VXM1*VYL*DY/DX
     $ + VXL1*VYM*DX/DY*GAMSQB
      WORK(MATRIX + LUPL + NC1U11) = WORK(MATRIX + LUPL + NC1U11)
```

```
     $ - VXM3*VYL*DY/DX
      WORK(MATRIX + LUPL + NC1U12) = WORK(MATRIX + LUPL + NC1U12)
     $ - HASQ*VXL2*VYL*DY
       WORK(MATRIX + LUPL + NC1U22) = WORK(MATRIX + LUPL + NC1U22)
     $ - VXM1*VYL*DY/DX
C
C     UPPER-RIGHT
C
       WORK(MATRIX + LUPR + NDIA11) = WORK(MATRIX + LUPR + NDIA11)
     $ + VXM3*VYL*DY/DX
     $ + VXR3*VYM*DX/DY*GAMSQ
     $ + VXR3*VYL*DX*DY*HASQ
       WORK(MATRIX + LUPR + NDIA12) = WORK(MATRIX + LUPR + NDIA12)
     $ - HASQ*VXR2*VYL*DY
      WORK(MATRIX + LUPR + NDIA22) = WORK(MATRIX + LUPR + NDIA22)
     $ + VXM1*VYL*DY/DX
     $ + VXR1*VYM*DX/DY*GAMSQB
      WORK(MATRIX + LUPL + NC1L12) = WORK(MATRIX + LUPL + NC1L12)
     $ + HASQ*VXR2*VYL*DY
C
C     EVALUATION OF THE RIGHT HAND SIDE
C
      IF (I .LT. M) GOTO 10
      DYFM = DY*F1((WORK(NY+J-1)+WORK(NY+J))/2.)*0.5*FAC
       WORK(NFR + LOWR + MN) =  WORK(NFR + LOWR + MN) + DYFM
       WORK(NFR + LUPR + MN) = WORK(NFR + LUPR + MN) + DYFM
10    IF (J .EQ. N .AND. I .LE. NX0 ) WORK(NFR + LUPL + MN) =
     S WORK(NFR + LUPL + MN)   - (2.*AA/X0**2)*VXL1*DX
      IF (J .EQ. N .AND. I .LE. NX0 ) WORK(NFR + LUPR + MN) =
     S WORK(NFR + LUPR + MN)   - (2.*AA/X0**2)*VXR1*DX
20    CONTINUE
30    CONTINUE
C
C     COMPUTATION OF THE ROW AND COLUMN NUMBERS OF THE
C     NON-ZERO ENTRIES
C
      IROW = NBOUND
      JCOL = IROW + NEL
      NBOUND = NBOUND + 2*NEL
      IF (NBOUND.GT.NWRK) CALL FATAL(30H UPPER BOUND OF WORK TOO SMALL)
C
      DO 50 L = 1,MN
      WORK(IROW+L + NDIA11) = L
      WORK(JCOL+L + NDIA11) = L
      WORK(IROW+L + NDIA12) = L
      WORK(JCOL+L + NDIA12) = L + MN
      WORK(IROW+L + NDIA22) = L + MN
      WORK(JCOL+L + NDIA22) = L + MN
C
      WORK(NFR + L+MN) = WORK(NFR + L+MN)*HASQ
      WORK(MATRIX + L + NDIA22) = WORK(MATRIX + L + NDIA22)*HASQ
```

```
C
C
      IF (L .GT. MN - M) GOTO 40
      WORK(IROW+L + NC2U11) = L
      WORK(JCOL+L + NC2U11) = L + M
C
      WORK(MATRIX + L + NC2U22) = WORK(MATRIX + L + NC2U22)*HASQ
C
      WORK(IROW+L + NC2U22) = L + MN
      WORK(JCOL+L + NC2U22) = L + MN + M
C
40    IF (L .EQ. MN) GOTO 50
C


      WORK(IROW+L + NC1U11) = L
      WORK(JCOL+L + NC1U11) = L + 1
      WORK(IROW+L + NC1L12) = L + 1
      WORK(JCOL+L + NC1L12) = L + MN
      WORK(IROW+L + NC1U12) = L
      WORK(JCOL+L + NC1U12) = L + MN + 1
C
      WORK(MATRIX + L + NC1U22) = WORK(MATRIX + L + NC1U22)*HASQ
C
      WORK(IROW+L + NC1U22) = L + MN
      WORK(JCOL+L + NC1U22) = L + MN + 1
50    CONTINUE
C
C     IMPLEMENTATION OF BOUNDARY CONDITIONS
C
C     V = 0 , X = 1
C
      DO 60 L = M,MN,M
      WORK(MATRIX + L + NDIA11) = 1.
      WORK(MATRIX + L + NDIA12) = 0.
      WORK(MATRIX + L + NC1L12 - 1) = 0.
      WORK(MATRIX + L + NC1U11 - 1) = 0.
      IF (L .NE. M) WORK(MATRIX + L + NC2U11 - M) = 0.
C
      IF (L .EQ. MN) GOTO 60
C
      WORK(MATRIX + L + NC1U11) = 0.
      WORK(MATRIX + L + NC1U12) = 0.
      WORK(MATRIX + L + NC2U11) = 0.
60    CONTINUE
C
C     V = 0 , Y = 1
C
```

```
      LOW = MN - M + 1
      LUP = MN - 1
      DO 70 L = LOW,LUP
      WORK(MATRIX + L + NDIA11) = 1.
      WORK(MATRIX + L + NDIA12) = 0.
      WORK(MATRIX + L + NC1L12 - 1) = 0.
      WORK(MATRIX + L + NC1U11 - 1) = 0.
      WORK(MATRIX + L + NC2U11 - M) = 0.
C
      WORK(MATRIX + L + NC1U11) = 0.
      WORK(MATRIX + L + NC1U12) = 0.
70    CONTINUE
C
C
C     IN ORDER TO AVOID SUPERFLUOUS MULTIPLICATIONS IN THE
C     SUBROUTINE MATVEC , ALL THE ZERO ENTRIES ARE ELIMINATED OUT
C     OF THE ARRAY WORK ; THIS IS DONE BY WRITING ALL THE NON-ZERO
C     ENTRIES PLUS THEIR ROW AND COLUMN NUMBERS TO A SCRATCH DISC
C     AND READING THEM AGAIN , THUS OVERWRITING THE ORIGINAL
C     WORKSPACE RESERVED FOR THE MATRIX
C
C
      REWIND 4
      IC = 0
      DO 80  L = 1,NEL
      IF (WORK(MATRIX + L) .EQ. 0.) GOTO 80
      IC = IC + 1
      WRITE(4) WORK(MATRIX + L),WORK(IROW+L),WORK(JCOL+L)
80    CONTINUE
      NEL = IC
      WRITE(6,1) NEL


      NEL = IC*3
      NBOUND = MATRIX + NEL
C
      REWIND 4
      DO 90  L = 3,NEL,3
      READ(4) WORK(MATRIX+L-2),WORK(MATRIX+L-1),WORK(MATRIX+L)
90    CONTINUE
1     FORMAT(3X,40H NUMBER OF NON-ZERO MATRIX ENTRIES            ,I5
2     FORMAT(3X,40H DIMENSION OF WORK SHOULD BE AT LEAST         ,I5
      RETURN
C     END
```

```
      SUBROUTINE SCALE(WORK)
C
C-----------------------------------------------------------------------
C
C     THIS SUBROUTINE SCALES THE LINEAR PROBLEM
C
C           A*X = B
C
C     TO THE PROBLEM
C
C           X = S*Y
C
C           S*A*S*Y = S*B
C
C     WHERE S IS A DIAGONAL MATRIX , WHOSE ENTRIES ARE DEFINED BY
C
C     S(I,I) = 1.0/SQRT(A(I,I))  , I = 1 , N
C
C-----------------------------------------------------------------------
C
      DIMENSION WORK(1)
      COMMON /WKSP/ MATRIX, NX, NY, NFYD, NFR,
     S NSC, NU, NG, NH, NEL, NBOUND, NWRK
      DO 10 L= 3,NEL,3
      I = IFIX(WORK(MATRIX+L-1))
      J = IFIX(WORK(MATRIX+L))
      IF (I.EQ.J) WORK(NSC + I) = 1./SQRT(WORK(MATRIX+L-2))
10    CONTINUE
      REWIND 4
      IC = 0
      DO 20 L= 3,NEL,3
      I = IFIX(WORK(MATRIX+L-1))
      J = IFIX(WORK(MATRIX+L))
      IF (I.EQ.J) GOTO 20
      WORK(MATRIX+L-2) = WORK(MATRIX+L-2)*WORK(NSC+I)*WORK(NSC+J)
      IC = IC + 1
      WRITE(4) WORK(MATRIX+L-2),WORK(MATRIX+L-1),WORK(MATRIX+L)
20    CONTINUE
      NBOUND = MATRIX + 3*IC
C
      NEL = IC*3
      REWIND 4
      DO 30 L = 3,NEL,3
30    READ(4) WORK(MATRIX + L-2),WORK(MATRIX + L-1),WORK(MATRIX + L)
      RETURN
      END
```

```
C
      SUBROUTINE CONGR(X,R,P,AP,N,TOL,RP,WORK)
      DIMENSION X(N),R(N),P(N),AP(N),WORK(1)
C
C---------------------------------------------------------------------
C
C     THIS SUBROUTINE SOLVES THE LINEAR SYSTEM
C
C           A*X = B
C
C     BY MEANS OF THE CONJUGATE GRADIENT METHOD
C
C     THE SUBROUTINE ONLY NEEDS INDIRECT ACCESS TO THE MATRIX A
C     BY MEANS OF A SUBROUTINE MATVEC(P,AP,N,WORK) WHICH COMPUTES THE
C     VECTOR A*P FOR ANY GIVEN VECTOR P AS INPUT PARAMETER
C
C---------------------------------------------------------------------
C
      IT = 0
      CALL MATVEC(X,P,N,WORK)
      DO 10 I = 1,N
      R(I) = R(I) - P(I)
      P(I) = R(I)
10    CONTINUE
C
      CALL INPROD(R,R,N,PR)
      GOTO 40
20    IT = IT + 1
      B = RP/PR
      PR = RP
      DO 30 I = 1,N
30    P(I) = R(I) + B*P(I)
C
40    CALL MATVEC(P,AP,N,WORK)
      CALL INPROD(P,AP,N,VV)
      A = PR/VV
       DO 50 I = 1,N
       X(I) = X(I) + A*P(I)
       R(I) = R(I) - A*AP(I)
 50    CONTINUE
      CALL INPROD(R,R,N,RP)
      CALL INPROD(X,X,N,XNORM)
C
      IF (RP .GT. TOL*(XNORM+1.) .AND.  IT .LT. N) GOTO 20
      RETURN
      END
```

```
C
      SUBROUTINE MATVEC(P,AP,N,WORK)
C
C-------------------------------------------------------------------
C
C     THIS SUBROUTINE COMPUTES THE MATRIX-VECTOR PRODUCT A*P ,
C     FOR ANY GIVEN VECTOR P
C
C-------------------------------------------------------------------
C
      DIMENSION P(N),AP(N)
      DIMENSION WORK(1)
      COMMON /WKSP/ MATRIX, NX, NY, NFYD, NFR,
     S NSC, NU, NG, NH, NEL, NBOUND, NWRK
      DO 10 I = 1,N
10    AP(I) = P(I)
      DO 20 L = 3,NEL,3
      AA = WORK(MATRIX + L-2)
      I = IFIX(WORK(MATRIX + L-1))
      J = IFIX(WORK(MATRIX + L))
      AP(I) = AP(I) + AA*P(J)
      AP(J) = AP(J) + AA*P(I)
20    CONTINUE
      RETURN


      END
C
C
      SUBROUTINE INPROD(X,Y,N,XY)
C
C-----------------------------------------------------------------------
C
C     THIS SUBROUTINE COMPUTES THE INNER PRODUCT OF TWO VECTORS
C
C-----------------------------------------------------------------------
C
      DIMENSION X(N) , Y(N)
      XY = 0.0
      DO 10 I = 1 , N
10    XY = XY + X(I)*Y(I)
      RETURN
      END
C
      SUBROUTINE FATAL(TEXT)
C
C-----------------------------------------------------------------------
C
C     THIS SUBROUTINE TERMINATES THE PROGRAM AFTER PRINTING A MESSAGE O
C     THE REASON
C
C-----------------------------------------------------------------------
C
      INTEGER TEXT(3)
      WRITE(6,1) TEXT
1     FORMAT(//,25H REASON OF STOPPING :         ,//,3A10,//)
      STOP
      END
```

```
      PROGRAM MCFOM(TAPE4,INPUT,TAPE5=INPUT,OUTPUT,TAPE6=OUTPUT)
      DIMENSION WORK(6905)
      CALL PLASMA(WORK,6905)
      STOP
      END


      FUNCTION F1(Y)
      F1 = 0.
      IF (Y.GT. 0.25 .AND. Y .LT. 0.3125) F1 = 1.0
      RETURN
      END


      SUBROUTINE OUT(X,Y,YDISC,V,FI,M,N,NDISC,AA,BETA,GAMMA,HA,X0)
      DIMENSION X(M),Y(N),YDISC(NDISC),V(M,N),FI(M,N)
C
C-----------------------------------------------------------------
C
C     THIS SUBROUTINE ENABLES THE USER TO MANIPULATE THE OUTPUT DATA AS
C     DESIRED
C
C-----------------------------------------------------------------
C
      WRITE(6,3)AA,BETA,GAMMA,HA,X0,YDISC
      TEXT1 = 10H V :
      TEXT2 = 10H FI :
      WRITE(6,1)TEXT1,Y
      DO 10 I = 1,M
10    WRITE(6,2) X(I) , (V(I,J),J=1,N)
      WRITE(6,1)TEXT2,Y
      DO 20 I = 1,M
20    WRITE(6,2) X(I) , (FI(I,J),J=1,N)
1     FORMAT(1H1,//,44X,A10,//,3X,8H Y       ,15F8.4,/,4H X    ,//)
2     FORMAT(F7.4,4X,15F8.2)
3     FORMAT(1H1,///,
     S 30H INTEGRAL VALUE                ,F8.4,//,
     S 30H HALL PARAMETER                ,F8.4,//,
     S 30H GAMMA                         ,F8.4,//,
     S 30H HARTMANN CONSTANT             ,F8.4,//,
     S 30H X0                            ,F8.4,//,
     S 30H JUMPING POINTS OF F1(Y)       ,12F8.4)
      RETURN
      END
```

```
   -0.0146
   13.3
    0.25
   23.0
    0.0625
15
    0.0
    0.02
    0.04
    0.0625
    0.08
    0.10
    0.15
    0.20
    0.25
    0.375
    0.500
    0.625
    0.750
    0.875
    1.000
15
    0.0
    0.10
    0.15
    0.20
    0.25
    0.28
    0.3125
    0.34
    0.42
    0.5
    0.6
    0.7
    0.8
    0.9
    1.000
 2
    0.25
    0.3125
```