JOINT INSTITUTE FOR LABORATORY ASTROPHYSICS

REPORT

JILA REPORT #91

PROGRAM FOR THE EVALUATION OF

REDUCED MATRIX ELEMENTS

by

J. Cooper

and

U. Palmer

NOTICE

This work was done in part under a NASA contract, however, we publish here so that the program may be made available to a wider constituency.

University of Colorado
Boulder, Colorado
November 17, 1967
Section 1. Description

A given atom is specified by the charge on the nucleus (ZZ), core-charge (CORECH) (i.e. CORECH = 1 for neutral atom, etc.) and ionization limit (FLIM; in wave numbers). The states of this atom involving transitions of the type \( \chi^N_1 \rightarrow \chi^N_{1'} \) and for which LS coupling is applicable are described by the total angular momentum J (TJ), the total spin S (S), the total angular momentum L(LCAP), the principal quantum number, n, and orbital angular momentum, \( \ell \), of the 'jumping' electron (N and L), the angular momentum of the core (K), the parity (LPARITY = 1 for odd, = 2 for even) and the energy of the state, E (ENERGY). The energy of bound states is in wave numbers above the ground state (but prefixed with a negative sign), and continuum states are in rydbergs (positive).

The program calculates the following:

(a) Reduced matrix elements between states E and E'

\[
\text{HINT (k)} = \text{TRANSIT (k)} \times \frac{\text{TRANSIT (k)}}{(2J + 1)}
\]

where

\[
\text{TRANSIT (k)} = (-1)^{J+L'\ell'+L+K+S+k}(2J + 1)(2J' + 1)(2L + 1)(2L' + 1)
\]

\[
(2\ell + 1)(2\ell' + 1) \right)^{1/2}
\]

\[
\delta(S, S')\delta(K, K')\delta_{LL'}\delta_{kk'}(\int_0^\infty r^k R(E, n\ell) R(E', n'\ell') \, dr)
\]

\[
= (K_{\ell}sLJ, E \mid G(k)k^k \mid K_{\ell'}'SL'J'E')
\]

with \( C^{(k)}_{\ell} = (4\pi/2k + 1)^{1/2} y_{\ell q} \)

if change in \( \ell \) is \( \pm 1 \), then \( k = 1 \) for DIPOLE case, and if change in \( \ell \) is 0 or \( \pm 2 \), then \( k = 2 \) for QUADRUPOLE.

* Work supported in part by NASA under contract NAS1-7173.
(b) **Radial matrix elements**

\[
\text{SUM (k)} = \int_0^\infty R(E, n\ell) \ R(E', n'\ell') \ \rho^k \, dr
\]

\(k=1\) for DIPOLE and \(k = 2\) for QUADRUPOLE. \(R(E', n'\ell')\) can be replaced by a continuum state \(G(E'\ell')\).

Radial wave functions are calculated from Schrodinger's equation

\[
\frac{d^2 R_{n\ell}}{dr^2} + \left[ V(r, \alpha) + E_{n\ell} - \ell(\ell + 1)/r^2 \right] R_{n\ell} = 0
\]

and \(V(r, \alpha)\) is the stretched Thomas-Fermi potential of Stewart and Rotenburg (Phys. Rev. **140**, A 1508 (1965)).

The main program is called ZAMNU. The input consists of a CARD 1 which specifies the atom, and this is followed by cards specifying two bound states (STATES 1 and 2) (each bound state requiring a CARD 2 and CARD 3 in order), and then cards specifying either bound or continuum states (CARD 2 only). The continuum states always follow a bound state and the radial wavefunctions are calculated for the same value of \(\alpha\) and \(\ell'\) as the last bound state. Since the same potential is used in the continuum case as for the last bound state, the principal quantum number of this state should, for accuracy, be as large as possible. (See input for ZAMNU, Section 2). The output from ZAMNU consists of calculation of HINT (k) and SUM (k) (DIPOLE AND QUADRUPOLE where applicable) for STATES 1 and 2 between themselves and each of the following states in turn.

An example of the OUTPUT FORMAT is as follows:
Charge of Nucleus $Z = 11.00$
Charge of Core $Z_C = 1.00$
Ionization Limit = 41449.65 CM-1

<table>
<thead>
<tr>
<th>K</th>
<th>N</th>
<th>L</th>
<th>S</th>
<th>CAPL</th>
<th>J</th>
<th>P</th>
<th>E(CM-1)</th>
<th>STATE 1 AND INPUT STATE</th>
<th>STATE 2 AND INPUT STATE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>4</td>
<td>1.5</td>
<td>34548.79 0.0000+000 -1.6716+000 0.0000+000 -1.0456+001</td>
<td>0.0000+000 2.0398-001</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>4</td>
<td>1.5</td>
<td>1.00 0.0000+000 3.0896-001 0.0000+000 2.0398-001</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>4</td>
<td>1.5</td>
<td>10.00 0.0000+000 5.6518-002 0.0000+000 3.2424-002</td>
<td></td>
</tr>
</tbody>
</table>

continuum states
The reduced matrix elements are listed under DIPOLE or QUADRUPOLE where applicable, and SUM refers to whichever reduced matrix element is relevant. The first two lines of output are the matrix elements of STATE 1 and STATE 2 with themselves in turn, and the third line is the matrix element of STATE 1 with STATE 2. The classification of states 1 and 2 are listed in the first two lines of output. After the third line of output each successive state is listed, as are the matrix elements with states 1 and 2 respectively. Notice that the continuum states have the same classification as the last bound state, but the energy is in Rydbergs.

Subroutines called from ZAMNU are:

(a) SUBROUTINE PARAMNU (P)

This determines the bound wavefunctions as described in (Section 3).

(b) FUNCTION TRANSIT (J POLE)

This computes the radial integrals for SUM (k) as described in (Section 5) and multiplies by the necessary vector coupling coefficients for the reduced matrix elements.

(c) SUBROUTINE ENERGYNU (P)

This determines the continuum wavefunctions as described in (Section 4).

(d) SUBROUTINES FERMI (R, Q, Z, ALF, V) and FERME (R, Q, Z, ALF, V, RZERO)

These generate the Thomas-Fermi potential as described by Stewart and Rotenburg for the bound and free state wavefunction calculations respectively. [The potentials returned differ by a factor of 2Z/r between the two subroutines] FUNCTION RKLDEQ (N, Y, F, X, H, NT) is used in FERMI and FERME.
(e) FUNCTION S3JO (FJ1, FJ2, FJ3), and FUNCTION S6J (FJ1, FJ2, FJ3, FL1, FL2, FL3) (with FUNCTION FCTRL (A) and FUNCTION DELSQ (A, B, C))

Determine the 3-j and 6-j vector coupling coefficients (see Edmonds: Angular Momentum in Quantum Mechanics).
Section 2        INPUT FOR ZAMNU

A. CARD 1 DETERMINES A CASE (A GIVEN ATOM)
B. BOUND STATES REQUIRE CARDS 2 AND 3
C. FREE STATES REQUIRE ONLY CARD 2 (BUT ARE ASSUMED TO HAVE SAME QUANTUM NUMBERS AS THE PREVIOUS BOUND STATE)
D. FOR TERMINATION OF A CASE SET $S =$ NEGATIVE NUMBER IN CARD 2. PROGRAM WILL PROCEED IMMEDIATELY TO READ A NEW CARD 1 SIGNIFYING A NEW CASE. AN END-OF-FILE CARD TERMINATES TOTAL JOB.

E. CONDITIONS:
THE FIRST TWO STATES MUST BE BOUND STATES. A CONTINUUM STATE WILL USE THE STEP AND ALPHA OF PREVIOUS BOUND STATE.

F. ERRORS
   BOUND STATE
   IHOW       REASON
   1          50 TRIES PER DAPLHA
   2          ALPHA IS NEGATIVE
   3          NO CONVERGENCE FOR 5 DALPHAS

   CONTINUUM STATE
   IHOW       REASON
   1          5TH NODE WITHIN LESS THAN 250 POINTS, STEP WAS HALVED 5 TIMES.

G. FORMATS
   501 FORMAT (5E15.0) FOR CARDS 1 AND 2
   500 FORMAT (8I10) FOR CARD 3
CARD 1

WORD

1 ZZ Charge of nucleus
2 CORECH Core charge
3 FLIM Ionization limit (in wave-numbers)

CARD 2

WORD

1 S Total spin
   For end of case (for particular Card 1) S must be negative
2 TJ Total angular momentum
3 ENERG Energy of State
   A. Bound State
   Energy must be given in wave numbers measured above ground state (with a negative sign)
   B. Continuum State
   Energy must be in positive rydbergs

CARD 3

WORD

1 K L value of core
2 N Principal quantum number
3 L Orbital angular momentum of 'jumping' electron
4 LCAP Orbital angular momentum
5 LPARITY Parity for state
   1 for odd
   2 for even
Section 3. DETERMINATION OF BOUND STATE WAVEFUNCTIONS (PARAMNU (P))

The problem is, given a value of energy $E_{n\ell}$ and a potential $V(\alpha, r)$ where $\alpha$ is a variable parameter, find $\alpha$, and hence $P_{n\ell}$, for which the following eigenvalue equation is satisfied:

$$\frac{d^2P_{n\ell}}{dr^2} + \left[ -V(r, \alpha) - \frac{\ell(\ell+1)}{r^2} + E_{n\ell} \right] P_{n\ell} = 0.$$ 

The subroutine will work for any potential $V(\alpha,r)$, but we apply it here specifically to the Stewart and Rotenburg potential (Phys. Rev. 140, A1508 (1965) --(SUBROUTINE FERMI calculate $V(\alpha,r)$).

The following initial conditions are used:

(i) For small $r$, $P_{n\ell} \sim r^{\ell+1}$

(ii) When $[ -V(r, \alpha) - \frac{\ell(\ell+1)}{r^2} + E_{n\ell} ] = 0$ we have $\frac{d^2P_{n\ell}}{dr^2} = 0$.

Then if $r_\alpha$ is the largest root of this equation, for $r > r_\alpha$ the solution for $P_{n\ell}$ is monotonically decreasing. Since $-V(r, \alpha) \leq 2Z/r$ where $Z$ is the nuclear charge, we must have $r_\alpha \leq 2Z/(-E_{n\ell})$.

Thus, $r_\alpha$ is found by an iterative procedure using $2Z/(-E_{n\ell})$ as the starting value.

(iii) Outside $r_\alpha$ the wavefunction goes asymptotically as $e^{-(-E_{n\ell})^{1/2}r}$.

We therefore know that the wavefunction is small at $r_{max}$ where

$$r_{max} = r_\alpha + 23.0/(-E_{n\ell})^{1/2}.$$ 

(iv) At large values of $r$, $V(r, \alpha) \rightarrow 2z/r$ where $z$ is the core charge.
Then
\[ P_{n\ell}(r) = e^{-\left(-E_{n\ell}\right)^{1/2}} r^\beta \phi(1,r,E) \]
and
\[ \frac{dP_{n\ell}(r)}{dr} = e^{-\left(-E_{n\ell}\right)^{1/2}} r^\beta \left[ \frac{\beta}{r} - \left(-E_{n\ell}\right)^{1/2} \right] \phi(1,r,E) + \phi(2,r,E) \]

\[ \phi(1,r,E) = \left\{ 1 + \frac{\gamma a_1}{r} + \frac{\gamma^2 a_2}{r^2} + \frac{\gamma^3 a_3}{r^3} + \ldots \right\} \]

\[ \phi(2,r,E) = -\frac{1}{r} \left( \frac{\gamma a_1}{r} + \frac{2\gamma^2 a_2}{r^2} + \frac{3\gamma^3 a_3}{r^3} + \ldots \right) \]

where
\[ \gamma = \frac{1}{\left(-E_{n\ell}\right)^{1/2}} ; \quad \beta = z/\left(-E_{n\ell}\right)^{1/2} ; \quad a_0 = 1 \]

and
\[ 2(n+1)a_{n+1} + \left[ n(n+1) - (2n+1)\beta + \beta^2 - \ell(\ell+1) \right] a_n = 0 \quad n \geq 0 \]

With these initial conditions, the integration of the eigenvalue equation is carried out as follows:

(a) An initial guess is made for \( \alpha \) (usually \( \alpha = 1 \)), and with this we determine \( r_{max} \) and the value of \( P_{n\ell} \) and \( \frac{dP_{n\ell}}{dr} \) at \( r_{max} \).

(b) Numerov integration procedures are used for speed and accuracy.

We use a scale of \( y = r^{1/2} \) because here the nodes are almost equally spaced so that the step-size need not be constantly changed to maintain numerical accuracy (Stewart and Rotanburg).
(c) A step-size STEPS is defined, such that \( \text{STEPS} = 8/z \cdot 10^{-6} \). For reasonable numerical accuracy we require about 1,000 points in the integration in the \( y \) scale. The required step size is then found by repeatedly doubling STEPS until \( (r_{\text{max}})^{1/2} / \text{STEP} < 1000 \).

(d) Outward integration from \( r = 0 \) is started by using \( r^{l+1} \) for the first \( l + 3 \) steps (which is found sufficient to start accurately the Numerov integration). Outward integration then proceeds by the Numerov method, counting nodes of the wavefunction, \( N \), until a point \( r_0 \) has been reached, where \( r_0 \) is the first maximum after \( N = n - l - 1 \) nodes. If this point is not reached within \( r_a \), \( \alpha \) is changed (so as to increase the potential) until it does.

(e) When \( r_0 < r_a \) has been found, inward integration starts from \( r_{\text{max}} \) using values of \( P_{n\ell} \) and \( \frac{dP_{n\ell}}{dr} \) at \( r_{\text{max}} \) to start the integration. Integration inwards proceeds until the point \( r_0 \) is reached. If it crosses the axis before \( r_0 \), the potential is decreased by changing \( \alpha \); \( r_{\text{max}} \) redetermined, and inward integration again started.

(f) When, for a given \( \alpha \), forward and backward integrations reach \( r_0 \), the logarithmic derivatives of the inward and outward integration are compared at \( r_0 \), i.e.

\[
\frac{1}{P_f} \left. \frac{dP_f}{dr} \right|_{r = r_0} - \frac{1}{P_b} \left. \frac{dP_b}{dr} \right|_{r = r_0} = \varepsilon(\alpha)
\]

A three-point formula is used to calculate the differential (which is in the \( r \) and not the \( y \) scale).
(g) \( \alpha \) is then changed to decrease \( |\varepsilon(\alpha)| \) until it changes sign (from e.g. positive to negative). The step size in \( \alpha \) is then decreased and \( \alpha \) again changed until \( \varepsilon(\alpha) \) changes sign. The procedure is repeated until there are two values \( \alpha_1 \) and \( \alpha_2 \) (differing by one step in \( \alpha \)) for which \( \varepsilon \) changes sign, and for which ratios of the wavefunction at two points, normally the second point of \( r^{k+1} \) and \( r_0 \) satisfy

\[
\left| \frac{P(r_0, \alpha_1)}{P(2, \alpha_1)} \cdot \frac{P(2, \alpha_2)}{P(r_0, \alpha_2)} - 1 \right| \leq 0.01 \text{ (say)}
\]

This indicates that the wave function is not changing wildly in the step of \( \alpha \) for which \( \varepsilon \) changes sign. The final value of \( \alpha \), \( \alpha_f \) say, is then obtained by linear interpolation of the values \( \varepsilon(\alpha_1) \) and \( \varepsilon(\alpha_2) \) (for \( \varepsilon(\alpha_f) = 0 \)). Normally the wave function is not too sensitive to \( \alpha \).

(h) In changing the step size in \( \alpha \), by amount \( \Delta \alpha \), it is essential to have a convergent procedure (i.e., we must not change back to an original value). To ensure this we start off with steps of \( \Delta \alpha \) (initially 0.1) in one direction (either increasing or decreasing \( \alpha \)). When we have to change direction, due, for example, to \( \varepsilon(\alpha) \) changing sign, the step is decreased to 0.1 \( \Delta \alpha \). This new step is used until we again have to change direction or when nine further steps in \( \alpha \) have been taken, at which time the step in \( \alpha \) is changed to 0.01 \( \Delta \alpha \), and so on. (Step sizes less than 0.001 are not normally needed.)
(i) Final inward and outward integrations are performed for $\alpha_f$.

The wavefunction is made continuous by multiplying the value $P_b$ for $r > r_0$ by $P_f(r_0)/P_b(r_0)$. These values of $P_{n\ell}$ are normalized by dividing by

$$\int_0^{r_{\text{max}}} P_{n\ell}^2 \, dr .$$

This integration is done in the $y$ scale by a Simpson rule method. The wavefunction $P_{n\ell}(y^2)$ in equal intervals in the $y$ scale is then stored, to be used to calculate radial integrals with wavefunctions of other values of $n$ and $\ell$.

Section 4. DETERMINATION OF CONTINUUM WAVEFUNCTIONS (ENERGYNU (P))

Here the problem is to solve Shrodinger's equation

$$\frac{d^2 G_{k\ell}}{dr^2} + \left[ -V(r, \alpha) - \frac{\ell(\ell+1)}{r^2} + E_{k\ell} \right] G_{k\ell} = 0$$

where $E_{k\ell} = k^2 > 0$.

$G_{k\ell}$ is normalized to unit amplitude such that $G_{k\ell}(r) \rightarrow \cos (kr + \delta)$. $r \rightarrow \infty$

The procedure is as follows:

(1) In determining the potential, $\alpha_f$ for the last bound state is used and $V(r, \alpha_f)$ is obtained from SUBROUTING FERME. Although as we proceed up a Rydberg series $\alpha_f$ is only slowly changing, for accuracy, the continuum state should be proceeded by a bound state of as high a principal quantum number as possible (and of course the same $\ell$).
(2) For convenience in calculating the radial integrals with the bound states (STATES 1 or 2), we use the step size corresponding to the larger of the step sizes for the two bound states (needed in TRANSIT (JPOLE) - see Section 5).

(3) Outward integration of the Schrödinger equation proceeds on this predetermined STEP size by the Numerov method (in the $y = r^{1/2}$ scale). $C_{k+1} = r^{\ell+1}$ is used for the first $\ell + 3$ points.

(4) There are at most 1000 points in the bound state wavefunction, so, to insure complete overlap, integration is carried out for 1000 points. During the integration each successive node in the wavefunction is counted; in particular the position of the 5th node is noted. (The 5th node is used since it is found to lie in a region where the potential is asymptotic).

Then (a) if the 5th node is between 250th and 1000th point, the values of the wavefunction at the 5th node are used for normalization of the 1000 points.

(b) If the 5th node has not been reached within 1000 points, these 1000 points are stored, integration then proceeds until the 5th node is reached (never more than 10,000 points), and the values at the 5th node are again used to normalize the original 1000 points.

(c) If in the first integration the 5th node is less than the 250th point, this indicates the original STEP size is too coarse. In this case the step size is halved and the integration is repeated, but retaining only every second point. This gives the step size which is
required in the determination of the radial integrals in subroutine TRANSIT. This process is repeated until the 5th node, which is used for normalization, is at more than the 250th point in the reduced step size. The integration then proceeds until 1000 points on the original step size are obtained. These 1000 points are again normalized using values at the 5th node.

(5) It is found that the 5th node is always in a region where the potential is almost asymptotic and slowly varying, so the WKB approximation may be used to normalize the wavefunction (see Appendix B). For unit amplitude at infinity this amounts to dividing the un-normalized \( G_{k\ell}(r) \) by \( Q \) where

\[
Q^2 = \frac{D^2A^2}{E_k}
\]

such that \( D \) is the derivative of the un-normalized wavefunction with respect to \( r \) (not \( y \)) at the 5th node (i.e. at \( r = R \), so that

\[
D = \frac{dG}{dr} \bigg|_{r = R} \quad \text{and} \quad A^2 = \left[ E_k / \{ E_k + V(\alpha_f, R) - \frac{\ell(\ell+1)}{R^2} \} \right]^{1/2}.
\]

This procedure gives 1000 points of continuum wavefunction, properly normalized, and on the correct step size for overlap with bound states 1 and 2 in TRANSIT. (The \( G_{k\ell}(r) \) are stored in the \( y = r^{1/2} \) scale.)
Section 5. RADIAL INTEGRALS (TRANSIT (JPOLE))

This is for evaluation of

\[
\text{SUM} \ (k) = \int_{0}^{\infty} P_{n_1 l_1} (r) \ P_{n_2 l_2} (r) \ r^k \ dr
\]

where \( k = 1 \) for dipole integrals and \( k = 2 \) for quadrupole integrals.

During the running of the program the wavefunctions for STATES 1 and 2 are first computed and then remain stored. Subsequent states (bound or free) are temporarily stored so that SUM \( (k) \) can be calculated. For each of these subsequent states SUM \( (k) \) with both STATES 1 and 2 are calculated, as indicated below.

(a) For bound states the wavefunctions \( P_{n_1 l_1} \) and \( P_{n_2 l_2} \) (Normalized for two states \( E_{n_1 l_1} \) and \( E_{n_2 l_2} \) are stored in the \( r^{1/2} (= y) \) scale \( E_{n_1 l_1} \) is either STATE 1 or 2). The spacing (STEP) in the \( y \) scale is some multiple of STEPS so that the integral is then evaluated by a simple Simpson rule (in the \( y \) scale), using the largest value of STEP (so that the values which are used correspond on the \( r \) scale).

(b) For a continuum state \( P_{n_2 l_2} \) is replaced by \( G_{k l_2} (r) \). \( G_{k l_2} \) is automatically calculated on the STEP size corresponding to the largest value of step for STATES 1 and 2, so the integral can be determined as in (a).

Section 6. ACKNOWLEDGMENT

Thanks are due to Dr. John Stewart for supplying part of the program and for his continuous helpful advice.
426  STEP = SSTEP(1)
       IFX = ISVX(1)
       IF (SSTEP(1) = GT. SSTEP(2)) GO TO 426
       STEP = SSTEP(2)
       IFX = ISVX(2)

426  CALL FNEKG(NOC) (P(1), NOC)
       GO TO 460

450  ALPHA = ALPHA
       STEP = H(0) * STEPS/ZT
       IFX = 1
       CALL PARAMOD (P(1), NOC)
       SSTEP(NOC) = STEP
       ISVX(NOC) = IFX

460  LIMOW(NOC) = THOW
       MP(NOC) = UK
       IFX(NOC) = IFX
       GO TO (100, 101, 200, 100, 100)

100  IF (LIMOW(NOC), EQ, 0) GO TO 105
       WRITE(6) * 62, NOC
       CALL EXIT

105  I(J) = NOC
       L(J) = NOC
       I(J) = M
       IF (NOC .LE. 2) J = 1

110  HINT(2) = TRANSIT(2)
       HINT(1) = SINT

111  FMT(J+1) = 6H*F14.4
       FMT(J+1) = 5H*F14.4
       D(R+1) = FMT(1) * N(NOC) * N(NOC) * S(NOC) * LCAP(NOC) * TJ(NOC) * L(NOC)
       D(R+1) = FMT(1) * FNEKG(NOC) * HINT(1) * HINT(2)

115  FMT(J-1) = D(I)*D(I)
       NOC = NOC + 1
       IF (NOC .GT. 4) GO TO 2
       N(NOC) = N(NOC-1)
       D(NOC) = D(NOC-1)
       LCAP(NOC) = LCAP(NOC-1)
       LPARITY(NOC) = LPARITY(NOC-1)
       D(R) = D(R)*NOC
       LTFNZ = 1
       L: = 1
GO TO 250
200 LEND = 2
LT2 = 3
250 KUREG = 1
KUEND = 2 * LEND
GO 300 LT1 = 1 * LEND
INDEXA = 2 * LT1 - 1
INDEXB = INDEXA + 1
IPR = LT1 * 0 + 6
PRINT = IPR - 1
IF ( LTHOW(LT2), GT, 0 ) GO TO 705
255 [SUM = 1
LDIFF = XAHSF(L(LT1) - L(LT2))
IF ( LDIFF G, GE, 3 ) GO TO 705
JPOLE = 1
IF ( XMOD[LDIFF * 2], EN, 0 ) JPOLE = 2
LTEM = XAHSF(LCA(LT1) = ICA(LT2))
TEMP = AHSF(TJ(LT1) = TJ(LT2))
GO TO (260 * 270) * JPOLE
260 IF ( LPARITY(LT1) * AND, LPARITY(LT2) ) GO TO 700
IF (( TEMP, NF, 0 ), AND, ( TEMP, NF, 1, 0 )) GO TO 700
IF ( S(LT1), NF, S(LT2) ) GO TO 700
IF ( K(LT1), NE, K(LT2) ) GO TO 700
IF (( LTEMP, NF, 0 ), AND, ( LTEMP, NF, 1 )) 700, 275
270 [PRINT = IPR + 1
INDEXA = INDEXA
INDEXB = INDEXA - 1
IF ( S(LT1), NF, S(LT2) ) GO TO 700
IF ( K(LT1), NE, K(LT2) ) GO TO 700
IF ( LPARITY(LT1), NF, LPARITY(LT2) ) GO TO 700
IF ( TEMP, GT, 2, 0 ) GO TO 700
IF ( LTEMP, GT, 2 ) 700, 275
700 TSEM = 2
275 HIT ( INDEXA ) = TRANSIT(JPOLE)
IF ( JPOLE, EQ, 1 ) GO TO 276
TEMP = 2, 99NF + 10 * ( ENERG(LT2) - ENERG(LT1) )
276 HIT ( INDEXB ) = SUM
GO TO 280
705 GO TO (701 + 702) * LT1
701 KUREG = 3
GO TO 300
702 KUEND = 2
GO TO 300
C 400 FORMAT (XI10)
 501 FORMAT (5F14.6)
 600 FORMAT (1H1, 6F14.6, 1X, 2H1, 1F14.6, 1X)
 701 FORMAT (1X, 10X)
 800 FORMAT (1H1, 10X)
 905 FORMAT (1H1, 10X)

C 3000 CALL EXIT
END
FUNCTION TRANSIT(JPOLE)

IMI = IFAX(LT1)
IM2 = IFAX(LT2)
IM = IM1 + IM2
FM = IM
IM1 = IM/IM2
IM2 = IM2/IM1
KM1 = K(MU)
KM2 = K(MU)
TL(MU) = L(MU)

TCAPL(MU) = L.CAP(LM1)
STEP = STEP * FM * FM
NDP = XM1NM0F(NP(LT1)/IM1*NP(LT2)/IM2)
FNP = NDP

SUM = NDP + 1
SUM = 0.75 * P(IM1,LT1) + P(IM2,LT2) + STEP**JPOLE
TEMPR = STEP

ISUM = ISUM + STEP**JPOLE + P(IM1,LT1) + P(IM2,LT2) + II

CONTINUE

GO TO (51,50,ISUM)

END
GO TO (100*200)*JPOLFE

100 MN = MN + 1

CUL3 = 1.0
LAGGER = MAXOF(L(L1),T13*L(L2))
FLAGGER = LAGGER

TEMPL = 1.0
IF(XM0)FL(LAGGER**2,\=EQ,1) TEMP = -1.0
COEF (2) = TEMP*SIGN(FLAGGER)
GO TO 300

200 COEF(K) = COEF(K)*(2*L(L1) + 1) * (2*L(L2) + 1)

CUL3 = 2.0
COEF (2) = SQRT(TL(L1) + 2*0*TL(L2))

300 IF(XM0) (MN,2,ED,1) SIGN = -1.0

COEF(1) = SQRT1(COEF0)

COEF (3) = SQRT(TL(L1) + T(L1)*L(L2) + TIME(L2) + T(L2) + S(L1))

COEF (4) = SQRT(TCPL(L1) + TCAP(L2) + COL3 + TL(L2) + TL(L1) + TK(L1))

TRANSIT = SUM

500 MN = 1.0

350 TRANSIT = TRANSIT + COEF4

TRANSIT = TRANSIT + SIGN

TRANSIT = TRANSIT*TRANSIT/(2.0*TJ(L1) + 1.0)

RETURN

END
SUBROUTINE PARAMNI(P)

NUMERIOUS METHODS IS USED FOR INTEGRATION
ARRAY P = WAVE FUNCTION
ALPHA AND DALPHA ARE INITIAL VALUES; ALFA AND NALFA ARE
THE WORKING VALUES
IF AT ANY TIME ALFA BECOMES NEGATIVE, THEN HOW WILL BE SET TO P
AND AN EXIT TO CALLING PROGRAM IS TAKEN
RA = INFECTION PT. AFTER WHICH P DECREASES MONOTONICALLY
KF = INDEX FOR R AT WHICH RAD. IN FORW. INTEGR. OCCURS
Y = SORT(R) SINCE SORT SCALE CONVENIENT
EPS = DIFFERENCE OF LOGARITHMIC DERIVATIVES AT R(KF) USING
VALUES OF P(R(KF)) AND P(R(KF+1)) RESULTING FROM
FORW. AND BACKW. INTEGRATIONS
DEC1 = GAMMA (= SORT(-F))
DERIV = LOGARITHMIC DERIV. ON P*R
IFOR = 1 ALPHA HAS NOT CHANGED IN FORW. INTEGRATION
= 2 ALPHA HAS CHANGED IN FORW. INTEGRATION
IFACK = 1 ALPHA HAS NOT CHANGED IN BACKW. INTEGRATION
= 2 ALPHA HAS CHANGED IN BACKW. INTEGRATION
INOR = ARRAY WHICH SAVES WAVE FUNCTION VALUES O(KF+1),
O(KF+2), O(KF+1), O(KF+2) CALCULATED IN FORW.
INTEGRATION

DIMENSION YS(1000,2), P(1000,2), Y(1000), D(1000), DALPHA(5),
TENN(2), N(2), E(2), EPS(2), EPSF(2)
DIMENSION KFSAVE(2), steps(2), IFEX(2)
COMMON /ZAMMY/ STEP, ALPHA, THO, L, E, J, K, THOW, LP1, COFGCH, TEX

DATA (THO = 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1),
CAY(1000), DALPHA(5),
TENN(2), N(2), E(2), EPS(2), EPSF(2),
KFSAVE(2), steps(2), IFEX(2)

INITIALIZATIONS AND CALCULATIONS WHICH DO NOT DEPEND ON ALPHA
CHANGES

DALPHA = LOOP INDEX FOR THE THREE POSSIBLE DALPHAS
ISearch = COUNTS NUMBER OF SEARCHES FOR RA PER IDALPHA;
IF GREATER THAN 25 THEN THO WILL BE SET TO 1
AND AN EXIT TO CALLING PROGRAM WILL BE TAKEN

C

DIM = 2, 1 * 72 / (-E)
ALFA = ALPHA
SDEC = NATOM - L + 1
FRONT = SORTF (-E)
GETA = C0 * FCH / FRONT
TERMF = 23.0 / FRONT
LAST = 1

C

LOOP ON DELTA ALPHA

00 2000 INALFA = 1, 5
LSARCH = 0
DALFA = DALPHA (INALFA)
TEOR = 1
THACK = 1
IF = 0
50 IF = IF + 1
IF (IF, GT, 2) GO TO 255
100 W = INIT
INTEMP = T101(F)
TIDDF = 0
LSARCH = LSARCH + 1
IF (LSARCH, LT, 50) GO TO 101
SHOW = 1
RETURN
101 IF (ALFA, GT, 0.51) GO TO 104
SHOW = 2
RETURN
104 CALL FERMT (R, 00, 7Z, ALFA, V)
FORE = CAN(L, U) + F
IF (FORE, GE, 70) GO TO 102
01 = 0.5 * R1
GO TO 104
102 W = 01
NMAX = RA + TERMF
103 NMAX = SORTF (NMAX/STEP)
NMAX = NMAX + 1
NIN (IF = NMAX
IF (NMAX, LT, 1000) GO TO 105
STEP = 6.0 + STEP
IF = 2 * GEX
GO TO 103

GENERATE AHS/SS: Y = SQUARE = R

105 DO 120 J = 1, NMAX
120 YSG(J*IF) = STEP * J**2
   STEPSV(IF) = STEP
   IFEX(IF) = IFEX

SET INITIAL VALUES FOR FORWARD INTEGRATION

POW = 1. + 0.75
CAV(1) = 0.9
C(1*IF) = YSG(1) ** POW
LSTART = 1. * 1
DO 107 MJ = 2, LSTART
   CALL FERM1(YSG(MJ*IF) * QZ * Z * ALFA * V)
   CAV(MJ) = CAYSQ(V, YSG(MJ*IF))
107 YSG(MJ*IF) = YSG(MJ*IF) ** POW
   YSTEP = CORTF(STEP)

SET INITIAL VALUES FOR BACKWD. INTEGRATION

DRLIV = HETA/RMAX - FROOT
RECI = 1.0 / (RMAX * FROOT)
CONTH = 0.0
COFFA = 1.0
PHI1 = 1.0
PHI2 = 0.0
TERMS = 1.0
RECP1 = RECI
DO 110 ICOM = 1, 100
   C1 = TCOM * (ICOM - 1)
   C2 = 2 * (ICOM - 1) + 1
   C3 = 2 * TCOM
   COFFA = COFFA * (C1 - C2 * HETA + HETA * HETA - (L*(L+1))) / C3
   TERM = COFFA * RECP1
   IF (AHSE(TERM) .GE. TERMS) GO TO 115
   PHI1 = PHI1 + TERM
   PHI2 = PHI2 + TCOM * TERM
   TERMS = AHSE(TERM)
   RECP1 = RECP1 * RECI
   CONTR = -PHI2 / (RMAX * PHI1)
110 CONTINUE
115 DERIV = DERIV * CONI

125 N = N + 1
CALL FERMI(YS0(N+1,IF),YS0(N+1,IF))
CAY(N+1) = CAYSO(YS0(N+1,IF))
DENOM = 1.0 - STEP12 * CAY(N+1)
ENUMERA = Q(N+1,IF) * (2.0 + 10.0 * STEP12 * CAY(N))
ENUMERH = Q(N-1,IF) * (1.0 - STEP12 * CAY(N-1))
Q(N+1,IF) = (ENUMERA - ENUMERH) / DENOM

127 IODE = IODE + 1

128 IF (YS0(N+1,IF) .GT. 0.0) GO TO 130

129 TEST FOR DO

130 IF (IODE .LE. NODES) GO TO 125

135 TEST FOR MAXIMUM

136 IF (((Q(N+1,IF) - Q(N,IF)) / (Q(N,IF) - Q(N-1,IF))) .GT. 0.0) GO TO 125

K = N
KFSAVE(1F) = KB
DFN(1*IF) = Q(KF,IF)
DFN(2*IF) = Q(KF,IF)
72 = Q(KF,IF)
GO TO 130

INC. ALPHA BY DALPHA, RA HAS BEEN REACHED before MAX, AFTER
CORRECT NO. OF NODES WAS REACHED

130 IFOR = 2
GO TO (140,135),TRACK

135 DALFA = 0.95 * DALFA
140 ALFA = ALFA + DALFA
TRACK = 1
GO TO 100

BACKWARD INTEGRATION

150 DO 160 JP = 1,2
KP = NMAX + 1 - JP
CALL FERMI(YS0(KP,IE),Q,Z7*ALFA,V)

160 CAY(KP) = CAYSO(V,YS0(KP,IE))
KFINAL = NMAX - 1
KSTART = KP + 1

200 DO 200 KK = KSTART,KFINAL
J = KFINAL + KSTART - KK
CALL FERMI(YS0(J-1,IE),Q,Z7*ALFA,V)
CAY(J-1) = CAYSO(V,YS0(J-1,IE))
DENOM = 1.0 - STEP12 * CAY(J-1)
ENUMERA = 0(J,IE) * (2.0 + 10.0 * STEP12 * CAY(J))
ENUMERH = 10(J,IE) * (1.0 - STEP12 * CAY(J))
Q(J-1,IE) = (ENUMERA - ENUMERH) / DENOM
IF(Q(J-1,IE) < 0.95) GO TO 200

A NODE WAS FOUND, DECREASE ALPHA

GO TO (170,145), IFOR

165 DALFA = 0.95 * DALFA
170 ALFA = ALFA - DALFA
IFOR = 1
THACK = 2
GO TO 100

200 CONTINUE

NO NODE WAS ENCOUNTERED IN BACKWARD INTEGRATION

$\theta(I) = \theta(KF, IF)$
$\theta(IE) = \theta(I) / \theta(I)$

CALCULATE EPS

$Epsilon(IE) = ((\theta(IE, IF) - \theta(IE)) / \theta(IE, IF)) = 1$

$Epsilon(IE) = 0(KE, IE) / \theta(KE, IE))$

SAVE ALFA OF EPS(IE)

$Tempa(IE) = Alfa$
GO TO (210, 400, 1) LAST

210 IF(IE, F0, 2) GO TO 255
TF(EPS(IE), 1) 211, 250, 250
GO TO (213, 212), IFOR

ALPHA WAS INCREASED BY DALPHA IN LAST FORWARD INTEGRATION,

THEREFORE DALPHA MUST BE DECREASED FOR NECESSARY DECREASE IN ALPHA

DALFA = 0.52 * DALFA

DECREASE ALPHA

ALFA = ALFA - DALFA

IFOR = 1
THACK = 2
GO TO 50

EPS(1) IS POSITIVE, INCREASE ALPHA FOR EPS(2)

GO TO (252, 251), TRACKE

ALPHA HAD KEF-1 DECREASED IN LAST BACKWARD INTEGRATION BY DALPHA,

SO DALPHA HAS TO BE DECREASED FOR THE NOW NECESSARY INCREASE IN

DALFA = 0.95 * DALFA
SPA = 1
IF OR = 2
ALFA = ALFA + DALFA
GO TO 50
255 IF = 1
C CHECK FOR CHANGE IN SIGN OF EPS(1) AND EPS(2)

IF (EPS(1) * EPS(2)) 300, 300, 259
259 ALFA = TEMP(2)
TEMP(1) = ALFA
MEN(1) = MEN(2)
W(1) = W(2)
EPS(1) = EPS(2)
T(EX(1) = T(EX(2)
XSAVE(1) = XSAVE(2)
QFD(1,1) = QFD(1,2)
QFD(2,1) = QFD(2,2)
STEPS(1) = STEPS(2)
NO 266 MM = 1, NMIX
O(MM,1) = O(MM,2)
265 VSN(MM,1) = VSN(MM,2)
GO TO 210
C
C COMPARE WAVE FUNCTIONS OF DIFFERENT ALPHAS

300 RATIO = QFD(1,1) / QFD(2,1)
WATIO = W(2,2) / W(1,2)
IF (ANSE(RATIO)RATIO = 1.0) .LE. 0.999) LAST = 2
C
C INTERPOLATE FOR NEW STARTING VALUE OF ALFA

ALFA = TEMP(1) + ((EPS(1) / (EPS(2) - EPS(1))) .5)
((TEMP(2) - TEMP(1)))
2000 CONTINUE
C
C NO CONVERGENCE FOR FIVE ALPHAS, RETURN TO CALLING PROGRAM

THW = 3
RETURN
C
C CHOOSE FINAL ALPHA THEREFORE FINAL WAVE FUNCTION
400 IUSE = 1
410 ALFA = TEMPA(IUSE)
    ALPHA = ALFA
    STEP = STEPSV(IUSE)
    TEx = TEx(IUSE)
    NMAX = NFMDO(IUSE)
    IK = NMAX
    KF = KFSAVF(IUSE)

CC
TRANSFORM Q TO P

DO 425 MZ = KF, NMAX
425 Q(MZ, IUSE) = Q(IUSE) * Q(MZ, IUSE)
430 450 MN = 1, NMAX
450 P(MN) = (YSO(MN, IUSE) * (0.75)) * Q(MN, IUSE)

CC
NORMALIZE

KKFND = NMAX = 1
SUMP1 = 0.75 * P(1) * P(1)
475 KK = 2 * KKFND
475 SUMP1 = SUMP1 + KK * P(KK) * P(KK)
SUMP1 = 2.0 * SUMP1 + P(NMAX) * P(NMAX) * (NMAX - 0.5)
RENMOR = SQRTF(STEP + SUMP1)
480 J1 = 1, NMAX
480 P(JJ) = P(JJ) / RENMOR
RETURN
END
SUBROUTINE ENRGYNU (P)

NUCLEAR'S METHOD IS USED FOR INTEGRATION
ARRAY P = WAVE FUNCTION

DIMENSION P(1000), YSO(1000), CAY(1000), Q(1000)
1,MY(3),DP(3)
COMMON/ZAMM,STEP,ALPHA,77,QQ,NATOM,LE,IK,THOW,LP1,COPECH,JEX

ARITHMETIC STATEMENT FUNCTIONS TO EVALUATE K(R)

CAYSO (V,YSO) = (16.0 * (L*(L+1)) * 3.0) / (4.0*YSO)
+ 4.0 * YSO * (V - E)

INITIALIZATION

SVSTEP = STEP
P(1) = L + 0.75
NODES = 5
NMAX = 1K = 1000
NHALF = 1
THALF = 0
MAT = 998
MSUR = 2
120 I*MOR = 1
MPCOUNT = 0
INODE = 0
STEP12 = STEP/12.0

GENERATE ARRAY OF YSO WITH CURRENT STEP

130 J = 1, NMAT
130 YSO(J) = STEP * J**2

CALCULATE FIRST (L+3) VALUES OF FUNCTION Q AND OF K(Y)

CAY(1) = 0.0
Q(1) = YSO(1) ** 2
I*START = 1 + 3
140 MJ = 2*I*START
CALL FEMME (YSO(MJ),Q0,27,ALPHA,VP,H2R0)
CAY(MJ) = CAYSO(Y*, YSO(MJ))

140 Q(MJ) = YSO(MJ) * DPW

C
FORWARD INTEGRATION WITH NUMERICAL METHOD

145 N = LSTART + 1

CALL FERME ( YSO(N), 00, Z, ALPHA, V, RZERO)
CAY(N) = CAYSO(V, YSO(N))
DENOM = 1.0 - STEP12 * CAY(N)
ENUMER = D(N-1) * (2.0 + 10.0 * STEP12 * CAY(N-1))
ENUMERB = D(N-2) * (1.0 - STEP12 * CAY(N-2))
Q(N) = (ENUMER - ENUMERB) / DENOM

C
TEST FOR NODE:

147 NMODE = INODE + 1
IF (INODE * NE, NODES) GO TO 1000
LOCNOD = N
IF (LOCNOD * GE, 250) GO TO 400
IF (INODE GT. 1) GO TO 300
NHALF = 2
STEP = STEP/4.0
THALF = NHALF + 1
MMD = 2 * NHALF
MMD = XMURF (1000, MMD)
NSHR = MMD + MMD
MEND = 1000 - NSHR
MAT = MEND
IF (THALF * 1.15) GO TO 120
THAL = 1
RETURN

C
SET UP ARRAY FOR DERIVATIVE CALCULATION

100 DO 390 JJ = 1, 3
    TEMP = LOCNOD + JJ - 3
    DY(JJ) = YS0(TEMP)
390  DS(JJ) = Q(TEMP) * DY(JJ) * 0.125
IF((NHALF .EQ. 1) .AND. (INORE .GT. 1)) GO TO 1001

1000 CONTINUE

1001 IF((INORE .GE. NODES)) GO TO 1500

C
C
C
C
C

NODES LT NODES

GO TO (1100, 1200), NHALF

C
C
C
C
C

ORIGINAL STEP

1100 GO TO (1125, 1135), IMORE

C
C
C
C
C

CHANGE Q TO P AND KEEP 1000 PTS. AT ORIGINAL STEP

1125 DO 1130 M = 1, 1000

1130 P(M) = Q(M) * YSQ(M) ** 0.25

C
C
C
C
C

CREATE NEW Y-SQUARE MESH

1135 DO 1140 M = 1, 1000

1140 YSQ(M) = STEP * (M + MAT) ** 2

DO 1145 M = 1, 2

MM = M * 1000 - MSUR

Q(M) = Q(MM)

1145 CAY(M) = CAY(MM)

IMORE = IMORE + 1

MAT = 1000 + MAT - MSUR

MMIN = 3

GO TO 145

C
C
C
C
C

STEP HAD BEEN REDUCED

IHALF COUNTS NO. OF REDUCTIONS

KEEP 1000 PTS. AT ORIGINAL STEP

EVERY 2nd IHALF TH POINT OF PRESENT INTEGRATION WILL THEREFORE

BE STORED

1200 DO 1225 M = MMIN, MEND, MD4

NP = NP + 1

P(NP) = Q(M) * YSQ(M) ** 0.25

IF(NP .GT. 1000) GO TO 1750

1225 CONTINUE
GO TO 1135

C NODES GE 5

1500 GO TO (1600,1200),NHALF

C ORIGINAL STEP

1400 GO TO (1700,1750),IMORE

C NODE 5 WAS BETWEEN 250 AND 1000 POINTS, CHANGE Q TO P

1700 DU 1710 M = 1.1000
1710 P(Q) = Q(M) * YSQ(M)*100.25

C COMPUTE FNORM

C THE DERIVATIVE IS CALCULATED AT THE FIRST VALUE OF FUNCTION P

C BEYOND FIFTH NODE

C THE DERIVATIVE IS THEN USED TO CALCULATE THE VALUE OF R OR YSQ

C AT WHICH THE FIFTH NODE OCCURS

1750 RHO = 0.0
200 DD 200 KK = 1.3
300 PROD = 1.0
400 DD 190 KI = 1.3
500 IF(KK *EQ. KI) GO TO 190
600 PROD = PROD * ((DP(KI) / (DP(KK) - DP(KI))))

190 CONTINUE

RHO = RHO + N*KK * PROD

290 CONTINUE

DPRIK = 0.0
300 DD 300 KK = 1.3
400 SUM = 0.0
500 PROD = 1.0
600 DD 250 KI = 1.3
700 IF(KK *EQ. KI) GO TO 250
800 SUM = SUM + N*KI
900 PROD = PROD * ((N*KK - N*KI))

250 CONTINUE

DPRIK = DPRIK + ((7.0 * RHO - SUM) * DP(KK) / PROD)

300 CONTINUE
CALL FERMF(RHO)*Q0*ZZ*ALPHA*V*ZERO)
  TI = SORTF(E / (F - V + L + LP1 / (RHO*RHO)))
FNORMR = SORTF(TI * F) * ARSF(DERIV)
STEP = SVSTEP
DO 180 K = 1,NMAX
  180 P(K) = P(K) / FNORMR
RETURN
END
SUBROUTINE FERMI(R,Q,Z,ALF,V)
   C
   FERMI FOR PARAMNH
   DIMENSION SOL(2),SOLP(2),CHI(200)
   DATA TEMP/0.0/
   CMU=8.8534/(7860.*1./3.0)
   XP=CMU/ALF
   Y=SQRT(XP)
   TF(W-TEMP)NN.F71 11
11 CONTINUE
   TEMP=0.
   STEP=100.
   KK=STEP+2.0
   CHI(KK)=0.
   IF (Q<2.E)61.60
60    X=1.-0.10*6.E66666667
   H1=0.99999996-09
   H2=2.994626F00
   R3=1.09834004-01
   R4=7.695530F00
   R5=5.5765651F01
   R6=4.7246300E02
   R7=3.0630161F03
   R8=1.5030340F04
   R9=5.527980F04
   N10=1.5167922F05
   N11=3.986517E05
   N12=4.597141E05
   N13=4.360661E5
   N14=3.412771E5
   N15=1.449474E00
   N16=2.798460F04
   P=((((((((((((((R16*X+R14)*X+R14)*X+R13)*X+R12)*X+R11)*X
   +R10)*X+R9)*X+R8)*X+R7)*X+R6)*X+R5)*X+R4)*X+R3)

   2*X+R2)*X+R1
53 CONTINUE
V=W-SQRT(P)
Y=V
CHI=V/P
N=STEP+1.
H=V/STEP
ST=0
SOL(1)=0.
2 CONTINUE
SOLP(1) = VAR * SOL(2)
SOLP(2) = 4 * SQRT(SOL(1) **3)
S = RKLDEW(2, SOLP, VAR, H, NT)
TF(S, 1, 1, 2, 3)
1 WRITE (61, 101)
101 FORMAT (1H0, 5X, 31HERROR IN FERMAT, S LESS THAN ONF)
CALL EXIT
3 CONTINUE
N = N + 1
CHI(N) = SOL(1)
IF (N = 1) 4, 4, 2
4 CONTINUE
IF (XP = P) 6, 13, 13
13 V = (Z**2/R) * 2.
GO TO 5
6 MM = STEP
A = (Y/Y2) * STEF10.
J = A
G = J
S = A * G
IF (J = MM) 70, 73, 73
70 CONTINUE
VP = 5 * (1 - S) * (2 - S) * CHI(J) * 5 * (2 - S) * CHI(J+1) * 5 * S * (S = 1) * CHI(J+2)
72 CONTINUE
V = (2 * P**2/R) * (VP + QX * XP/P)
5 V = 10. / 2.0
RETURN
72 VP = (1 - S) * CHI(J)
1 IF (VP = QX * XP/P) 13, 13, 72
61 TE {1} = 62, 63, 63
63 T = 6 * LOG(N)
A1 = 2.4025073F-01
A2 = 0.7517329F00
A3 = 3.4419590F00
A4 = 6.134012E00
A5 = 7.127224F00
A6 = 6.352231F00
A7 = 4.2617929F00
A8 = 2.1357434F00
x9=7.6614453E+01
x10=2.0617123E+01
x11=3.6440191E+02
x12=3.8489733E+03
x13=1.8925049E+04
x14=0.
x15=0.
x16=0.
x=((((((x16+x81)x+x14)x+x13)x+x12)x+x11)x
+1+x10)x+x9)x+x8)x+x7)x+x6)x+x5)x+x4)x+x3)x+x2)x+x1)
go TO 53
62 x=A1.0G(9)
x1=2.0398509E+02
x2=2.0614454E+00
x3=9.0429095E+01
x4=3.0314427E+01
x5=3.9737421E+00
x6=2.0070116E+02
x7=1.7161375E+03
x8=2.1825265E+03
x9=3.0667491E+04
x10=2.367505E+05
x11=7.862251E+07
x12=0.
x13=0.
x14=0.
x15=0.
x16=0.
x=((((((x16+x81)x+x14)x+x13)x+x12)x+x11)x
+1+x10)x+x9)x+x8)x+x7)x+x6)x+x5)x+x4)x+x3)x+x2)x+x1)
p=EXP(p)
go TO 53
END
SUBROUTINE FERME (RQ, Z, ALF, VRZERO)
C
FERME FOR ENERGY 
DIMENSION SOL(2), SOLP(2), CHI(200)
DATA TEMP/0.0/
CMO=0.8534/(Z**2*(1.31))
XP0=(CMO+ALF)
Y=SORT(XP)
IF(N=TEMP)11,21,11
11 CONTINUE
TEMP=0
STEP=198
KK=STEP+20
CHI(KK)=7
IF(Q<5)A1,61,60
50 X=(1.+Q)**.666666667
H1=9.959999609
H2=2.9425626600
H3=1.8653000601
H4=7.7095549600
H5=5.576561501
H6=3.7425200602
H7=3.863916103
H8=1.503024064
H9=5.5227984604
A1=1.515092205
H11=3.8056117065
H12=4.897144206
H13=4.833631805
A14=3.432177105
H15=1.440473405
H16=2.746402704
P=(((((((((16*X+15)*X+14)*X+13)*X+12)*X+11)*X
1+10)*X+9)*X+8)*X+7)*X+6)*X+5)*X+4)*X+3
2*X+2)*X+1
53 CONTINUE
VAR=SORT(P)
Y2=VAR
CHIPZ2=0/P
N=STEP+1
HE=VAR/STEP
NT=0
SOL(1)=0.
$\text{SOL}(2) = 2.6 \times 10^{15} \text{XZ}$
$\text{CHI}(N) = \text{SOL}(1)$

2 CONTINUE
$\text{SOL}(1) = \text{VAR} \cdot \text{SOL}(2)$
$\text{SOL}(2) = 4 \cdot \text{SORT}(\text{SOL}(1) \cdot 3)$
$S = \text{RNLDER}(2 \cdot \text{SOL}(1) \cdot \text{SOL} \cdot \text{VAR} \cdot \text{INT})$
IF $(S \neq 1, 1, 2) \cdot 3$
1 WRITE(61, 101)
101 FORMAT(1X, 'ERROR IN FERMI. S LESS THAN ONE')
CALL EXIT
3 CONTINUE
$N = N + 1$
$\text{CHI}(N) = \text{SOL}(1)$
IF $(N - 1) \cdot 4 \cdot 4, 12$
4 CONTINUE
21 CONTINUE
IF $(X \neq P) \cdot 6 \cdot 13 \cdot 13$
13 $V = -(Z \cdot Q / P) \cdot 2^{*}$
GO TO 5
6 MM = STEP
$A = (Y / Y7) \cdot S \cdot TEV + 1$
J = A
$P = J$
$S = A - G$
IF $(J = MM) \cdot 7 \cdot 9, 73$
10 CONTINUE
$VP = 5 \cdot (1 - S) \cdot (2 - S) \cdot \text{CHI}(J) \cdot S \cdot (2 - S) \cdot \text{CHI}(J + 1) \cdot 5 \cdot S \cdot (S - 1) \cdot \text{CHI}(J + 2)$
72 CONTINUE
$V = -(2 \cdot Z / P) \cdot (VP \cdot N \cdot X / P / P)$
5 CONTINUE
PZERO = 0.1
$\text{CHI} = 1.0$
RETURN
13 $VP = (1 - S) \cdot \text{CHI}(J)$
TF $(VP \times (P / P / P = 0) \cdot 13 \cdot 72$
61 IF $(Q - 1) \cdot 62, 63$
53 $X \times \text{ALO}(1) \cdot 1$
$H1 = 2.402593 \times 01$
$H2 = 4.751732 \times 00$
$H3 = 3.441959 \times 00$
$H4 = 6.139401 \times 00$
$H5 = 1.127022 \times 00$
$H6 = 6.357023 \times 00$
$H7 = 4.281792 \times 00$
x8=2.1357434F00
x9=7.8616453E-01
x10=-2.0617123E-01
x11=-3.6440131E-02
x12=3.889733E-03
x13=-1.202509E-04
x14=0.
x15=0.
x16=0.

p=(((((((R16+x+R15)*x+x+R14)*x+R13)*x+R12)*x+R11)*x
1+x+R10)*x+R9)*x+R8)*x+R7)*x+R6)*x+R5)*x+R4)*x+R3)*x+R2)*x+R1

GO TO 53

62
x=A1.06(C)

h1=2.0398550F-02
h2=-2.0504454F-00
h3=-9.0425095E-01
h4=3.0314829F-01
h5=3.978721F-02
h6=1.4070116E-02
h7=7.161753E-03
h8=2.142265F-03
h9=-3.096749E-04
h10=2.367450AF-05
h11=7.862251F-07
h12=0.
h13=0.
h14=0.
h15=0.
h16=0.

p=(((((((R16+x+R15)*x+x+R14)*x+R13)*x+R12)*x+R11)*x
1+x+R10)*x+R9)*x+R8)*x+R7)*x+R6)*x+R5)*x+R4)*x+R3)*x+R2)*x+R1

p=EXP(p)

GO TO 53
FUNCTION RKLDEF0(N,Y,F,X,H,NT)

RKLDEF0 FOR FERMI AND FERME

DIMENSION Y(10),F(10),Q(10)

N=N+1
GO TO (1,2,3,4,NT)

1 N=0
X=X+H/2
GO TO 5

A=1.5
GO TO 5

2 A=2.9289331891
GO TO 5

3 A=1.7071067812
X=X+H/2
GO TO 10

5 N=1
Y(I)=Y(I)+A*(F(I)*Q(I)/N(I))/3
N=N+1
GO TO 6

6 RETURN
EN0
FUNCTION S3J0(FJ1,FJ2,FJ3)

G=(FJ1*FJ2*FJ3)/2.0
N = G + 0.1
M = 2.0 .0 G 0.1
IF(MOD(M,2) =EQ. 0)GO TO 5
S3J0 = 0.0
RETURN

5 DELTA = DELSQ(FJ1,FJ2,FJ3)
DELTA = SQRT(DELTA)
S3J0 = (-1.0)**N * DELTA * FCTRL(G) / (FCTRL(G-FJ1)**FCTRL(G-FJ2)
1
RETURN
6 CONTINUE
END
FUNCTION SUM (F1, F2, F3, FL1, FL2, FL3)

AA = A
FL = DELSQ (F1, F2, F3) * DELSQ (FL1, FL2, FL3)
10: IF FL = 0.0 THEN 213, 213
210 A = F1 + F2 + F3 + 0.02
B = F1 + FL2 + FL3 + 0.02
C = F1 + F2 + FL1 + 0.02
D = F1 + FL2 + F3 + 0.02
E = FJ1 + FJ2 + FL1 + FL2 + 0.02
F = FJ2 + FJ3 + FL2 + FL3 + 0.02
G = FJ3 + FJ1 + FL3 + FL1 + 0.02
11 = MAX1 (A, B, C, D)
12 = MIN1 (E, F, G)
11 = (I2 - 1) + 213, 211, 211
211 IF (I2 = 1) THEN 217, 217
212 F1 = FLOAT (1)
F2 = CTRL (F1 = A) * CTRL (F1 = B) * CTRL (F1 = C) * CTRL (F1 = D)
F3 = CTRL (F = F1) * CTRL (F = F2) * CTRL (F = FL1)
212 A = A + B + (F1 * 1.0) * MOD (F1) * CTRL (F1 + 1.0) / (F2 * F3)
213 A = A + B + (F1 + 1.0) / (F2 * F3)
214 RETURN
END
FUNCTION DFLS01(A, B, C)

C

IF (A MOD (A+B+C+0.1+1.0)=0.2) 108, 107, 107
107 DFLS0=0.0
RETURN
108 DFLS0=FCTRL(A+B-C)*FCTRL(A-H+C)*FCTRL(-A+H+C)/FCTRL(A+H+C+1.0)
111 RETURN
END
FUNCTION FCTHL(A)

DIMENSION FCTI(20)

DATA ((FCTI(I), I=1,20) = 1.0,2.0,6.0,24.0,120.0,720.0,5040.0,
1 40320.0,362880.0,3628800.0,39916800.0,479001600.0,
2 6227020800.0,87178291200.0,1307674368000.0,
3 2.09227799561E13, 3.55647428096E14, 6.02373705728E15,
4 1.216654940983E17*, 2.4329020081766E18)

IF (A) 50 A>0.7
50 IF (A.GE.(=0.1)) GO TO 60
   FCTRL = 0.0
   RETURN
60 FCTRL = 1.0
   RETURN
70 T = A * 0.1
   IF (T.FLE.0) Goto 60
   IF (T-20.0) 140.140.130
130 F=20.0
   FCTHL=FCTI(20)
   J=1:1
   F=F+1.0
131 FCTHL=FCTRL*F
   GO TO 150
140 FCTHL=FCTI(T)
150 RETURN
   END
APPENDIX B
NORMALISATION OF CONTINUUM WAVE FUNCTION

In the asymptotic region where the potential is slowly varying the radial Schrödinger equation is

\[ \frac{d^2 G_{k\ell}}{dr^2} + K(r)^2 G_{k\ell} = 0 \quad \text{where} \quad K(r)^2 = \left[-V(r, \alpha) - \frac{\ell(\ell+1)}{r^2} + E_k \right] \]

The WKB solutions of this equation behaving asymptotically as sine and cosine at large and are

\[ F(r) = \left( E_k^{1/4} / k^{1/2} \right) \cos \int K dr \quad \text{and} \quad G(r) = \left( E_k^{1/4} / k^{1/2} \right) \sin \int K dr \]

In this asymptotic region the solution can be expressed in terms of a sum of the above \( F(r) \) and \( G(r) \).

i.e. \( G_{k\ell}(r) = \alpha F(r) + \beta G(r) \)

Since \( F \) and \( G \) behave as sine and cosine, to normalize \( G_{k\ell}(r) \) so that it behaves as \( \cos(kr + \delta) \) at infinity, we have to divide by \( (\alpha^2 + \beta^2)^{1/2} \).

At a node \( (r = R) \) in the un-normalized wave-function (usually the 5th)

\[ G_{k\ell}(R) = 0 = \alpha F(R) + \beta G(R) \]

and \( D = \left. \frac{dG_{k\ell}}{dr} \right|_{r = R} = \alpha \left. \frac{dF}{dr} \right|_{r = R} + \beta \left. \frac{dG}{dr} \right|_{r = R} \)

But \( F^2 + G^2 = E_k^{1/2} / K \)

and \( F \left. \frac{dG}{dr} - G \left. \frac{dF}{dr} \right| = -\sqrt{E_k} \)

Thus \( Q = (\alpha^2 + \beta^2)^{1/2} = D / (K^{1/2} E_k^{1/4}) \)