

C. S. Davis

LA-5280

C.3

CIC-14 REPORT COLLECTION  
**REPRODUCTION  
COPY**

Polynomial Solutions of the  
Schrödinger Equation Applied to  
Photon Cross Sections in Atoms



**Los Alamos**  
**scientific laboratory**  
of the University of California  
LOS ALAMOS, NEW MEXICO 87544



This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Atomic Energy Commission, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

In the interest of prompt distribution, this LAMS report was not edited by the Technical Information staff.

Printed in the United States of America. Available from  
National Technical Information Service  
U. S. Department of Commerce  
5285 Port Royal Road  
Springfield, Virginia 22151  
Price: Printed Copy \$4.00; Microfiche \$0.95

LA-5280

UC-32

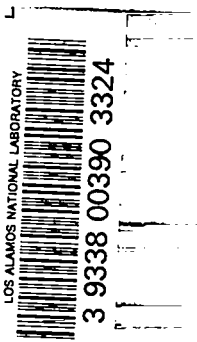
ISSUED: August 1973



# Polynomial Solutions of the Schrödinger Equation Applied to Photon Cross Sections in Atoms

by

A. L. Merts  
Walter Matuska, Jr.



**POLYNOMIAL SOLUTIONS OF THE SCHRÖDINGER EQUATION  
APPLIED TO PHOTON CROSS SECTIONS IN ATOMS**

by

A. L. Merts and Walter Matuska, Jr.

**ABSTRACT**

Solutions of the Schrödinger equation with a realistic potential are carried out in detail. To check our methods, we have calculated a few bound-bound, bound-free, and free-free cross sections and compared our values with existing calculations and experimental data. These comparisons, along with a listing of the computer code in its bound-free form, are included.

**I. INTRODUCTION**

This report intends to show how the CDC-7600 code DEGA-A (Dense Electron Gas Approximation-Absorption) computes the bound-free, free-free, and bound-bound absorption coefficients,  $\sigma_{bf}$ ,  $\sigma_{ff}$ , and  $\sigma_{bb}$ , respectively, as a function of photon energy,  $h\nu$ , from the given potential function  $V(r)$  and the electron occupancy of the atom. Assuming  $V(r)$  to be exact, the code will do its computation at any desired accuracy, within the machine limits, because the Schrödinger equation is solved using exact power-series expansions, not difference equations.

As the code is now written, any potential function can be used if it adheres to the three following requirements.

1.  $\lim_{r \rightarrow 0} V(r)r^2 = 0$ .

2. After some finite value of  $r$ ,  $R_1$ , the potential must be  $V(r) = -k/r$ .

3.  $V(r)$  must be a negative, monotonically, increasing function.

Although it is not necessary to the internal structure of the code, we would also like  $V(r)$  to be a function of temperature, density, and the atomic number of the atom.

Throughout this report,  $[0, R_1]$  is called Region 1 and  $[R_1, \infty)$  Region 2.

The following  $V(r)$  is the one most often used in the code.

Experience has shown that a parameterized potential works well and gives energy levels close to those that Herman and Skillman<sup>1</sup> calculated using their Hartree-Fock-Slater method for isolated neutral atoms.

The potential form for an isolated atom is

$$V(r) = Ze/r(1 + \alpha r)^2 \quad (1)$$

where  $\alpha = 0.6057Z^{1/3}$ . However, the form actually used in this report, which is also valid for a compressed atom, is

$$V(r) = -(Z^*e)/R_1 \left( \frac{R_1}{r} + \frac{r^2}{2R_1^2} - 3/2 \right) \quad (2)$$

for  $r_0 \leq r \leq R_1$ ,

where

$Z^*$  is the "effective number" of free electrons,

$R_1$  is the radius of the sphere representing the spherical atomic volume,

$r_0$  is the value of the radial distance for which Eq. (2) is equal to the value given in Eq. (3) below, and  $e$  is the electronic charge.

In the inner region, the potential is given by

$$V(r) = -Ze/r(1 + \alpha r)^2 - Z^*e/r \left( \frac{r^2}{2R_1^2} - B_0 \right) \quad (3)$$

for  $0 \leq r \leq r_0$ .

$$B_0 = \left(\frac{Z}{Z^*}\right) \left(\frac{R_1}{r_0}\right) / (1 + \alpha r_0)^2 - \frac{R_1}{r_0} + 3/2 \quad (4)$$

$$Z^* = Z / (1 + \alpha r_0)^2 \left[ \left(\frac{2\alpha r_0}{1 + \alpha r_0}\right) + 1 \right] \quad (5)$$

After the form of the potential in the two regions is chosen, Eqs. (4) and (5) follow from the potential's continuity and its relation to charge density through Poisson's equation. In the potential outlined above, the value of  $R_1$  is determined by the density of the material being considered. The value of  $Z^*$  is chosen by an iterative procedure so that at some finite temperature  $T$ , and for an atom occupying a spherical volume of radius  $R_1$ , we have

$$Z = N(Z^*) = \int_0^{R_1} \left[ \int_0^\infty n(r, T, P) dP \right] 4\pi r^2 dr \quad (6)$$

where  $n$  is the Fermi-Dirac distribution function, representing the number of electrons at point  $r$ , having momentum between  $P$  and  $P + dP$ . Using Eq. (5) we have also determined  $r_0$ .

We must correct the above potential for self-interaction. This is done in the simplest possible fashion. We replace  $Z$  in Eqs. (2) through (5) with  $(Z - 1)$  and add the term  $-1/r$  to Eqs. (2) and (3). The potential for  $r > R_1$  is defined as  $-1/r$ . This then represents our potential function used to calculate the one-electron energies and the one-electron orbitals from which we calculate the cross sections and  $f$ -values.

This potential can be shown to satisfy the three previously stated conditions.

The hydrogenic potential  $V(r) = -1/r$  for  $0 \leq r \leq \infty$  is often used to check parts of the code because analytic solutions for this potential are known.

The code considers the Schrödinger equation in the form

$$\varphi''(r) + \left[ \lambda - 2V(r) - \frac{\ell(\ell+1)}{r^2} \right] \varphi(r) = 0, \quad 0 \leq r \leq \infty \quad (7)$$

where

$r\varphi(r)$  is the radial wave function (however,  $\varphi(r)$  will be called the wave function throughout the rest of this report),

$\lambda$  is the energy eigenvalue, and

$\ell$  is the angular-momentum quantum number.

2

Equation (7) has an infinite (in some cases finite, but large) number of discrete bound solutions (negative  $\lambda$ ) commonly denoted by 1s, 2s, 2p, 3s, 3p, 3d, etc. Of this infinite sequence, we calculate only the solutions allowed by the electron occupancy.  $\sigma_{bb}$  can be evaluated only at discrete values of  $h\nu$ , because  $h\nu = |\lambda_1 - \lambda_2|$  where  $\lambda_1$  and  $\lambda_2$  both represent bound solutions. By using a line profile, we then distribute each  $\sigma_{bb}$  over a narrow range of  $h\nu$ . However, every positive  $\lambda$ , given  $\ell$ , is an eigenvalue for a free state. This allows us to choose any finite number of positive  $\lambda$ 's. Therefore, we can evaluate  $\sigma_{bf}$  at any energy  $h\nu$  above the threshold, and  $\sigma_{ff}$  at any desired value of  $h\nu$ . Here  $h\nu = |\lambda_1 - \lambda_2|$ ,  $\lambda_1$  represents either a bound or a free solution, and  $\lambda_2$  represents a free solution.

## II. THE METHOD OF SOLUTION OF THE SCHRÖDINGER EQUATION

### A. The Potential Approximation

To solve Eq. (7) on the computer, we choose a finite  $R > R_1$  to approximate  $r = \infty$ , and divide  $[0, R]$  into a finite number of intervals. For each of these intervals,  $r^2$  times the potential is approximated by a parabola to some specified degree of accuracy,  $\xi$ . This series of fits is started at  $r = 0$  with an interval of arbitrary length. In this first interval, we approximate the potential with

$$V(r) \approx c_1 + \frac{c_2}{r} + \frac{c_3}{r^2} \quad (8)$$

Let  $r_1$  be the left end point of this interval,  $r_3$  the right end point, and  $r_2$  the midpoint. To evaluate the  $c$ 's, we solve the set of equations

$$c_1 r_j^2 + c_2 r_j + c_3 = V(r_j) r_j^2 \quad \text{for } j = 1, 2, 3. \quad (9)$$

These equations have no difficulty at  $r = 0$ , and they fit the potential exactly at  $r_1$ ,  $r_2$ , and  $r_3$ . Because  $V(r)$  is a smooth, monotonic function, we can check our fit by checking the validity of

$$\left| c_1 + \frac{c_2}{r} + \frac{c_3}{r^2} - V(r) \right| < |\xi V(r)| \quad (10)$$

at several points between  $r_1$  and  $r_3$ . If this inequality is not satisfied for all points, we decrease the length of the interval until inequality [Eq. (10)] is satisfied for all points checked. This process is continued until Region 1 is complete. Here  $c_1$ ,  $c_2$ , and  $c_3$  should actually be

thought of as  $c_{1,i}$ ,  $c_{2,i}$ , and  $c_{3,i}$ , where the subscript  $i$  denotes the  $i$ th interval.

In Region 2, obviously,  $c_2 = -1$  and  $c_1 = c_3 = 0$ . The lengths of the intervals in this region are governed only by an additional condition discussed later. This condition applies to all intervals.

As the code now exists, for a given set of conditions, all solutions to the Schrödinger equation are found by using the same set of intervals over Region 1; however, each solution has its own set of intervals in Region 2. The code could also be written so that each solution would have its own set of intervals over the entire range of  $r$ . It would be impractical to require all solutions to have the same set of intervals in Region 2 because the maximum interval lengths allowed for the various solutions in Region 2 are so different.

## B. The Expansion of the Radial Wave Function

In the expansion of the wave function, we will need the condition  $c_{3,1} = 0$ . This is satisfied if the first requirement on the potential is met.

We assume the wave function to have the form

$$\varphi(r) = \sum_{j=1}^{\infty} a_{ji} r^{j-1} \text{ for } i = 1 \quad (11)$$

in the first interval. Dropping the subscript  $i$ , we substitute the power series and its second derivative for  $\varphi(r)$  and  $\varphi''(r)$  and the approximation for  $V(r)$  in Eq. (8) into the Schrödinger equation to get

$$\begin{aligned} & \left\{ -[2c_3 + \ell(\ell + 1)] a_1 \right\} / r^2 \\ & + \left\{ -2c_2 a_1 - [2c_3 + \ell(\ell + 1)] a_2 \right\} / r \\ & + \sum_{j=3}^{\infty} \left\{ (\lambda - 2c_1) a_{j-2} - 2c_2 a_{j-1} \right. \\ & \left. + [(j-2)(j-1) - 2c_3 - \ell(\ell + 1)] a_j \right\} r^{j-3} = 0 \quad (12) \end{aligned}$$

Here we assume that  $c_3 = 0$  and  $a_j = 0$  for  $1 \leq j \leq \ell + 1$ . Also  $a_{\ell+2}$  is arbitrary because any constant times a solution eigenfunction  $\varphi(r)$  is still a solution. As each term in Eq. (12) must be zero for every  $r$  in the interval, we now have the recursion relation

$$a_j = \frac{2c_2 a_{j-1} - (\lambda - 2c_1) a_{j-2}}{(j-2)(j-1) - \ell(\ell + 1)} \text{ for } j > \ell + 2 \quad (13)$$

The power-series expansions in all other intervals are also expanded about one of the interval's end points. If expanding about the left end point, we make the substitution  $d = r - \rho_i$  and require that  $0 \leq d \leq \rho_{i+1} - \rho_i$  where the  $\rho$ 's are the end points of the intervals. When it is necessary to expand about the right end point, the roles of  $\rho_i$  and  $\rho_{i+1}$  are switched and  $\rho_i - \rho_{i+1} \leq d \leq 0$ . Again omitting the subscript  $i$ , the Schrödinger equation can now be written as

$$(\rho^2 + 2\rho d + d^2)\varphi''(d) + (B_1 + B_2 d + B_3 d^2)\varphi(d) = 0 \quad ,$$

where

$$B_1 = b_1 \rho^2 + b_2 \rho + b_3$$

$$B_2 = 2b_1 \rho + b_2$$

$$B_3 = b_1$$

and

(14)

$$b_1 = \lambda - 2c_1$$

$$b_2 = -2c_2$$

$$b_3 = -[2c_3 + \ell(\ell + 1)] \quad .$$

Here we assume the wave function to have the form

$$\varphi(d) = \sum_{j=1}^{\infty} a_{ji} d^{j-1} \text{ for } i > 1 \quad (15)$$

Substituting the power series and its second derivative for  $\varphi(d)$  and  $\varphi''(d)$ , we get

$$\begin{aligned} & [2\rho^2 a_3 + B_1 a_1] + [6\rho^2 a_4 + 4\rho a_3 + B_1 a_2 + B_2 a_1] d \\ & + \sum_{j=5}^{\infty} \left[ (j-2)(j-1)\rho^2 a_j + (j-3)(j-2)2\rho a_{j-1} \right. \\ & \left. + (j-4)(j-3)a_{j-2} + B_1 a_{j-2} \right. \\ & \left. + B_2 a_{j-3} + B_3 a_{j-4} \right] d^{j-3} = 0 \quad (16) \end{aligned}$$

with the subscript  $i$  omitted. Again, because each term of Eq. (16) must be zero for all values of  $d$ , we have the recursion relations

$$a_3 = -B_1 a_1 / 2\rho^2$$

$$a_4 = -[B_2 a_1 + B_1 a_2 + 4\rho a_3] / 6\rho^2$$

$$a_j = -\left\{ B_3 a_{j-4} + B_2 a_{j-3} + [B_1 + (j-4)(j-3)] a_{j-2} + 2(j-3)(j-2)\rho a_{j-1} \right\} / (j-2)(j-1)\rho^2$$

$$\text{for } j > 4 \quad (17)$$

when  $a_1$  and  $a_2$  are known.

We can easily show that  $a_1 = \varphi(d)$  and  $a_2 = \varphi'(d)$  at  $d = 0$ . Therefore,  $a_1$  and  $a_2$  will always be known by the right-boundary condition or can be evaluated with the power-series expansion of the wave function in the previous interval.

In practice, we start with  $a_j = 0$  for  $1 \leq j \leq \ell + 1$  and  $a_{\ell+2}$  an arbitrary nonzero constant to start the power-series expansion in the first interval. Then we evaluate this power-series and its first derivative at its right end point. This determines  $a_1$  and  $a_2$  for the second interval. This process is repeated until we reach some  $\hat{r}$ ,  $0 < \hat{r} < R$ , where we evaluate  $\varphi_f(\hat{r})$  and  $\varphi'_f(\hat{r})$ ; the determination of  $\hat{r}$  is discussed later. At the right boundary,  $R$ ,  $\varphi(R)$ , and  $\varphi'(R)$  are given by the boundary condition. This enables us to start our successive power-series expansions at  $R$  and work our way backward to  $\hat{r}$  where we evaluate  $\varphi_b(\hat{r})$  and  $\varphi'_b(\hat{r})$ . We have a solution when  $\varphi'_f(\hat{r}) = \varphi'_b(\hat{r})$  for  $\varphi_f(\hat{r}) = \varphi_b(\hat{r})$ .  $\varphi_f(\hat{r})$  is simply the value of the wave function at  $\hat{r}$  found by forward expansions, and  $\varphi_b(\hat{r})$  is the value of the same wave function at  $\hat{r}$  found by backward expansions.

### C. Determination of the Expansion Interval Length

In real life, these power series can have only a finite number of terms. The maximum number of terms in the present code is 50. This leads to the other condition regulating the maximum length of a particular interval. Let us define

$$\tilde{V}(r) = \left[ \lambda - 2V(r) - \frac{\ell(\ell+1)}{r^2} \right] \quad (18)$$

Using this definition, we can write the Schrödinger equation as

$$\varphi''(r) \pm h^2 \varphi(r) = 0, \text{ where } h^2 = \pm \tilde{V}(r) \quad (19)$$

choosing the sign so that  $h^2 > 0$ . If  $h$  is assumed to be constant locally, the solution is either

$$\varphi(r) = A e^{-hr} + B e^{hr} \quad (20)$$

or

$$\varphi(r) = A \sin hr + B \cos hr, \quad (21)$$

depending upon the sign. Expanding the decreasing exponential solution in a Taylor series about an end point of the interval, we get

$$\varphi(r) = A e^{-h\rho_i} \sum_{j=0}^{\infty} (-1)^j \frac{(h\Delta r)^j}{j!} \quad (22)$$

where  $\Delta r = r - \rho_i$  is the maximum allowable interval length. With this series, we can estimate the largest value of  $(h\Delta r)$  that allows the series to converge to a specified accuracy in the allotted number of terms. In Region 1, all solutions are considered in estimating the maximum interval length; however, in Region 2 only one solution at a time is considered. The code then plugs the largest encountered value of  $h$  into

$$\Delta r = \frac{C}{h} \quad (23)$$

This  $C$  is an input parameter, and for a 50-term expansion and 8-place accuracy a conservative value for  $C$  is 6.

The sinusoidal and increasing exponential solutions have a series similar to Eq. (22), and the interval length is also estimated by Eq. (23).

### D. Finding the Join Point $\hat{r}$ for Forward and Backward Integration

Let us consider the question of stability. Here again, Eqs. (20) and (21) can be thought of as local solutions to the Schrödinger equation. If an error is introduced in Eq. (20) during forward integration (increasing  $r$ ) when the first term is the desired solution, this error will grow exponentially. However, any error introduced will diminish exponentially during backward integration. Likewise, when the second term is the desired solution, backward integration is unstable and forward integration causes any error to diminish exponentially. Equation (21) is considered stable for integration in either direction. Any error introduced into the integration will not grow exponentially; however, once an error is introduced it cannot be diminished as was the case with Eq. (20). Figure 1 shows the regions of stable and unstable integration for the four possible combinations of given conditions.

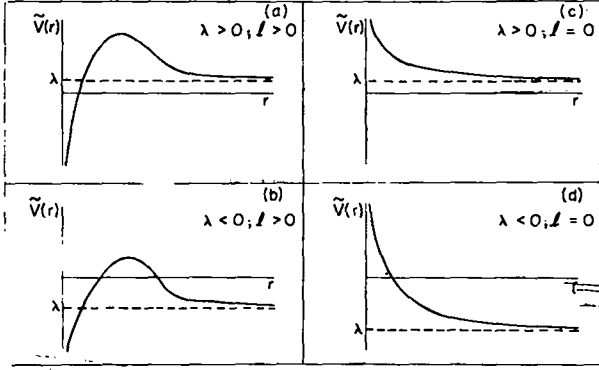


Fig. 1.

Integration is stable in both directions for  $\tilde{V}(r) > 0$ ; integration is stable in only one direction for  $\tilde{V}(r) < 0$ .  $\lambda$  indicates the asymptotic limit of  $\tilde{V}(r)$ .

For the cases shown in Figs. 1a and 1b,  $\hat{r}$  is chosen such that  $\tilde{V}(\hat{r})$  is the maximum value of  $\tilde{V}(r)$ . For the case shown in Fig. 1c,  $\hat{r}$  is the center of the rightmost interval, and for the case shown in Fig. 1d,  $\hat{r}$  is chosen such that  $\tilde{V}(\hat{r}) = 0$ . For positive  $\lambda$ , the value of  $\lambda$  is known and is used to find  $\hat{r}$ ; however, for negative  $\lambda$  one has only a maximum and minimum guess at its value at this stage, so the minimum guess is used in evaluating  $\hat{r}$ .

## E. Boundary Condition at R

1. **Bound States—Negative  $\lambda$ .** The two right-boundary conditions used in the code are discussed below; however, this method of solving the Schrödinger equation is not limited to these two conditions. For the bound state, the Schrödinger equation becomes

$$\varphi''(r) - |\lambda|\varphi(r) = 0 \text{ as } r \rightarrow \infty \quad (24)$$

Note that this is true only in Region 2. Because  $\varphi(R)$  is arbitrary and

$$\varphi(r) = Ae^{-\sqrt{|\lambda|}r} \quad (25)$$

is the desired solution to Eq. (24), we derive the boundary condition

$$\varphi'(R) = -\sqrt{|\lambda|} \varphi(R) \quad (26)$$

Although we have no exact relation between  $R$  and the accuracy desired, we regulate  $R$  by the following method. First, we find the largest root of  $\tilde{V}(r)$ .  $R$  is then taken as some constant (typically 25) times this root. Then, after

the Schrödinger equation is solved using this  $R$ , we check the validity of

$$|\varphi(r)|_{\max} < \gamma|\varphi(R)| \quad (27)$$

( $10^{-15}$  is a typical value for  $\gamma$ ). If this condition is not satisfied, we simply keep increasing  $R$  until it is. A quick method for getting a value near  $|\varphi(r)|_{\max}$  is to check  $\varphi(r)$  at the end points of the intervals.

2. **Free States—Positive  $\lambda$ .** The sinusoidal condition given by Carson, Mayers, and Stibbs<sup>2</sup> is used on the right boundary in the free case. This boundary condition also applies only in Region 2.

For large  $r$ , the solution may be written

$$\varphi(r) = M(r) \cos[kr + \delta(r)] \quad (28)$$

in which  $k = \sqrt{\lambda}$ ,

$$\lim_{r \rightarrow \infty} M(r) = M, \quad (29)$$

and

$$\lim_{r \rightarrow \infty} \frac{\delta(r)}{r} = 0$$

$M$  fixes the scale of the solution. We shall require the normalization  $M = \sqrt{2/\pi k}$ .

In Region 2 where  $V(r) = -c/r$  for  $c > 0$ , we assume a solution of the form

$$\varphi(r) = A(r) \cos\left(kr + \frac{c}{k} \ln r\right) + B(r) \sin\left(kr + \frac{c}{k} \ln r\right) \quad (30)$$

Now substitute Eq. (30) into the Schrödinger equation. Then asymptotic expansions for  $A(r)$  and  $B(r)$  exist in the form

$$A(r) = \sum a_n/r^n, \quad B(r) = \sum b_n/r^n, \quad (31)$$

where

$$a_{n+1} = \left\{ \left[ \ell(\ell+1) + \frac{c^2}{k^2} - n(n+1) \right] b_n - \frac{c}{k} (2n+1) a_n \right\} / 2k(n+1)$$

$$b_{n+1} = \left\{ \left[ n(n+1) - \frac{c^2}{k^2} - \ell(\ell+1) \right] a_n - \frac{c}{k} (2n+1) b_n \right\} / 2k(n+1) \quad (32)$$



and

$$a_0 = M, b_0 = 0 \quad (33)$$

The sums in Eq. (31) are only semiconvergent. Therefore  $A(r)$  and  $B(r)$  must be evaluated at an  $r$  large enough that the sums converge to the desired accuracy.

Equation (30) can also be written as

$$\varphi(r) = M(r) \cos \theta(r) \quad (34)$$

where

$$M(r) = \sqrt{A(r)^2 + B(r)^2} \quad (35)$$

and

$$\theta(r) = kr + \frac{c}{k} + \ln r + \tan^{-1} \left[ \frac{A(r)}{B(r)} \right]$$

The asymptotic series expansion for the solution of  $\varphi(r)$  given by Eq. (34) is uniquely determined up to some constant phase  $\alpha_q$ ; therefore, we can write

$$\varphi(r) = M(r) \cos [\theta(r) + \alpha_q] \quad (36)$$

instead of Eq. (34), and we also have

$$\begin{aligned} \varphi'(r) &= M'(r) \cos [\theta(r) + \alpha_q] \\ &\quad - M(r)\theta'(r) \sin [\theta(r) + \alpha_q] \end{aligned} \quad (37)$$

For sufficiently large  $R$ ,  $\varphi(R)$  and  $\varphi'(R)$  are the desired right-boundary conditions. The guesses at  $R$  may have to be increased several times before Eq. (31) is satisfied.

The phase  $\alpha_q$  is determined by an iteration process that is explained later.

### III. DETERMINATION OF THE EIGENVALUES AND PHASE FACTORS

#### A. The Eigenvalue

As stated previously, for the bound state,  $\lambda$  is varied until we find a  $[\varphi(\lambda, r), \lambda]$  that solves the Schrödinger equation. We start with a minimum and a maximum guess at  $\lambda$ . This difference in  $\lambda$  is divided into a specified number of logarithmically equal  $\lambda$ -intervals. We define

$$F(\lambda) = \frac{\varphi_b(\lambda, \hat{r})}{\varphi_f(\lambda, \hat{r})} \varphi_f'(\lambda, \hat{r}) - \varphi_b'(\lambda, \hat{r}) \quad (38)$$

at the end points of these  $\lambda$ -intervals and look for a sign change in  $F(\lambda)$  in each. If we detect a sign change in an interval, we use the Regula Falsa<sup>3</sup> method to find the root of  $F(\lambda)$  in that interval. As  $F(\lambda)$  also changes sign through poles, we check to reject these intervals.

When we find a  $[\varphi(\lambda, r), \lambda]$  solution, we must determine whether it is the desired one for specified quantum numbers  $n$  and  $\ell$ , where  $\ell \leq n - 1$ . We have the desired solution when

$$I = n - \ell - 1 \quad (39)$$

where  $I$  is the number of roots in  $\varphi(r)$ .  $I$  is found using a Sturm Sequence.<sup>3</sup> If Eq. (39) is not satisfied, we reject this solution and continue our search. The value of  $\lambda$  so obtained is called the eigenvalue.

#### B. Phase Determination

In the free state,  $\lambda$  is specified and the phase  $\alpha_q$  at the right boundary is varied over the range  $0 \leq \alpha_q \leq \pi$  until we find a  $[\varphi(\alpha_q, r), \alpha_q]$  that solves the Schrödinger equation. To find this solution, we again use the Regula Falsa method to find the root of  $F(\alpha_q)$  where

$$F(\alpha_q) = \frac{\varphi_b(\alpha_q, \hat{r})}{\varphi_f(\alpha_q, \hat{r})} \varphi_f'(\alpha_q, \hat{r}) - \varphi_b'(\alpha_q, \hat{r}) \quad (40)$$

Here, we need not check for poles or unwanted solutions as was necessary for the bound states.

#### C. Normalization

Even though the wave functions need not be normalized to be solutions to the Schrödinger equation, they must be normalized to produce correct absorption coefficients. We define  $\varphi(r)$  as normalized when, for bound states:

$$\int_0^{\infty} \varphi(r) \varphi(r) dr = 1 \quad (41)$$

and, for free states:

$$\int_0^{\infty} \varphi_\lambda \varphi_\lambda' dr = \delta(\lambda - \lambda') \quad (42)$$

The free states have this normality built into the right-boundary condition. However, for each bound state, we evaluate

$$\int_0^{\infty} \varphi(r)\varphi(r) dr = \beta^2 \quad , \quad (43)$$

and the normalized wave function is then  $\varphi(r)/\beta$ . Equation (43) is evaluated by polynomial multiplication and integration over each interval of  $r$ .

#### IV. EVALUATION OF MATRIX ELEMENTS

Now that we have found all of the necessary solutions to the Schrödinger equation, we compute the matrix elements,  $H_{mn}$ , used in evaluating the absorption coefficients. The matrix elements are found by

$$H_{mn} = 2 \int_0^{\infty} \varphi_m(r) \frac{\partial V(r)}{\partial r} \varphi_n(r) dr \quad , \quad (44)$$

where  $m$  and  $n$  refer to eigen solutions of the Schrödinger equation. If one wave function is bound and the other is free,  $H_{mn}$  is used in evaluating  $\sigma_{bf}$ . Likewise, if both are bound, the result is  $\sigma_{bb}$ , and if both are free, the result is  $\sigma_{ff}$ .

Equation (44) is evaluated by summing the integrals calculated in each expansion interval. Because  $\partial V(r)/\partial r$  decreases as  $r^{-2}$  and  $\varphi(r)$  decreases exponentially for bound states, this integral converges rapidly to a specified accuracy at some finite value of  $r$ , provided at least one  $\varphi(r)$  represents a bound state. However, if both wave functions represent free states, the convergence is much slower, because each bound wave function asymptotically approaches a constant amplitude. Here, convergence is achieved only through the  $r^{-2}$  decrease in  $\partial V(r)/\partial r$ .

In the first interval we have

$$\frac{\partial V(r)}{\partial r} = -c_{2,1} r^{-2} \quad , \quad (45)$$

multiplied by the polynomial representations of  $\varphi_m(r)$  and  $\varphi_n(r)$  that can be integrated easily. In the rest of the intervals, we have

$$\frac{\partial V(d)}{\partial d} = -\frac{c_{2,i}(\rho_i + d) + 2c_{3,i}}{(\rho_i + d)^3} \quad , \quad (46)$$

which, by continued long division, can be written as a converging power series in  $d$  when  $|d/\rho_i| < 1$ . Now, multiplying these three polynomials, we get a polynomial that can be integrated easily.

#### V. EXAMPLES

##### A. Bound-Free Absorptions

In the code,  $\lambda$ ,  $r$ ,  $V(r)$ , and  $H_{mn}$  are dimensionless parameters. Both  $\lambda$  and  $V(r)$ , when multiplied by one Rydberg, are energies expressed in Rydbergs, and  $r$ , when multiplied by one Bohr radius, is a length expressed in Bohr radii.

The bound-free absorption coefficient is given by

$$\sigma_{bf}(h\nu) = 10.756 \times 10^6 \sum \frac{\ell_{\max} H_{mn}^2 \eta}{|\lambda_m - \lambda_n|^3 g} \quad , \quad (47)$$

where

$\ell_{\max}$  is the maximum of  $\ell_m$  and  $\ell_n$ .

$\eta$  is the number of electrons in the bound state that can make this transition.

$g$  is the maximum possible degeneracy given by  $2(2\ell + 1)$  for the  $\ell$  of the bound state.

Here  $\lambda_m$ ,  $\lambda_n$ , and  $H_{mn}$  are dimensionless numbers and  $\sigma_{bf}$  is expressed in barns per atom. Also when  $h\nu = 13.605 |\lambda_m - \lambda_n|$ ,  $h\nu$  is expressed in electron volts. The sum in Eq. (47) is over all possible bound-free transitions at the given  $h\nu$ .

Figures 2 through 7 show examples of bound-free absorptions for cold, normal-density beryllium, carbon, aluminum, iron, copper, and lead as computed by DEGA-A. In these figures, the continuous line was computed by DEGA-A and the X's are experimental data given by Storm and Israel.<sup>4</sup> In Fig. 5, the three experimental points at the m edge for iron were given by Carter and Givens.<sup>5</sup> In Fig. 4, DEGA-A was compared with calculations by Barfield, Koontz, and Huebner<sup>6</sup> for aluminum at low photon energies.

Even though the bound-free cross sections have been computed down to the lowest edge in these examples, we make no claims about the accuracy of the copper and lead cross sections at these low photon energies.

None of the cross sections in this report include electron spin or relativistic effect.

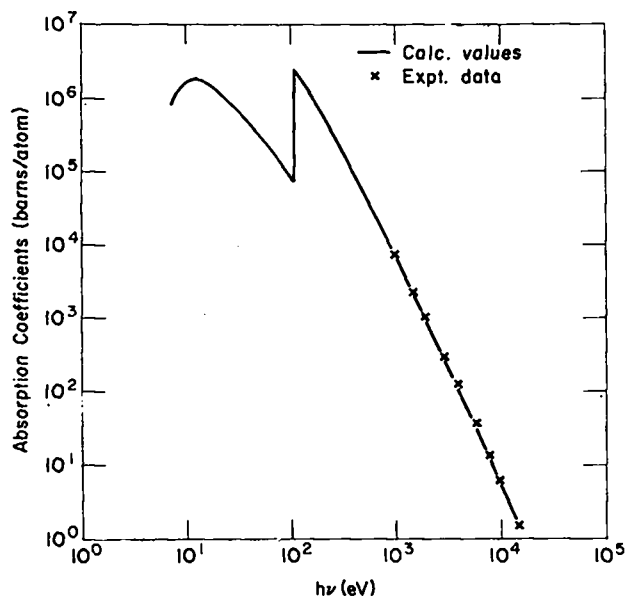


Fig. 2.

Bound-free absorption coefficients for cold, normal density ( $1.845 \text{ g/cm}^3$ ) Be. ( $Z = 4$ ,  $r_0 = 1.406$ ,  $R_1 = 2.355$ )

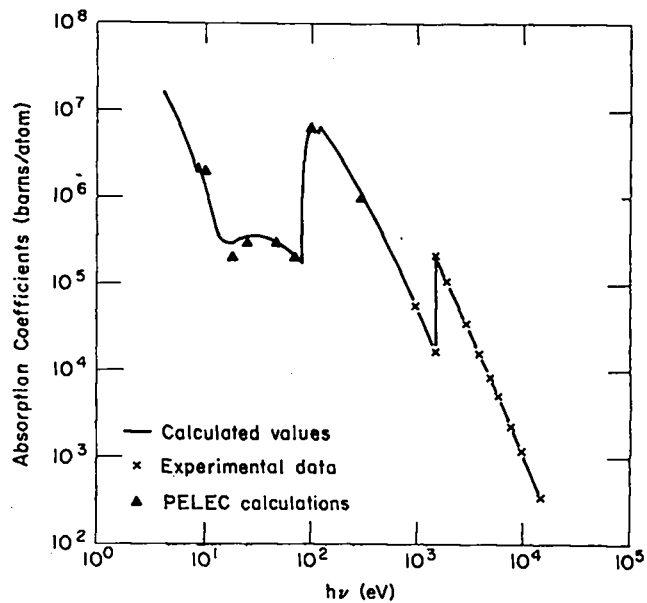


Fig. 4.

Bound-free absorption coefficients for cold, normal density ( $2.699 \text{ g/cm}^3$ ) Al. ( $Z = 13$ ,  $r_0 = 1.699$ ,  $R_1 = 2.990$ )

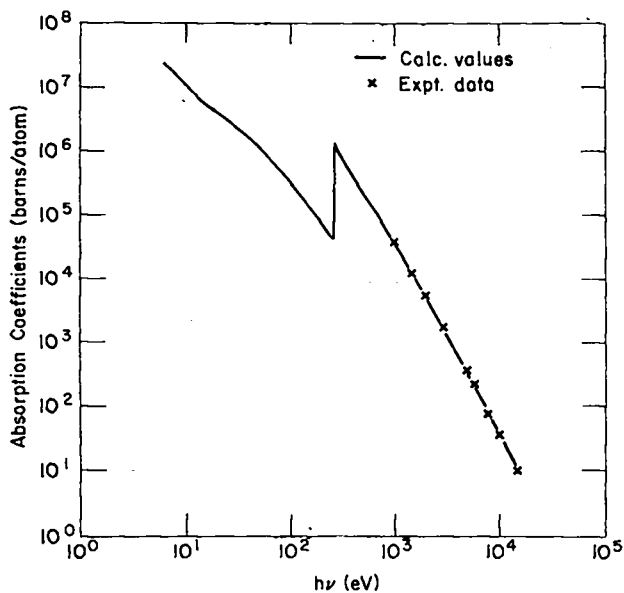


Fig. 3.

Bound-free absorption coefficients for cold, normal density ( $2.25 \text{ g/cm}^3$ ) C. ( $Z = 6$ ,  $r_0 = 1.442$ ,  $R_1 = 2.426$ )

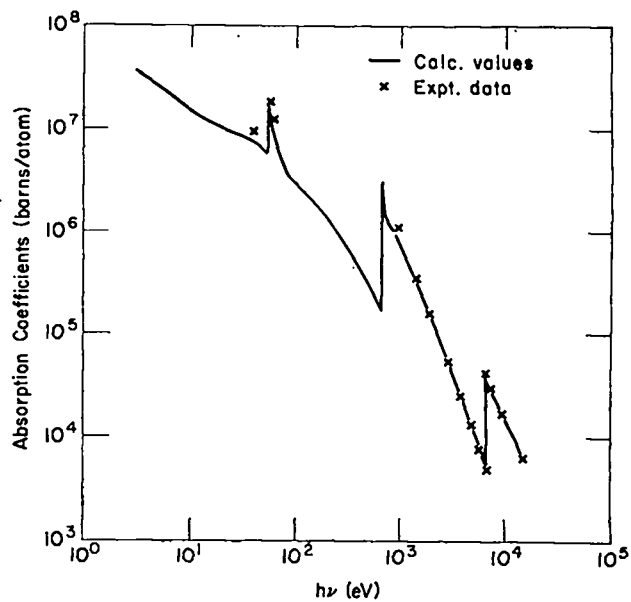


Fig. 5.

Bound-free absorption coefficients for cold, normal density ( $7.85 \text{ g/cm}^3$ ) Fe. ( $Z = 26$ ,  $r_0 = 1.512$ ,  $R_1 = 2.670$ )

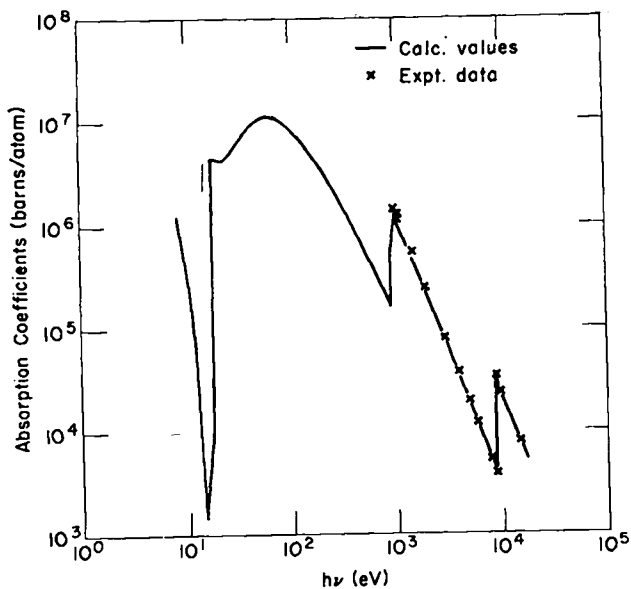


Fig. 6.

Bound-free absorption coefficients for cold, normal density ( $8.89 \text{ g/cm}^3$ ) Cu. ( $Z = 29$ ,  $r_0 = 1.517$ ,  $R_1 = 2.674$ )

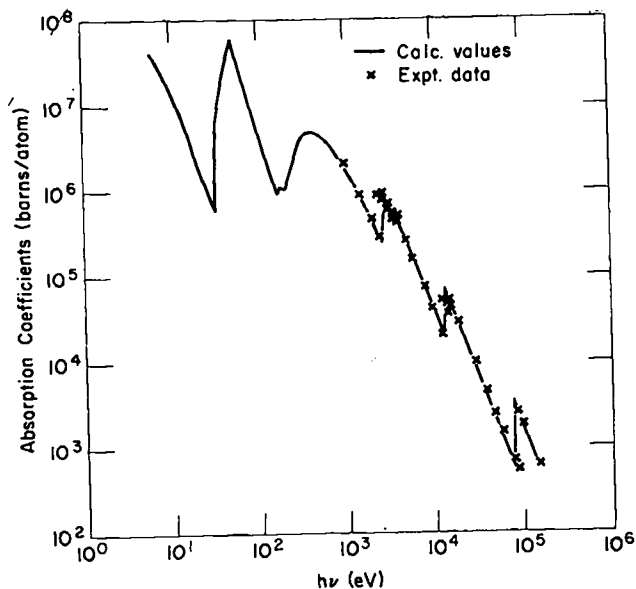


Fig. 7.

Bound-free absorption coefficients for cold, normal density ( $11.34 \text{ g/cm}^3$ ) Pb. ( $Z = 82$ ,  $r_0 = 2.003$ ,  $R_1 = 3.656$ )

## B. Bound-Bound Absorptions

In checking the bound-bound case, we consider hydrogen, lithium, beryllium, sodium, potassium, and strontium. A formula similar to Eq. (47) can be derived for  $\sigma_{bb}(h\nu)$ .<sup>2</sup> However, because the data found in the literature<sup>7</sup> appeared in a different form, we used the formula

$$\tau = \frac{4H_{mn}^2}{|\lambda_m - \lambda_n|^4 (4\lambda_{\max}^2 - 1)} \quad (48)$$

to make our numbers directly comparable. See Table I.

Squares of dipole moments for hydrogen were computed by

$$M^2 = \frac{4H_{mn}^2}{|\lambda_m - \lambda_n|^4} \quad (49)$$

These results checked with the values of the squares of dipole moments\* given by Bethe and Salpeter.<sup>8</sup> Note that Eq. (49) is not defined by the method presented in this report when  $\lambda_m = \lambda_n$ .

To check the code to more digits than given by Bethe and Salpeter (the CDC 7600 is a 14-digit machine),  $H_{1s,2p}$  was computed to the maximum possible accuracy using the hydrogenic potential and was compared with its analytic solution. We know that for hydrogen

$$\varphi_{1s} = 2re^{-r} \quad (50)$$

and

$$\varphi_{2p} = \frac{1}{2\sqrt{6}} r^2 e^{-r/2}$$

When these analytic expressions for  $\varphi$  are used in Eq. (44) along with  $V(r) = -1/r$ , we obtain

$$H_{1s,2p} = \frac{8}{9\sqrt{6}} = 0.36288736930121$$

and DEGA-A computed

$$H_{1s,2p} = 0.36288736930118$$

\*Compare with Table 13, page 264, Bethe and Salpeter.<sup>8</sup>

TABLE I  
TRANSITION INTEGRAL  $\tau$   
FOR COLD, NORMAL DENSITY ELEMENTS

Element	Transition	Transition Integral $\tau$			
		Screened-Hydrogenic	Coulomb Approx	Self-Cons Field	DEGA-A
Li I <sup>a</sup>	2p-2s	5.96	5.42	5.5-5.6	4.26
	3p-2s	0.011	0.016	0.011-0.020	0.071
	3s-2p	1.72	2.39		1.75
	4s-2p	0.105	0.177		0.148
	5s-2p	0.029	0.056		0.044
	6s-2p	0.013	0.025		0.020
	3d-2p	1.28	1.14		0.748
Be I <sup>b</sup>	4d-2p	0.19	0.18		0.148
	2s2p-2s <sup>2</sup>	2.27	2.03	1.86	3.29
Na I <sup>c</sup>	3p-3s	3.41	6.0	6.7	3.63
	4p-3s	0.211	0.047	0.051	0.026
	4s-3p	2.06	6.09	6.2	3.60
K I <sup>d</sup>	4p-4s	4.61	8.05	9.05	7.77
Sr I <sup>e</sup>	5d-5p	0.483	0.42		1.39

<sup>a</sup>Z=3,  $r_0=1.928$ , and  $R_1=3.263$ .

<sup>b</sup>Z=4,  $r_0=1.406$ , and  $R_1=2.355$ .

<sup>c</sup>Z=11,  $r_0=2.264$ , and  $R_1=3.986$ .

<sup>d</sup>Z=19,  $r_0=2.714$ , and  $R_1=4.950$ .

<sup>e</sup>Z=38,  $r_0=2.475$ , and  $R_1=4.518$ .

### C. Free-Free Absorptions

1. **Discussion of Gaunt Factors.** A formula similar to Eq. (47)<sup>2</sup> can be derived for  $\sigma_{ff}(h\nu)$  in terms of Gaunt factors  $g_{ff}$ ; however, here we only compute Gaunt factors for hydrogen and sodium and compare some of these results with Karzas and Latter.<sup>9</sup> Gaunt factors can be evaluated with

$$g_{ff}(\lambda_a, h\nu) = \frac{\pi\sqrt{3}}{8(Z^{eff})^2}$$

$$\sum_{\ell=0}^{L'} \left[ (\ell+1) H_{\ell+1, \ell}(\lambda_a, h\nu)^2 + \ell H_{\ell-1, \ell}(\lambda_a, h\nu)^2 \right] \quad (51)$$

when the sum converges rapidly enough to be practical. Here  $\lambda_a$  is the initial energy of the electron,  $h\nu$  is the photon energy,  $Z^{eff}$  is the effective number of free elec-

trons in the atom under consideration, and  $L'$  is a finite integer approximation to infinity. The matrix elements are still defined by Eq. (44), but we rewrite the equation as

$$H_{mn}(\lambda_a, h\nu) = 2 \int_0^{\infty} \varphi_m(\lambda_a, r) \frac{\partial V(r)}{\partial r} \varphi_n(\lambda_a + h\nu, r) dr \quad (52)$$

to define the association between energy levels and quantum numbers.

The sum in Eq. (51) is not always converging rapidly. Results for other than a coulomb potential can often be obtained in these cases by making use of the formula<sup>10</sup>

$$g_{ff}(\lambda_a, h\nu) = g_{ff}^c(\lambda_a, h\nu) + \frac{\pi\sqrt{3}}{8} \sum_{\ell=0}^L \left\{ (\ell+1) \left[ H_{\ell+1, \ell}(\lambda_a, h\nu)^2 - H_{\ell+1, \ell}^c(\lambda_a, h\nu)^2 \right] + \ell \left[ H_{\ell-1, \ell}(\lambda_a, h\nu)^2 - H_{\ell-1, \ell}^c(\lambda_a, h\nu)^2 \right] \right\}, \quad (53)$$

where  $g_{ff}^c$  is the coulombic Gaunt factor for the initial energy and photon energy under consideration,  $H_{mn}^c$  is a coulombic matrix element defined by Eq. (52) using  $V(r) = -1/r$ , and  $L$  is an integer sufficiently large so that the sum has converged to a predetermined accuracy. A graph of  $g_{ff}^c(\lambda_a, h\nu)$  is given by Karzas and Latter who circumvented the slow convergence problem in Eq. (51) by using hypergeometric functions. Applications of this procedure are limited to the coulomb potential.

The method for obtaining a noncoulombic  $\sigma_{ff}(h\nu)$ , as described in this report, never requires the evaluation of a  $Z^{eff}(\lambda_a, h\nu)$ . In terms of coulombic Gaunt factors,  $g_{ff}^c(\lambda_a, h\nu)$ ,  $Z^{eff}(\lambda_a, h\nu)$  must also be evaluated and then  $Z^{eff}(\lambda_a, h\nu) g_{ff}^c(\lambda_a, h\nu)$  must be used as the desired Gaunt factor. However, Eqs. (51) and (53) give the desired noncoulombic  $g_{ff}(\lambda_a, h\nu)$  for  $Z^{eff} = 1$  when the matrix elements are computed by DEGA-A using a realistic potential.

**2. Checks on Method.** For the first check on the code, we computed several Gaunt factors with Eq. (51) using the coulomb potential,  $V(r) = -1/r$ , and compared these results with the Karzas and Latter graph. Here comparisons were made up to, at most, three significant figures, which is the maximum accuracy for reading the graph. Table II shows that for several cases we were able to reproduce the Karzas and Latter values. In the remaining cases, Eq. (51) had not converged for  $\ell = 34$ , the maximum value of  $\ell$  used in these calculations.

Cold, normal-density sodium ( $Z = 11$ ,  $r_0 = 2.264$ ,  $R_1 = 3.986$ ) was arbitrarily used in the next check (Fig. 8). Here, for low photon energies and low initial energies, the Gaunt factors using the sodium potential and  $Z^{eff} = 1$  should approach the coulombic Gaunt factors. Also, for large photon energies and large initial energies, the Gaunt factors using the sodium potential and  $Z^{eff} = 11$  should approach the coulombic Gaunt factors. In both cases, the sodium Gaunt factor for  $(\lambda_a, h\nu)$  is compared with the coulombic Gaunt factor for  $(\lambda_a/(Z^{eff})^2, h\nu/(Z^{eff})^2)$ .

**3. Discussion of Screening.** The general screening effects are noted for the small electron kinetic energy and the small photon energy shown in Fig. 8. As the photon energy is held fixed and the electron energy is increased, the calculated Gaunt factor increases faster than would be expected from the coulombic case. The qualitative reason for this follows. As the electron energy is increased, the electron wave function samples in greater and greater detail the structure of the atom (the shielding), and as a result the effective charge  $Z^{eff}$  increases with increasing energy. The result is that, relative to a coulomb potential, the cross section is raised. However, as the energy of the electron and the photon increase, we eventually arrive at a

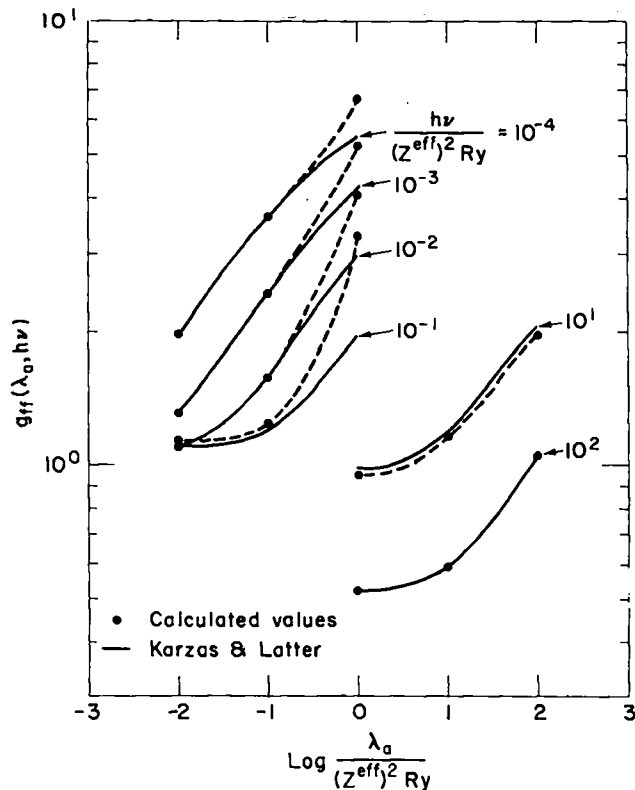


Fig. 8.

A comparison of coulombic and shielded potential (NaI) Gaunt factors.

point where the wave function of the electron oscillates sufficiently rapidly that it is essentially seeing the coulomb field of the nucleus. The effective charge is then the nuclear charge and we have agreement with the coulomb field results. This corresponds to the lower right-hand curves of Fig. 8.

**4. Numerical Results.** The purpose of Table II is to illustrate a trend in the convergence rate for each of Eqs. (51) and (53) as a function of  $(\lambda_a, h\nu)$ . An integer in parentheses indicates the value of  $\ell$  for which the series was terminated. For example, where  $(\lambda_a, h\nu) = (1.0, 0.01)$ , Eq. (51) has summed to 3.07 by 34 terms, whereas Eq. (53) has summed to within 2% of the expected (converged) value of 4.10 by three terms. Here convergence means no change in the sum to the accuracy given in Table II. Also, for the coulombic case at this energy pair, Eq. (51) has not converged for  $\ell = 34$ . For the (1.0, 10) entry, both Eqs. (51) and (53) have converged by  $\ell = 3$  for the sodium case, and Eq. (51) has converged by  $\ell = 3$  in the coulombic case.

Table II also shows that neither Eq. (51) nor (53) has converged for the entries labeled with footnote b. The use

TABLE II  
CALCULATIONS OF COLD, NORMAL DENSITY SODIUM  
AND COULOMBIC GAUNT FACTORS<sup>a</sup>

$\lambda_n$ (Ry)	$h\nu$ (Ry)	$g_{ff}^c(\lambda_n, h\nu)$ from Karzas and Latter	$g_{ff}^c(\lambda_n, h\nu)$ Calculated Using Eq. (51)	$g_{ff}(\lambda_n, h\nu)$ for Na Calculated Using Eq. (51)	$g_{ff}(\lambda_n, h\nu)$ for Na Calculated Using Eq. (53)		
0.01	0.0001	1.97	0.83(34)	0.83(34)	1.97(3)	1.97(34)	
	0.001	1.32	1.20(34)	1.20(34)	1.32(3)	1.32(34)	
	0.01	1.11	1.11(29)	1.09(29)	1.09(3)	1.09(29)	
	0.1	1.08	1.08(14)	1.11(14)	1.11(3)	1.11(14)	
	1.0	1.10	1.10(3)	1.51(3)	1.51(3)	1.51(9)	
0.1	0.0001	3.65	1.34(34)	1.34(34)	3.64(3)	3.65(34)	
	0.001	2.46	1.41(34)	1.40(34)	2.45(3)	2.45(34)	
	0.01	1.58	1.53(34)	1.53(34)	1.58(3)	1.58(34)	
	0.1	1.21	1.21(24)	1.24(24)	1.24(3)	1.24(24)	
	1.0	1.13	1.13(3)	1.99(3)	1.99(3)	1.99(11)	
1.0	0.0001	5.5	1.92(34)	3.01(34)	6.50(3)	6.59(25)	6.59(34)
	0.001	4.2	1.92(34)	3.01(34)	5.20(3)	5.29(25)	5.29(34)
	0.01	3.0	1.97(34)	3.07(34)	4.02(3)	4.10(25)	4.10(34)
	0.1	1.95	1.89(34)	3.23(34)	3.22(3)	3.29(25)	3.29(34)
	1.0	1.31	1.31(15)	5.50(15)	5.44(3)	5.50(15)	5.50(22)
	10.	0.97	0.97(3)	37.4(3)	37.4(3)	37.4(7)	
10.0	0.01	4.55	2.23(34)	27.5(34)	26.7(3)	29.8(25)	29.8(34)
	0.1	3.25	2.19(34)	27.6(34)	25.5(3)	28.6(25)	28.7(34)
	1.0	2.09	2.06(34)	29.6(34)	26.6(3)	29.6(25)	29.6(34)
	10.	1.20	1.20(10)	50.6(10)	49.4(3)	50.6(10)	50.6(16)
	100.	0.59	0.59(3)	106.7(3)	106.7(3)	106.7(7)	
100.0	0.1	4.55	2.28(34)	102.2(34)	82.6(3)	104.3(25)	104.5(34)
	1.	3.30	2.31(34)	102.4(34)	81.7(3)	103.2(25)	103.4(34)
	10.	2.08	2.05(34)	106.1(34)	85.8(3)	106.0(25)	106.1(34)
	100.	1.06	1.06(10)	125.4(10)	121.0(3)	125.4(10)	125.4(15)
	1000.	0.42	0.42(3)	117.7(3)	117.7(3)	117.7(7)	
1000.0	10.0 <sup>b</sup>	3.30	2.31(34)	187.3(34)	122.4(3)	185.7(25)	188.3(34)
	100. <sup>b</sup>	2.07	1.89(17)	187.7(34)	129.0(3)	172.8(10)	183.1(17)
	1000.	0.99	0.99(10)	141.3(10)	135.2(3)	141.3(10)	141.3(15)
	10,000.	0.36	0.36(3)	74.1(3)	74.1(3)	74.1(7)	
10,000.0	100.0 <sup>b</sup>	3.30	2.31(34)	244.8(34)	132.1(3)	235.4(25)	245.8(34)
	1000. <sup>b</sup>	2.06	2.03(34)	232.2(34)	139.5(3)	228.1(25)	232.2(34)
	10,000.	0.98	0.98(10)	128.1(10)	122.4(3)	128.1(10)	128.1(15)
	100,000.	0.35	0.35(3)	51.8(3)	51.8(3)	51.8(7)	

<sup>a</sup>All values in this table, with the exception of the Karzas and Latter entries, were computed with the DEGA-A code. An integer in parentheses is the value of  $l$  for which either Eq. (51) or (53) was terminated to obtain the Gaunt factor. These integers illustrate the speed of convergence of Eqs. (51) and (53).

<sup>b</sup>Neither Eq. (51) nor (53) has converged.

of a geometric sum enables the extrapolation of Eq. (51) to results that should represent a lower bound to the infinite sum. Numerical checks indicate that this extrapolation is good to 10% or better. To derive this extrapolation formula, let us simplify Eq. (51) by setting

$$Y_{\ell}^1 = (\ell + 1) H_{\ell+1, \ell}(\lambda_a, h\nu)^2$$

and (54)

$$Y_{\ell}^2 = \ell H_{\ell-1, \ell}(\lambda_a, h\nu)^2 .$$

From calculations, we note that both  $(\log Y_{\ell}^1, \ell)$  and  $(\log Y_{\ell}^2, \ell)$  approximate straight lines for large values of  $\ell$ . (The  $(\log Y_{\ell}, \ell)$  pair is not to be confused with the  $(\lambda_a, h\nu)$  pair.) Figure 9 illustrates this point for  $(1000, 100)$ . The straight line for  $Y_{\ell}^1$  is written as

$$\log Y_{\ell}^1 = s(\ell - \ell') + q , \quad (55)$$

and two known points,  $(\log Y_{\ell'}^1, \ell')$  and  $(\log Y_{\ell'+1}^1, \ell'+1)$ , are chosen to evaluate  $s$  and  $q$ . Equation (55) can also be written as

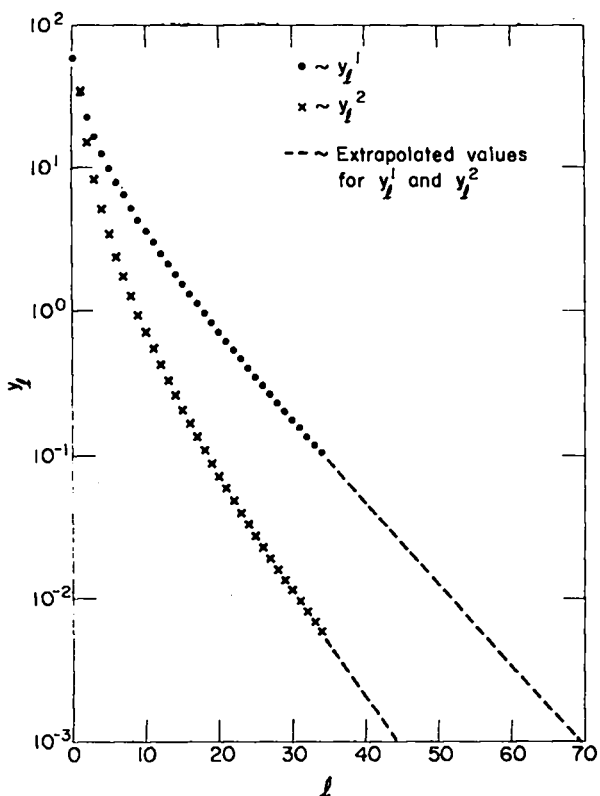


Fig. 9.

A graphic representation of the terms in Eq. (51) and the extrapolation indicated in Eq. (56).

$$Y_{\ell}^1 = e^q (e^s)^{\ell - \ell'} . \quad (56)$$

Now using Eq. (56) to take the infinite sum for all  $\ell \geq \ell'$ , we have the extrapolation formula

$$\sum_{\ell=\ell'}^{\infty} Y_{\ell}^1 = e^q / (1 - e^s) . \quad (57)$$

The above procedure is repeated using  $Y_{\ell}^2$ . These two extrapolated values are added to the number obtained by summing Eq. (51) to  $\ell = \ell' - 1$ . The results obtained by using this extrapolation formula are given in Table III.

TABLE III

EXTRAPOLATED VALUES OF GAUNT FACTORS

$\lambda_a$	$h\nu$	$G_{ff}(\lambda_a, h\nu)$ for Na Calculated Using Eq. (51); $\ell$ in Parentheses	$G_{ff}(\lambda_a, h\nu)$ for Na Calculated Using Extrapolated Form of Eq. (51); $\ell'$ in Parentheses	$G_{ff}(\lambda_a, h\nu)$ for Na Calculated Using Extrapolated Form of Eq. (51); $\ell'$ in Parentheses
1000.0	10.0	187.3 (34)	189.5 (27)	189.9 (34)
	100.	187.7 (34)	188.3 (27)	188.3 (34)
10,000.0	100.0	244.8 (34)	259.0 (27)	261.7 (34)
	1000.	232.2 (34)	234.2 (27)	234.3 (34)

#### REFERENCES

1. F. Herman and S. Skillman, *Atomic Structure Calculations* (Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1963).
2. T. R. Carson, D. F. Mayers, and D. W. N. Stibbs, "The Calculation of Stellar Radiative Opacity," *Mon. Not. R. Astr. Soc.* 140, 483 (1968).
3. A. S. Householder, *Principles of Numerical Analysis* (McGraw-Hill Book Company Inc., New York, 1953).
4. E. Storm and H. I. Israel, "Photo Cross Sections from 0.001 to 100 MeV for Elements 1 through 100," Los Alamos Scientific Laboratory report LA-3753 (November 15, 1967).
5. D. E. Carter and M. P. Givens "Soft X-Ray Absorption of Thin Films of Iron and Iron Oxide," *Phys. Rev.* 101, 1469 (1956).
6. W. D. Barfield, G. D. Koontz, and W. F. Huebner, "Fits to New Calculations of Photoionization Cross Sections for Low Z-Elements," *J. Quant. Spectrosc. Radiat. Transfer* 12, 1409 (1972).



7. A. M. Naqvi, "Calculations and Applications of Screened Hydrogenic Wave Functions," *J. Quant. Spectrosc. Radiat. Transfer* 4, 597 (1964).
8. H. A. Bethe and E. E. Salpeter, *Quantum Mechanics of One- and Two-Electron Atoms* (Academic Press Inc., New York, 1957).
9. W. J. Karzas and R. Latter, "Electron Radiative Transitions in a Coulomb Field," *Apl. J., Suppl.* 6, 167 (1961). Also see "Free-Free Gaunt Factors," The Rand Corporation, Santa Monica, California, report RM-2010-AEC (November 8, 1957).
10. R. R. Johnston, "Free-Free Radiative Transitions—A Survey of Theoretical Results," *J. Quant. Spectrosc. Radiat. Transfer* 7, 815 (1967).

**APPENDIX  
DEGA-A**

DEGA-A in its present form is a research and not a production code. The version, whose listing follows, only calculates bound-free absorptions. The code lists these absorption coefficients and makes plots as seen in the previous section. With slight modifications in the coding, bound-bound and free-free absorptions can be calculated.

There are several free parameters that regulate the accuracy of the code. Most of these parameters are set at the beginning of the code; however, a few are found throughout the code. The values in the listing will give at least four-place accuracy. The user should feel free to vary these parameters as he pleases and at his own risk.

Data are read into the bound-free version of DEGA-A with the following FORTRAN statements.

```

2 FORMAT (8F10.3)
7 FORMAT (16I5)
133 FORMAT (F15.5,2I5)
  READ 2, Z, RR2, RR
  IF (Z.LT.0.0) end job
  READ 7, ISSMAX
  READ 7, (NSUBSHL (ISS), ISS = 1, ISSMAX)
  READ 7, KWKB
  DO 132 K = 1, KWKB
132 READ 133, XLAMWKB(K), LWKB(K), NWKB(K)
  READ 7, IHNUMAX
  READ 2, (HNUVEC(IHNU), ACOFVEC(IHNU),
    IHNU = 1, IHNUMAX)

```

**Z** is the atomic number of the element under consideration.

**RR2** is the same as  $r_0$  on page 1.

**RR** is the same as  $R_1$  on page 1. Units for RR2 and RR are number of Bohr Radii.

**ISSMAX** is the number of subshells under consideration.

**NSUBSHL(ISS)** is an array that contains the number of electrons in each subshell. The entries are read into this array in the order 1s, 2s, 2p, 3s, 3p, 3d, etc., up to the last occupied subshell. If any previous subshell is vacant, it must be assigned the value zero.

**KWKB** is the number of bound wave functions to be computed. The necessary free wave functions are generated internally by the code.

**XLAMWKB(K)** is an array that contains guesses at the eigenvalues of the bound states. Units are number of Rydbergs.

**LWKB(K)** is an array that contains the quantum numbers  $\ell$ .  $\ell = 0$  for s - states,  $\ell = 1$  for p - states,  $\ell = 2$  for d - states, etc.

**NWKB(K)** is an array that contains the quantum numbers  $n$ .  $n = 1$  for the 1s state,  $n = 2$  for the 2s and 2p states,  $n = 3$  for the 3s, 3p, and 3d states, etc.

For each K, XLAMWKB(K), LWKB(K), and NWKB(K) should be consistent with Eq. (39). These guesses at bound eigenvalues can be read in any order. NSUBSHL(ISS) should be defined for each of these guesses at a bound eigenvalue.

**IHNUMAX** is the number of  $(h\nu, \sigma_{bf}(h\nu))$  pairs from a separate source that one wants to compare with the results of DEGA-A. This option is illustrated by the X's in Figs. 2 through 7. If IHNUMAX equals zero, no  $(h\nu, \sigma_{bf}(h\nu))$  pairs will be read.

**HNUVEC(IHNU)** is the array that contains the  $h\nu$ 's given in electron-volts.

**ACOFVEC(IHNU)** is the array that contains the  $\sigma_{bf}(h\nu)$ 's given in barns/atom.

Data decks may be stacked one behind the other. The job terminates normally when it encounters a negative Z.

DEGA-A  
(LP-129)

PROGRAM DEGAA(INP, OUT, FILM)	DEGA	00002
DIMENSION XLAMARY(100), LARY(101), ALFAARY(100), IMAXARY(100)	DEGA	00003
1 NSUBSHL(28), NSHELL(100)	DEGA	00004
COMMON/SCRATCH/SCRATCH(1204)	DEGA	00005
DIMENSION AM1S(401), AM2S(401), JMAXS(401)	DEGA	00006
EQUIVALENCE (SCRATCH(1), AM1S(1)), (SCRATCH(402), AM2S(1)),	DEGA	00007
1 (SCRATCH(803), JMAXS(1))	DEGA	00008
DIMENSION AROOT(2)	DEGA	00009
DIMENSION PDU4(2)	DEGA	00010
COMMON/CB3/ Z, ZM1, RR2, RR, A0	DEGA	00011
COMMON/AIMAX/ AF(51,402), JMAXF(402), IMAX,	DEGA	00012
1 IMAXF, IMAXFP1, IMAXB, IMAXBP1, C(3,400)	DEGA	00013
2 CF(3,402), ATOP(401), ATOPF(402)	DEGA	00014
DIMENSION AB(51,402), JMAXB(402), CB(3,402), ATOPB(402)	DEGA	00015
EQUIVALENCE(AF(1), AB(1)), (JMAXF(1), JMAXB(1)),	DEGA	00016
1 (CF(1), CB(1)), (ATOPF(1), ATOPB(1))	DEGA	00017
COMMON/EPSCONV/ FBMAX, EMJ6, EMATELE, EMDVDD, MAXDIM	DEGA	00018
DIMENSION AXLAMB(100)	DEGA	00019
DIMENSION XLAMWKB(100), LWKB(101), NWKR(100)	DEGA	00020
COMMON/PI/PI, TWOSPI	DEGA	00021
COMMON/CB1/ HNUVEC(500), ACOFVEC(500), NOPTS, IHNUMAX, MM(100),	DEGA	00022
1 NN(100), MNMAX, I4MIN, I4MAX	DEGA	00023
COMMON/METHODS/METHOD, TEMPLAM, CASF, CAPFP, THETA, THETAP	DEGA	00024
PJ = 3.1415926535898 \$ TWOSPI = 2.0/PI	DEGA	00025
MAXITER = 100	DEGA	00026
MAXDIM = 100	DEGA	00027
EPSCONV = 1.0E-5	DEGA	00028
FRMAX = 1.0E200	DEGA	00029
EM16 = 1.0E-8	DEGA	00030
EMATELE = 1.0E-5	DEGA	00031
EMDVDD = 1.0E-8	DEGA	00032
RMAXFAC = 25.0	DEGA	00033
EPHI = 1.0E-15	DEGA	00034
DRMIN = 1.0E-8	DEGA	00035
DVMAX = 1.0E-4	DEGA	00036
ATOPFAC = .5	DEGA	00037
DHNU1 = .1	DEGA	00038
ZZZFAC = 10.0	DEGA	00039
NODIVNG = 70	DEGA	00040
XLAMFC1 = 3.0	DEGA	00041
XLAMFC2 = .2	DEGA	00042
C MAKE XLAMFC1 .GT. 1.0 AND XLAMFC2 .LT. 1.0	DEGA	00043
CALL ADV(15)	DEGA	00044
1 FORMAT(E24.14, 2I5)	DEGA	00045
2 FORMAT(BF10.3)	DEGA	00046
3 FORMAT(5E25.14)	DEGA	00047
7 FORMAT(16I5)	DEGA	00048
9 FORMAT(BF10.3)	DEGA	00049
997 CONTINUE	DEGA	00050
MFAC = 1	DEGA	00051
NOPTS = 500	DEGA	00052
KPHIMAX = 0	DEGA	00053
IFRROR = 0	DEGA	00054
RMIN = 0.0	DEGA	00055
READ 2, Z, RR2, RR	DEGA	00056
IF(Z .LT. 0.0) GO TO 998	DEGA	00057
RMAX = RR	DEGA	00058
PRINT 3, Z, RR2, RR	DEGA	00059
ZM1 = Z - 1.0	DEGA	00060
A0 = .6057*Z**.333333	DEGA	00061
READ 7, ISSMAX	DEGA	00062
PRINT 7, ISSMAX	DEGA	00063

	READ 7, (NSUBSHL(ISS), ISS=1, ISSMAX)	DEGA	00064
	PRINT 7, (NSUBSHL(ISS), ISS=1, ISSMAX)	DEGA	00065
	READ 7, KWKB	DEGA	00066
	PRINT 7, KWKB	DEGA	00067
133	FORMAT(F15.5, 2I5)	DEGA	00068
	DO 132 K=1, KWKB	DEGA	00069
	READ 133, XLAMWKB(K), LWKB(K), NWKB(K)	DEGA	00070
	PRINT 1, XLAMWKB(K), LWKB(K), NWKB(K)	DEGA	00071
	XLAMARY(K) = XLAMWKB(K)	DEGA	00072
	LARY(K) = LWKB(K)	DEGA	00073
132	NSHELL(K) = NWKB(K)	DEGA	00074
	LARY(KWKB + 1) = -1	DEGA	00075
	IF(RR2 .GE. 0.0) SCRATCH(1) = V1(RR2)	DEGA	00076
	CALL YYY(RMIN,RMAX,DRMIN,DVMAX,ATOPFAC,I2,IERROR,	DEGA	00077
1	XLAMWKB, LWKB, KWKB, ZZZFAC)	DEGA	00078
	IF(IERROR .NE. 0) GO TO 99	DEGA	00079
	I2P1 = I2 + 1	DEGA	00080
	ISKIP = 3*MAXDIM + 4	DEGA	00081
	IECS = 1 - ISKIP	DEGA	00082
	DO 118 I=1, I2	DEGA	00083
	IECS = IECS + ISKIP	DEGA	00084
	CALL ECWR(ATOP(I+1), IECS, 1, IE)	DEGA	00085
118	CALL ECWR(C(1,I), IECS+1, 3, IE)	DEGA	00086
	PRINT 7, IMAX, I2	DEGA	00087
	IMAXP1 = IMAX + 1	DEGA	00088
	IF(IMAXP1 .GT. 400) GO TO 109	DEGA	00089
	DO 108 I=IMAXP1, 400	DEGA	00090
	C(1,I) = 0.0	DEGA	00091
	C(2,I) = -1.0	DEGA	00092
108	C(3,I) = 0.0	DEGA	00093
109	ATOP(1) = 0.0	DEGA	00094
	DO 10 I=1, IMAX	DEGA	00095
10	PRINT 3, C(1,I), C(2,I), C(3,I), ATOP(I+1)	DEGA	00096
	GO TO 49	DEGA	00097
135	CONTINUE	DEGA	00098
	CALL ZZZ(XLAMARY, LARY, NSHELL, KPHIMAX, HNUVEC, NOPTS,	DEGA	00099
1	IHNUMAX, DHNUI, ZZZFAC)	DEGA	00100
136	FORMAT(///)	DEGA	00101
	PRINT 136	DEGA	00102
	DO 137 KPHI=1, KPHIMAX	DEGA	00103
137	PRINT 133, XLAMARY(KPHI), LARY(KPHI), NSHELL(KPHI)	DEGA	00104
	PRINT 136	DEGA	00105
	PRINT 3, (HNUVEC(IHNU), IHNU=1, IHNUMAX)	DEGA	00106
	PRINT 136	DEGA	00107
	DO 138 IHNU=1, IHNUMAX	DEGA	00108
138	ACOFVEC(IHNU) = 0.0	DEGA	00109
	KWKBMAX = 0	DEGA	00110
	DO 139 KPHI=1, KPHIMAX	DEGA	00111
	KWKBMAX = KWKBMAX + 1	DEGA	00112
	NWKB(KWKBMAX) = NSHELL(KPHI)	DEGA	00113
	LWKB(KWKBMAX) = 1000*(LARY(KPHI) + 1) + LARY(KPHI)	DEGA	00114
	XLAMWKB(KWKBMAX) = XLAMARY(KPHI)	DEGA	00115
	IF(LARY(KPHI) .EQ. 0) GO TO 139	DEGA	00116
	KWKBMAX = KWKBMAX + 1	DEGA	00117
	NWKB(KWKBMAX) = NSHELL(KPHI)	DEGA	00118
	LWKB(KWKBMAX) = 1000*(LARY(KPHI) - 1) + LARY(KPHI)	DEGA	00119
	XLAMWKB(KWKBMAX) = XLAMARY(KPHI)	DEGA	00120
139	CONTINUE	DEGA	00121
	KWKB = 0	DEGA	00122
	I4MIN = 1	DEGA	00123
140	KWKB = KWKB + 1	DEGA	00124
	XLAMARY(1) = XLAMWKB(KWKB)	DEGA	00125

	NSHELL(1) = NWKB(KWKB)	DEGA	00126
	LPOS = LWKB(KWKB)/1000	DEGA	00127
	LARY(1) = LWKB(KWKB) - LPOS*1000	DEGA	00128
141	IF(1.0000001*ABS(XLAMARY(1)) .LT. HNUVFC(I4MIN)) GO TO 142	DEGA	00129
	I4MIN = I4MIN + 1	DEGA	00130
	GO TO 141	DEGA	00131
142	I4MAX = IHNUMAX	DEGA	00132
	MNMAX = 0	DEGA	00133
	DO 143 I4=I4MIN, I4MAX	DEGA	00134
	MNMAX = MNMAX + 1	DEGA	00135
	MNMAXP1 = MNMAX + 1	DEGA	00136
	MM(MNMAX) = 1	DEGA	00137
	NN(MNMAX) = MNMAXP1	DEGA	00138
	XLAMARY(MNMAXP1) = HNUVEC(I4) + XLAMARY(1)	DEGA	00139
	LARY(MNMAXP1) = LPOS	DEGA	00140
143	NSHELL(MNMAXP1) = 0	DEGA	00141
	LARY(MNMAXP1 + 1) = -2	DEGA	00142
	KPHIMAX = 0	DEGA	00143
49	CONTINUE	DEGA	00144
	KPHIMP1 = KPHIMAX + 1	DEGA	00145
	L = LARY(KPHIMP1)	DEGA	00146
	IF(L .EQ. -1) GO TO 135	DEGA	00147
	IF(L .EQ. -2) GO TO 99	DEGA	00148
	METHOD = 2	DEGA	00149
	IF(XLAMARY(KPHIMP1) .LT. 0.0) METHOD = 1	DEGA	00150
	XL = L	DEGA	00151
	XLLP1 = L*(L+1)	DEGA	00152
	XLLM1 = XL*(XL-1.0)	DEGA	00153
	TWOXL = 2.0*XL	DEGA	00154
	LM2 = L-2	DEGA	00155
	LM1 = L-1	DEGA	00156
	LP1 = L+1	DEGA	00157
	XLPI = LP1	DEGA	00158
	IF (METHOD .EQ. 1) GO TO 107	DEGA	00159
	XLAMBDA = XLAMARY(KPHIMP1)	DEGA	00160
	SMALLK = SQRT(XLAMBDA)	DEGA	00161
	RMAX = (XLLP1 + 1.0/XLAMBDA - 30.0)*SMALLK	DEGA	00162
	RTMP = 10.0/XLAMBDA	DEGA	00163
	IF(RTMP .GT. RMAX) RMAX = RTMP	DEGA	00164
	IF(ATOP(I2+1) .GT. RMAX) RMAX = 1.05*ATOP(I2+1)	DEGA	00165
	IF(RMAX .LT. 1.01) RMAX = 1.01	DEGA	00166
	GO TO 122	DEGA	00167
146	RMAX = 1.2*RMAX	DEGA	00168
	IERROR = 0	DEGA	00169
	GO TO 110	DEGA	00170
107	AROOT(1) = 0.0	DEGA	00171
	AROOT(2) = 0.0	DEGA	00172
	IMAX = I2 + 1	DEGA	00173
	ATOP(IMAX+1) = 1000.0	DEGA	00174
	CALL ROOTDIV(XLAMARY(KPHIMP1), XLLP1, AROOT, ICNTR, ICNTD, ICASE,	DEGA	00175
	1 IERROR, 1)	DEGA	00176
	PRINT 3, XLAMARY(KPHIMP1), AROOT(1), AROOT(2)	DEGA	00177
	IF(IERROR .EQ. 0) GO TO 151	DEGA	00178
	PRINT 152, IERROR	DEGA	00179
152	FORMAT(15, * RMAX SET TO 1.1*ATOP(I2+1)*)	DEGA	00180
	IERROR = 0	DEGA	00181
	RMAX = 1.1*ATOP(I2+1)	DEGA	00182
	GO TO 122	DEGA	00183
151	RMAX = AROOT(2)	DEGA	00184
	IF(AROOT(1) .GT. RMAX) RMAX = AROOT(1)	DEGA	00185
	RMAX = RMAXFAC*RMAX	DEGA	00186
	IF(RMAX .LT. 1.01*ATOP(I2+1)) RMAX = 1.01*ATOP(I2+1)	DEGA	00187

PRINT 153, RMAX	DEGA	00188
153 FORMAT(* RMAX = *, E20.10)	DEGA	00189
122 IMAX = I2	DEGA	00190
110 DR = ATOPFAC*ATOP(IMAX+1)	DEGA	00191
RTMP = ATOP(IMAX + 1)	DEGA	00192
SMALLK = SQRT(ABS(XLAMARY(KPHIMP1) + (2.0 - XLLP1/RTMP)/RTMP))	DEGA	00193
RTMP = ATOP(IMAX+1) + .5*(ATOP(IMAX+1) - ATOP(IMAX))	DEGA	00194
SMALLK1 = SQRT(ABS(XLAMARY(KPHIMP1) + (2.0 - XLLP1/RTMP)/RTMP))	DEGA	00195
IF(SMALLK1 .GT. SMALLK) SMALLK = SMALLK1	DEGA	00196
DRMAX = 6.28/SMALLK	DEGA	00197
IF(DR .GT. DRMAX) DR = DRMAX	DEGA	00198
IMAX = IMAX + 1	DEGA	00199
ATOP(IMAX+1) = ATOP(IMAX) + DR	DEGA	00200
IF(IMAX .EQ. 400) 124, 123	DEGA	00201
124 PRINT 125	DEGA	00202
125 FORMAT(* 400 INTERVALS WILL NOT SPAN (0,RMAX)*)	DEGA	00203
GO TO 999	DEGA	00204
123 IF(ATOP(IMAX+1) .GE. RMAX) 111, 110	DEGA	00205
111 ATOP(IMAX+1) = RMAX	DEGA	00206
CAPR = RMAX	DEGA	00207
ATOPIMP = ATOP(IMAX + 1)	DEGA	00208
ATOP(IMAX+1) = CAPR	DEGA	00209
GO TO (68,69), METHOD	DEGA	00210
68 NODIV = NODIVNG	DEGA	00211
NODIVP1 = NODIV + 1	DEGA	00212
AXLAMB(1) = XLAMFC1*XLAMARY(KPHIMP1)	DEGA	00213
AXLAMB(NODIVP1) = XLAMFC2*XLAMARY(KPHIMP1)	DEGA	00214
DAXLAMB = ABS(ALOG(ABS(AXLAMB(1))))	DEGA	00215
1 = ALOG(ABS(AXLAMB(NODIVP1)))/NODIV	DEGA	00216
IF(ABS(AXLAMB(1)) .GT. ABS(AXLAMB(NODIVP1))) DAXLAMB = -DAXLAMB	DEGA	00217
DO 60 I=2, NODIV	DEGA	00218
60 AXLAMB(I) = -EXP(ALOG(ABS(AXLAMB(I-1))) + DAXLAMB)	DEGA	00219
XLAMBDA = AXLAMB(I)	DEGA	00220
GO TO 70	DEGA	00221
69 TEMPLAM = XLAMBDA	DEGA	00222
NODIV = 4 \$ NODIVP1 = NODIV + 1	DEGA	00223
AXLAMB(1) = 0.0 \$ DXLAM = PI/4.0	DEGA	00224
DO 71 I=1, NODIV	DEGA	00225
71 AXLAMB(I+1) = AXLAMB(I) + DXLAM	DEGA	00226
CALL CARSON(XLAMBDA, C(2,IMAX), MFAC, CAPR, XLLP1,	DEGA	00227
1 CAPF, CAPFP, THETA, THETAP, IERROR)	DEGA	00228
IF(IERROR .EQ. 12) GO TO 146	DEGA	00229
70 CONTINUE	DEGA	00230
AROOT(1) = 0.0 \$ AROOT(2) = 0.0	DEGA	00231
CALL ROOTDIV(XLAMBDA, XLLP1, AROOT, ICNTR, ICNTD, ICASE,	DEGA	00232
1 IERROR, 0)	DEGA	00233
PRINT 3, XLAMBDA, AROOT(1), AROOT(2)	DEGA	00234
J25SET = 1	DEGA	00235
80 J25FST = 1	DEGA	00236
J25MIN = J25SET	DEGA	00237
J25MAX = NODIVP1	DEGA	00238
GO TO 90	DEGA	00239
81 J25FST = 2	DEGA	00240
J25MIN = NODIVP1+1	DEGA	00241
J25MAX = NODIVP1 + 20	DEGA	00242
D25 = (XLAMSTP - XLAMSTT)/19.0	DEGA	00243
AXLAMB(J25MIN) = XLAMSTT	DEGA	00244
AXLAMB(J25MAX) = XLAMSTP	DEGA	00245
I79MIN = J25MIN+1	DEGA	00246
I79MAX = J25MAX - 1	DEGA	00247
DO 79 I79=I79MIN, I79MAX	DEGA	00248
79 AXLAMB(I79) = AXLAMB(I79 - 1) + D25	DEGA	00249

90	DO 25 J= J25MIN, J25MAX	DEGA	00250
	XLAMSTP = AXLAMB(J)	DEGA	00251
	CALL TAYLORF(XL, XLAMSTP, IERROR, AF, JMAXF, IMAXF, CF, ATOFF)	DEGA	00252
	IRKW = IMAXFP1 + 1	DEGA	00253
	CALL TAYLORB(XL, XLAMSTP, IERROR, AB(1, IRKW),	DEGA	00254
	1 JMAXH(IRKW), IMAXB, CB(1, IRKW), ATOPR(IRKW))	DEGA	00255
	CALL BOUNDRY(IFCONV, FBSTP)	DEGA	00256
	TROUBLE MAY ARISE (MISS AN FIGENVALUE) IF FBSTP=0 TWICE	DEGA	00257
C	IN A ROW.	DEGA	00258
C	*****	DEGA	00259
C	HERE, LATER, FIND NEW ROOTS AND CHECK STABILITY CONDITIONS.	DEGA	00260
	IF(IFCONV .EQ. 1) 73, 74	DEGA	00261
73	CAPLAMB = XLAMSTP	DEGA	00262
	GO TO 75	DEGA	00263
74	IF(J .EQ. J25MIN) 26, 27	DEGA	00264
27	IF(FBSTP*FBSTP .LE. 0.0) 28, 26	DEGA	00265
28	XLAMMID = .5*(XLAMSTP + XLAMSTT)	DEGA	00266
	CALL TAYLORF(XL, XLAMMID, IERROR, AF, JMAXF, IMAXF, CF, ATOFF)	DEGA	00267
	IRKW = IMAXFP1 + 1	DEGA	00268
	CALL TAYLORB(XL, XLAMMID, IERROR, AB(1, IRKW),	DEGA	00269
	1 JMAXB(IRKW), IMAXB, CB(1, IRKW), ATOPR(IRKW))	DEGA	00270
	CALL BOUNDRY(IFCONV, FBMID)	DEGA	00271
	IF(FBSTT .GT. FBSTP) 29, 30	DEGA	00272
29	FBTOP = FBSTT \$ FBBOT = FBSTP \$ GO TO 31	DEGA	00273
30	FBTOP = FBSTP \$ FBBOT = FBSTT	DEGA	00274
31	IF((FBTOP .GT. FBMID) .AND. (FBMID .GT. FBBOT)) 32, 57	DEGA	00275
57	PRINT 58	DEGA	00276
	PRINT 58	DEGA	00277
	PRINT 59	DEGA	00278
	PRINT 58	DEGA	00279
	PRINT 58	DEGA	00280
58	FORMAT(* XXX*)	DEGA	00281
59	FORMAT(* FBMID IS NOT BETWEEN FBTOP AND FBBOT. MAKE SURE A VALUE	DEGA	00282
	1 OF LAMBDA WAS NOT MISSED.*)	DEGA	00283
	GO TO 87	DEGA	00284
32	XLAMB0 = XLAMSTT \$ XLAMB1 = XLAMSTP	DEGA	00285
	FRO = FBSTT \$ FB1 = FBSTP	DEGA	00286
C	REGULA DOES NOT CHECK STABILITY CONDITIONS.	DEGA	00287
	CALL REGULA(XLAMB0, XLAMB1, FRO, FB1, CAPLAMB, MAXITER, XL, IERROR)	DEGA	00288
C	HERE, LATER, FIND NEW ROOTS AND CHECK STABILITY CONDITIONS.	DEGA	00289
	IF(IERROR .NE. 0) GO TO 87	DEGA	00290
	IF(XLAMSTT .LT. XLAMSTP) GO TO 105	DEGA	00291
	XLAMB0 = XLAMSTP	DEGA	00292
	XLAMB1 = XLAMSTT	DEGA	00293
	GO TO 104	DEGA	00294
105	XLAMB0 = XLAMSTT	DEGA	00295
	XLAMB1 = XLAMSTP	DEGA	00296
104	IF(XLAMBO .LE. CAPLAMB .AND. CAPLAMB .LE. XLAMB1) 75, 86	DEGA	00297
86	PRINT 58	DEGA	00298
	PRINT 58	DEGA	00299
	PRINT 88	DEGA	00300
88	FORMAT(* CAPLAMB IS NOT BETWEEN XLAMSTT AND XLAMSTP. XLAMSTT, CA	DEGA	00301
	1PLAMB, AND XLAMSTP ARE *)	DEGA	00302
	PRINT 3, XLAMSTT, CAPLAMB, XLAMSTP	DEGA	00303
	PRINT 58	DEGA	00304
	PRINT 58	DEGA	00305
87	IF(J25FST .EQ. 1) 82, 84	DEGA	00306
82	J25SET = J	DEGA	00307
	PRINT 83	DEGA	00308
83	FORMAT(* DIVIDE THIS INTERVAL INTO 19 EQUAL INTERVALS AND TRY AGA	DEGA	00309
	1IN. *)	DEGA	00310
	PRINT 58	DEGA	00311

PRINT 58	DEGA	00312
GO TO 81	DEGA	00313
84 PRINT 85	DEGA	00314
85 FORMAT(* THIS INTERVAL FAILED FOR THE SECOND TIME. FORGET IT AND	DEGA	00315
1 GO TO THE NEXT INTERVAL. *)	DEGA	00316
PRINT 58	DEGA	00317
PRINT 58	DEGA	00318
GO TO 80	DEGA	00319
75 GO TO(65, 66), METHOD	DEGA	00320
66 ALPHA = CAPLAMB \$ CAPLAMB = XLAMBDA	DEGA	00321
PRINT 67, ALPHA, CAPLAMB	DEGA	00322
67 FORMAT(* ALPHA =*, E24.14, * FOR XLAMBDA =*, E24.14)	DEGA	00323
GO TO 127	DEGA	00324
65 CONTINUE	DEGA	00325
IV1 = 0	DEGA	00326
D = ATOPF(2)	DEGA	00327
DLP1 = D*LP1	DEGA	00328
JMAXROT = JMAXF(1) - 1	DEGA	00329
PDUM(1) = AF(1,2)/DLP1	DEGA	00330
PDUM(2) = (AF(2,2) - XLPI*AF(1,2)/D)/DLP1	DEGA	00331
CALL STURMSQ(AF(2,1), JMAXROT, PDUM, D, IV2, IERROR)	DEGA	00332
IV1 = IV1 + IV2	DEGA	00333
IF(IMAXF .EQ. 1) GO TO 98	DEGA	00334
DO 91 IF=2, IMAXF	DEGA	00335
D = ATOPF(IF+1) - ATOPF(IF)	DEGA	00336
CALL STURMSQ(AF(1,IF), JMAXF(IF), AF(1,IF+1), D, IV2, IERROR)	DEGA	00337
91 IV1 = IV1 + IV2	DEGA	00338
98 CONTINUE	DEGA	00339
DO 95 IR=1, IMAXB	DEGA	00340
IBKW = IMAXFP1 + IR	DEGA	00341
D = ATOPB(IBKW + 1) - ATOPB(IBKW)	DEGA	00342
CALL STURMSQ(AB(1,IBKW), JMAXB(IBKW), AB(1,IBKW+1),	DEGA	00343
1 D, IV2, IERROR)	DEGA	00344
95 IV1 = IV1 + IV2	DEGA	00345
PRINT 128, IV1	DEGA	00346
128 FORMAT(* THIS WAVE FUNCTION HAS*, I5, * CROSSINGS.*)	DEGA	00347
127 IBKW = IMAXFP1 + IMAXBPI	DEGA	00348
FAC = AB(1,IBKW)/AF(1,IMAXFP1)	DEGA	00349
DO 33 I=1, IMAXF	DEGA	00350
JJJ = JMAXF(I)	DEGA	00351
DO 33 JF=1, JJJ	DEGA	00352
33 AF(JF,I) = FAC*AF(JF,I)	DEGA	00353
IF(CAPLAMB .LT. 0.0) CALL NORMPHI(CAPLAMB, L)	DEGA	00354
PRINT 3, CAPLAMB	DEGA	00355
KPHIMAX = KPHIMAX + 1	DEGA	00356
GO TO(129,130), METHOD	DEGA	00357
129 IV1L1 = IV1 + L + 1	DEGA	00358
IF(NSHELL(KPHIMAX) .EQ. IV1L1) GO TO 154	DEGA	00359
IV1 = NSHELL(KPHIMAX) - L - 1	DEGA	00360
KPHIMAX = KPHIMAX - 1	DEGA	00361
PRINT 134, IV1	DEGA	00362
134 FORMAT(* THE PREVIOUS WAVE FUNCTION SHOULD HAVE HAD*, I5,	DEGA	00363
1 * CROSSINGS.*/ * FORGET THE LAST EIGENVALUE AND TRY SOMEMORE.*/	DEGA	00364
GO TO 26	DEGA	00365
154 IRKW = IMAXFP1 + 1	DEGA	00366
PHIMIN9 = ABS(AB(1,IRKW))	DEGA	00367
PHIMAX9 = PHIMIN9	DEGA	00368
IF(IMAXB .EQ. 1) GO TO 155	DEGA	00369
DO 156 IR=2, IMAXB	DEGA	00370
IRKW = IMAXFP1 + IR	DEGA	00371
PHI9 = ABS(AB(1,IRKW))	DEGA	00372
IF(PHI9 .GT. PHIMAX9) PHIMAX9 = PHI9	DEGA	00373



156	CONTINUE	DEGA	00374
155	DO 157 IF=1, IMAXF	DEGA	00375
	PHI9 = ABS(AF(1,IF))	DEGA	00376
	IF(PHI9 .GT. PHIMAX9) PHIMAX9 = PHI9	DEGA	00377
157	CONTINUE	DEGA	00378
	PRINT 158, PHIMAX9, PHIMIN9	DEGA	00379
158	FORMAT(* PHI MAX AND MIN ARE*, 2E20.10)	DEGA	00380
	IF(PHIMIN9 .LT. PHIMAX9*EPI) GO TO 131	DEGA	00381
	PRINT 159	DEGA	00382
159	FORMAT(* RMAX IS NOT BIG ENOUGH*)	DEGA	00383
	KPHIMAX = KPHIMAX - 1	DEGA	00384
	GO TO 146	DEGA	00385
130	NSHELL(KPHIMAX) = 0	DEGA	00386
131	IMAXARY(KPHIMAX) = IMAX	DEGA	00387
	XLAMARY(KPHIMAX) = CAPLAMB	DEGA	00388
	LARY(KPHIMAX) = L	DEGA	00389
	IF(CAPLAMB .LT. 0.0) 112, 113	DEGA	00390
112	ALFAARY(KPHIMAX) = 0.0	DEGA	00391
	GO TO 114	DEGA	00392
113	ALFAARY(KPHIMAX) = ALPHA	DEGA	00393
114	CONTINUE	DEGA	00394
	JMAXS(1) = JMAXF(1)	DEGA	00395
	DO 115 I=1, IMAXF	DEGA	00396
	AM1S(I) = AF(1,I)	DEGA	00397
115	AM2S(I) = AF(2,I)	DEGA	00398
	IC = IMAXB(1)	DEGA	00399
	ID = IMAXF	DEGA	00400
	DO 116 I=1, IMAXB	DEGA	00401
	IC = IC-1	DEGA	00402
	ID = ID+1	DEGA	00403
	IBKW = IMAXFP1 + IC	DEGA	00404
	AM1S(ID) = AB(1,IBKW)	DEGA	00405
	AM2S(ID) = AB(2,IBKW)	DEGA	00406
116	JMAXS(ID) = JMAXB(1,IBKW)	DEGA	00407
	DO 117 I=2, IMAXFP1	DEGA	00408
117	JMAXS(I) = 0	DEGA	00409
	IMAXB(1) = IMAXB + IMAXF	DEGA	00410
	IFCS = KPHIMAX + 4	DEGA	00411
	IECS1 = IECS + MAXDIM	DEGA	00412
	IECS2 = IECS1 + MAXDIM	DEGA	00413
	CALL ECWR(JMAXS(1), IECS, 1, IE)	DEGA	00414
	CALL ECWR(AM1S(1), IECS1, 1, IE)	DEGA	00415
	CALL ECWR(AM2S(1), IECS2, 1, IE)	DEGA	00416
	DO 121 I=3, I2P1	DEGA	00417
	IFCS = IECS + ISKIP	DEGA	00418
	IECS1 = IECS + MAXDIM	DEGA	00419
	IECS2 = IECS1 + MAXDIM	DEGA	00420
	CALL ECWR(JMAXS(I), IECS, 1, IE)	DEGA	00421
	CALL ECWR(AM1S(I), IECS1, 1, IE)	DEGA	00422
121	CALL ECWR(AM2S(I), IECS2, 1, IE)	DEGA	00423
	IF(IMAX .EQ. I2) GO TO 126	DEGA	00424
	IECS = 4*(400 - I2)	DEGA	00425
	IECS = IECS*(KPHIMAX - 1) + 1	DEGA	00426
	IFCS = IECS + I2*ISKIP	DEGA	00427
	IJUMP = 400 - I2	DEGA	00428
	INUM = IMAX - I2	DEGA	00429
	I2P2 = I2+2	DEGA	00430
	CALL ECWR(ATOP(I2P2), IECS, INUM, IE)	DEGA	00431
	IECS = IECS + IJUMP	DEGA	00432
	CALL ECWR(JMAXS(I2P2), IECS, INUM, IE)	DEGA	00433
	IECS = IECS + IJUMP	DEGA	00434
	CALL ECWR(AM1S(I2P2), IECS, INUM, IE)	DEGA	00435

IECS = IECS + IJUMP	DEGA	00436
CALL ECWR(AM2S(I2P2), IECS, INUM, IE)	DEGA	00437
126 GO TO 61	DEGA	00438
26 XLAMSTT = XLAMSTP \$ FBSTT = FBSTP	DEGA	00439
25 CONTINUE	DEGA	00440
IF(J25FST.EQ. 2) 80, 998	DEGA	00441
61 ATOP(IMAX+1) = ATOPTMP	DEGA	00442
GO TO 49	DEGA	00443
99 IF(KPHIMAX.EQ. 0) GO TO 999	DEGA	00444
DO 120 KPHI = 1, KPHIMAX	DEGA	00445
PRINT 7, LARY(KPHI)	DEGA	00446
120 PRINT 3, XLAMARY(KPHI), ALFAARY(KPHI)	DEGA	00447
CALL MATELE(KPHIMAX, XLAMARY, LARY, ALFAARY, IMAXARY, I2,	DEGA	00448
1 NSUBSHL, ISSMAX, NSHELL)	DEGA	00449
999 CONTINUE	DEGA	00450
IF(KWKB.LT. KWKBMAX) GO TO 140	DEGA	00451
145 FORMAT(15X, * EV*, 10X, * BARNS/ATOM*)	DEGA	00452
PRINT 145	DEGA	00453
YTOP = ALOG10(ACOFVEC(1))	DEGA	00454
YBOT = YTOP	DEGA	00455
DO 144 IHNU=1, IHNUMAX	DEGA	00456
HNUVEC(IHNU) = 13.605*HNUVEC(IHNU)	DEGA	00457
PRINT 3, HNUVEC(IHNU), ACOFVEC(IHNU)	DEGA	00458
ACOFVEC(IHNU) = ALOG10(ACOFVEC(IHNU))	DEGA	00459
IF(ACOFVEC(IHNU).GT. YTOP) YTOP = ACOFVEC(IHNU)	DEGA	00460
IF(ACOFVEC(IHNU).LT. YBOT) YBOT = ACOFVEC(IHNU)	DEGA	00461
144 HNUVEC(IHNU) = ALOG10(HNUVEC(IHNU))	DEGA	00462
CALL ADV(2)	DEGA	00463
CALL DGA(120, 980, 50, 310, HNUVEC(1), HNUVEC(IHNUMAX), YTOP, YBOT)	DEGA	00464
CALL DLGLG	DEGA	00465
CALL SBLOG	DEGA	00466
CALL SLLOG	DEGA	00467
CALL PLOT(IHNUMAX, HNUVEC, 1, ACOFVEC, 1, 42, 1)	DEGA	00468
READ 7, IHNUMAX	DEGA	00469
PRINT 7, IHNUMAX	DEGA	00470
IF(IHNUMAX.EQ. 0) GO TO 997	DEGA	00471
READ 2, (HNUVEC(IHNU), ACOFVEC(IHNU)), IHNU=1, IHNUMAX)	DEGA	00472
DO 150 IHNU=1, IHNUMAX	DEGA	00473
PRINT 3, HNUVEC(IHNU), ACOFVEC(IHNU)	DEGA	00474
HNUVEC(IHNU) = ALOG10(HNUVEC(IHNU))	DEGA	00475
150 ACOFVEC(IHNU) = ALOG10(ACOFVEC(IHNU))	DEGA	00476
CALL PLOT(IHNUMAX, HNUVEC, 1, ACOFVEC, 1, 55, 0)	DEGA	00477
GO TO 997	DEGA	00478
998 CONTINUE	DEGA	00479
CALL ADV(15)	DEGA	00480
CALL EMPTY	DEGA	00481
END	DEGA	00482
SUBROUTINE ROOTDIV(XLAMBDA, XLLP1, AROOT, ICNTR, ICNTD, ICASE,	DEGA	00483
1 IERROR, IFSTOP)	DEGA	00484
DIMENSION AROOT(2)	DEGA	00485
COMMON/AIMAX/ AF(51,402), JMAXF(402), IMAX,	DEGA	00486
1 IMAXF, IMAXFP1, IMAXB, IMAXBp1, C(3,400),	DEGA	00487
2 CF(3,402), ATOP(401), ATOPF(402)	DEGA	00488
DIMENSION AB(51,402), JMAXB(402), CB(3,402), ATOPB(402)	DEGA	00489
EQUIVALENCE(AF(1), AB(1)), (JMAXF(1), JMAXB(1)),	DEGA	00490
1 (CF(1), CB(1)), (ATOPF(1), ATOPB(1))	DEGA	00491
ICNTR = 0 \$ ICNTD = 0 \$ IERROR = 0	DEGA	00492
DO 10 I=1, IMAX	DEGA	00493
II=I	DEGA	00494
CALL BINOM (XLAMBDA, XLLP1, II, NORROOTS, ROOT1, ROOT2, NODIV, DIV1)	DEGA	00495
NOROOT1 = NORROOTS + 1	DEGA	00496
GO TO(14, 11, 12), NOROOT1	DEGA	00497

11 IF((ATOP(I) .LE. ROOT1) .AND. (ROOT1 .LT. ATOP(I+1)))13,14	DEGA	00498
13 ICNTR = ICNTR + 1	DEGA	00499
AROOT(ICNTR) = ROOT1	DEGA	00500
IF(ICNTR .EQ. 2)15, 14	DEGA	00501
12 IF(ROOT1 .LT. ROOT2) 28, 27	DEGA	00502
27 TEMP = ROOT1 \$ ROOT1 = ROOT2 \$ ROOT2 = TEMP	DEGA	00503
28 IF((ATOP(I) .LE. ROOT1) .AND. (ROOT1 .LT. ATOP(I+1)))16,17	DEGA	00504
16 ICNTR = ICNTR + 1	DEGA	00505
AROOT(ICNTR) = ROOT1	DEGA	00506
IF(ICNTR .EQ. 2)15, 17	DEGA	00507
17 IF((ATOP(I) .LE. ROOT2) .AND. (ROOT2 .LT. ATOP(I+1)))18,14	DEGA	00508
18 ICNTR = ICNTR + 1	DEGA	00509
AROOT(ICNTR) = ROOT2	DEGA	00510
IF(ICNTR .EQ. 2)15, 14	DEGA	00511
15 IF(ICNTD .EQ. 1)19, 20	DEGA	00512
20 IF(NODIV .EQ. 1)21, 19	DEGA	00513
21 IF((ATOP(I) .LE. DIV1) .AND. (DIV1 .LT. ATOP(I+1)))22, 19	DEGA	00514
22 ICNTD = 1 \$ ADIV = DIV1 \$ IDIV = I \$ GO TO 19	DEGA	00515
14 IF(ICNTD .EQ. 1)10, 23	DEGA	00516
23 IF(NODIV .EQ. 1)24, 10	DEGA	00517
24 IF((ATOP(I) .LE. DIV1) .AND. (DIV1 .LT. ATOP(I+1)))25, 10	DEGA	00518
25 ICNTD = 1 \$ ADIV = DIV1 \$ IDIV = I	DEGA	00519
10 CONTINUE	DEGA	00520
19 IF((ICNTD .EQ. 1) .AND. (ICNTR .EQ. 2))29, 44	DEGA	00521
44 IF((ICNTD .EQ. 1) .AND. (ICNTR .EQ. 0)) 45, 26	DEGA	00522
45 ICASE = 5 \$ GO TO 31	DEGA	00523
29 ICASE = 4	DEGA	00524
IF((AROOT(1) .LT. ADIV) .AND. (ADIV .LT. AROOT(2)))31, 30	DEGA	00525
30 IERROR = 2	DEGA	00526
PRINT 2	DEGA	00527
2 FORMAT(* IERROR=2, ICASE=4, MAX IS NOT BETWEEN THE TOW ROOTS,*)	DEGA	00528
RETURN	DEGA	00529
26 IF((ICNTD .EQ. 0) .AND. (ICNTR .EQ. 0))34, 35	DEGA	00530
34 ICASE = 1 \$ IDIV = IMAX	DEGA	00531
ADIV = (ATOP(IMAX) + ATOP(IMAX+1))/2.0	DEGA	00532
GO TO 31	DEGA	00533
35 IF((ICNTD .EQ. 0) .AND. (ICNTR .EQ. 1))37, 40	DEGA	00534
37 ICASE = 2	DEGA	00535
IF(XLPL1 .LT. .25) 47, 46	DEGA	00536
47 IMAXP1 = IMAX + 1	DEGA	00537
DO 48 I=2, IMAXP1	DEGA	00538
IF(AROOT(I) .LT. ATOP(I)) 49, 48	DEGA	00539
49 IDIV = I-1	DEGA	00540
ADIV = AROOT(1)	DEGA	00541
GO TO 31	DEGA	00542
48 CONTINUE	DEGA	00543
IDIV = IMAX	DEGA	00544
GO TO 38	DEGA	00545
46 IDIV = IMAX	DEGA	00546
IF(AROOT(1) .LT. ATOP(IMAX))38, 39	DEGA	00547
38 ADIV = (ATOP(IMAX) + ATOP(IMAX+1))/2.0	DEGA	00548
GO TO 31	DEGA	00549
39 ADIV = (AROOT(1) + ATOP(IMAX+1))/2.0	DEGA	00550
GO TO 31	DEGA	00551
40 IF((ICNTD .EQ. 1) .AND. (ICNTR .EQ. 1))41, 42	DEGA	00552
42 IERROR = 1	DEGA	00553
PRINT 1, ICNTD, ICNTR	DEGA	00554
1 FORMAT(* IERROR=1, ICNTD =*, I2, *, ICNTR =*, I2)	DEGA	00555
RETURN	DEGA	00556
41 ICASE = 3	DEGA	00557
IF(AROOT(1) .LT. ADIV) 31, 43	DEGA	00558
43 IERROR = 3	DEGA	00559

PRINT 3, AROOT(1), ADIV	DEGA	00560
3 FORMAT(* IERROR=3, ICASE=3, AROOT(1),GE,ADIV, AROOT(1) =*,E20,10,	DEGA	00561
1 *, ADIV =*, E20,10)	DEGA	00562
RETURN	DEGA	00563
31 IF(I*STOP.EQ. 1) RETURN	DEGA	00564
IMAXF = IDIV \$ IMAXFP1 = IMAXF + 1	DEGA	00565
IMAXB = IMAX + 1 - IDIV \$ IMAXBP1 = IMAXB + 1	DEGA	00566
DO 32 I=1, IMAXF	DEGA	00567
ATOPF(I) = ATOP(I)	DEGA	00568
CF(1,I) = C(1,I) \$ CF(2,I) = C(2,I)	DEGA	00569
32 CF(3,I) = C(3,I)	DEGA	00570
ATOPF(IMAXFP1) = ADIV	DEGA	00571
II = IMAX + 1 \$ III = IMAX + 2	DEGA	00572
DO 33 I=1, IMAXB	DEGA	00573
III = III-1	DEGA	00574
IRKW = IMAXFP1 + I	DEGA	00575
ATOPB(IRKW) = ATOP(III)	DEGA	00576
II = II - 1	DEGA	00577
CR(1,IRKW) = C(1,II) \$ CB(2,IRKW) = C(2,II)	DEGA	00578
33 CB(3,IRKW) = C(3,II)	DEGA	00579
IRKW = IMAXFP1 + IMAXBP1	DEGA	00580
ATOPB(IRKW) = ADIV	DEGA	00581
RETURN	DEGA	00582
END	DEGA	00583
SUBROUTINE BINOM(XLAMBD, XLLP1, I, NORROOTS, ROOT1, ROOT2,	DEGA	00584
1 NODIV, DIV1)	DEGA	00585
COMMON/AIMAX/ AF(51,402), JMAXF(402), IMAX,	DEGA	00586
1 IMAXF, IMAXFP1, IMAXB, IMAXBP1, C(3,400),	DEGA	00587
2 CF(3,402), ATOP(401), ATOPF(402)	DEGA	00588
DIMENSION AB(51,402), JMAXB(402), CB(3,402), ATOPB(402)	DEGA	00589
EQUIVALENCE(AF(1), AB(1)), (JMAXF(1), JMAXB(1)),	DEGA	00590
1 (CF(1), CB(1)), (ATOPF(1), ATOPB(1))	DEGA	00591
B1 = -(2.0*C(3,I) + XLLP1)	DEGA	00592
B2 = -2.0*C(2,I)	DEGA	00593
B3 = XLAMBD - 2.0*C(1,I)	DEGA	00594
ARSR1 = ABS(B1)	DEGA	00595
IF(ARSR1.LT. ABS(B2)*1.0E-12) 1, 2	DEGA	00596
2 IF(ARSR1.LT. ABS(B3)*1.0E-12) 1, 3	DEGA	00597
1 ROOT1 = -B2/B3	DEGA	00598
NORROOTS = 1	DEGA	00599
NODIV = 0	DEGA	00600
GO TO 6	DEGA	00601
3 RAD = B2*B2 - 4.0*B1*B3	DEGA	00602
TWOB1 = 2.0*B1	DEGA	00603
NODIV = 1	DEGA	00604
DIV1 = -TWOB1/B2	DEGA	00605
IF(RAD.LT. 0.0) 4, 5	DEGA	00606
4 NORROOTS = 0	DEGA	00607
GO TO 6	DEGA	00608
5 NORROOTS = 2	DEGA	00609
RAD = SQRT(RAD)	DEGA	00610
T1 = ABS(TWOB1)	DEGA	00611
T2 = ABS(B2 + RAD)	DEGA	00612
IF(T2.LT. T1*1.0E-20) 7, 8	DEGA	00613
9 FORMAT( I5, 3E20,10, * XXXXX*)	DEGA	00614
7 I1 = 1	DEGA	00615
PRINT 9, I1, TWOB1, B2, RAD	DEGA	00616
ROOT1 = 1.0E100	DEGA	00617
GO TO 10	DEGA	00618
8 CONTINUE	DEGA	00619
ROOT1 = -TWOB1/(B2 + RAD)	DEGA	00620
10 T1 = ABS(TWOB1)	DEGA	00621

T2 = ABS(-B2 + RAD)	DEGA	00622
IF (T2 .LT. T1*1.0E-20) 11, 12	DEGA	00623
11 I1 = 2	DEGA	00624
PRINT 9, I1, TWOB1, B2, RAD	DEGA	00625
ROOT2 = 1.0E100	DEGA	00626
GO TO 6	DEGA	00627
12 CONTINUE	DEGA	00628
ROOT2 = TWOB1/(-B2 + RAD)	DEGA	00629
6 RETURN	DEGA	00630
END	DEGA	00631
SUBROUTINE REGULA(XLAMB0,XLAMB1,FB0,FB1,CAPLAMB,MAXITER,XL,IERROR)	DEGA	00632
COMMON/AMAX/ AF(51,402), JMAXF(402), IMAX,	DEGA	00633
1 IMAXF, IMAXFP1, IMAXB, IMAXBP1, C(3,400),	DEGA	00634
2 CF(3,402), ATOP(401), ATOPF(402)	DEGA	00635
DIMENSION AB(51,402), JMAXB(402), CB(3,402), ATOPB(402)	DEGA	00636
EQUIVALENCE(AF(1), AB(1)), (JMAXF(1), JMAXB(1)),	DEGA	00637
1 (CF(1), CB(1)), (ATOPF(1), ATOPB(1))	DEGA	00638
COMMON/EPS/ EPSCONV, FBMAX	DEGA	00639
1 FORMAT(2E24.14, 4E20.10)	DEGA	00640
IERROR = 0	DEGA	00641
DO 14 ITER = 1, MAXITER	DEGA	00642
IF (FB1 .EQ. FB0) 20, 21	DEGA	00643
20 IRKW = IMAXFP1 + IMAXBP1	DEGA	00644
AFAC = AB(1,IBKW)/AF(1,IMAXFP1)*AF(2,IMAXFP1)	DEGA	00645
FBOUNDC = FBOUND/ABS(AFAC)	DEGA	00646
CAPLAMB = .5*(XLAMB1 + XLAMB0)	DEGA	00647
PRINT 22	DEGA	00648
PRINT 22	DEGA	00649
PRINT 23, FBOUNDC, EPSCONV	DEGA	00650
PRINT 24, CAPLAMB, XLAMB1, XLAMB0	DEGA	00651
PRINT 22	DEGA	00652
PRINT 22	DEGA	00653
RETURN	DEGA	00654
22 FORMAT(* XXX*)	DEGA	00655
23 FORMAT(* FB1=FB0, FBOUNDC = *.E20.10,* EPSCONV = *.E20.10)	DEGA	00656
24 FORMAT(* CAPLAMB = *.E24.14,4H = (.E24.14,2H +,E24.14,4H )/2)	DEGA	00657
21 ONEGAMA = (XLAMB1 - XLAMB0)/(FB1 - FB0)	DEGA	00658
XLAMB2 = XLAMB1 - ONEGAMA*FB1	DEGA	00659
IF (XLAMB2.EQ.XLAMB1.OR.XLAMB2.EQ.XLAMB0) GO TO 15	DEGA	00660
CALL TAYLORF(XL, XLAMB2, IERROR, AF, JMAXF, IMAXF, CF, ATOPF)	DEGA	00661
IF (IERROR .NE. 0) RETURN	DEGA	00662
IRKW = IMAXFP1 + 1	DEGA	00663
CALL TAYLORB(XL, XLAMB2, IERROR, AB(1,IBKW),	DEGA	00664
1 JMAXB(1,IBKW), IMAXB, CB(1,IBKW), ATOPB(1,IBKW))	DEGA	00665
IF (IERROR .NE. 0) RETURN	DEGA	00666
CALL BOUNDRY(IFCONV, FBOUND)	DEGA	00667
IF (IFCONV .EQ. 1) 15, 16	DEGA	00668
15 CAPLAMB = XLAMB2 \$ RETURN	DEGA	00669
16 IF (ABS(FBOUND) .GT. FBMAX) 17, 19	DEGA	00670
17 IERROR = 7	DEGA	00671
PRINT 7	DEGA	00672
7 FORMAT(* IERROR = 7, FBOUND .GT. FBMAX*)	DEGA	00673
RETURN	DEGA	00674
19 XLAMB0 = XLAMB1 \$ XLAMB1 = XLAMB2	DEGA	00675
FB0 = FB1	DEGA	00676
14 FB1 = FBOUND	DEGA	00677
IERROR = 8	DEGA	00678
PRINT 8, MAXITER	DEGA	00679
8 FORMAT (* PROBLEM DOES NOT CONVERGE WITHIN*.15,* ITERATIONS*)	DEGA	00680
RETURN	DEGA	00681
END	DEGA	00682
SUBROUTINE TAYLORF(XL, ALPHLAM, IERROR, A, JMAX, IMAX, C, ATOP)	DEGA	00683

DIMENSION A(51,1), JMAX(1), C(3,1), ATOP(1)	DEGA	00684
COMMON/METHODS/METHOD, TEMPLAM, CAPF, CAPFP, THETA, THETAP	DEGA	00685
COMMON/EPS/EPSCONV, FBMAX, EM16	DEGA	00686
GO TO (60, 61), METHOD	DEGA	00687
60 XLAMBDA = ALPHLAM	DEGA	00688
GO TO 62	DEGA	00689
61 ALPHA = ALPHLAM	DEGA	00690
XLAMBDA = TEMPLAM	DEGA	00691
62 CONTINUE	DEGA	00692
DO 30 I=1, IMAX	DEGA	00693
30 JMAX(I) = 0	DEGA	00694
IERRQR = 0	DEGA	00695
D = ATOP(2)	DEGA	00696
*****	DEGA	00697
C LEFT BOUNDARY CONDITIONS. THESE ARE ONLY TRUE FOR	DEGA	00698
C PHI(0)=0 WHERE PHI(R) IS THE EIGENFUNCTION.	DEGA	00699
*****	DEGA	00700
A(1,1) = 0.0	DEGA	00701
A(2,1) = 1.0	DEGA	00702
*****	DEGA	00703
CALL TAYLOR1(A(1,1), JMAX(1), D, XLAMBDA,	DEGA	00704
1 XL, C(1,1), C(2,1), EM16, IERROR)	DEGA	00705
IF(IERROR.EQ. 9) RETURN	DEGA	00706
RP = ATOP(2)	DEGA	00707
CALL POLYOP(A(1,1), JMAX(1), RR, P)	DEGA	00708
A(1,2) = P*RR**XL	DEGA	00709
CALL POLY1P(A(1,1), JMAX(1), RR, DERIVP)	DEGA	00710
A(2,2) = (DERIVP*RR + P*XL)*RR**(XL-1.0)	DEGA	00711
IF(IMAX.EQ. 1) GO TO 37	DEGA	00712
IBOT = 2	DEGA	00713
GO TO 58	DEGA	00714
ENTRY TAYLORB	DEGA	00715
DO 54 I=1, IMAX	DEGA	00716
54 JMAX(I) = 0	DEGA	00717
GO TO (65, 66), METHOD	DEGA	00718
65 XLAMBDA = ALPHLAM	DEGA	00719
GO TO 67	DEGA	00720
66 ALPHA = ALPHLAM	DEGA	00721
XLAMBDA = TEMPLAM	DEGA	00722
67 CONTINUE	DEGA	00723
IERROR = 0	DEGA	00724
IPOP = 1	DEGA	00725
*****	DEGA	00726
RIGHT BOUNDARY ,R.B. CONDITIONS. HERE A(1,1) IS AN	DEGA	00727
ARBITRARY CONSTANT, THE MAGNITUDE OF PHI(R.B.)	DEGA	00728
THE DERIVATIVE OF PHI(R.B.) NORMALIZED TO A(1,1).	DEGA	00729
*****	DEGA	00730
GO TO (63, 64), METHOD	DEGA	00731
63 CONTINUE	DEGA	00732
A(1,1) = 1.0E-140	DEGA	00733
A(2,1) = -SQRT(ABS(XLAMBDA))*A(1,1)	DEGA	00734
GO TO 58	DEGA	00735
64 THETA1 = THETA + ALPHA	DEGA	00736
COSTHE1 = COS(THETA1)	DEGA	00737
A(1,1) = CAPF*COSTHE1	DEGA	00738
A(2,1) = CAPFP*COSTHE1 - CAPF*THETAP*SIN(THETA1)	DEGA	00739
*****	DEGA	00740
58 XLL = XL*(XL+1.0)	DEGA	00741
DO 25 I=IBOT, IMAX	DEGA	00742
D = ATOP(I+1) - ATOP(I)	DEGA	00743
CALL TAYLORS(A(1,1), JMAX(I), ATOP(I), D, XLAMBDA,	DEGA	00744
1 XLL, C(1,I), C(2,I), C(3,I), EM16, IERROR)	DEGA	00745

	IF(IERROR.EQ.10) RETURN	DEGA	00746
	CALL POLYOP(A(1,I),JMAX(I),D,A(I,I+1))	DEGA	00747
	CALL POLYIP(A(1,I),JMAX(I),D,A(2,I+1))	DEGA	00748
25	CONTINUE	DEGA	00749
37	RETURN	DEGA	00750
	END	DEGA	00751
	SUBROUTINE BOUNDRY(IFCONV,FBOUND)	DEGA	00752
	COMMON/AMAX/ AF(51,402),JMAXF(402),IMAX,	DEGA	00753
1	IMAXF,IMAXFP1,IMAXB,IMAXBP1,C(3,400),	DEGA	00754
2	CF(3,402),ATOP(401),ATOPF(402)	DEGA	00755
	DIMENSION AB(51,402),JMAXB(402),CB(3,402),ATOPB(402)	DEGA	00756
	EQUIVALENCE(AF(1),AB(1)),(JMAXF(1),JMAXB(1)),	DEGA	00757
1	(CF(1),CB(1)),(ATOPF(1),ATOPB(1))	DEGA	00758
	COMMON/EPSCONV,FBMAX	DEGA	00759
C	YOU ARE LIVING DANGEROUSLY IF YOU LET IMAXF=1. THIS MAY RESU	DEGA	00760
C	IN AN UNDETECTED DIVISION BY ZERO OR AN UNDETECTED LOSS OF ACCURAC	DEGA	00761
	IF(IMAXF.EQ.1) 1, 2	DEGA	00762
2	IF(ABS(AF(1,IMAXFP1)).LT.ABS(AF(1,IMAXF))*1.0E-4) 3, 1	DEGA	00763
1	IRKW = IMAXFP1 + IMAXB	DEGA	00764
	IRKWP1 = IRKW + 1	DEGA	00765
	IF(ABS(AB(1,IRKWP1)).LT.ABS(AB(1,IRKW))*1.0E-4) 3, 5	DEGA	00766
3	PRINT 4	DEGA	00767
4	FORMAT(* IN DANGER OF DIVIDING BY ZERO, OR ATLEAST LOOSING ACCU	DEGA	00768
	1RACY: *)	DEGA	00769
5	IRKW = IMAXFP1 + IMAXBP1	DEGA	00770
	FAC = AB(1,IRKW)/AF(1,IMAXFP1)	DEGA	00771
	AFAC = FAC*AF(2,IMAXFP1)	DEGA	00772
	FBOUND = AFAC - AB(2,IRKW)	DEGA	00773
	EPSC = ABS(AFAC)*EPSCONV	DEGA	00774
	IF(ABS(FBOUND).LT.EPSC) 6, 7	DEGA	00775
6	IFCONV = 1 \$ GO TO 8	DEGA	00776
7	IFCONV = 0	DEGA	00777
8	RETURN	DEGA	00778
	END	DEGA	00779
	SUBROUTINE STURMSG(A1,JMAX1,A2,D,IV2,IERROR)	DEGA	00780
	DIMENSION A1(2),A2(2),CPI(51),CPIPI(51)	DEGA	00781
	COMMON/SCRATCH/SCRATCH(604)	DEGA	00782
	EQUIVALENCE(SCRATCH(1),CPI(1)),(SCRATCH(52),CPIPI(1))	DEGA	00783
	IERROR = 0	DEGA	00784
	JMAXI = JMAX1	DEGA	00785
	IF(D.LT.0.0) 11, 12	DEGA	00786
11	ISIGN = -1	DEGA	00787
	GO TO 13	DEGA	00788
12	ISIGN = 1	DEGA	00789
13	IV2 = 0	DEGA	00790
	J1 = JMAXI+1	DEGA	00791
	DO 14 J=1,JMAXI	DEGA	00792
	J1 = J1 - 1	DEGA	00793
	CPI(J) = A1(J1)	DEGA	00794
14	CPIPI(J) = (J1-1)*CPI(J)	DEGA	00795
	JMAXIP1 = JMAXI-1	DEGA	00796
	P0 = CPI(JMAXI)	DEGA	00797
	PD = A2(1)	DEGA	00798
	P10 = CPIPI(JMAXIP1)	DEGA	00799
	P1D = A2(2)	DEGA	00800
	IF(P10.EQ.0.0) GO TO 15	DEGA	00801
	IF(P0*P10.LT.0.0) 15, 16	DEGA	00802
15	IV2 = IV2 + 1	DEGA	00803
16	IF(P1D.EQ.0.0) GO TO 17	DEGA	00804
	IF(P0*P1D.LT.0.0) 17, 18	DEGA	00805
17	IV2 = IV2 - 1	DEGA	00806
18	FAC = CPI(1)/CPIPI(1)	DEGA	00807

ZERO = ABS(CPI(1))*1.0E-11	DEGA	00808
DO 19 J1=2, JMAXIP1	DEGA	00809
19 CPI(J1) = CPI(J1) - FAC*CPIPI(J1)	DEGA	00810
NOZERO = 1	DEGA	00811
DO 20 J=2, JMAXI	DEGA	00812
IF(ABS(CPI(J)) .LT. ZERO) 20, 21	DEGA	00813
20 NOZERO = NOZERO + 1	DEGA	00814
GO TO 99	DEGA	00815
21 JMAXI = JMAXI - NOZERO	DEGA	00816
DO 22 J=1, JMAXI	DEGA	00817
22 CPI(J) = CPI(J+NOZERO)	DEGA	00818
IF(JMAXI .LT. JMAXIP1) 23, 18	DEGA	00819
23 JMIN = JMAXI + 1	DEGA	00820
DO 24 J=JMIN, JMAXIP1	DEGA	00821
24 CPI(J) = 0.0	DEGA	00822
DO 25 J=1, JMAXIP1	DEGA	00823
TEMP = CPI(J)	DEGA	00824
CPI(J) = CPIPI(J)	DEGA	00825
25 CPIPI(J) = -TEMP	DEGA	00826
JTEMP = JMAXI	DEGA	00827
JMAXI = JMAXIP1	DEGA	00828
JMAXIP1 = JTEMP	DEGA	00829
P0 = P10	DEGA	00830
PD = P1D	DEGA	00831
P10 = CPIPI(JMAXIP1)	DEGA	00832
P1D = CPIPI(1)	DEGA	00833
IF(JMAXIP1 .EQ. 1) 28, 26	DEGA	00834
26 DO 27 J1=2, JMAXIP1	DEGA	00835
27 P1D = P1D*U + CPIPI(J1)	DEGA	00836
28 IF(P10 .EQ. 0.0) GO TO 29	DEGA	00837
IF(PU*P10 .LT. 0.0) 29, 30	DEGA	00838
29 IV2 = IV2 + 1	DEGA	00839
30 IF(P1D .EQ. 0.0) GO TO 31	DEGA	00840
IF(PU*P1D .LT. 0.0) 31, 35	DEGA	00841
31 IV2 = IV2 - 1	DEGA	00842
35 IF(JMAXIP1 .EQ. 1) 99, 18	DEGA	00843
99 IV2 = ISIGN*IV2	DEGA	00844
IF(IV2 .LT. 0) 32, 34	DEGA	00845
32 IERROR = 5	DEGA	00846
PRINT 33	DEGA	00847
33 FORMAT( * IV2 IS LESS THAN 0. * )	DEGA	00848
34 RETURN	DEGA	00849
END	DEGA	00850
SUBROUTINE YYY(RMIN, RMAX, DRMIN, DVMAX, ATOPFAC, I2, IERROR,	DEGA	00851
1 XLAMWKB, LWKB, KWKB, ZZZFAC)	DEGA	00852
DIMENSION XLAMWKB(1), LWKB(1)	DEGA	00853
COMMON/AMAX/ AF(51,402), JMAXF(402), IMAX,	DEGA	00854
1 IMAXF, IMAXFP1, IMAXB, IMAXBp1, C(3,400),	DEGA	00855
2 CF(3,402), ATOP(401), ATOPF(402)	DEGA	00856
DIMENSION AB(51,402), JMAXB(402), CB(3,402), ATOPB(402)	DEGA	00857
EQUIVALENCE(AF(1), AB(1)), (JMAXF(1), JMAXB(1)),	DEGA	00858
1 (CF(1), CB(1)), (ATOPF(1), ATOPB(1))	DEGA	00859
DIMENSION A(3, 4)	DEGA	00860
DIMENSION XLAMDIM(10), XLDIM(10)	DEGA	00861
COMMON/CB3/ Z, ZM1, RR2, RR, A0	DEGA	00862
2 FORMAT(/////)	DEGA	00863
IERROR = 0	DEGA	00864
XLAMMAX = 0.0	DEGA	00865
LMAX = 0	DEGA	00866
DO 65 I=1, 10	DEGA	00867
XLAMDIM(I) = 0.0	DEGA	00868
65 XLDIM(I) = I*(I-1)	DEGA	00869



DO 66 I=1, KWKB	DEGA	00870
IF(XLAMWKB(I) .LT. XLAMMAX) XLAMMAX = XLAMWKB(I)	DEGA	00871
IF(LWKB(I) .GT. LMAX) LMAX = LWKB(I)	DEGA	00872
J = LWKB(I) + 1	DEGA	00873
IF(XLAMWKB(I) .LT. XLAMDIM(J)) XLAMDIM(J) = XLAMWKB(I)	DEGA	00874
66 CONTINUE	DEGA	00875
LMAXP1 = LMAX + 1	DEGA	00876
XLDIM(LMAX + 2) = LMAXP1*(LMAX + 2)	DEGA	00877
LMAX = LMAX + 3	DEGA	00878
XLAMDIM(LMAX) = ZZZFAC*ABS(XLAMMAX)	DEGA	00879
XLDIM(LMAX) = 0.0	DEGA	00880
68 FORMAT(2E20.10)	DEGA	00881
DO 69 L=1, LMAX	DEGA	00882
69 PRINT 68, XLAMDIM(L), XLDIM(L)	DEGA	00883
IF(RR2 .GT. 0.0) GO TO 62	DEGA	00884
IMAX = 1	DEGA	00885
DR = ABS(RR2)	DEGA	00886
ATOP(2) = DR	DEGA	00887
C(1,1) = 0.0	DEGA	00888
C(2,1) = -1.0	DEGA	00889
C(3,1) = 0.0	DEGA	00890
63 DR = ATOPFAC*ATOP(IMAX+1)	DEGA	00891
IF(DR .GT. DRMAX) DR = DRMAX	DEGA	00892
IMAX = IMAX + 1	DEGA	00893
ATOP(IMAX+1) = ATOP(IMAX) + DR	DEGA	00894
C(1,IMAX) = 0.0	DEGA	00895
C(2,IMAX) = -1.0	DEGA	00896
C(3,IMAX) = 0.0	DEGA	00897
IF(ATOP(IMAX+1) .GE. RMAX) 64, 63	DEGA	00898
64 ATOP(IMAX+1) = RMAX	DEGA	00899
I2 = IMAX	DEGA	00900
RETURN	DEGA	00901
62 IMAX = 0	DEGA	00902
I2 = 0	DEGA	00903
IREG1 = 2	DEGA	00904
IREG2 = 1	DEGA	00905
OPAF = 1.0 + ATOPFAC	DEGA	00906
RROT = RMIN	DEGA	00907
IF(RR2 .GE. RMAX) 50, 51	DEGA	00908
50 RTOP = RMAX	DEGA	00909
IFINISH = 1	DEGA	00910
GO TO 37	DEGA	00911
51 RTOP = RR2	DEGA	00912
IFINISH = 0	DEGA	00913
37 DRTB = RTOP - RROT	DEGA	00914
R1 = .3*DRTB	DEGA	00915
R2 = .6*DRTB	DEGA	00916
CALL DRMAXSB(DRMAX, R1, R2, XLAMDIM, XLDIM, LMAX)	DEGA	00917
IF(DRTB .LE. DRMAX) GO TO 21	DEGA	00918
RTOP = RROT + DRMAX	DEGA	00919
IFINISH = 0	DEGA	00920
21 IF(IREG2 .EQ. 3) 60, 59	DEGA	00921
60 IF(I2 .EQ. 0) I2 = IMAX	DEGA	00922
IMAX = IMAX + 1	DEGA	00923
C(1,IMAX) = 0.0	DEGA	00924
C(2,IMAX) = -1.0	DEGA	00925
C(3,IMAX) = 0.0	DEGA	00926
GO TO 61	DEGA	00927
59 DR = RTOP - RROT	DEGA	00928
DDR = DR/9.0	DEGA	00929
IF(DR .LT. DRMIN) 22, 23	DEGA	00930
22 PRINT 3	DEGA	00931

```

3 FORMAT(* AN INTERVAL ALONG THE R-AXIS GOT TOO SMALL.*)
PRINT 4, DR, DRMIN
4 FORMAT(* DR = **,E20,10,*,*,10X,*DRMIN = **,E20,10)
IFRROR = 21
6 FORMAT (* IERROR = *, I5)
PRINT 6, IERROR
PRINT 2
GO TO 99
23 R1 = RROT $ R3 = RTOP $ R2 = (R1 + R3)/2.0
IF(R1 .LT. 1.0E-20) 33, 34
33 A(3,4) = 0.0 $ GO TO 35
34 A(3,4) = R1*R1*V(R1)
35 A(2,4) = R2*R2*V(R2)
A(1,4) = R3*R3*V(R3)
A(1,1) = R3*R3 $ A(2,1) = R2*R2 $ A(3,1) = R1*R1
A(1,2) = R3 $ A(2,2) = R2 $ A(3,2) = R1
A(1,3) = 1.0 $ A(2,3) = 1.0 $ A(3,3) = 1.0
CALL MATPAC(-1, 3, 1, DET, 0.0, IF SING)
CHAT1 = A(1,4) $ CHAT2 = A(2,4) $ CHAT3 = A(3,4)
C*****
C R=V IS NEVER USED. WE WILL ONLY BE WORKING WITH VS
C WHERE V(R) .NE. 0 FOR R .NE. 0.
C*****
R = RROT
DO 24 J=1, 8
R = R + DDR
VR = V(R)
P = CHAT1 + CHAT2/R + CHAT3/(R**P)
XJ = J
C 1 FORMAT(4E20,10)
C PRINT 1, XJ, R, VR, P
IF(ABS(VR - P)/VR) .LT. DVMAX) 24, 25
24 CONTINUE
C*****
IREG2 = IREG1
IMAX = IMAX + 1
C(1,IMAX) = CHAT1 $ C(2,IMAX) = CHAT2 $ C(3,IMAX) = CHAT3
61 ATOP(IMAX + 1) = RTOP
IF (.IFINISH .EQ. 1) 99, 31
31 IF(IMAX .LT. 40) 28, 32
32 PRINT 7
7 FORMAT(* THE MAXIMUM NUMBER OF INTERVALS WILL NOT SPAN (RMIN, RMAX
1).*)
IFRROR = 22
PRINT 6, IERROR
PRINT 2
GO TO 99
28 RROT = RTOP
IF(IREG1 .EQ. 3) 58, 57
58 DRTOP = ATOPFAC*RTOP
R1 = RTOP
R2 = RTOP + .5*(RTOP - ATOP(IMAX))
CALL DRMAXSB(DRMAX, R1, R2, XLAMDIM, XLDIM, LMAX)
IF(DRTOP .GT. DRMAX) DRTOP = DRMAX
RTOP = RTOP + DRTOP
GO TO 52
C*****
C THIS CARD DETERMINES MAXIMUM INTERVAL LENGTH.
C*****
57 IF(IMAX .EQ. 1) 40, 41

```

```

DEGA 00932
DEGA 00933
DEGA 00934
DEGA 00935
DEGA 00936
DEGA 00937
DEGA 00938
DEGA 00939
DEGA 00940
DEGA 00941
DEGA 00942
DEGA 00943
DEGA 00944
DEGA 00945
DEGA 00946
DEGA 00947
DEGA 00948
DEGA 00949
DEGA 00950
DEGA 00951
DEGA 00952
DEGA 00953
DEGA 00954
DEGA 00955
DEGA 00956
DEGA 00957
DEGA 00958
DEGA 00959
DEGA 00960
DEGA 00961
DEGA 00962
DEGA 00963
DEGA 00964
DEGA 00965
DEGA 00966
DEGA 00967
DEGA 00968
DEGA 00969
DEGA 00970
DEGA 00971
DEGA 00972
DEGA 00973
DEGA 00974
DEGA 00975
DEGA 00976
DEGA 00977
DEGA 00978
DEGA 00979
DEGA 00980
DEGA 00981
DEGA 00982
DEGA 00983
DEGA 00984
DEGA 00985
DEGA 00986
DEGA 00987
DEGA 00988
DEGA 00989
DEGA 00990
DEGA 00991
DEGA 00992
DEGA 00993

```

40	DR1 = DR	DEGA	00994
	DR2 = DR	DEGA	00995
	DR3 = DR	DEGA	00996
	DR4 = DR	DEGA	00997
	DRTMP = DR	DEGA	00998
	GO TO 42	DEGA	00999
41	DR5 = DR4	DEGA	01000
	DR4 = DR3	DEGA	01001
	DR3 = DR2	DEGA	01002
	DR2 = DR1	DEGA	01003
	DR1 = DR	DEGA	01004
	DRTMP = (DR1 + DR2 + DR3 + DR4 + DR5)/5.0	DEGA	01005
42	IF(2.0*DRTMP.LT.ATOPFAC*RTOP) 46, 47	DEGA	01006
46	RTOP = RTOP + 2.0*DRTMP	DEGA	01007
	GO TO 48	DEGA	01008
47	RTOP = OPAF*RTOP	DEGA	01009
48	R1 = RBOT	DEGA	01010
	R2 = RBOT + .5*(RBOT - ATOP(IMAX))	DEGA	01011
	CALL DRMAXSB(DRMAX, R1, R2, XLAMDIM, XLDIM, LMAX)	DEGA	01012
	IF(RTOP - RBOT.GT.DRMAX) RTOP = RBOT + DRMAX	DEGA	01013
	GO TO (54, 53), IREG1	DEGA	01014
54	IF(RTOP.GE. RR2) 55, 52	DEGA	01015
55	RTOP = RR2	DEGA	01016
	IREG1 = 2	DEGA	01017
	GO TO 52	DEGA	01018
53	IF(RTOP.GE. RR) 56, 52	DEGA	01019
56	RTOP = RR	DEGA	01020
	IREG1 = 3	DEGA	01021
C*****		DEGA	01022
52	IF(RMAX.LE. RTOP) 29, 26	DEGA	01023
29	RTOP = RMAX \$ IFINISH = 1	DEGA	01024
	GO TO 21	DEGA	01025
25	IPEG1 = IREG2	DEGA	01026
	IF(IMAX.EQ. 0) 43, 44	DEGA	01027
44	IF(DR.LT. .5*DRTMP) 43, 45	DEGA	01028
43	RTOP = R2	DEGA	01029
	GO TO 26	DEGA	01030
45	RTOP = RBOT + .75*DR	DEGA	01031
26	IFINISH = 0	DEGA	01032
	GO TO 21	DEGA	01033
99	IF(I2.EQ. 0) I2 = IMAX	DEGA	01034
	RETURN	DEGA	01035
	END	DEGA	01036
	SUBROUTINE DRMAXSB(DRMAX, R1, R2, XLAMDIM, XLDIM, LMAX)	DEGA	01037
	DIMENSION XLAMDIM(1), XLDIM(1)	DEGA	01038
	SMALLK = 0.0	DEGA	01039
	R12 = 1.0/R1**2	DEGA	01040
	R22 = 1.0/R2**2	DEGA	01041
	TVR1 = 2.0*V(R1)	DEGA	01042
	TVR2 = 2.0*V(R2)	DEGA	01043
	DO 70 L=1, LMAX	DEGA	01044
	SK = ABS(XLAMDIM(L) - TVR1 - XLDIM(L)*R12)	DEGA	01045
	IF(SK.GT. SMALLK) SMALLK = SK	DEGA	01046
	SK = ABS(XLAMDIM(L) - TVR2 - XLDIM(L)*R22)	DEGA	01047
	IF(SK.GT. SMALLK) SMALLK = SK	DEGA	01048
70	CONTINUE	DEGA	01049
	DRMAX = 6.28/SORT(SMALLK)	DEGA	01050
	RETURN	DEGA	01051
	END	DEGA	01052
	FUNCTION V1(R)	DEGA	01053
	COMMON/CB3/ Z, ZM1, RR2, RR, A0	DEGA	01054
	FNZZ = 1.0/(1.0 + A0*RR2)	DEGA	01055

FNZ = FNZZ**2*(2.0*A0*RR2*FNZZ + 1.0)*ZM1	DEGA	01056
A1 = RR/RR2*(ZM1/FNZ*FNZZ**2 - 1.0) + 1.5	DEGA	01057
VV = ZM1/RR	DEGA	01058
A0RR = A0*RR	DEGA	01059
FNZSRRM = -FNZ/RR	DEGA	01060
ENTRY V	DEGA	01061
IF(R .GT. RR) GO TO 3	DEGA	01062
X1 = R/RR	DEGA	01063
IF(R .GT. RR2) GO TO 2	DEGA	01064
V1 = VV/(X1*(1.0 + A0RR*X1)**2)	DEGA	01065
V1 = -V1 + FNZSRRM*(X1**2/2.0 - A1) - 1.0/R	DEGA	01066
RETURN	DEGA	01067
2 V1 = FNZSRRM*(1.0/X1 + X1**2/2.0 - 1.5) - 1.0/R	DEGA	01068
RETURN	DEGA	01069
3 V1 = -1.0/R	DEGA	01070
RETURN	DEGA	01071
END	DEGA	01072
SUBROUTINE MATPAC (IJOB, A, N, M, DET, EP, IF SING)	DEGA	01073
DIMENSION A(3, 4)	DEGA	01074
IF SING = 0	DEGA	01075
DET = 1.	DEGA	01076
NP1 = N+1	DEGA	01077
NPM = N+M	DEGA	01078
NM1 = N-1	DEGA	01079
IF(IJOB) 2, 1, 2	DEGA	01080
1 DO 3 I=1, N	DEGA	01081
NP1 = N+I	DEGA	01082
A(I, NP1) = 1.	DEGA	01083
IP1 = I+1	DEGA	01084
IF(N - IP1) 2, 19, 19	DEGA	01085
19 DO 3 J=IP1, N	DEGA	01086
NPJ = N+J	DEGA	01087
A(I, NPJ) = 0.	DEGA	01088
3 A(J, NP1) = 0.	DEGA	01089
2 DO 4 J=1, NM1	DEGA	01090
C = ABS(A(J, J))	DEGA	01091
JP1 = J+1	DEGA	01092
DO 5 I=JP1, N	DEGA	01093
D = ABS(A(I, J))	DEGA	01094
IF(C<D) 6, 5, 5	DEGA	01095
6 DET = -DET	DEGA	01096
DO 7 K=J, NPM	DEGA	01097
B = A(I, K)	DEGA	01098
A(I, K) = A(J, K)	DEGA	01099
7 A(J, K) = B	DEGA	01100
C = D	DEGA	01101
5 CONTINUE	DEGA	01102
IF(ABS(A(J, J))-EP) 14, 15, 15	DEGA	01103
14 DET = 0.	DEGA	01104
IF(IJOB) 16, 16, 17	DEGA	01105
16 IF SING = 1	DEGA	01106
17 RETURN	DEGA	01107
15 DO 4 I= JP1, N	DEGA	01108
CONST = A(I, J)/A(J, J)	DEGA	01109
DO 4 K= JP1, NPM	DEGA	01110
4 A(I, K) = A(I, K) - CONST*A(J, K)	DEGA	01111
IF(ABS(A(N, N)) - EP) 14, 18, 18	DEGA	01112
18 DO 11 I=1, N	DEGA	01113
11 DET = DET*A(I, I)	DEGA	01114
IF(IJOB) 10, 10, 17	DEGA	01115
10 DO 12 I=1, N	DEGA	01116
K = N-I+1	DEGA	01117

KP1 = K+1	DEGA	01118
DO 12 L=NP1, NPM	DEGA	01119
S = 0.	DEGA	01120
IF( N = KP1) 12, 20, 20	DEGA	01121
20 DO 13 J=KP1, N	DEGA	01122
13 S = S+A(K,J)*A(J,L)	DEGA	01123
12 A(K,L) = (A(K,L)-S)/A(K,K)	DEGA	01124
RETURN	DEGA	01125
END	DEGA	01126
SUBROUTINE NORMPHI(CAPLAMB, L)	DEGA	01127
COMMON/AIMAX/ AF(51,402), JMAXF(402), IMAX,	DEGA	01128
1 IMAXF, IMAXFP1, IMAXB, IMAXBPI, C(3,400),	DEGA	01129
2 CF(3,402), ATOP(401), ATOPF(402)	DEGA	01130
DIMENSION AB(51,402), JMAXB(402), CB(3,402), ATOPB(402)	DEGA	01131
EQUIVALENCE(AF(1), AB(1)), (JMAXF(1), JMAXB(1)),	DEGA	01132
1 (CF(1), CB(1)), (ATOPF(1), ATOPB(1))	DEGA	01133
DIMENSION CC(101)	DEGA	01134
COMMON/SCRATCH/SCRATCH(604)	DEGA	01135
EQUIVALENCE(SCRATCH(1), CC(1))	DEGA	01136
SQAB = SQRT(ABS(CAPLAMB))	DEGA	01137
IBKW = IMAXFP1 + 1	DEGA	01138
EXPSQ = EXP(-SQAB*ATOPB(IBKW))	DEGA	01139
CAPA = AB(1,IBKW)/EXPSQ	DEGA	01140
GUNDA2 = (CAPA*EXPSQ)**2/(2.0*SQAB)	DEGA	01141
D = ATOPF(2)	DEGA	01142
CALL POLYMUL(AF(1,1), JMAXF(1), AF(1,1), JMAXF(1), CC, NC)	DEGA	01143
L2 = 2*L	DEGA	01144
LD = L2 + NC	DEGA	01145
SUM = CC(NC)/LD	DEGA	01146
N = NC	DEGA	01147
NCM1 = NC - 1	DEGA	01148
DO 26 NN=1, NCM1	DEGA	01149
N = N - 1	DEGA	01150
LD = LD - 1	DEGA	01151
26 SUM = SUM*D + CC(N)/LD	DEGA	01152
GUNDA2 = GUNDA2 + SUM*D**(L2 + 1)	DEGA	01153
IF(IMAXF .EQ. 1) GO TO 27	DEGA	01154
DO 20 I=2, IMAXF	DEGA	01155
D = ATOPF(I+1) - ATOPF(I)	DEGA	01156
CALL POLYMUL(AF(1,I), JMAXF(I), AF(1,I), JMAXF(I), CC, NC)	DEGA	01157
SUM = CC(NC)/NC	DEGA	01158
N = NC	DEGA	01159
NCM1 = NC - 1	DEGA	01160
DO 21 NN=1, NCM1	DEGA	01161
N = N - 1	DEGA	01162
21 SUM = SUM*D + CC(N)/N	DEGA	01163
20 GUNDA2 = GUNDA2 + SUM*D	DEGA	01164
27 DO 22 I=1, IMAXB	DEGA	01165
IBKW = IMAXFP1 + I	DEGA	01166
D = ATOPB(IBKW + 1) - ATOPB(IBKW)	DEGA	01167
CALL POLYMUL(AB(1,IBKW), JMAXB(IBKW),	DEGA	01168
1 AB(1,IBKW), JMAXB(IBKW), CC, NC)	DEGA	01169
SUM = -CC(NC)/NC	DEGA	01170
N = NC	DEGA	01171
NCM1 = NC - 1	DEGA	01172
DO 23 NN=1, NCM1	DEGA	01173
N = N - 1	DEGA	01174
23 SUM = SUM*D - CC(N)/N	DEGA	01175
22 GUNDA2 = GUNDA2 + SUM*D	DEGA	01176
GUNDA = SQRT(GUNDA2)	DEGA	01177
DO 24 I=1, IMAXF	DEGA	01178
JJJ = JMAXF(I)	DEGA	01179

DO 24 J=1, JJJ	DEGA	01180
24 AF(J,I) = AF(J,I)/GUNDA	DEGA	01181
DO 25 I=1, IMAXB	DEGA	01182
IRKW = IM, XFP1 + I	DEGA	01183
JJJ = JMAXB(IRKW)	DEGA	01184
DO 25 J=1, JJJ	DEGA	01185
25 AR(J,IRKW) = AB(J,IRKW)/GUNDA	DEGA	01186
RETURN	DEGA	01187
END	DEGA	01188
SUBROUTINE POLYMUL (A, LM, B, LN, C, LL)	DEGA	01189
DIMENSION A(1), B(1), C(1)	DEGA	01190
LL = LM + LN - 1	DEGA	01191
MMIN = 1	DEGA	01192
DO 1 L=1, LL	DEGA	01193
IF (L.GT. LM) 3, 2	DEGA	01194
2 MMAX = L	DEGA	01195
3 IF (L.GT. LN) 5, 4	DEGA	01196
4 NMAXP1 = L + 1	DEGA	01197
GO TO 6	DEGA	01198
5 MMIN = MMIN + 1	DEGA	01199
6 C(L) = 0.0	DEGA	01200
N = NMAXP1	DEGA	01201
DO 1 M=MMIN, MMAX	DEGA	01202
N = N - 1	DEGA	01203
1 C(L) = C(L) + A(M)*B(N)	DEGA	01204
RETURN	DEGA	01205
END	DEGA	01206
SUBROUTINE MATELE(KPHIMAX, XLAMARY, LARY, ALFAARY, IMAXARY, I2)	DEGA	01207
1 NSUBSHL, ISSMAX, NSHELL	DEGA	01208
DIMENSION JMAX(400), AM1S(400), AM2S(400)	DEGA	01209
EQUIVALENCE (SCRATCH(1), ATOPI1), (SCRATCH(2), C1),	DEGA	01210
1 (SCRATCH(3), C2), (SCRATCH(4), C3), (SCRATCH(5), JMAX(1)),	DEGA	01211
2 (SCRATCH(405), AM1S(1)), (SCRATCH(805), AM2S(1)),	DEGA	01212
DIMENSION XLAMARY(1), LARY(1), ALFAARY(1), IMAXARY(1)	DEGA	01213
1 NSUBSHL(1), NSHELL(1)	DEGA	01214
COMMON/AIMAX/ A(51,100), CAPO(100), IFSTOP(100),	DEGA	01215
1 XLLP1(100), S1(300), S2(300), DVPOLY(70), SCRATCH(1204)	DEGA	01216
2 , A1M(400), A2M(400), AMT(51), AMS(51)	DEGA	01217
3 , A1N(400), A2N(400), ANT(51), ANS(51)	DEGA	01218
4 , ATOPP1M(400), ATOPP1N(400), JMAXM(400), JMAXN(400)	DEGA	01219
COMMON/CB1/ HNUVEC(500), ACOFVEC(500), NOPTS, IHNUMAX, MM(100),	DEGA	01220
1 NN(100), MNMAX, I4MIN, I4MAX	DEGA	01221
COMMON/EPSPCONV, FBMAX, EM16, EMATELE, EMDVDD, MAXDIM	DEGA	01222
1 FORMAT(16I5)	DEGA	01223
2 FORMAT (25I5)	DEGA	01224
3 FORMAT(5E25.14)	DEGA	01225
4 FORMAT(2E25.14, I5)	DEGA	01226
5 FORMAT(/)	DEGA	01227
6 FORMAT (4E25.14, 2I5)	DEGA	01228
PRINT1, MNMAX	DEGA	01229
PRINT 1, (MM(MN), NN(MN), MN=1, MNMAX)	DEGA	01230
KPHIM4 = KPHIMAX + 4	DEGA	01231
MAXDIM4 = MAXDIM + 4	DEGA	01232
ISKIP = 3*MAXDIM + 4	DEGA	01233
IECS = 1	DEGA	01234
IECS1 = IECS + MAXDIM4	DEGA	01235
IECS2 = IECS1 + MAXDIM	DEGA	01236
CALL ECRD (SCRATCH(1), IECS, KPHIM4, IE)	DEGA	01237
CALL ECRD(SCRATCH(405), IECS1, KPHIMAX, IE)	DEGA	01238
CALL ECRD(SCRATCH(805), IECS2, KPHIMAX, IE)	DEGA	01239
DO 31 KPHI=1, KPHIMAX	DEGA	01240
A(1,KPHI) = AM1S(KPHI)	DEGA	01241

31	A(2,KPHI) = A42S(KPHI)	DEGA	01242
	DO 21 KPHI=1, KPHIMAX	DEGA	01243
	XL = LARY(KPHI)	DEGA	01244
	IERROR = 0	DEGA	01245
	CALL TAYLOR1 (A(1,KPHI), JMAX(KPHI), ATOPI1,	DEGA	01246
1	XLAMARY(KPHI), XL, C1, C2, EM16, IERROR)	DEGA	01247
	JMAXKP = JMAX(KPHI)	DEGA	01248
21	XLLP1(KPHI) = XL*(XL + 1.0)	DEGA	01249
	DO 20 MN=1, MNMAX	DEGA	01250
	M = MM(MN)	DEGA	01251
	N = NN(MN)	DEGA	01252
	LNLMI = LARY(N) + LARY(M) + 1	DEGA	01253
	DEOM = LNLMI - 1	DEGA	01254
	CALL POLYMUL (A(2,M), JMAX(M)-1, A(2,N), JMAX(N)-1, S2, IS2)	DEGA	01255
	DO 30 J=1, IS2	DEGA	01256
	DEOM = DEOM + 1.0	DEGA	01257
30	S2(J) = S2(J)/DEOM	DEGA	01258
	CALL POLYOP (S2, IS2, ATOPI1, P)	DEGA	01259
20	CAPO(MN) = -C2*P*ATOPI1**LNLMI	DEGA	01260
	DO 36 MN=1, MNMAX	DEGA	01261
36	IFSTOP(MN) = 0	DEGA	01262
	DO 24 I=2, I2	DEGA	01263
	ATOPI = ATOPI1	DEGA	01264
	IFCS = IECS + ISKIP	DEGA	01265
	IECS1 = IECS + MAXDIM4	DEGA	01266
	IFCS2 = IECS1 + MAXDIM	DEGA	01267
	CALL ECRD (SCRATCH(1), IECS, KPHIM4, IE)	DEGA	01268
	CALL ECRD(SCRATCH(405), IECS1, KPHIMAX, IE)	DEGA	01269
	CALL ECRD(SCRATCH(805), IECS2, KPHIMAX, IE)	DEGA	01270
	DO 25 KPHI=1, KPHIMAX	DEGA	01271
	A(1, KPHI) = AM1S(KPHI)	DEGA	01272
25	A(2, KPHI) = AM2S(KPHI)	DEGA	01273
	D = ATOPI - ATOPI1	DEGA	01274
	DO 27 KPHI=1, KPHIMAX	DEGA	01275
	IERROR = 0	DEGA	01276
27	CALL TAYLORS(A(1,KPHI), JMAX(KPHI), ATOPI1,	DEGA	01277
1	D, XLAMARY(KPHI), XLLP1(KPHI), C1, C2, C3, EM16, IERROR)	DEGA	01278
	CALL DVDD(ATOPI1, D, C2, C3, DVPOLY, IDVDD, EMDVDD)	DEGA	01279
	MLAST = 0	DEGA	01280
	DO 28 MN=1, MNMAX	DEGA	01281
	IF(IFSTOP(MN) .EQ. 5) GO TO 28	DEGA	01282
	M = MM(MN)	DEGA	01283
	N = NN(MN)	DEGA	01284
	IF(M .EQ. MLAST) GO TO 33	DEGA	01285
	MLAST = M	DEGA	01286
	CALL POLYMUL(A(1,M), JMAX(M), DVPOLY, IDVDD, S1, IS1)	DEGA	01287
33	CALL POLYMUL(A(1,N), JMAX(N), S1, IS1, S2, IS2)	DEGA	01288
	CALL POLYINT(S2, IS2, D, P)	DEGA	01289
	CAPO(MN) = CAPO(MN) - P	DEGA	01290
	IF(ABS(P) .LT. ABS(CAPO(MN))*EMATELE) 34, 35	DEGA	01291
34	IFSTOP(MN) = IFSTOP(MN) + 1	DEGA	01292
	GO TO 28	DEGA	01293
35	IFSTOP(MN) = 0	DEGA	01294
28	CONTINUE	DEGA	01295
24	CONTINUE	DEGA	01296
	ATOPI2 = ATOPI1	DEGA	01297
	ISKIP = 3*MAXDIM + 4	DEGA	01298
	IJUMP = 400 - I2	DEGA	01299
	MLAST = 0	DEGA	01300
	NLAST = 0	DEGA	01301
	DO 40 MN=1, MNMAX	DEGA	01302
	IF(IFSTOP(MN) .EQ. 5) GO TO 40	DEGA	01303

M = MM(MN)	DEGA	01304
N = NN(MN)	DEGA	01305
IF(M .EQ. MLAST) 42, 41	DEGA	01306
41 MLAST = M	DEGA	01307
IMMAX = IMAXARY(M) - I2	DEGA	01308
IF(IMMAX .EQ. 0) GO TO 60	DEGA	01309
IECS = 4*(400 - I2)	DEGA	01310
IECS = IECS*(M - 1) + 1	DEGA	01311
IECS = IECS + I2*ISKIP	DEGA	01312
CALL ECRD(ATOPPI1(M), IECS, IMMAX, IE)	DEGA	01313
IECS = IECS + IJUMP	DEGA	01314
CALL ECRD(JMAXM(1), IECS, IMMAX, IE)	DEGA	01315
IECS = IECS + IJUMP	DEGA	01316
CALL ECRD(A1M(1), IECS, IMMAX, IE)	DEGA	01317
IECS = IECS + IJUMP	DEGA	01318
CALL ECRD(A2M(1), IECS, IMMAX, IE)	DEGA	01319
60 CONTINUE	DEGA	01320
42 IF(N .EQ. NLAST) 45, 44	DEGA	01321
44 NLAST = N	DEGA	01322
INMAX = IMAXARY(N) - I2	DEGA	01323
IF(INMAX .EQ. 0) GO TO 61	DEGA	01324
IECS = 4*(400 - I2)	DEGA	01325
IECS = IECS*(N - 1) + 1	DEGA	01326
IECS = IECS + I2*ISKIP	DEGA	01327
CALL ECRD(ATOPPI1(N), IECS, INMAX, IE)	DEGA	01328
IECS = IECS + IJUMP	DEGA	01329
CALL ECRD(JMAXN(1), IECS, INMAX, IE)	DEGA	01330
IECS = IECS + IJUMP	DEGA	01331
CALL ECRD(A1N(1), IECS, INMAX, IE)	DEGA	01332
IECS = IECS + IJUMP	DEGA	01333
CALL ECRD(A2N(1), IECS, INMAX, IE)	DEGA	01334
61 CONTINUE	DEGA	01335
45 IM = 1	DEGA	01336
IN = 1	DEGA	01337
ATOPI1 = ATOPI2	DEGA	01338
ALASTM = ATOPI2	DEGA	01339
ALASTN = ATOPI2	DEGA	01340
IMFIRST = 0	DEGA	01341
INFIRST = 0	DEGA	01342
CALL FINDATP(IM, IMMAX, XLLPI(M), ALASTM, ATOPI1,	DEGA	01343
1 XLAMARY(M), IERROR)	DEGA	01344
IF(IERROR .EQ. 14) GO TO 100	DEGA	01345
CALL FINDATP(IN, INMAX, XLLPI(N), ALASTN, ATOPI1,	DEGA	01346
1 XLAMARY(N), IERROR)	DEGA	01347
IF(IERROR .EQ. 14) GO TO 100	DEGA	01348
47 CONTINUE	DEGA	01349
ATOPI1 = ATOPI1	DEGA	01350
IF(ABS(ALASTM-ALASTN) .LT. (ALASTM+ALASTN)*.5E-12 ) 48, 49	DEGA	01351
48 INFIRST = 0	DEGA	01352
IMFIRST = 0	DEGA	01353
ATOPI1 = .5*(ALASTM + ALASTN)	DEGA	01354
CALL FINDA12(IM, IMMAX, XLAMARY(M), ALFAARY(M), XLLPI(M),	DEGA	01355
1 ATOPI1, A1M, A2M, JMAXM, AMS, JMSMAX, IERROR)	DEGA	01356
IF(IERROR .EQ. 12) GO TO 100	DEGA	01357
CALL FINDA12(IN, INMAX, XLAMARY(N), ALFAARY(N), XLLPI(N),	DEGA	01358
1 ATOPI1, A1N, A2N, JMAXN, ANS, JNSMAX, IERROR)	DEGA	01359
IF(IERROR .EQ. 12) GO TO 100	DEGA	01360
IM = IM + 1	DEGA	01361
IN = IN + 1	DEGA	01362
CALL FINDATP(IM, IMMAX, XLLPI(M), ALASTM, ATOPI1,	DEGA	01363
1 XLAMARY(M), IERROR)	DEGA	01364
IF(IERROR .EQ. 14) GO TO 100	DEGA	01365



CALL FINDATP(IN, INMAX, XLLP1(N), ALASTN, ATOPPI1,	DEGA	01366
1 XLAMARY(N), IERROR)	DEGA	01367
IF(IERROR.EQ. 14) 100, 50	DEGA	01368
49 IF(ALASTM.GT. ALASTN) 51, 52	DEGA	01369
51 IMFIRST = 0	DEGA	01370
IMFIRST = IMFIRST + 1	DEGA	01371
ATOPI1 = ALASTN	DEGA	01372
DM = ATOPI1 - ALASTM	DEGA	01373
IF(IMFIRST.EQ. 1) 53, 54	DEGA	01374
53 CALL FINDA12(IM, IMMAX, XLAMARY(M), ALFAARY(M),	DEGA	01375
1 XLLP1(M), ALASTM, A1M, A2M, JMAXM, AMT, JMTMAX, IERROR)	DEGA	01376
IF(IERROR.EQ. 12) GO TO 100	DEGA	01377
IERROR = 0	DEGA	01378
CALL TAYLORS(AMT, JMTMAX, ALASTM, DM, XLAMARY(M),	DEGA	01379
1 XLLP1(M), 0.0, -1.0, 0.0, EM16, IERROR)	DEGA	01380
IF(IERROR.EQ. 10) 100, 54	DEGA	01381
54 CALL POLYOP(AMT, JMTMAX, DM, AMS(1))	DEGA	01382
CALL POLY1P(AMT, JMTMAX, DM, AMS(2))	DEGA	01383
JMSMAX = 0	DEGA	01384
CALL FINDA12(IN, INMAX, XLAMARY(N), ALFAARY(N), XLLP1(N),	DEGA	01385
1 ATOPI1, A1N, A2N, JMAXN, ANS, JNSMAX, IERROR)	DEGA	01386
IF(IERROR.EQ. 12) GO TO 100	DEGA	01387
IN = IN + 1	DEGA	01388
CALL FINDATP(IN, INMAX, XLLP1(N), ALASTN, ATOPPI1,	DEGA	01389
1 XLAMARY(N), IERROR)	DEGA	01390
IF(IERROR.EQ. 14) 100, 50	DEGA	01391
52 IMFIRST = 0	DEGA	01392
IMFIRST = IMFIRST + 1	DEGA	01393
ATOPI1 = ALASTM	DEGA	01394
DN = ATOPI1 - ALASTN	DEGA	01395
IF(IMFIRST.EQ. 1) 55, 56	DEGA	01396
55 CALL FINDA12(IN, INMAX, XLAMARY(N), ALFAARY(N),	DEGA	01397
1 XLLP1(N), ALASTN, A1N, A2N, JMAXN, ANT, JNTMAX, IERROR)	DEGA	01398
IF(IERROR.EQ. 12) GO TO 100	DEGA	01399
IERROR = 0	DEGA	01400
CALL TAYLORS(ANT, JNTMAX, ALASTN, DN, XLAMARY(N),	DEGA	01401
1 XLLP1(N), 0.0, -1.0, 0.0, EM16, IERROR)	DEGA	01402
IF(IERROR.EQ. 10) 100, 56	DEGA	01403
56 CALL POLYOP(ANT, JNTMAX, DN, ANS(1))	DEGA	01404
CALL POLY1P(ANT, JNTMAX, DN, ANS(2))	DEGA	01405
JNSMAX = 0	DEGA	01406
CALL FINDA12(IM, IMMAX, XLAMARY(M), ALFAARY(M), XLLP1(M),	DEGA	01407
1 ATOPI1, A1M, A2M, JMAXM, AMS, JMSMAX, IERROR)	DEGA	01408
IF(IERROR.EQ. 12) GO TO 100	DEGA	01409
IM = IM + 1	DEGA	01410
CALL FINDATP(IM, IMMAX, XLLP1(M), ALASTM,	DEGA	01411
1 ATOPPI1, XLAMARY(M), IERROR)	DEGA	01412
IF(IERROR.EQ. 14) 100, 50	DEGA	01413
50 D = ATOPI1 - ATOPI1	DEGA	01414
IERROR = 0	DEGA	01415
CALL TAYLORS(AMS, JMSMAX, ATOPI1, D, XLAMARY(M),	DEGA	01416
1 XLLP1(M), 0.0, -1.0, 0.0, EM16, IERROR)	DEGA	01417
IF(IERROR.EQ. 10) GO TO 100	DEGA	01418
CALL TAYLORS(ANS, JNSMAX, ATOPI1, D, XLAMARY(N),	DEGA	01419
1 XLLP1(N), 0.0, -1.0, 0.0, EM16, IERROR)	DEGA	01420
IF(IERROR.EQ. 10) GO TO 100	DEGA	01421
CALL DVDD(ATOPPI1, D, -1.0, 0.0, DVPOLY, IDVDD, EMDVDD)	DEGA	01422
CALL POLYMUL(AMS, JMSMAX, DVPOLY, IDVDD, S1, IS1)	DEGA	01423
CALL POLYMUL(ANS, JNSMAX, S1, IS1, S2, IS2)	DEGA	01424
CALL POLYINT(S2, IS2, D, P)	DEGA	01425
CAPO(MN) = CAPO(MN) - P	DEGA	01426
IF(ABS(P) .LT. ABS(CAPO(MN))*EMATELE) 58, 57	DEGA	01427

PRIN! 7, ISSMAX

DEGA 00003

58	IFSTOP(MN) = IFSTOP(MN) + 1	DEGA	01428
	IF(IFSTOP(MN) .EQ. 5) 40, 47	DEGA	01429
57	IFSTOP(MN) = 0	DEGA	01430
	GO TO 47	DEGA	01431
100	PRINT 101	DEGA	01432
101	FORMAT(* ERROR IN REGION 3*)	DEGA	01433
40	CONTINUE	DEGA	01434
	DO 29 MN=1, MNMAX	DEGA	01435
	CAPO(MN) = 2.0*CAPO(MN)	DEGA	01436
29	PRINT 3, CAPO(MN)	DEGA	01437
64	FORMAT(E24.14, * EV*, I5, E24.14, * EV*, I5,	DEGA	01438
	1 E24.14, * EV*, E24.14, * BARN/ATOM*)	DEGA	01439
65	FORMAT(E24.14, * BARN/ATOM*)	DEGA	01440
66	FORMAT(/)	DEGA	01441
	DO 62 MN=1, MNMAX	DEGA	01442
	M = MM(MN)	DEGA	01443
	N = NN(MN)	DEGA	01444
	DE = ABS(XLAMARY(M) - XLAMARY(N))	DEGA	01445
	HNU = DE*13.605	DEGA	01446
	DE3 = DE**3	DEGA	01447
	LMAX = LARY(M)	DEGA	01448
	IF(LARY(N) .GT. LMAX) LMAX = LARY(N)	DEGA	01449
	THE GA AND NOE ARE ONLY GOOD HERE FOR THE BOUND-FREE CASE.	DEGA	01450
	MNSS = N	DEGA	01451
	IF(XLAMARY(M) .LT. 0.0) MNSS = M	DEGA	01452
	ISS = NSHELL(MNSS)*(NSHELL(MNSS) - 1)/2 + LARY(MNSS) + 1	DEGA	01453
	NOE = NSURSHL(ISS)	DEGA	01454
	GA = 2*(2*LARY(MNSS) + 1)	DEGA	01455
	DSIGMA = 10.756E+6*NOE*LMAX*CAPO(MN)**2/(DE3*GA)	DEGA	01456
	I4 = I4MIN + MN - 1	DEGA	01457
	ACOFVEC(I4) = ACOFVEC(I4) + DSIGMA	DEGA	01458
	EM = XLAMARY(M)*13.605	DEGA	01459
	EN = XLAMARY(N)*13.605	DEGA	01460
62	CONTINUE	DEGA	01461
	RETURN	DEGA	01462
	END	DEGA	01463
	SUBROUTINE ZZZ(XLAMWKB, LWKB, NWKB, KWKB, HNUVEC,	DEGA	01464
	1 NOPTS, IHNUMAX, DHNUI, ZZZFAC)	DEGA	01465
	COMMON/SCRATCH/SCRATCH(1204)	DEGA	01466
	DIMENSION XLAMWKB(1), LWKB(1), NWKB(1), HNUVEC(1)	DEGA	01467
	FAC = ALOG(10.0)	DEGA	01468
	DHNU = DHNUI	DEGA	01469
	DO 6 K=1, KWKB	DEGA	01470
	6 XLAMWKB(K) = ABS(XLAMWKB(K))	DEGA	01471
	CALL SORT1(KWKB, 2, XLAMWKB, SCRATCH, LWKB, NWKB,	DEGA	01472
	XLAMWKB(KWKB+1) = ZZZFAC*XLAMWKB(KWKB)	DEGA	01473
	3 IHNUMAX = 0	DEGA	01474
	DO 1 I=1, KWKB	DEGA	01475
	EPS = .01	DEGA	01476
	XNUM = ALOG10(ABS(XLAMWKB(I+1))) - ALOG10(ABS(XLAMWKB(I) + EPS))	DEGA	01477
	NOINT = XNUM/DHNU	DEGA	01478
	IF(NOINT .EQ. 0) NOINT = 1	DEGA	01479
	NOINT = NOINT + 1	DEGA	01480
	SDHNU = XNUM/NOINT	DEGA	01481
	IHNUMAX = IHNUMAX + 1	DEGA	01482
	IF(IHNUMAX .GT. NOPTS) 4, 5	DEGA	01483
	4 DHNU = 1.2*DHNU	DEGA	01484
	GO TO 3	DEGA	01485
	5 HNUVEC(IHNUMAX) = XLAMWKB(I) + EPS	DEGA	01486
	NOINTM1 = NOINT - 1	DEGA	01487
	DO 2 INT=1, NOINTM1	DEGA	01488
	IHNUMAX = IHNUMAX + 1	DEGA	01489

```

IF(IHNUMAX .GT. NOPTS) GO TO 4
XNUM = FAC*(ALOG10(ABS(XLAMWKB(I) + EPS)) + INT*SDHNU)
2 HNUVEC(IHNUMAX) = EXP(XNUM)
IHNUMAX = IHNUMAX + 1
IF(IHNUMAX .GT. NOPTS) GO TO 4
1 HNUVEC(IHNUMAX) = XLAMWKB(I+1)
DO 7 K=1, KWKB
7 XLAMWKB(K) = -XLAMWKB(K)
RETURN
END
SUBROUTINE CARSON(XLAMBD, C2, MFAC, CAPR, XLLP1,
1 CAPF, CAPFP, THETA, THETAP, IERROR)
COMMON/PI/PI, TWOSPI
IERROR = 0
SMALLC = ABS(C2)
SMALLK = SQRT(XLAMRDA)
CAPM = SQRT(TWOSPI/SMALLK)
CAPM = MFAC*CAPM
AN = CAPM $ BN = 0.0
CSK = SMALLC/SMALLK $ CSK2 = CSK*CSK
ACON = XLLP1 + CSK2 $ BCON = -(CSK2 + XLLP1)
DEOM = 1.0 $ TWOK = 2.0*SMALLK
CAPA = AN $ CAPB = BN $ CAPAP = 0.0 $ CAPBP = 0.0
DO 63 NP1 = 1, 15
DEOM = DEOM*CAPR
N = NP1 - 1
XNP1 = NP1
XNNP1 = N*XNP1
TWNOP1 = (2.0*N + 1.0)*CSK
XDEOM = TWOK*XNP1
ANP1 = ((ACON - XNNP1)*BN - TWONP1*AN)/XDEOM
BNP1 = ((XNNP1 + BCON)*AN - TWONP1*BN)/XDEOM
AADD = ANP1/DEOM $ BADD = BNP1/DEOM
APADD = XNP1*AADD $ BPADD = XNP1*BADD
CAPA = CAPA + AADD $ CAPB = CAPB + BADD
CAPAP = CAPAP + APADD $ CAPBP = CAPBP + BPADD
CAPAB = ABS(CAPA) + ABS(CAPB)
CAPABP = ABS(CAPAP) + ABS(CAPBP)
IF( ABS(AADD) .LT. CAPAB*1.0E-4 .AND.
1 ABS(BADD) .LT. CAPAB*1.0E-4 .AND.
2 ABS(APADD) .LT. CAPABP*1.0E-4 .AND.
3 ABS(BPADD) .LT. CAPABP*1.0E-4) GO TO 62
AN = ANP1
63 BN = BNP1
PRINT 64
64 FORMAT(* CAPA, CAPB, CAPAP, AND CAPBP DO NOT CONVERGE.*)
IERROR = 12
RETURN
C*****
C RUN THIS CODE FOR CAPR .GT. 1.0
C*****
62 CAPAP = -CAPAP/CAPR
CAPBP = -CAPBP/CAPR
CAPF2 = CAPA*CAPA + CAPB*CAPB
CAPF = SQRT(CAPF2)
CAPFP = (CAPA*CAPAP + CAPB*CAPBP)/CAPF
THETA = SMALLK*CAPR + CSK*ALOG(CAPR) + ATAN2(CAPA, CAPB)
THETAP = SMALLK + CSK/CAPR + (CAPAP*CAPB - CAPBP*CAPA)/CAPF2
RETURN
END
SUBROUTINE TAYLOR1(A, JMAX, D, XLAMBD, XL, C1, C2, EM16, IERROR)
DIMENSION A(1)
PRINT 7, ISSMAX

```

```

DEGA 01490
DEGA 01491
DEGA 01492
DEGA 01493
DEGA 01494
DEGA 01495
DEGA 01496
DEGA 01497
DEGA 01498
DEGA 01499
DEGA 01500
DEGA 01501
DEGA 01502
DEGA 01503
DEGA 01504
DEGA 01505
DEGA 01506
DEGA 01507
DEGA 01508
DEGA 01509
DEGA 01510
DEGA 01511
DEGA 01512
DEGA 01513
DEGA 01514
DEGA 01515
DEGA 01516
DEGA 01517
DEGA 01518
DEGA 01519
DEGA 01520
DEGA 01521
DEGA 01522
DEGA 01523
DEGA 01524
DEGA 01525
DEGA 01526
DEGA 01527
DEGA 01528
DEGA 01529
DEGA 01530
DEGA 01531
DEGA 01532
DEGA 01533
DEGA 01534
DEGA 01535
DEGA 01536
DEGA 01537
DEGA 01538
DEGA 01539
DEGA 01540
DEGA 01541
DEGA 01542
DEGA 01543
DEGA 01544
DEGA 01545
DEGA 01546
DEGA 01547
DEGA 01548
DEGA 01549
DEGA 01550
DEGA 01551
DEGA
DEGA

```

IERROR = 0	DEGA	01552
DEOM = 0.0	DEGA	01553
DDEOM = 2.0*XL	DEGA	01554
B1 = 2.0*C1 - XLAMBDA	DEGA	01555
B2 = 2.0*C2	DEGA	01556
IF(JMAX .GT. 2) GO TO 40	DEGA	01557
IFCONV = 0	DEGA	01558
FAC = 1.0	DEGA	01559
AMAX = A(2)	DEGA	01560
AMAXM16 = AMAX*EM16	DEGA	01561
DO 20 J=3, 50	DEGA	01562
DDEOM = DDEOM + 2.0	DEGA	01563
DEOM = DEOM + DDEOM	DEGA	01564
JM1 = J-1	DEGA	01565
A(J) = (B1*A(J-2) + B2*A(JM1))/DEOM	DEGA	01566
FAC = FAC*D	DEGA	01567
ARSA = JM1*ABS(A(J))*FAC	DEGA	01568
IF(ARSA .GT. AMAX) 30, 31	DEGA	01569
30 AMAX = ARSA	DEGA	01570
AMAXM16 = AMAX*EM16	DEGA	01571
GO TO 32	DEGA	01572
31 IF(ARSA .LT. AMAXM16) 33, 32	DEGA	01573
32 IFCONV = 0	DEGA	01574
GO TO 20	DEGA	01575
33 IF(IFCONV .EQ. 1) 35, 34	DEGA	01576
35 JMAX = J	DEGA	01577
RETURN	DEGA	01578
34 IFCONV = 1	DEGA	01579
20 CONTINUE	DEGA	01580
IERROR = 9	DEGA	01581
PRINT 9, IERROR	DEGA	01582
9 FORMAT (15, * TAYLOR SERIES DOES NOT CONVERGE IN SUBROUTINE TAYLOR1	DEGA	01583
1. DO-LOOP 20.*)	DEGA	01584
RETURN	DEGA	01585
40 DO 41 J=3, JMAX	DEGA	01586
DDEOM = DDEOM + 2.0	DEGA	01587
DEOM = DEOM + DDEOM	DEGA	01588
JM1 = J-1	DEGA	01589
41 A(J) = (B1*A(J-2) + B2*A(JM1))/DEOM	DEGA	01590
RETURN	DEGA	01591
END	DEGA	01592
SUBROUTINE TAYLORS(A, JMAX, RR, D, XLAMBDA,	DEGA	01593
1 XLLP1, C1, C2, C3, EM16, IERROR)	DEGA	01594
DIMENSION A(1)	DEGA	01595
IERROR = 0	DEGA	01596
RR2 = RR*RR	DEGA	01597
TWORR = 2.0*RR	DEGA	01598
SB1 = XLAMBDA - 2.0*C1	DEGA	01599
SB2 = -2.0*C2	DEGA	01600
SB3 = -(2.0*C3 + XLLP1)	DEGA	01601
B1 = SB1*RR2 + SB2*RR + SB3	DEGA	01602
B2 = SB1*TWORR + SB2	DEGA	01603
B3 = SB1	DEGA	01604
A(3) = -B1*A(1) / (2.0*RR2)	DEGA	01605
A(4) = -(B2*A(1) + B1*A(2) + 2.0*TWORR*A(3)) / (6.0*RR2)	DEGA	01606
IF(JMAX .GT. 5) GO TO 60	DEGA	01607
ABSD = ABS(D)	DEGA	01608
FAC = 1.0	DEGA	01609
AMAX = ABS(A(2))	DEGA	01610
AMAXM16 = AMAX*EM16	DEGA	01611
FAC = FAC*ABSD	DEGA	01612
ABS3 = 2.0*ABS(A(3))*FAC	DEGA	01613

IF(ABS3 .GT. AMAX) 54, 55	DEGA	01614
54 AMAX = ABS3	DEGA	01615
AMAXM16 = AMAX*EM16	DEGA	01616
55 FAC = FAC*ABSD	DEGA	01617
ARS4 = 3.0*ABS(A(4))*FAC	DEGA	01618
IF(ABS4 .GT. AMAX) 56, 57	DEGA	01619
56 AMAX = ABS4	DEGA	01620
AMAXM16 = AMAX*EM16	DEGA	01621
57 DO 26 J=6, 50, 2	DEGA	01622
JM1=J-1 \$ JM2 = J-2 \$ JM3 = J-3	DEGA	01623
JM4 = J-4 \$ JM5 = J-5	DEGA	01624
DEOM = JM3*JM2*RR2	DEGA	01625
A(JM1) = -(B3*A(JM5) + B2*A(JM4) + (B1+JM5*JM4)*A(JM3)	DEGA	01626
1 +JM4*JM3*TWORR*A(JM2) )/DEOM	DEGA	01627
DEOM = JM2*JM1*RR2	DEGA	01628
A(J) = -(B3*A(JM4) + B2*A(JM3) + (B1+JM4*JM3)*A(JM2)	DEGA	01629
1 +JM3*JM2*TWORR*A(JM1) )/DEOM	DEGA	01630
FAC = FAC*ABSD	DEGA	01631
ARSA1 = JM2*ABS(A(JM1))*FAC	DEGA	01632
FAC = FAC*ABSD	DEGA	01633
ARSA = JM1*ABS(A(J))*FAC	DEGA	01634
IF(ABSA .GT. AMAX) 45, 44	DEGA	01635
45 AMAX = ABSA	DEGA	01636
AMAXM16 = AMAX*EM16	DEGA	01637
IF(ABSA1 .GT. AMAX) 46, 26	DEGA	01638
46 AMAX = ABSA1 \$ AMAXM16 = AMAX*EM16 \$ GO TO 26	DEGA	01639
44 IF(ABSA1 .GT. AMAX) 46, 47	DEGA	01640
47 IF(ABSA .LT. AMAXM16) 48, 26	DEGA	01641
48 IF(ABSA1 .LT. AMAXM16) 49, 26	DEGA	01642
49 JMAX = J	DEGA	01643
RETURN	DEGA	01644
26 CONTINUE	DEGA	01645
IERROR = 10	DEGA	01646
PRINT 10, IERROR, I	DEGA	01647
10 FORMAT (215, * TAYLOR SERIES DOES NOT CONVERGE IN SUBROUTINE TAYLOR	DEGA	01648
15, DO-LOOP 26 IN LOOP 25.*)	DEGA	01649
JMAX = 50	DEGA	01650
IERROR = 0	DEGA	01651
1 FORMAT(4E24,14)	DEGA	01652
PRINT 1, AMAX, AMAXM16, ABSA, ABSA1	DEGA	01653
RETURN	DEGA	01654
60 DO 27 J=6, JMAX, 2	DEGA	01655
JM1=J-1 \$ JM2 = J-2 \$ JM3 = J-3	DEGA	01656
JM4 = J-4 \$ JM5 = J-5	DEGA	01657
DEOM = JM3*JM2*RR2	DEGA	01658
A(JM1) = -(B3*A(JM5) + B2*A(JM4) + (B1+JM5*JM4)*A(JM3)	DEGA	01659
1 +JM4*JM3*TWORR*A(JM2) )/DEOM	DEGA	01660
DEOM = JM2*JM1*RR2	DEGA	01661
27 A(J) = -(B3*A(JM4) + B2*A(JM3) + (B1+JM4*JM3)*A(JM2)	DEGA	01662
1 +JM3*JM2*TWORR*A(JM1) )/DEOM	DEGA	01663
RETURN	DEGA	01664
END	DEGA	01665
SUBROUTINE POLYOP( A, JMAX, D, P)	DEGA	01666
DIMENSION A(1)	DEGA	01667
M = JMAX	DEGA	01668
P = A(M)	DEGA	01669
JUP = M - 1	DEGA	01670
IF(JUP .LT. 1) RETURN	DEGA	01671
DO 1 J=1, JUP	DEGA	01672
M = M - 1	DEGA	01673
1 P = P*D + A(M)	DEGA	01674
RETURN	DEGA	01675
PRINT 7, ISSMAX	DEGA	0006J

ENTRY POLYIP	DEGA	01676
M = JMAX	DEGA	01677
P = (M-1)*A(M)	DEGA	01678
JIP = M-2	DEGA	01679
IF(JUP .LT. 1) RETURN	DEGA	01680
DO 2 J=1, JUP	DEGA	01681
M = M-1	DEGA	01682
2 P = P*D + (M-1)*A(M)	DEGA	01683
RETURN	DEGA	01684
ENTRY POLYINT	DEGA	01685
M = JMAX	DEGA	01686
P = A(M)/M	DEGA	01687
JIP = M-1	DEGA	01688
IF(JUP .LT. 1) GO TO 4	DEGA	01689
DO 3 J=1, JUP	DEGA	01690
M = M-1	DEGA	01691
3 P = P*D + A(M)/M	DEGA	01692
4 P = P*D	DEGA	01693
RETURN	DEGA	01694
END	DEGA	01695
SUBROUTINE DVDD( ATOPI, D, C2, C3, DVPOLY, IDVDD, EMDVDD)	DEGA	01696
COMMON/SCRATCH/SCRATCH(604)	DEGA	01697
DIMENSION DVPOLY(1)	DEGA	01698
OATOPI = -1.0/ATOPI	DEGA	01699
DN = 1.0	DEGA	01700
IF(C3 .EQ. 0.0) 10, 20	DEGA	01701
10 IDVDD = 1	DEGA	01702
DVPOLY(1) = -C2/ATOPI**2	DEGA	01703
EPSILON = ABS(DVPOLY(1))*EMDVDD	DEGA	01704
TERMLST = DVPOLY(1)	DEGA	01705
30 IDVDD = IDVDD + 1	DEGA	01706
TERMLST = TERMLST*OATOPI	DEGA	01707
DN = DN*D	DEGA	01708
DVPOLY(IDVDD) = IDVDD*TERMLST	DEGA	01709
IF(ABS(DVPOLY(IDVDD)*DN) .LT. EPSILON) 40, 30	DEGA	01710
20 ISCH = 1	DEGA	01711
ICNT1 = 1	DEGA	01712
ICNT2 = 1	DEGA	01713
SCRATCH(1) = -1.0/ATOPI**3	DEGA	01714
EPSILON = ABS(SCRATCH(1))*EMDVDD	DEGA	01715
TERMLST = SCRATCH(1)	DEGA	01716
50 ISCH = ISCH + 1	DEGA	01717
ICNT2 = ICNT2 + 1	DEGA	01718
ICNT1 = ICNT1 + ICNT2	DEGA	01719
TERMLST = TERMLST*OATOPI	DEGA	01720
DN = DN*D	DEGA	01721
SCRATCH(ISCH) = ICNT1*TERMLST	DEGA	01722
IF(ABS(SCRATCH(ISCH)*DN) .LT. EPSILON) 60, 50	DEGA	01723
60 SCRATCH(499) = C2*ATOPI + 2.0*C3	DEGA	01724
SCRATCH(500) = C2	DEGA	01725
CALL POLYMUL(SCRATCH(1), ISCH, SCRATCH(499), 2, DVPOLY, IDVDD)	DEGA	01726
40 RETURN	DEGA	01727
END	DEGA	01728
SUBROUTINE FINDA12(I, IMAX, XLAM, ALFA, XLLP1, ATOPIPI,	DEGA	01729
1 A1, A2, JMAX, AS, JSMAX, IERROR)	DEGA	01730
DIMENSION A1(1), A2(1), JMAX(1), AS(2)	DEGA	01731
IERROR = 0	DEGA	01732
IF(I .GT. IMAX) 2, 1	DEGA	01733
1 AS(1) = A1(I)	DEGA	01734
AS(2) = A2(I)	DEGA	01735
JSMAX = JMAX(I)	DEGA	01736
RETURN	DEGA	01737

2 CALL CARSON(XLAM, -1.0, 1, ATOP1, XLLP1, -----	DEGA	01738
1 CAPF, CAPFP, THETA, THETAP, IERROR) -----	DEGA	01739
IF(IERROR .EQ. 12) RETURN -----	DEGA	01740
THETA1 = THETA + ALFA -----	DEGA	01741
COSTHE1 = COS(THETA1) -----	DEGA	01742
AS(1) = CAPF*COSTHE1 -----	DEGA	01743
AS(2) = CAPFP*COSTHE1 - CAPF*THETAP*SIN(THETA1) -----	DEGA	01744
JSMAX = 0 -----	DEGA	01745
RETURN -----	DEGA	01746
END -----	DEGA	01747
SUBROUTINE FINDATP(IM, IMAX, XLLP1, ALAST, ATOP1, XLAMB, IERROR) -----	DEGA	01748
DIMENSION ATOP1(1) -----	DEGA	01749
IERROR = 0 -----	DEGA	01750
IF(IM .GT. IMAX) 2, 1 -----	DEGA	01751
1 ALAST = ATOP1(IM) -----	DEGA	01752
RETURN -----	DEGA	01753
2 IF(XLAMB .GT. 0.0) 4, 3 -----	DEGA	01754
3 IERROR = 14 -----	DEGA	01755
RETURN -----	DEGA	01756
4 DR = .2*ALAST -----	DEGA	01757
R1 = 1.05*ALAST -----	DEGA	01758
CALL DRMAXSB(DRMAX, ALAST, R1, XLAMB, XLLP1, 1) -----	DEGA	01759
IF(DR .GT. DRMAX) DR = DRMAX -----	DEGA	01760
ALAST = ALAST + DR -----	DEGA	01761
RETURN -----	DEGA	01762
END -----	DEGA	01763

PRINT 7, ISSMAX

DEGA 00003

01/09/73 \*LASL MCH1 .16 01/09/73 MACH. 1 ECS ON  
 18.56.54.J15WM2W 01/09/73  
 18.56.54.\$JOB(NAME=J15WM, AC=J15D, USF=DEGAA, TL=  
 18.56.54.1 SC=156000, PL=200, MX=66,  
 18.56.54.S1 CL=U, PR=6, CAT=6, UA=8715C131)  
 18.56.55.ASSIGNMT, OLDPL(PLB,LF223L00,SHB)  
 18.56.55.65 ASSIGNED  
 18.56.55.LF223L00  
 18.56.55.ASSIGNMT, NEWPL(NLB,SHB)  
 18.56.56.66 ASSIGNED  
 18.56.56.LD274L00  
 18.56.56.UPDATE(F,S)  
 18.57.07. DECK STRUCTURE CHANGED  
 18.57.07. DECK STRUCTURE CHANGED  
 18.57.14. UPDATING FINISHED  
 18.57.15.REWIND(SOURCE)  
 18.57.16.UPDATE(N,I=SOURCE)  
 18.57.38. UPDATING FINISHED  
 18.57.38.COPYSBF(COMPILER,OUTPUT)  
 18.57.43.CP 00005.336 SEC.  
 18.57.43.PP 00057.626 SEC.  
 18.57.43.SS 00127.002 SEC.  
 18.57.43.  
 18.57.43.  
 18.57.43.  
 18.57.43. 2W 2W 2W