

Lawrence Berkeley National Laboratory

Recent Work

Title

A COMPUTER PROGRAM TO CALCULATE LOW ENERGY ELECTRON DIFFRACTION INTENSITIES.

Permalink

<https://escholarship.org/uc/item/5cr8590n>

Author

Martin, Mario.

Publication Date

1972-12-01

A COMPUTER PROGRAM TO CALCULATE
LOW ENERGY ELECTRON DIFFRACTION INTENSITIES

Marlo Martin

December 1972

Prepared for the U.S. Atomic Energy
Commission under Contract W-7405-ENG-48

TWO-WEEK LOAN COPY

This is a Library Circulating Copy
which may be borrowed for two weeks.
For a personal retention copy, call
Tech. Info. Division, Ext. 5545



DISCLAIMER

This document was prepared as an account of work sponsored by the United States Government. While this document is believed to contain correct information, neither the United States Government nor any agency thereof, nor the Regents of the University of California, nor any of their employees, makes any warranty, express or implied, or assumes any legal 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. Reference herein to any specific commercial product, process, or service by its trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or the Regents of the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof or the Regents of the University of California.

-1-

A COMPUTER PROGRAM TO CALCULATE
LOW ENERGY ELECTRON DIFFRACTION INTENSITIES

Marlo Martin

Inorganic Materials Research Division, Lawrence Berkeley Laboratory
and Department of Chemistry; University of California,
Berkeley, California

Introduction

This report consists of a detailed writeup of a multiple scattering Low Energy Electron Diffraction program. The first section exhibits the essential features of the formalism used in the computation, and references are made to the numbered equations in a later section where the program is described in detail. This report is not intended as a substitute for a proper derivation of the formulas used. The reader is referred especially to references 1, 2 and 3 for this purpose. Section II presents the commonly used models used to account for the "inner potential" and the "inelastic damping" of the scattered electron. Section III is concerned with the presentation of the formalism for introducing temperature dependence into the multiple scattering problem. In Appendix A, the computer program is described taking each subroutine separately. Appendix B is included which reproduces a listing of the program plus sample input and output formats.

I. The t-matrix method for LEED calculations.

The LEED calculation and the computer program which is the subject of this report is based on the t-matrix approach to the electron-crystal scattering problem. The method to be followed was first set forth by Beeby¹ who derived the multiple scattering, or t-matrix method, but did not include the inelastic processes which are responsible for the severe attenuation of the reflected elastic electron intensities. This attenuation of the elastically scattered electrons was introduced phenomenologically by Duke and Tucker² who related an electron-electron mean free path for inelastic collisions, λ_{ee} , to the complex electron self energy of an electron moving through a crystal. It is the presence of this damping factor in an exponential which rapidly attenuates the elastic electron beams and makes multiple scattering summations converge rapidly. We employ the notation used by Laramore and Duke³ in our description of the t-matrix method and the computer program we have constructed.

The differential cross section of an elastically reflected LEED beam is proportional to the observed intensity.

$$\left. \frac{d^2\sigma}{dE d\Omega} \right|_{\substack{\vec{k}_i \rightarrow \vec{k}_f}} = \frac{m^2}{(2\pi\hbar^2)^2} \delta(E_f - E_i) |I(\vec{k}_f, \vec{k}_i; E)|^2. \quad (1)$$

The differential cross section is taken with respect to the energy of the incident (and emerging) beam and the solid angle $d\Omega$. The delta function restricts this description to elastic scattering processes for which the incident and emerging propagation vectors at points far outside the

crystal surface are $\tilde{k}_i$ and $\tilde{k}_f$, respectively. The mass of an electron is denoted by m . In Figure 1 the propagation vectors and their projections onto the surface are shown.

The quantity $I(\tilde{k}_f, \tilde{k}_i; E)$ is obtained by employing the two-dimensional periodicity of the surface, resulting in the expression

$$I(\tilde{k}_f, \tilde{k}_i; E) = \frac{(2\pi)^2}{A} \sum_{\lambda} \sum_{\tilde{g}} e^{-i\tilde{g} \cdot \tilde{a}_{\lambda}} \delta(\tilde{k}_{f\parallel} - \tilde{k}_{i\parallel} - \tilde{g}) e^{-i(\tilde{k}_{f\perp} - \tilde{k}_{i\perp})d_{\lambda}} \cdot T_{\lambda}(\tilde{k}_f, \tilde{k}_i; E) \quad (2)$$

The normalization factor $(2\pi)^2/A$ (where A is the area of the two dimensional unit cell of the surface layer) and the summation over the two dimensional reciprocal lattice vectors $\tilde{g}$ results from performing a summation over the periodic lattice positions in the surface layer. This also gives rise to the delta function which restricts the difference between the components of the wavevectors $\tilde{k}_i$ and $\tilde{k}_f$ parallel to the surface ($\tilde{k}_{i\parallel}$ and $\tilde{k}_{f\parallel}$) to be a reciprocal lattice vector $\tilde{g}$. In other words, this delta function, which is obtained from imposing the condition of an infinitely extended perfect two dimensional lattice, describes the experimentally observed pattern of discrete spots observed on the LEED screen. The two dimensional vector $\tilde{a}_{\lambda}$ is the shift of the origin position of the λ th plane parallel to the surface from an arbitrary zero position (often taken to be the origin position of the surface plane). We choose a coordinate system in which the $(\hat{x}, \hat{y})$ directions are in a plane parallel to the surface and $\hat{z}$ is the inward

normal to the surface. The remaining exponential term in eq. (2) accounts for the phase difference between the normal components of the incident and emerging beams, $k_{f\perp}$ and $k_{i\perp}$, due to the perpendicular distance between the origin and the λ th plane, d_λ .

The factor $T(\underline{k}_f, \underline{k}_i; E)$ is the t-matrix for the λ th layer. That is to say, it is the matrix element for all orders of scattering by the λ th layer, in the presence of all the other layers, between incident and final plane wave states of energy E denoted by the asymptotic wave vectors $\underline{k}_i$ and $\underline{k}_f$. This factor can be conveniently expressed in terms of two intermediate quantities whose physical significance can readily be appreciated. We introduce $\tau_\lambda(\underline{k}_2, \underline{k}_1; E)$ as the t-matrix for scattering an initial plane wave $\underline{k}_1$ into a final plane wave $\underline{k}_2$ by a single atomic layer parallel to the surface. This includes the effects of all possible multiple scattering events within the λ th plane and ignores all scatterings between different planes. The second factor $G^{\lambda\lambda'}(\underline{k}, \underline{k}_i; E)$ is the propagator for scattering between planes λ' and λ . It contains the phase and amplitude factor for the sum of all spherically scattered waves measured at the origin of the λ th plane due to all the ion core locations in plane λ' . Detailed expressions for τ_λ and $G^{\lambda\lambda'}$ will be presented.

$$T_\lambda(\underline{k}_f, \underline{k}_i; E) = \tau_\lambda(\underline{k}_f, \underline{k}_i; E) + \sum_{\lambda' \neq \lambda} \sum_{\underline{k}} \tau_\lambda(\underline{k}_f, \underline{k}; E) G^{\lambda\lambda'}(\underline{k}, \underline{k}_i; E) \cdot$$

$$T_{\lambda'}(\underline{k}, \underline{k}_i; E) \quad (3)$$

The first term in this expression contains the effect of scattering of the incident beam, $\underline{k}_i$, into the final beam, $\underline{k}_f$, by means of all single and multiple scatterings within layer λ . The second term represents the effects of electrons scattered from all planes $\lambda' \neq \lambda$ summed over all propagation directions $\underline{k}$ and impinging on plane λ , and include all subsequent multiple scatterings within plane λ . When the inelastic damping factor is sufficiently large and the t-matrices, τ_λ , are sufficiently weak, this expression can be expanded in a perturbation series by always expressing the final T_λ -factor in terms of the right hand side of equation (3). It will be seen later, when we work with a particular matrix representation of this equation, that the inversion of a rather large matrix will yield the complete multiple scattering result for T_λ .

The layer t-matrix $\tau_\lambda(\underline{k}_2, \underline{k}_1; E)$ appearing in eq. (3) can finally be related to the scattering of electrons from individual ion cores of the crystal. We write the t-matrix for ion-core scattering from plane wave states $\underline{k}_1$ to $\underline{k}_2$ as $t_\lambda(\underline{k}_2, \underline{k}_1; E)$. We further restrict all ions within the plane labelled by λ to be of the same material. The resultant simplification in the calculation causes no unnecessary limitation in generality since additional planes, λ' (now referred to as sub-planes), made up of different types of ions can coexist even at the same perpendicular distance from the surface, i.e., $d_{\lambda'} = d_\lambda$. The expression for τ_λ can be written in a manner similar to eq. (3).

$$\tau_\lambda(\underline{k}_f, \underline{k}_i; E) = t_\lambda(\underline{k}_f, \underline{k}_i; E) + \sum_{\underline{k}} t_\lambda(\underline{k}_f, \underline{k}; E) G^{sp}(\underline{k}, \underline{k}_i; E) \cdot \tau_\lambda(\underline{k}, \underline{k}_i; E) \quad (4)$$

The structure of this equation is similar to that of eq. (3) with G^{sp} being interpreted as the sub-plane propagator which contains the phase and amplitude factor for the sum of all spherically scattered waves impinging on an ion-core due to scattering from all other ion cores in the same sub-plane. A schematic diagram of the relationship between these quantities is furnished in Figure 3.

A perturbation method can be applied to solve eq. (4) as long as relatively weak scattering materials are used.⁴ A more general technique involves a matrix inversion, derived by Beeby,¹ in which all quantities appearing in eq. (4) are written in an appropriate matrix representation (to be specified below). Using a simplified notation, we write formally,

$$\begin{aligned}\tau_\lambda &= t_\lambda + t_\lambda G^{sp} t_\lambda + t_\lambda G^{sp} t_\lambda G^{sp} t_\lambda + \dots \\ &= t_\lambda (I + (G^{sp} t_\lambda) + (G^{sp} t_\lambda)^2 + (G^{sp} t_\lambda)^3 + \dots)\end{aligned}\tag{5a}$$

This series can be summed to yield the exact expression,

$$\tau_\lambda = t_\lambda [I - G^{sp} t_\lambda]^{-1}\tag{5b}$$

In order to proceed further we must specify the matrix representation to be used with all the foregoing equations. It is convenient to use an angular momentum representation in which these quantities are expressed in terms of summations involving spherical harmonics. In what follows, we write the conventional complex spherical harmonic $Y_{\ell m}(\theta, \phi) \equiv Y_L(\underline{k})$, where $\underline{k}$ is a vector whose angular components are θ and ϕ . The subscript L is written to replace the combined (ℓ, m) subscripts. Values of L

corresponding to various values of ℓ and m are listed explicitly in the writeup for the spherical harmonic subroutine (YLM) appearing later in this report.

The t -matrix $t_{\lambda}(k_f, k_i; E)$ can be expanded in a series of Legendre polynomials $P_{\ell}(\cos \theta)$, where θ is the angle between k_f and k_i . When the ion-core potential is spherically symmetrical there is no functional dependence of t_{λ} on the azimuthal angle ϕ . Due to this lack of dependence on the quantum number m , designation of ℓ or L alone suffices as a label for such quantities.

$$t_{\lambda}(k_f, k_i; E) = \sum_{\ell=0}^{\infty} t_{\lambda}^{\ell}(E) P_{\ell}(\cos \theta) \quad (6)$$

The Legendre polynomials can be expressed in terms of the spherical harmonics whose angular arguments are the vectors k_f and k_i by the formula,⁵

$$\frac{(2\ell + 1)}{4\pi} P_{\ell}(\cos \theta) = \sum_{m=-\ell}^{+\ell} Y_{\ell m}^*(k_f) Y_{\ell m}(k_i) \quad (7)$$

The (ℓ, m) indices will be combined as indicated in the previous paragraph and the expansion can be rewritten as,

$$t_{\lambda}(k_f, k_i; E) = \sum_L t_{\lambda}^L(E) Y_L^*(k_f) Y_L(k_i) \quad (8a)$$

For the quantities $\tau_{\lambda}(k_f, k_i; E)$ and $T_{\lambda}(k_f, k_i; E)$ one can no longer assume spherical symmetry and the expansion in terms of Legendre

polynomials is not possible. However, it can be shown that any such function which depends on $\underline{k}_i$ and $\underline{k}_f$ (and not simply $\underline{k}_f - \underline{k}_i$) can be expanded in terms of a double sum over the spherical harmonics,^{1,3}

$$T_{\lambda}(\underline{k}_f, \underline{k}_i; E) = \sum_L \sum_{L'} T_{\lambda}^{LL'}(E) Y_L^*(\underline{k}_f) Y_{L'}(\underline{k}_i) \quad (8b)$$

$$T_{\lambda}(\underline{k}_f, \underline{k}_i; E) = \sum_L \sum_{L'} T_{\lambda}^{LL'}(E) Y_L^*(\underline{k}_f) Y_{L'}(\underline{k}_i) \quad (8c)$$

Substitution of equation (8c) into equation (2) allows us to write

$$I(\underline{k}_f, \underline{k}_i; E) = \left| \sum_L \sum_{L'} Y_L^*(\underline{k}_f) Y_{L'}(\underline{k}_i) \sum_{\lambda} e^{-i(\underline{k}_f - \underline{k}_i) \cdot \underline{d}_{\lambda}} T_{\lambda}^{LL'}(E) \cdot \frac{(2\pi)^2}{A} \sum_{\underline{g}} (\underline{k}_f - \underline{k}_i - \underline{g}) e^{-i\underline{g} \cdot \underline{a}_{\lambda}} \right|^2 \quad (9)$$

A similar substitution in equations (3) and (4) allows us to factor out the spherical harmonics and to work directly with the matrices $T_{\lambda}^{LL'}(E)$ and $T_{\lambda}^{LL'}(E)$. For a given maximum ℓ -value of $\ell_{\max}$ the maximum value $L_{\max} = (\ell_{\max} + 1)^2$ become $(L_{\max} \times L_{\max})$ square matrices. Thus, the complexity and cost of the computer calculation will vary approximately as $(\ell_{\max} + 1)^4$. The value of $\ell_{\max}$ is set, in a particular problem, by the requirement that the diagonal matrix element $t_{\lambda}^L(E)$ for L corresponding to $(\ell_{\max} + 1)$ be negligible. The t -matrix elements for the ion core scattering problem can be directly related to the phase shifts for the scattering amplitude $f(\theta)$,⁶

$$f(\theta) = \frac{1}{k} \sum_{\ell} (2\ell + 1) P_{\ell}(\cos \theta) \left[\frac{e^{i2\delta_{\ell}} - 1}{2i} \right], \quad (10)$$

by the equation

$$f(k_f, k_i) = - \frac{m}{2\pi\hbar^2} t(k_f, k_i). \quad (11)$$

Equating (10) and (11) and using the relationship of eq. (7), the ℓ -component of the expansion (eq. 8a) becomes,

$$t_{\lambda}^{\ell}(E) = i \frac{4\pi^2\hbar^2}{mk} \left(e^{i2\delta_{\ell}(E)} - 1 \right), \quad (12)$$

where $\delta_{\ell}(E)$ is the energy dependent phase shift that must be calculated separately from an assumed form for the ion-core potential. For the purposes of the computer program being described here, the $\delta_{\ell}(E)$ are supplied as necessary input parameters, and the reader is referred to the literature for various approximations which find frequent use in LEED calculations.^{2,7,8}

We now write explicitly the matrix forms of equations (3) and (4). These functions can be written so that the independent variable is k , which is the absolute value of k_i and k_f , and is related to the real part of the renormalized energy of the electron in the crystal by $E = \frac{\hbar^2}{2m} k^2$.

$$\tau_{\lambda}^{LL'}(k) = \tau_{\lambda}^{LL'}(k) + \sum_{\lambda' \neq \lambda} \sum_{L'', L'''} \tau_{\lambda}^{LL''}(k) G_{L''L'''}^{\lambda\lambda'}(k_1) \tau_{\lambda'}^{L'''L'}(k) \quad (13)$$

$$\tau_{\lambda}^{LL'}(k) = \tau_{\lambda}^{LL'}(k) + \sum_{L'', L'''} \tau_{\lambda}^{LL''}(k) G_{L''L'''}^{sp}(k_1) \tau_{\lambda}^{L'''L'}(k) \quad (14)$$

There are a number of manipulations necessary in order to transform equations (3) and (4) into (13) and (14), and these are detailed in the paper by Laramore and Duke.³ The simplicity of the last two equations and their straightforward substitution in eq. (9) are offset by the complexity of the matrix forms of the propagators $G_{LL'}^{sp}(k_1)$ and $G_{LL'}^{\lambda\lambda'}(k_1)$. Explicitly they are,

$$G_{LL'}^{sp}(k_1) = \sum_{P \neq 0} e^{-k_1 \cdot P} \sum_{L''} \left\{ \frac{(-1)^{L''}}{2\pi^2} \left[\frac{-m\pi i k}{\hbar^2} \right] h_{L''}^{(1)}(kP) \cdot Y_{L''}(P) I(L, L', L'') \right\} \quad (15)$$

and

$$G_{LL'}^{\lambda\lambda'}(k_1) = \sum_{\tilde{P}} e^{i\tilde{k}_1 [\tilde{P} + \tilde{d}_{\tilde{\lambda}}, -\tilde{d}_{\tilde{\lambda}}]} \sum_{L''} \left\{ \frac{(-1)^{L''}}{2\pi^2} \left[\frac{-m\pi i k}{\hbar^2} \right] h_{L''}^{(1)}(k | \tilde{P} + \tilde{d}_{\tilde{\lambda}}, -\tilde{d}_{\tilde{\lambda}} |) \cdot Y_{L''}(\tilde{P} + \tilde{d}_{\tilde{\lambda}}, -\tilde{d}_{\tilde{\lambda}}) I(L, L', L'') \right\} \quad (16)$$

A derivation of these results is presented in reference 3.

The vector $\tilde{P}$ is the distance from the origin in a plane parallel to the surface to any of the other lattice points in the same plane.

The summation in eq. (15) extends over all such lattice points except the one at the origin. In eq. (16), the summation includes $\underline{p} = 0$ since in that case the difference between the origin positions of planes λ and λ' must be non-zero. When $\underline{d}_\lambda$ is written as a vector it denotes the three dimensional vector from the origin chosen for the problem to the origin of the λ th subplane, $\underline{d}_\lambda = (\underline{a}_\lambda, d_\lambda)$, where $\underline{a}_\lambda$ is the displacement parallel to the crystal surface and d_λ is in the z-direction (inward normal to the surface). The spherical hankel function of the first kind ($h_{\ell''}^{(1)}(kP)$) describes the phase and amplitude of an outgoing spherical wave of energy $E = \frac{\hbar^2 k^2}{2m}$. The quantity $I(L, L', L'')$ is the angular integral of a product of three spherical harmonics,

$$\begin{aligned} I(L, L', L'') &= \int d\Omega \ Y_L(\Omega) \ Y_{L'}^*(\Omega) \ Y_{L''}^*(\Omega) \\ &= [(2\ell'+1)(2\ell''+1)/4\pi(2\ell+1)]^{\frac{1}{2}} \cdot \\ &\quad C(\ell', \ell'', \ell; m', m'', m) C(\ell', \ell'', \ell; 0, 0, 0). \end{aligned} \quad (17)$$

The present section reproduces the results of the t-matrix formalism which are to be incorporated into a computer program. There are two further points to be discussed in the next two sections before we can proceed to the details of the actual program. These involve the inner potential and inelastic damping factor, and the inclusion of temperature dependence in the formalism for the reflected beam intensities.

II. The inner potential and inelastic damping.

The propagation of an electron inside a metal differs from the propagation of an electron in a vacuum. There are three principle mechanisms operating on an electron moving through a metallic crystal. The first is a surface dipole layer due to the fact that metallic electron wavefunctions extend a short distance outside the surface layer of ion cores. This gives rise to the work function which presents a barrier (typically 2 to 4 eV in height) that must be overcome by an electron if it is to escape the metal surface. Conversely, an external electron entering the surface will experience an increase in its kinetic energy upon crossing the dipole layer region. This simplified model is sufficient for use with LEED calculations, though more detailed theoretical treatments are available.⁹

The second mechanism that alters the propagation of an electron inside a metallic crystal is the presence of positively charged ion cores. The third mechanism is due to the presence of valence and conduction electrons, which tend to screen the attractive potentials due to the ion cores. The screening is a dynamic effect and is dependent on the local free electron density and the momentum of the incident electron. The behavior of an electron moving through a uniform electron gas (the positive charge is assumed to be distributed uniformly in order to maintain charge neutrality) is well understood and yields a model which is useful for the understanding of the more complex problem at hand.¹⁰

Electron states at the fermi surface of a metal are infinitely long lived. Electrons injected with higher energies form what are known as quasiparticle states, which possess finite lifetimes.¹¹ These quasiparticles may be thought of as a "bare" electron surrounded by a "cloud" of electrons which interact with it, and the dynamical behavior of such an entity is considerably more complex than that of a single free electron. The finite lifetime of a quasiparticle results when part of its energy is given up in the creation of an electron-hole pair, or by plasmon excitations. Since we are normally concerned with elastically diffracted electrons in LEED analysis, the effect of a finite lifetime is to diminish the elastic beam intensity as the beam travels through a metal. The quasiparticle energy can be expressed as the energy of a free electron whose wavevector outside the electron gas is $\tilde{k}$ plus a self-energy correction $\Sigma(\tilde{k})$, which is a complex quantity.

Figure 4 shows the relationship of the real parts of the various energies considered. ϵ is the kinetic energy of the electron far outside the crystal and ϕ is the work function. The real part of the electron self energy correction, $\Sigma_1(\tilde{k}, \epsilon)$, is a negative quantity as measured from the Fermi level. The real part of the quasiparticle energy, $E(\tilde{k}, \epsilon)$, is the effective "free particle" energy of the quasiparticle in the solid.

$$E(\tilde{k}, \epsilon) = \epsilon - \Sigma_1(\tilde{k}, \epsilon) \quad (18)$$

The imaginary part of the quasiparticle energy $\Gamma(\tilde{k}, \epsilon)$ is simply the negative of the imaginary part of the self energy correction, $\Sigma_2(\tilde{k}, \epsilon)$.

$$\Gamma(\underline{k}, \epsilon) = -\Sigma_2(\underline{k}, \epsilon) \quad (19)$$

Finally, the complex propagation vector inside the electron gas is obtained from

$$[k_1(\epsilon) + ik_2(\epsilon)]^2 = \frac{2m}{\hbar^2} [E(\underline{k}, \epsilon) + i\Gamma(\underline{k}, \epsilon)] \quad (20)$$

Lundqvist has performed calculations¹⁰ to determine $\Sigma(\underline{k}, \epsilon)$ for a uniform electron gas of different densities. The real part, $\Sigma_1(\underline{k}, \epsilon)$, is a negative quantity and can be identified with the inner potential correction familiar in LEED studies if we write $V_0(\epsilon) = -\Sigma_1(\underline{k}, \epsilon)$. The imaginary part, $\Sigma_2(\underline{k}, \epsilon)$ is a positive quantity that goes to zero at the fermi energy, and attains a nearly constant value for energies greater than twice this value. The inner potential $V_0(\epsilon)$ is nearly constant below an energy equal to twice the fermi energy, and diminishes at higher energies, approaching an asymptotic $1/k$ behavior.

It should be stressed that the foregoing discussion concerns a simplified model of the metallic crystal in which it is approximated as a uniform electron gas. While the concepts are useful, the validity of the detailed form of $\Sigma(\underline{k}, \epsilon)$ for the uniform electron gas is questionable when applied to a real metal. The positive charge in a real metal is localized at the ion cores instead of being uniformly distributed, and the excitation of core electrons presents an additional inelastic loss mechanism. In the computer program under discussion here, we have made provisions for the calculation of $k_1(\epsilon) + ik_2(\epsilon)$

from the Lundqvist model,¹⁰ and also from an optical model potential.

The optical model potential treats the real and imaginary parts of the quasiparticle energy as constants, independent of the incident electron energy ϵ over the LEED range, in which case

$$E(\tilde{k}, \epsilon) = \epsilon + V_0 \quad (21a)$$

$$\Gamma(\tilde{k}, \epsilon) = -\Sigma_2 \quad (21b)$$

Comparisons between theoretical calculations and experimental LEED profiles have not yet been able to resolve which of these treatments is most accurate. In Figure 5 we plot the (00) beam of the aluminum (100) surface using both approximations to allow direct comparison between the two approaches.

III. Temperature dependence of LEED calculations.

Laramore and Duke^{3,12} have investigated the effects of lattice vibrations on the elastic LEED beam intensities. The result of their work is that temperature effects can be quite simply accounted for by introducing a quantity

$$b_{\lambda}(k_f, k_i; E) = \exp(-W_{\lambda}(k_f - k_i)) t_{\lambda}(k_f, k_i; E) \quad (22)$$

where $W_{\lambda}(k_f - k_i)$ is the Debye-Waller factor of the lattice. If the expression for b_{λ} is inserted for t_{λ} in equations (4) and (5), the effects of thermal lattice motions on the LEED beams are properly taken into consideration.

The partial wave decomposition for b_{λ} is similar to that for t_{λ} of eq. (6a), except that b_{λ} need no longer possess the same spherical symmetry of t_{λ} and the ion-core potential.

$$b_{\lambda}(k_f, k_i; E) = \sum_L \sum_{L'} b_{\lambda}^{LL'}(k) Y_L^*(k_f) Y_{L'}(k_i) \quad (23)$$

In the model we use here, W_{λ} is taken to be a function possessing spherical symmetry; i.e., the Debye-Waller factor is independent of direction in the crystal.

$$W_{\lambda}(k_f - k_i) = W_{\lambda}(|k_f - k_i|) \quad (24)$$

In such a case, b_{λ} and t_{λ} are both of the same symmetry and the matrix $b_{\lambda}^{LL'}$ is diagonal, as is $t_{\lambda}^{LL'}$. A further simplification is achieved if we approximate the phonon spectrum by a Debye model.³

$$W_{\lambda}(\underline{k}_f - \underline{k}_i) = \frac{3h^2(\underline{k}_f - \underline{k}_i)^2}{2M k_B \Theta_D^{\lambda}} \left[\frac{1}{4} + \left(\frac{T}{\Theta_D^{\lambda}} \right)^2 \right] \int_0^{\Theta_D^{\lambda}/T} \frac{x}{e^x - 1} dx \quad (25)$$

In this equation, M is the ionic mass, k_B is the Boltzmann constant, T is the absolute temperature of the crystal and Θ_D^{λ} is the effective Debye temperature of the λ th subplane of the crystal. The propagation vectors $\underline{k}_i$ and $\underline{k}_f$ refer to the incident and final waves scattered by a given lattice site, and do not represent the initial and final wave-vectors for scattering from the entire crystal. Thus, $\underline{k}_i$ and $\underline{k}_f$ are different for each order of multiple scattering and the Debye-Waller factor cannot be factored out of the final expression as in the case of single or kinematic scattering.

The expression for W of eq. (25) can be written

$$W_{\lambda}(\underline{k}_f - \underline{k}_i) = (\underline{k}_f - \underline{k}_i)^2 \bar{W}_{\lambda}$$

Since only elastic collision are considered, $|\underline{k}_i| = |\underline{k}_f| = k$ and

$$W_{\lambda}(\underline{k}_f - \underline{k}_i) = 2k^2 \bar{W}_{\lambda} (1 - \cos \theta),$$

where θ is the angle between the vectors $\underline{k}_i$ and $\underline{k}_f$. Using this expression for W_{λ} in Eq. (22) and expanding t_{λ} in spherical harmonics, one obtains

$$b_{\lambda}(\underline{k}_1, \underline{k}_2) = e^{-2k^2 \bar{W}_{\lambda}} \sum_{\ell, m} t_{\lambda}^{\ell}(k) e^{2k^2 \bar{W}_{\lambda} \cos \theta} Y_{\ell m}^*(\underline{k}_2) Y_{\ell m}(\underline{k}_1) \quad (26)$$

The exponential term inside the summation can be expanded in a series of Legendre polynomials or of spherical harmonics,

$$e^{2k^2\bar{w}_\lambda \cos \theta} = \sum_{\ell'} A_{\ell'} P_{\ell'}(\cos \theta) = \sum_{\ell'} \frac{4\pi}{2\ell'+1} A_{\ell'} \sum_{m'=-\ell'}^{\ell'} Y_{\ell'm'}^*(k_2) Y_{\ell'm'}(k_1), \quad (27a)$$

where

$$A_{\ell'} = \left(\frac{2\ell'+1}{2} \right) \int_{-1}^1 e^{2k^2\bar{w}_\lambda x} P_{\ell'}(x) dx \quad (27b)$$

Substitution of eq. (27a) into (26) yields,

$$b_\lambda(k_1, k_2) = e^{-2k^2\bar{w}_\lambda} \sum_{\ell} t_\lambda^\ell(k) \sum_{\ell'} \frac{4\pi}{2\ell'+1} A_{\ell'} \sum_{m, m'} Y_{\ell m}^*(k_2) Y_{\ell'm'}(k_1) Y_{\ell m}(k_1) Y_{\ell'm'}^*(k_2) \quad (28)$$

The summation over m and m' is performed using standard formulas for such products¹³ and the double summation can be expressed as

$$\sum_{\ell''} \sum_{m''=-\ell''}^{+\ell''} Y_{\ell''m''}^*(k_2) Y_{\ell''m''}(k_1) \delta(\ell, \ell', \ell'') \quad (29)$$

where the delta function is unity for any combination of ℓ'' -values which satisfy the combination rule for vectors $\ell'' = |\ell - \ell'|, \dots, \ell + \ell'$, and is zero when ℓ'' is outside these bounds. The expression for b_λ can then be simplified to

$$b_{\lambda}(\tilde{k}_1, \tilde{k}_2) = \sum_{\ell'', m''} \left\{ e^{-2k^2 \bar{W}_{\lambda}} \sum_{\ell=0}^{\ell''+\ell} \sum_{\ell' = |\ell''-\ell|}^{\ell''+\ell} \left[t_{\lambda}^{\ell}(\tilde{k}) \left(\frac{4\pi}{2\ell'+1} \right) A_{\ell'} \right] \right\} \cdot$$

$$Y_{\ell'' m''}^{*}(\tilde{k}_2) Y_{\ell'' m''}(\tilde{k}_1) \quad (30)$$

The quantity inside the braces thus represents the matrix element of b_{λ} corresponding to ℓ'' and m'' , and these are the values b_{λ}^L calculated by subroutine (FILLB) to be described shortly.

References

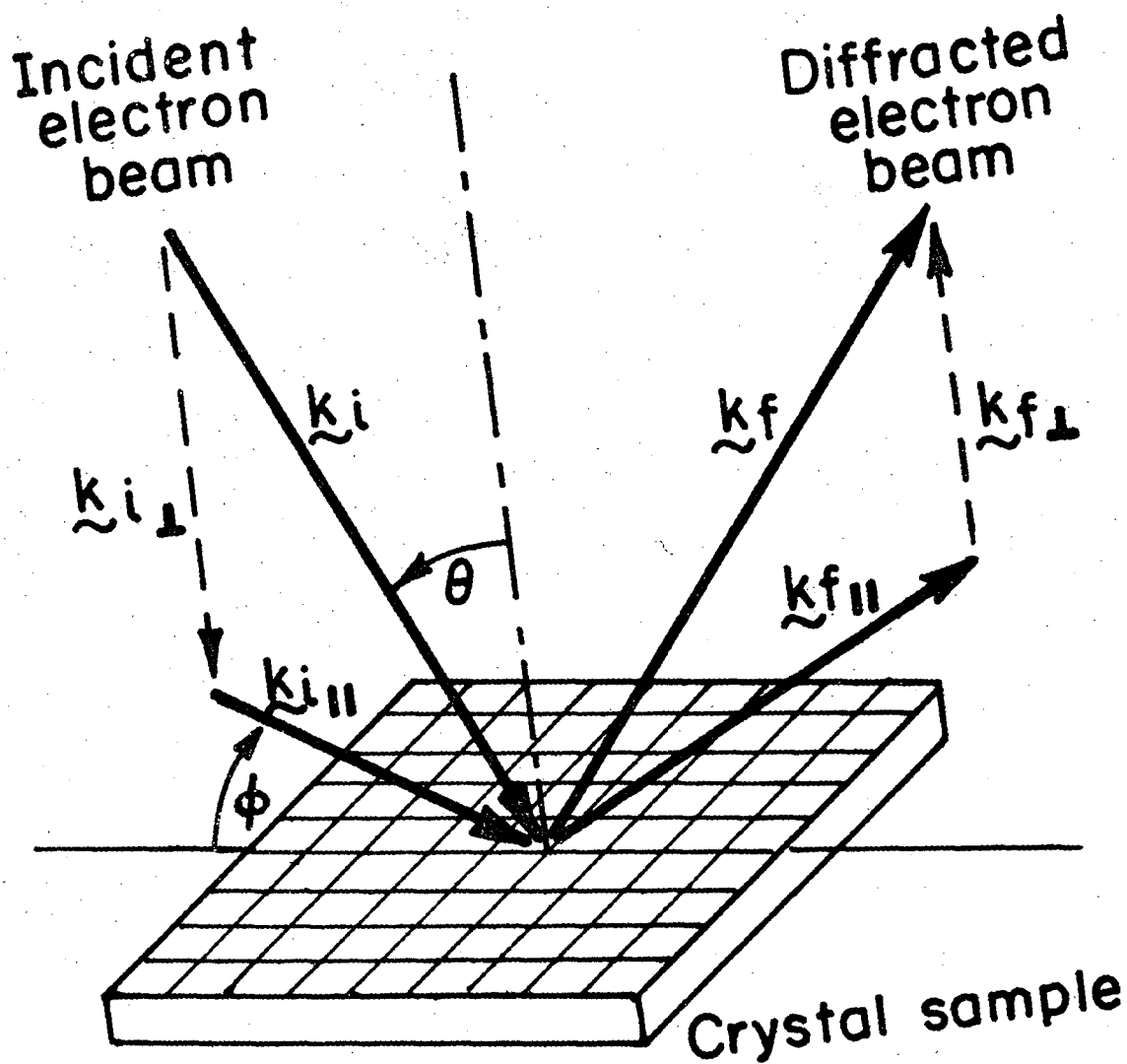
1. J. L. Beeby, J. Phys. (Proc. Phys. Soc.) [2], 1, 82 (1968).
2. C. B. Duke and C. W. Tucker, Jr., Surface Sci. 15, 231 (1969).
3. G. E. Laramore and C. B. Duke, Phys. Rev. B2, 4783 (1970).
4. J. B. Pendry, J. Phys. (Sol. St. Phys.) 4, 3095 (1971).
5. See, for example, J. D. Jackson, "Classical Electrodynamics" (p. 67, 68) Wiley, New York (1962).
6. L. I. Schiff, "Quantum Mechanics", third edition, p. 321, McGraw-Hill, New York (1968).
7. S. Y. Tong and T. N. Rhodin, Phys. Rev. Lett. 26, 711 (1971).
8. D. W. Jepsen, P. M. Marcus and F. Jona, Phys. Rev. Lett. 26, 1365 (1971); Phys. Rev. B5, 3933 (1972).
9. N. D. Lang and W. Kohn, Phys. Rev. B3, 1215 (1971).
10. B. I. Lundqvist, Phys. Stat. Sol. 32, 273 (1969).
11. D. Pines and P. Nozières, "The Theory of Quantum Liquids", Vol. 1, Benjamin, New York (1966).
12. C. B. Duke and G. E. Laramore, Phys. Rev. B2, 4765 (1970).
13. A. R. Edmonds, "Angular Momentum in Quantum Mechanics", Princeton (1957).

Figure Captions

- Fig. 1 The incident electron beam $\vec{k}_i$ and one of the diffracted beams $\vec{k}_f$, and their projections onto the surface $\vec{k}_{i\parallel}$ and $\vec{k}_{f\parallel}$ are shown. The angles θ and ϕ are used to describe the orientation of the incident beam.
- Fig. 2 The white circles represent the real space lattice positions for the surface layer of a crystal, with the central ion-core chosen as the origin. The black circles show the lattice positions of the λ th layer which are shifted with respect to the origin by a vector $\vec{a}_\lambda$ which is parallel to the surface layer.
- Fig. 3 Schematic view of the relationships between the t-matrices; (a) t_λ for a single ion-core; (b) τ_λ for the multiple scattering of $\vec{k}_i$ between all ion-cores in the λ th subplane. The subplane propagator G^{sp} accounts for the phase and amplitude of the scattered wave between the ion-core encounters; (c) T_λ for the multiple scattering between subplanes (τ_λ) including all possible scattering paths ending on layer λ . The interplane propagator $G^{\lambda\lambda'}$ accounts for the phase and amplitude of the scattered wave between subplane encounters.

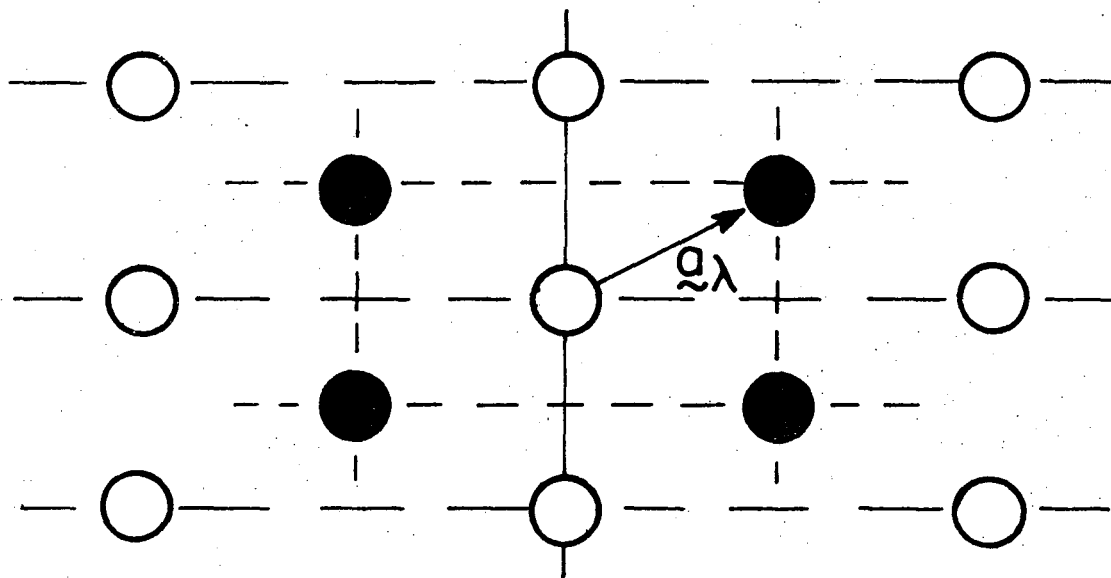
Fig. 4 Relationships between the real parts of the energies for quasiparticle motion are shown. ϵ is the kinetic energy of an electron travelling in the vacuum, and $E(k, \epsilon)$ is the quasiparticle energy inside the electron gas, where k is the propagation vector in the vacuum. The metallic work function ϕ and the real part of the renormalized electron energy inside the metal, $\Sigma_1(k, \epsilon)$, are measured with respect to the Fermi level ϵ_F .

Fig. 5 The (00) beam of the aluminum (100) surface is plotted using two approximations for the electron self energy. The solid curve uses the Lundqvist model of $\Sigma(E)$ for a free electron gas. The dashed curve uses the optical model potential with the real part of the self energy, $\Sigma_1 = -11$ eV.



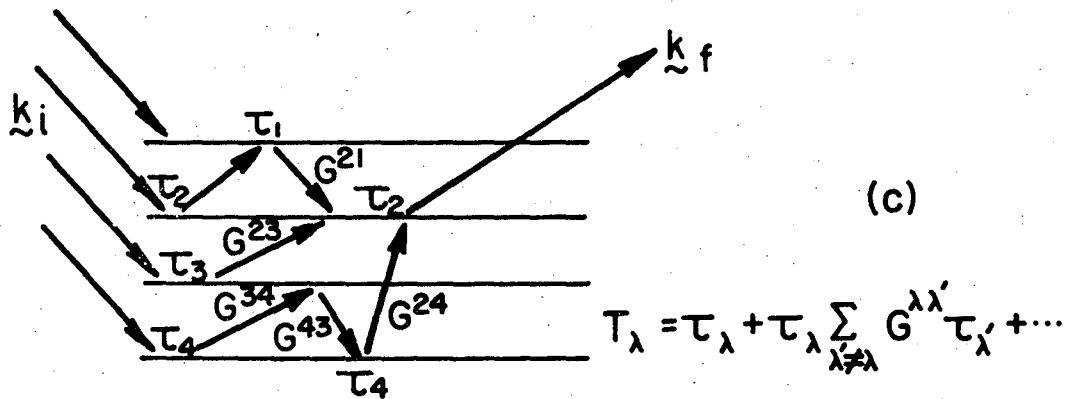
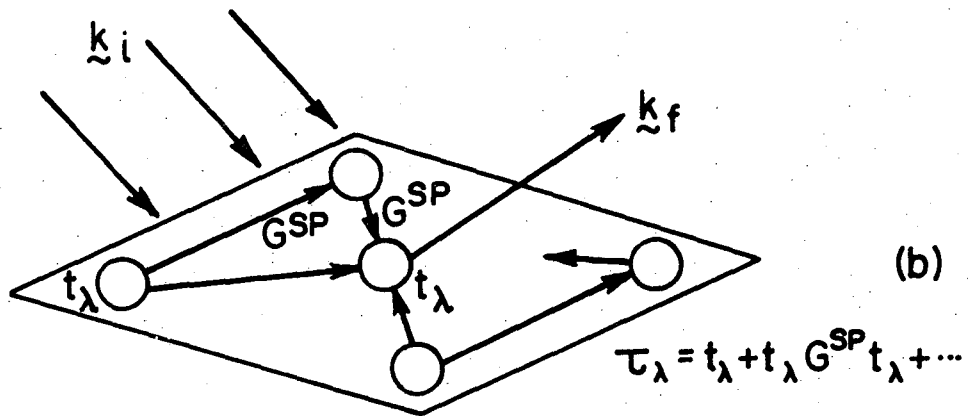
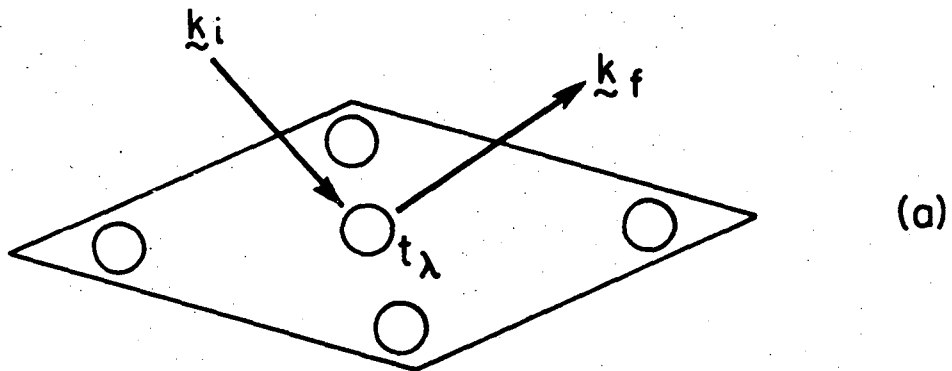
XBL7211-7333

Fig. 1



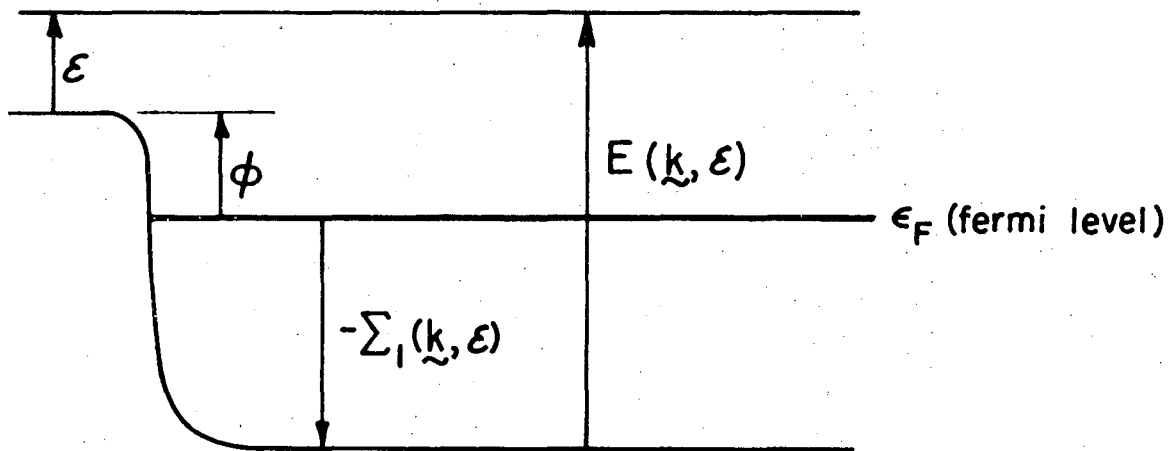
XBL 7211-7334

Fig. 2



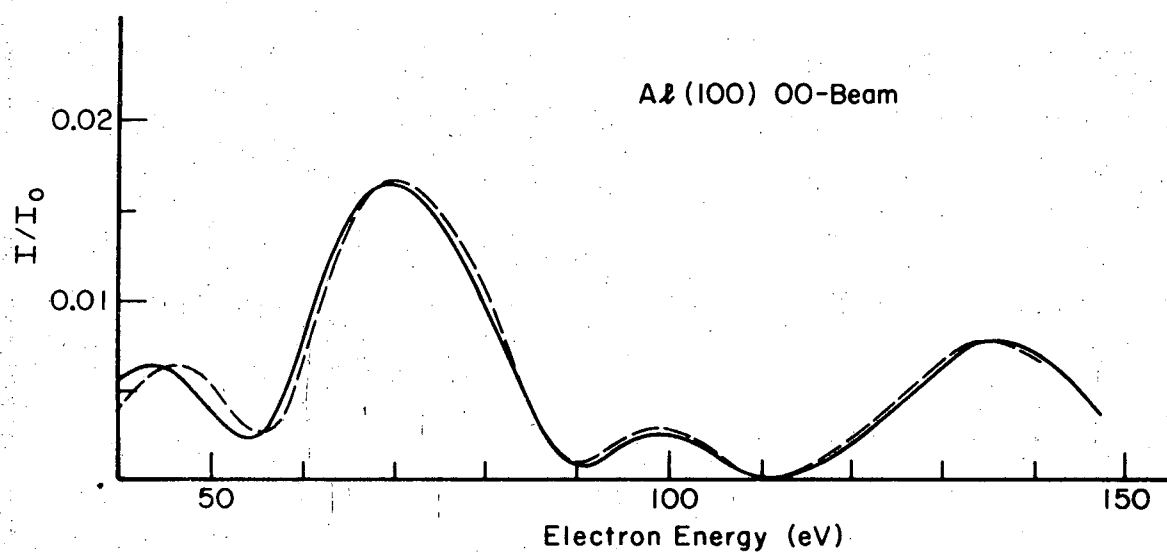
XBL7211-7335

Fig. 3



XBL 7211-7336

Fig. 4



XBL 727-6580

Fig. 5

Appendix A. Organization of the Computer Program

The LEED program which is the subject of this report has been written in Fortran IV for use on a CDC 7600 computer. Use has been made of the extended core facility of this computer and provisions will have to be made for this is a different machine is to be used. If this is to be done, care must be taken in the handling of all variables appearing in the LARGE statements.

The program is written using overlays to minimize necessary core requirements. The 0-level overlay contains a driver program (LEED) and two subroutines that are used in both overlays that are called by LEED. In addition, it contains a number of COMMON block definitions through which most of the subroutines are connected. The first overlay called consists of a program (READIN) which causes input data cards to be read and several arrays to be initialized. The second overlay (LOOP), also called by LEED, causes the multiple scattering problem to be solved at a sequence of incident electron energies as specified in the input data.

OVERLAY 0	
LEED	ENERGY
	YLM

OVERLAY 1	OVERLAY 2
READIN	LOOP
ATOMIC { STACK } { COEFT } { CLEB } LATTICE BEAM { DEBYE } { CUBINT } { GRATE } TABLE	{ FILLB } { MOX } KINEM GSTACK SCATT TSUM INV CINV INVERT EXCH ARITH TEST PERTB EXCX ARITX GREEN HANKEL

OVERLAY (0)

LEED

This is the driver program that controls the calling of the two overlays and causes punched card output to be produced. All common blocks through which data are passed between the overlays and sub-routines are defined here.

ENERGY

This subroutine is furnished with the incident electron energy (NRG) in electron volts. It then computes the real and imaginary parts of the quasiparticle energy $E(\epsilon)$ and $\Gamma(\epsilon)$ as defined in equations 17 or 19 (ESIG and SIG2). Also calculated are the real and imaginary components of the quasiparticle propagation vector $k_1(\epsilon)$ and $k_2(\epsilon)$ (XK and DAMP), and the complex renormalization constant ZNORM. The quantity NRGY is an index that is used to label the atomic t_{λ}^L -values as a function of the incident energy.

The subroutine is not completely general. The data in arrays SG1, SG2, ZRL, and ZIM are taken from the work of Lundqvist¹⁰ and refer to an electron gas whose density is that of the conduction electrons in aluminum, as does the fermi energy (EFERMI).

Provisions are also made for calculating $E(\epsilon)$ and $\Gamma(\epsilon)$ by using an optical model potential (eq. 19) in which the inner potential V_0 and the imaginary part of the self energy $i\Sigma_2$ are supplied as constant values in the input data cards (POTL and GAMA). An integer variable

read into the program causes the Lundqvist model to be calculated if
 SIGMA = 0 and the optical model potential is calculated if SIGMA = 1.

YLM

The spherical harmonics $Y_{\ell m}(\theta, \phi)$ are computed for $m = -\ell, -\ell+1, \dots, +\ell$
 for $\ell = 0, 1, \dots, LMAX$. The resulting values are stored in a complex
 array Y. The input angles (θ, ϕ) are given by specifying the three com-
 ponents of a vector $\tilde{k}$, of arbitrary magnitude, having the same angular
 directions. If CONJ equals -1, the complex conjugate of the entire
 array is taken, $Y_{\ell m}^*(\theta, \phi)$.

The linear array $Y_L(\tilde{k}) = Y_{\ell m}(\tilde{k})$ is stored using the algorithm
 $L = \ell^2 + \ell + 1 + m$, or ...

L	ℓ	m
1	0	0
2	1	-1
3	1	0
4	1	1
5	2	-2
6	2	-1
7	2	0
8	2	1
9	2	2
10	3	-3
11	3	-2
	.	
	.	
	.	

OVERLAY (1)

READIN

This program reads in the following quantities:

DATE ... alphanumeric data specifying date of calculation.

MULT = 0 for kinematic scattering

= 1 for multiple scattering

MAXR ... maximum value (in Angstroms) of radial distance
between ion cores to be included in the propagator
summations 15 and 16.

TIME = 0 ... computer time intervals for each of the major
segments of the calculation are not printed.

= 1 ... computer time intervals are printed on the output.

SIGMA = 0 ... specifies Lundqvist model for electron self-
energy.

= 1 ... specifies optical model potential for electron
self-energy.

= 2 ... specifies a constant elastic electron damping
length.

POTL ... inner potential in electron volts (a negative number)
... ignored if SIGMA = 0

GAMA ... ignored if SIGMA = 0

... imaginary part of self-energy (electron volts) if SIGMA = 1

... damping length (angstroms) if SIGMA = 2

NDIF ... the number of iterations to be carried out in the
perturbation expansion of eq. 13.

The following subroutines, each of which reads in and organizes further data, are read in by READIN:

ATOMIC
STACK
LATTICE
BEAM
DEBYE

The subroutine continues by calling ENERGY to ascertain if NLOW is within the range of energy increments for which the $t_{\lambda}^L(k)$ are supplied, and stops the program if it is too large. The energy and intensity arrays E, EN, and S are initialized to zero and the direction of the incident $\tilde{k}$ -vector is calculated and stored in V1. Then, YLM is called to store the spherical harmonic array for the incident beam in YIN.

Subroutine TABLE is now called and control is returned to LEED. At this point all arrays are prepared for one or a series of LEED calculations at various energies.

ATOMIC

The angular momentum matrix elements, $t_{\lambda}^L(k)$, defined in eq. 12 and used in the summation of eq. 8a, are stored as a function of the incident electron energy. These values are read in at 2 eV intervals and stacked in the array T(I,J,K). The index K specifies the angular momentum quantum number $\ell \sim (K=\ell+1)$, and due to the spherical symmetry assumed for the ion core scattering centers, there is no azimuthal quantum number dependence (independent of m). The J-index labels the energy

intervals, and the I-dimension of the array allows several different ion-core types to be read in to the calculation. Presently a maximum of 2 types is permitted.

This subroutine first reads ITYPE, the number of ion core types to be included in the calculations. ITYPE must presently equal 1 or 2. The maximum value of the angular momentum quantum number, ℓ , is read in as LMAX. This establishes the dimensions of all arrays in the program.

For each ion-core type, an identifying TITLE card, the atomic weight of the ion-core (ATWT) and the following parameters are read in:

LTOP = total number of ion-core t-values read in at each energy (LTOP = $\ell+1$). Note that LTOP > LMAX, otherwise the program will subsequently set LMAX = LTOP - 1.

LOWEN = lowest energy value (in eV) for which a member of the array $t_{\lambda}^{\ell}(k)$ is read in.

NNRG = total number of energy values for which array member $t_{\lambda}^{\ell}(k)$ are read in.

Finally the array T(I,J,K) is read in and the subroutine returns to READIN.

STACK

The expression $I(L,L',L'')$ is calculated by function COEFT and stored here in real array C. For each ℓ -value there are $2\ell+1$ values of the azimuthal quantum number m. Thus, the corresponding maximum value $L_{\max} = (\ell_{\max} + 1)^2$. If the quantity $I(L,L',L'')$ were to be

stored as a 3-dimensional array, its total storage requirements would be $(\ell_{\max}+1)^2 \cdot (\ell_{\max}+1)^2 \cdot (2\ell_{\max}+1)^2$. For a typical value $\ell_{\max} = 4$, this amounts to storage of 50625 numbers, many of which are zero! We reduce the size of the array C to a tractable value by storing only the non-zero values in C (NUM).

The L'' -values for which $I(L, L', L'')$ are non-zero are recorded in array LL (NUM). For given values of L and L' (eqs. 15 and 16), the number of non-zero values in the summation over L'' is stored as KX(ISTEP). ISTEP ranges from 1 to $(\ell_{\max}+1)^4$ as the one dimensional index corresponding to the two angular momentum indices L and L' . Specifically, we have

$$\text{ISTEP} = (L-1) \cdot \text{LMAX3} + L'$$

$$\text{where } \text{LMAX3} = (2\ell_{\max}-1)^2$$

$$\text{and } L = \ell^2 + \ell + m + 1$$

$$L' = \ell'^2 + \ell' + m' + 1.$$

The three arrays KX, LL and C are used in subroutine GREEN for the summations in the subplane and interplane propagator expressions (eqs. 15 and 16).

COEFT

This function delivers the expression $I(L, L', L'')$ of equation 17, which is stored by STACK in array C. It employs a library function CLEB to calculate the Clebsch-Gordan coefficients.

LATTICE

The purpose of this routine is to read in the lattice geometry and to calculate the two dimensional reciprocal lattice vectors parallel to

the surface. The following parameters are read:

TYPE ... specifies the type of crystal structure.

Allowable input in A3 format is,

BCC - for body centered cubic lattices

FCC - for face centered cubic lattices

HCP - for hexagonal close packed lattices

GRI - for graphite

FACE ... can be 100, 110 or 111 for BCC or FCC. Unnecessary input for hexagonal surfaces which can presently only be used for the 0001 face.

ALAT ... lattice constant a_0 in Angstroms.

HEX ... Distance between hexagonal stacked planes for a hexagonal structure. This quantity is ignored if the type is BCC or FCC.

LAYER ... total number of crystal layers parallel to the surface over which the calculation is to be performed. Since each such layer contains ion cores of a single type, LAYER more precisely specifies the number of subplanes.

The following corrections are read in layer by layer, starting with the outermost layer: KT, ITP, MM, SHIFT. These values cause the geometry of a given layer to be altered and the origin of the subplane to be shifted. When the program encounters a blank card, the remaining layers are left unaltered from the bulk layer geometry. If the surface is simply a truncation of the bulk structure, the first card should be blank.

KT ... denotes the ion-core type of the subplane and is stored in KTYP.

ITP ... denotes the subplane geometry type, stored in ILYR.

MM ... an array of dimension 4, these quantities specify the primitive lattice translations in a given subplane in terms of the primitive lattice vectors parallel to the surface, $\underline{A}$ and $\underline{B}$.

$$\underline{A}' = M(1) \cdot \underline{A} + M(2) \cdot \underline{B}$$

$$\underline{B}' = M(3) \cdot \underline{A} + M(4) \cdot \underline{B}$$

SHIFT(I) ... A 3-member array which expresses the deviation of the subplane. The x and y components of the shifts are specified as a decimal fraction of the lattice constant a_0 (ALAT). The z-component (I=3) is specified as a decimal fraction of the interplane distance. The coordinate system is such that z is zero at the surface and increases in the direction away from the surface into the bulk. A positive value for the z-component (SHIFT(3)) indicates an expansion over the bulk interlayer separation, and a negative value is a contraction.

The subroutine continues by filling the arrays $A(I,J)$ and $B(I,J)$. These are the two dimensional primitive lattice translations within the I th subplane. The index J specifies the x and y components for $J=1$ and 2 respectively. The location of the origin of each subplane is stored as a three dimensional vector in $D(I,J)$. Again, I specifies the subplane and $J=1,2,3$ indicates the x,y or z component of the vector. The (x,y) components of $D(I,J)$ are the vectors $\underline{a}_\lambda$ appearing in eq. 2, and the z -component is d_λ from the same equation. In eq. 16, the three dimensional vector has been referred to as $\underline{d}_\lambda = (\underline{a}_\lambda, d_\lambda)$.

Finally the reciprocal lattice vectors are calculated for a two dimensional mesh whose primitive translations, $A(I,J)$ and $B(I,J)$, have been stored earlier in this routine. Details of the surface geometry are printed on the output before a return is made to READIN.

BEAM

The angular coordinates of the incident beam are read in degrees, $\theta = \text{ITHT}$ and $\phi = \text{IPHI}$. The parameters specifying the energy values over which the calculation is to be performed are read:

NLOW = low incident electron energy value in eV.

NNUM = total number of energies to be calculated.

NDELTA = energy increment (in eV) between successive calculations.

This subroutine reads ITP and NN, which are instructions about which reciprocal lattice vectors, $\underline{g}$, are to be used in the computation of

of various LEED beams.

ITP ... indicates the type of subplane geometry (which must have previously been defined in the input to subroutine LATTICE).

NN ... an array composed of six pairs of integers. These integer pairs are the beam indices in standard notation ... i.e., (00), (01), (21), etc. Each of the pairs listed will appear in the calculation. Each set of indices refers to the reciprocal lattice vectors $\tilde{g}$ of the subplane geometry given by the corresponding parameter ITP.

The two dimensional arrays VXX(I,J), where I=1,2 are the x and y components, stores the parallel components of the k-vector $\tilde{k}_{\parallel}$ (eq. 2). For the index J=1,2,3,... components are stored for all beams specified in the input instructions.

DEBYE

This subroutine performs the operations indicated in eq. 25 and calculates the coefficient of the factor $(\tilde{k}_f - \tilde{k}_i)^2$. The quantity stored is W(I), where I is the subplane index and

$$W_I(\tilde{k}_f - \tilde{k}_i) = W(I) \cdot (\tilde{k}_f - \tilde{k}_i)^2.$$

The lattice temperature (TEMP) and the bulk Debye temperature (BULK) are read in (degrees Kelvin). The integral

$$\int_0^{\Theta^\lambda/D/T} \frac{x}{e^x - 1} dx$$

is performed numerically by calling subroutine CUBINT, which is a standard library routine.

TABLE

The summations that appear in eqs. 15 and 16 over the planar lattice vector $\tilde{P}$ are essential in order to correctly account for the scattering from each ion-core. The values $\tilde{P}$ assumes are determined by the lattice structure and are calculated in this subroutine. The vector quantities $\tilde{P}$ and $\tilde{P} + \tilde{d}_\lambda, -\tilde{d}_\lambda$ are necessary as arguments for the spherical harmonics $Y_{L''}(\tilde{P})$ and $Y_{L''}(\tilde{P} + \tilde{d}_\lambda, -\tilde{d}_\lambda)$. These values are stored serially in the array RVEC, where the x, y and z components of $\tilde{P}$ and $\tilde{P} + \tilde{d}_\lambda, -\tilde{d}_\lambda$ occur in that order. Since the direction of the incident beam $\tilde{k}_1$ is fixed during a calculation over a number of incident energies ($|\tilde{k}| = \sqrt{\frac{2m}{\hbar^2}} E$), we store the scalar product $\tilde{k}_1^0 \cdot \tilde{P}$ and $\tilde{k}_1^0 \cdot (\tilde{P} + \tilde{d}_\lambda, -\tilde{d}_\lambda)$ in array XMAG.* Finally, the magnitudes $|\tilde{P}|$ and $|\tilde{P} + \tilde{d}_\lambda, -\tilde{d}_\lambda|$ are stored in RMAG for use as an argument to the hankel function $h_{\ell''}^{(1)}(k|\tilde{P} + \tilde{d}_\lambda, -\tilde{d}_\lambda|)$. The array dimensions required by the program are printed and compared with the actual storage allotted in each case for arrays C, LL, KX, RMAG, XMAG, RVEC and G.

* $\tilde{k}_1^0$ is the value of the incident wave vector calculated for the lowest energy specified, NLOW.

OVERLAY (2)

LOOP

This overlay performs a series of intensity calculations over a number of incident electron energies. The number of energy values is supplied by NNUM. The lowest energy calculated is NLOW and the increment between energies is NDELTA. The variable NRG is set equal to the energy value being calculated in a given pass through the loop.

Subroutine ENERGY is called and calculates the real and imaginary parts of the renormalized electron energy and propagation vector. It also supplies NRGY, which is the index of the array $t_g(E)$ corresponding to the value $E = \text{NRGY}$. From this the incident electron propagation vector $V1$ is computed and stored. The three components depend on the incident beam angles and the real part of the propagation vector inside the crystal. The spherical harmonics corresponding to the incident direction are computed by YLM and stored in YIN.

The reciprocal lattice vectors VXX have been calculated previously (subroutine BEAM) for a specified set of beam indices. These are added to the x,y-components of the incident beam $V1$ to yield the outgoing propagation vector components $V2$, $V2 = k_{1||} + g$. The third component of $V2$ (in the perpendicular direction) is calculated from the condition that the scattering be elastic. The outgoing wavevector $V2$ defines a direction which is used as input to subroutine YLM to calculate the outgoing spherical harmonics $YOUT = Y_L^*(k_f)$.

Subroutine FILLB is called to compute the array $BBB = b_{\lambda}^{\ell}$ as defined by eq. 30. This corrects the ion-core t-matrices (t_{λ}^{ℓ}) for finite temperature effects. Subroutine GREEN(1) is called to compute the subplane propagator $G_{sp}^{LL'}$, which is stored in a linear array (G) in extended core memory of the CDC 7600.

PERTB performs a perturbation series expansion using the subplane propagator and the quantities b_{λ}^{ℓ} in a manner similar to that shown in eq. 5a.

$$\tau_{\lambda} = b_{\lambda} + b_{\lambda} G_{sp}^{sp} b_{\lambda} + b_{\lambda} G_{sp}^{sp} b_{\lambda} G_{sp}^{sp} b_{\lambda} + \dots$$

The same quantity (τ_{λ}) is then computed by an exact matrix inversion method (eq. 5b) by calling INV. The matrix inversion result is always employed, but both results are printed on the output. This is a check on the convergence of the iterative expansion method. If the two values for τ_{λ} so calculated are in serious disagreement it is an indication that the iterative method of solving for $T_{\lambda}^{LL'}$ (eq. 13) may be non-convergent, and that a complete matrix inversion treatment of the problem may be necessary.

The values $T_{\lambda}^{LL'}$ are calculated in the single scattering (kinematic) approximation by KINEM. Using these values, subroutine TSUM calculates the differential scattering cross section for each specified beam using equations 1 and 2.

If a multiple scattering computation is requested (MULT > 1), all interplane propagators are calculated by GREEN and stored linearly in array G.

The second order diffraction computation is performed by sub-routine GSTACK, after which TSUM calculates the cross section for each beam. Subsequent iterations are made (to the diffraction limit specified by NDIF) by subroutines SCATT and TSUM.

If the parameter TIME $\neq$ 0, central processor times in seconds are stored and printed at the end of each LOOP calculation. This indicates the amount of time spent in performing each of the above steps.

At the end of the series of energy loop calculations the final results for the scattering cross sections of each beam, S, as a function of the incident beam energy, E, are printed as output.

Control is then passed back to the main program LEED which punches the same output if requested.

FILLB

The array BBB corresponding to the temperature renormalized ion-core t-matrices, $b_{\lambda} = e^{-W_{\lambda}(\tilde{k}_f - \tilde{k}_i)} t_{\lambda}$, is computed in this subroutine. Each of the diagonal matrix elements b_{λ}^L can be written as a linear combination of the diagonal unrenormalized matrix elements t_{λ}^L ,

$$b_{\lambda}^{\ell} = e^{-\beta} \sum_{\ell''=0}^{\infty} \sum_{\ell'=\left|\ell-\ell''\right|}^{\ell+\ell''} B_{\ell'} t_{\lambda}^{\ell''}$$

The first call to this subroutine computes an array CP of the coefficients of the Legendre polynomials. Using these values the coefficients $e^{-\beta} B_{\ell}$, are computed and stored in array BCF. Subroutine MOX performs the summation and delivers the result b_{λ}^{ℓ} . Since this value is (2m+1)-fold

degenerate (depends only on the ℓ quantum number and not on m), the final array BBB is filled including the degeneracies.

KINEM, GSTACK and SCATT

These three subroutines are used with TSUM to calculate values of $T_\lambda(k_f, k_i; E)$ for each beam. KINEM is used to obtain T_λ in the kinematic scattering approximation whereby this quantity is simply replaced by the ion-core t-matrix renormalized for lattice motion,

$$T_\lambda = \sum_{LL'} b_\lambda^{LL'} \delta_{LL'} Y_L^*(k_f) Y_{L'}(k_i)$$

GSTACK is called to compute $T_\lambda^{LL'}$ in the approximation of double diffraction. The quantities $Q_\lambda^{L''L'}$ and $S_\lambda^{L''L'}$ are stored linearly in a single complex array G, which resides in extended core memory of the CDC 7600 computer.

$$Q_\lambda^{L''L'} = \delta^{L''L'} + S_\lambda^{(2)L''L'}$$

where
$$S_\lambda^{(2)L''L'} = \sum_{\lambda' \neq \lambda} \sum_{L'''} G_{\lambda\lambda'}^{L''L'''} \tau_{\lambda'}^{L''L'}$$

The arrays $Q_\lambda^{L''L'}$ are passed to TSUM where T_λ is obtained from

$$T_\lambda = \sum_{L, L'} \sum_{L''} \tau_\lambda^{LL''} Q_\lambda^{L''L'} Y_L^*(k_f) Y_{L'}(k_i)$$

SCATT adds one more term to the perturbation expansion for $Q_\lambda^{L''L'}$. The previous final term is $S_\lambda^{L''L'}$ and the additional term becomes

$$S_{\lambda}^{(3)L''L'} = \sum_{L,L''} \sum_{\mu \neq \lambda} G_{\lambda\mu}^{L''L'''} \tau_{\mu}^{L'''L} S_{\mu}^{(2)LL'}$$

The process is repeated for each iteration (i) and one obtains $S_{\lambda}^{(i+1)L''L'}$ from each previous $S_{\lambda}^{(i)L''L'}$. The matrices delivered to TSUM are obtained from the summation

$$Q_{\lambda}^{(i)L''L'} = \delta^{L''L'} + S_{\lambda}^{(2)L''L'} + S_{\lambda}^{(3)L''L'} + \dots + S_{\lambda}^{(i)L''L'}$$

TSUM (ITR, FTR, JINDX)

If the iteration ITR = 1, this subroutine uses the values $T_{\lambda}(k_f, k_i; E)$ for each beam obtained from the kinematic approximation of KINEM. The scattering cross section is then calculated for each beam by means of equations 1 and 2.

If ITR > 1, the quantities T_{λ} are recomputed from the intermediate arrays $Q_{\lambda}^{L''L'}$ supplied by subroutines GSTACK and SCATT.

$$T_{\lambda} = \sum_{L,L'} \sum_{L''} \tau_{\lambda}^{LL''} Q_{\lambda}^{L''L'} Y_{L'}^*(k_f) Y_{L''}(k_i)$$

These values are then used to recompute the beam scattering cross sections from equations 1 and 2.

INV

This subroutine performs the matrix inversion for the subplane scattering matrix (see equation 5b).

$$\tau_{\lambda}^{LL'} = b_{\lambda}^L (I - G_{sp} b_{\lambda})_{LL'}^{-1}$$

Array $G_{sp}^{LL'}$ must appear in the first $(\ell_{max}+1)^4$ locations of the complex array G (stored in extended core memory). This subroutine computes the matrix in parentheses and stacks it rowwise in the singly dimensioned array Q (also in extended core memory).

Subroutine CINV is called to perform the actual inversion of the matrix Q. The inverted matrix is then delivered in array Q, this time stacked columnwise. The matrix elements $\tau_{\lambda}^{LL'}$ are obtained by multiplying b_{λ}^L by Q. Subroutine TEST is called to print out results.

TEST

This is an auxiliary subroutine used to calculate and print out values for τ_{λ} . This can be called by INV and PERTB so that the matrix inversion method and the iterative method of obtaining τ_{λ} can be directly compared.

CINV

A complex square matrix with $(\ell_{max}+1)^2$ elements on a side is supplied by the real array Q in the extended core memory of the CDC 7600 computer. Real and imaginary arrays are constructed, and the complex matrix is inverted. All computations are performed directly from the extended core storage locations using the following subroutines, EXCH, INVERT, ARITH.

The resulting inverted array is stored columnwise in array Q. The original array, Q, to be inverted, must be stored rowwise in the singly dimensioned array.

EXCH(N1,N2)

The real matrix whose first word address is Q(N2+1) is transposed and stored at Q(N1+1). N1 may equal N2.

INVERT(N1)

A real matrix inversion is performed on the matrix stored at Q(N1+1). The inverted matrix is returned in the same location.

ARITH(M1,OP,M2,M3,SIGN)

OP may equal 1 indicating a matrix addition, or 2, indicating a matrix multiplication, between the two real matrices stored at Q(M1+1) and Q(M2+1). The result is delivered in location Q(M3+1).

SIGN can equal +1.0 or -1.0 and can be used to change the sign of the product matrix when a multiplication is performed.

Care must be taken when OP=2 is called for that the matrix at Q(M1+1) is stored rowwise and that stored at Q(M2+1) is columnwise. The resultant matrix at Q(M3+1) is delivered columnwise.

PERTB

This subroutine performs a perturbation expansion (see eq. 5a) to obtain the matrix $\tau_{\lambda}^{LL'}$. The same matrix is calculated by a matrix inversion process (INV). This subroutine may be removed from the program along with the Fortran instruction that calls it in program LOOP. It is retained solely for purposes of checking the convergence

of the iterative expansion as an approximation to the true matrix inversion method.

Use is made of subroutines EXCX, which transposes a matrix stored in complex array G, and ARITX which performs matrix additions and multiplications. EXCX and ARITX are the counterparts of EXCH and ARITH that operate on the large core array G (instead of Q). Up to 20 iterations are performed, with the computation terminating when the convergence of successive iterations is within 10^{-4} .

GREEN (INDEX, IBEGIN)

This subroutine is called repeatedly by LOOP to calculate the complex matrices G_{LL}^{sp} and $G_{LL}^{\lambda\lambda'}$. When INDEX=1, the subplane propagator G_{LL}^{sp} is computed from eq. 15 and stored in the first $(\ell_{\max}+1)^4$ locations of array G (stored in extended core memory). When INDEX > 1, the array $G_{LL}^{\lambda\lambda'}$ is stored beginning at G(IBEGIN+1) and is calculated from eq. 16.

The positions of all ion-cores have previously been calculated from the lattice geometry and stored in arrays RMAG and RVEC (See subroutine TABLE). The spherical harmonics $Y_{L''}(\tilde{P} + \tilde{d}_{\lambda}, -\tilde{d}_{\lambda})$ are computed and stored in array YINT. At each value for the incident energy, each lattice point must be summed over to obtain its contribution to the subplane or interplane propagators. Due to the necessity of computing a complete set of spherical harmonics and the spherical hankel function $h_{\ell''}^{(1)}(k|\tilde{P} + \tilde{d}_{\lambda}, -\tilde{d}_{\lambda}|)$ at each such point,

the major portion of the computing time for a given problem is spent within this subroutine and auxiliary routines called by it.

HANKEL

The spherical hankel functions $h^{(1)}_{\lambda}(K|P+d_{\lambda}, -d_{\lambda}|)$ needed to evaluate the expression in eq. (16) are calculated for use in subroutine GREEN. They are delivered by the complex array H.

	PROGRAM LEEC(INPUT,CUTPUT,PUNCH)	LEED.3
C		LEED.4
	CCMPCN/BUFF/LCG,ABC,KKK	LEED.5
	CCMPCN/BLCCKA/A(10,2),B(10,2),D(10,3),RA(10,2),RB(10,2),ALAT,SPACEA,2	A.3
2	BASIS(7),NPASIS	B.2
	CCMPCN/BLCCKE/T(2,2C,7),ILYR(10),LSYM(10)	B.3
	CCMPLEX T	C.2
	CCMPCN/BLCCKC/V1(3),V2(3,12),V3(3,12),VX,VY,VZ,SQK(12),GCK(12),	C.3
2	YIN(49),YOUT(49,12),XKZERO,AREAZ	C.4
	CCMPLEX YIN,YOUT	D.2
	CCMPON/BLCCKD/NLCW,ANUM,NDELTA,ARG,NRGY,LAYER,AUMMAX,NCIF,IST(6,5)	D.3
2	,LR	E.2
	CCMPCN/BLCCKE/LMAX,LTCP,LMAX1,LMAX2,LMAX3,LMAX4,LX	E.3
2	,ICNE,ITWO,ITHREE	F.2
	CCMPCN/BLCCKF/ITHT,IPHI,THTT,DIST,CSTH(12),VXX(2,12),KNLP,JM(12),	F.3
2	JA(12),IQ(12),KC(12)	G.2
	CCMPON/BLCCKG/ESIG,SIG2,XK,DAMP,ZNORM,PCTL,GAPA,SIGMA,LCWEN,NNRG	G.3
2	,LAMBDA	G.4
	REAL LAMBDA	G.5
	INTEGER SIGMA	G.6
	CCMPLEX ZACRM	H.2
	CCMPCN/BLCCKH/KX(24C1),C(7532),LL(7532),YINT(169)	H.3
	CCMPLEX YINT	J.2
	CCMPCN/BLCCKJ/MAXX(51),RMAG(9000),XMAG(9000),MAXR	K.2
	CCMPON/BLCCKK/BBB(36,2),TAU(36,36,2),TT(10,12),KTYP(10)	K.3
	CCMPLEX PBB,TAL,TT	L.2
	CCMPCN/BLCCKL/ARG,H(169),XD(9),PATH1(10),PATH2(10,12)	L.3
	CCMPLEX ARG,H	M.2
	CCMPCN/BLCCKM/RATIC,XCON,KSTAR	M.3
	CCMPLEX KSTAR,XCON	N.2
	CCMPCN/BLCCKN/F(51),EN(51),S(12,51)	X.2
	CCMPON/BLCCKX/TIME,ITEM,MLLT,ISCALE,IMAX,ITYPE,LSUM,ILCCP,IPWR	X.3
	INTEGER TIME	DEH.2
	CCMPON/BLCCKH/W(10),ATHT(10),TEPP,SUMF,BULK,CIN(10),COUT(10,12)	LEED.21
	INTEGER HAH(9)	CONST.2
	CCMPCN/CJNST/PI,PI4,PI42,CON1,CCN2,CCN3,CON4,CCNSTT,IM,CNST	CONST.3
	CCMPLEX IM	LARGE.2
	LARGE C.QM(1296),RVEC(30000),C.G(49000)	DIM.2
C		DIM.3
C	DIMENSIONS OF KX,C,AND LL.	DIM.4
	NM3=24C1 \$ NM1=7532 \$ NM2=7532	DIM.5
C	DIMENSIONS OF RMAG AND XMAG	DIM.6
5	NM4=9000 \$ NM5=9000	DIM.7
C	DIMENSIONS OF RVEC AND G-ARRAY IN LARGE CORE MEMORY.	DIM.8
6	NM6=30000 \$ NM7=49000	DIM.9
C		LEED.25
C	PROGRAM HANDLES MAXIMUM OF 10 LAYERS	LEED.26
C	LMAX=5	LEED.27
C		LEED.28
C		LEED.29
11	LX=36	LEED.30
11	ISCALE=1	LEED.31
12	IMAX=51	LEED.32
13	IM=CCFLX(C.C,1.0)	LEED.33
16	PI=3.141592654	LEED.34
17	PI4=4*PI	LEED.35
21	PI42=PI4**2	LEED.36

```

23      CCN1=2.*PI/SCRT(150.0)          LEFC.37
27      CCN2=PI/.50.0                  LEFC.38
30      CCN2=PI/1EC.0                  LEFC.39
32      CCN4=CCN2**2                   LEFC.40
33      CCNSTT=CCN1**4*PI4**3*PI       LEFC.41
36      CAST=PI42*CCN1**4              LEFC.42
40      ITEM=C                          LEFC.43
41      READ 810, JPUNCH, HAH          LEFC.44
      C                                LEFC.45
51      100 CONTINUE                   LEFC.46
      C                                LEFC.47
      C      OVERLAY(1,C)  READS IN PARAMETERS AND INITIALIZES ARRAYS ***** LEFC.48
      C                                LEFC.49
      C                                LEFC.50
51      CALL CVERLAY(2LT1,1,0,6HMECALL) LEFC.51
54      IF(ITEM.EC.2) STCP              LEFC.52
60      IF(ITEM.NE.5) GO TO 110         LEFC.53
62      PRINT 800                       LEFC.54
65      STCP                            LEFC.55
      C                                LEFC.56
70      110 CONTINUE                   LEFC.57
      C                                LEFC.58
      C      OVERLAY(2,0)  PERFORMS ENERGY LOOP CALCULATIONS ***** LEFC.59
      C                                LEFC.60
70      CALL CVERLAY(2LT1,2,0,6HMECALL) LEFC.61
      C                                LEFC.62
      C      PUNCH OUTPLT  (LEED CURVES) LEFC.63
      C                                LEFC.64
73      IF(JPUNCH.EC.0) GO TO 120      LEFC.65
74      PUNCH 812, HAH                  LEFC.66
102     DO 115 I=1,KNUM                 LEFC.67
104     PUNCH 802, IQ(I),JM(I),JN(I),LMAX,LAYER,ITHT,IPHI,ANUM LEFC.68
132     PUNCH 801, (E(J),S(I,J), J=1,NNUM) LEFC.69
153     115 CCATINUE                   LEFC.70
156     120 CCNTINUE                   LEFC.71
      C                                LEFC.72
      C                                LEFC.73
      801 FORMAT( 4(F7.2,E13.5) )      LEFC.74
      802 FORMAT( 2X*TYPE=*12,* BEAM(*213,*) L=*12,* LYR=*12,* THETA=*13, LEFC.75
      2 * PHI=*13,* NO. OF POINTS =* 13 ) LEFC.76
      810 FORMAT( 5X 12, 5A8 )         LEFC.77
      812 FORMAT( 5A8 )                LEFC.78
      C                                LEFC.79
      800 FORMAT( // 10X*$$$$$ RUN ABORTED $$$$$ ARRAY DIMENSIONS EXCEEDED. LEFC.80
      2 * )                             LEFC.81
      C                                LEFC.82
156     END

```

PROGRAM LENGTH INCLUDING I/O BUFFERS
001776

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

100	-	000052	110	-	000071	120	-	000157	800	-	000237
801	-	000214	802	-	000217	810	-	000232	812	-	000235

	SUBROUTINE ENERGY(ANG,NRGY)	ENERGY.2
	COMMON/BLCKG/ESTG,SG2,XX,DAMP,ZNCRN,PCTL,GAMA,SIGMA,LCWEN,NNRG	G.2
2	,LAMECA	G.3
	REAL LAMHCA	G.4
	INTGCR SIGMA	G.5
	COMPLEX ZNCRN	G.6
C		ENERGY.4
	COMPLEX Y,Z	ENERGY.5
	DIMENSION SG1(13),SG2(13),ZKL(13),ZIM(13)	ENERGY.6
	DATA SG1/.83,.85,.85,.75,.60,.58,.53,.48,.44,.41,.38,.36,.34/	ENERGY.7
	DATA SG2/.04,.15,.33,.40,.43,.44,.44,.44,.43,.41,.40,.39,.37/	ENERGY.8
	DATA ZPL/.0,.70,.82,.80,.90,.92,.93,.94,.95,.96,.96,.96,.97/	ENERGY.9
	DATA ZIM/.0,.35,.30,.20,.23,.21,.19,.17,.16,.14,.13,.13,.12/	ENERGY.10
	PI=3.141592654	ENERGY.11
5	LCAL=2*PI/SQRT(150.)	ENERGY.12
12	CCNVRG=1.0E-04	ENERGY.13
C		ENERGY.14
13	FFERMI=11.9	ENERGY.15
C		ENERGY.16
C		ENERGY.17
15	IF(SIGMA.EQ.1) GO TO 150	ENERGY.18
20	IF(SIGMA.EQ.2) GO TO 170	ENERGY.19
22	XKK=0.0	ENERGY.20
22	DPP=0.0	ENERGY.21
23	XX=SQRT(NRG/FFERMI)	ENERGY.22
C		ENERGY.23
31	100 CONTINUE	ENERGY.24
31	I=1	ENERGY.25
32	X=1.50	ENERGY.26
34	110 CONTINUE	ENERGY.27
34	IF(X.GE.+.5) GO TO 145	ENERGY.28
37	IF(XX.LE.X) GO TO 120	ENERGY.29
41	X=X+0.25	ENERGY.30
42	I=I+1	ENERGY.31
44	GO TO 110	ENERGY.32
45	120 CONTINUE	ENERGY.33
C		ENERGY.34
45	D=X-XX	ENERGY.35
47	IF(X.LE.1.50) I=2	ENERGY.36
55	SC1=SG1(I)-SG1(I-1)	ENERGY.37
57	SC2=SG2(I)-SG2(I-1)	ENERGY.38
61	Z1=ZKL(I)-ZKL(I-1)	ENERGY.39
63	Z2=ZIM(I)-ZIM(I-1)	ENERGY.40
65	C1=4*D*SC1	ENERGY.41
67	C2=4*D*SC2	ENERGY.42
70	O3=4*D*Z1	ENERGY.43
72	O4=4*D*Z2	ENERGY.44
73	IF(X.LE.1.50) I=1	ENERGY.45
77	EEEE=(SG1(I)-C1)*FFERMI+NRG	ENERGY.46
103	SSSS=(SG2(I)-C2)*FFERMI	ENERGY.47
106	Y=CMPLX(EEEE,SSSS)	ENERGY.48
111	Z=CCN1*CSQRT(Y)	ENERGY.49
122	XY=REAL(Z)	ENERGY.50
122	DCCC=AIMAG(Z)	ENERGY.51
124	IF(ABS(XY-XKK).LE.CCNVRG) GO TO 140	ENERGY.52
132	130 CONTINUE	ENERGY.53
132	DPP=DCCC	ENERGY.54
134	XKK=XY	ENERGY.55

136	XX=SQRT(EEEE/EFERMI)	ENERGY.56
143	GC TO 100	ENERGY.57
	C	ENERGY.58
145	140 CONTINUE	ENERGY.59
145	IF(ABS(CCCC-PPP).LE.CONVRG) GO TO 150	ENERGY.60
152	GC TO 130	ENERGY.61
152	145 CONTINUE	ENERGY.62
152	EEEE=C.C	ENERGY.63
	C	ENERGY.64
153	150 CONTINUE	ENERGY.65
154	ZR= ZRL(I)-C3	ENERGY.66
156	ZI=-(ZIM(I)-C4)	ENERGY.67
160	ZACRM=CMPLX(ZR,ZI)	ENERGY.68
	C	ENERGY.69
	C	ENERGY.70
	C	ENERGY.71
163	GC TO 180	ENERGY.72
	C	ENERGY.73
	C	ENERGY.74
	C	ENERGY.75
163	160 CONTINUE	ENERGY.76
	C	ENERGY.77
163	EEEE=NRG-PCTL	ENERGY.78
165	SSSS=GAMA	ENERGY.79
167	Y=CMPLX(EEEE,SSSS)	ENERGY.80
171	Z=CCN1*CSQRT(Y)	ENERGY.81
202	XY=REAL(Z)	ENERGY.82
202	CCCC=A*IMAG(Z)	ENERGY.83
204	ZACRM=CMPLX(1.0,0.0)	ENERGY.84
207	GC TO 180	ENERGY.85
	C	ENERGY.86
	C	ENERGY.87
	C	ENERGY.88
211	170 CONTINUE	ENERGY.89
211	EEEE=NRG-PCTL	ENERGY.90
213	AK1=CCN1**2*EEEE	ENERGY.91
216	AK2=CCN1*SQRT(EEEE)/LAMBDA	ENERGY.92
221	Y=CMPLX(AK1,AK2)	ENERGY.93
224	Z=CSQRT(Y)	ENERGY.94
227	XY=REAL(Z)	ENERGY.95
230	CCCC=A*IMAG(Z)	ENERGY.96
231	ZACRM=CMPLX(1.0,C.C)	ENERGY.97
234	PRINT 900, LAMBDA	ENERGY.98
	900 FORMAT(/ 10X*LAMBDA=* F10.5)	ENERGY.99
245	GC TO 180	ENERGY.100
	C	ENERGY.101
	C	ENERGY.102
	C	ENERGY.103
245	180 CONTINUE	ENERGY.104
246	A=EEEE/2	ENERGY.105
250	N=2*N	ENERGY.106
252	IF((EEEE-N).GT.1.0) N=N+2	ENERGY.107
257	NRGY=(A-LCWRN)/2+1	ENERGY.108
262	IF(NRGY.LT.1) NRGY=0	ENERGY.109
264	IF(NRGY.GT.ANRG) NRGY=0	ENERGY.110
267	ESIG=EEEE	ENERGY.111
270	SIG2=SSSS	ENERGY.112
272	XK=XY	ENERGY.113

273 DAMP=DDDD
 C
 C
 275 RETURN
 276 END

ENERGY.114
 ENERGY.115
 ENERGY.116
 ENERGY.117
 ENERGY.118

SUBPROGRAM LENGTH
 000501

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

100	-	000032	110	-	000035	120	-	000046	130	-	000134
140	-	000146	145	-	000153	150	-	000154	160	-	000164
170	-	000212	180	-	000246	900	-	000315			

BLOCK NAMES AND LENGTHS
 BLOCKG - 000014

VARIABLE ASSIGNMENTS

AK1	-	000476	AK2	-	000477	CONVRG	-	000450	CON1	-	000447
D	-	000457	DAMP	-	000003501	DDDD	-	000473	DPP	-	000453
D1	-	000464	D2	-	000465	D3	-	000466	D4	-	000467
EEEE	-	000470	EFERMI	-	000451	ESIG	-	000000501	GAMA	-	000007501
I	-	000455	LAMBOA	-	000013501	LOWEN	-	000011501	N	-	000500
NNRG	-	000012501	PI	-	000446	POTL	-	000006501	SD1	-	000460
SD2	-	000461	SG1	-	000362	SG2	-	000377	SIGMA	-	000010501
SIG2	-	000001501	SSSS	-	000471	X	-	000456	KK	-	000002501
XKK	-	000452	XX	-	000454	XY	-	000472	Y	-	000356
Z	-	000360	ZI	-	000475	ZIM	-	000431	ZNORM	-	000004501
ZR	-	000474	ZRL	-	000414	Z1	-	000462	Z2	-	000463

START OF CONSTANTS
 000301

START OF TEMPORARIES
 000322

START OF INDIRECTS
 000344

UNUSED COMPILER SPACE
 123600

[illegible]

70	INC=0	YLM.60
71	LX2=LMAX, **2	YLM.61
73	CC TC I=1, LX2	YLM.62
101	Y(1)=(0., 0.0)	YLM.63
102	105 CCNTINUE	YLM.64
	C	YLM.65
110	110 CCNTINUE	YLM.66
110	X=CMPLX(CPH, SPH)	YLM.67
	C	YLM.68
	C	YLM.69
	C	YLM.70
113	P(1)=1.0	YLM.71
114	Y(1)=1./SQRT(PI4)	YLM.72
124	P(2)=CTH	YLM.73
125	Y(2)=SQRT(3.0/PI4)*P(2)	YLM.74
	C	YLM.75
137	CC 120 L=2, LMAX	YLM.76
141	L2=L-1	YLM.77
142	L1=L+1	YLM.78
143	LL=L*(L+1)+1	YLM.79
146	CFT=SQRT((2*L+1)/PI4)	YLM.80
161	P(LL)=((2*L2+1)*CTH*P(L1)-L2*P(L2))/L	YLM.81
173	Y(LL)=CFT*P(LL)	YLM.82
176	120 CCNTINUE	YLM.83
200	IF(INC.EQ.0) GC TC 140	YLM.84
	C	YLM.85
	C	YLM.86
	C	YLM.87
201	XX=(1.0, 0.0)	YLM.88
204	CC 130 M=1, LMAX	YLM.89
205	XX=XX*X	YLM.90
213	SIGN=(-1.0)**M	YLM.91
216	NN=(M+1)**2	YLM.92
220	NN=NN-2*M	YLM.93
222	M2=2*M+1	YLM.94
223	CF1=DSQRT(M2*FAC(M2)/PI4)	YLM.95
252	CF2=SIGN*FAC(M+1)*2.0**M	YLM.96
266	Y(NN)=CF1/CF2*STH**M*XX	YLM.97
304	Y(NM)=SIGN*CCNJG(Y(NN))	YLM.98
315	IF(M.EC.LMAX) GO TO 140	YLM.99
	C	YLM.100
317	CC 130 L=M, LMX	YLM.101
320	L1=L+1	YLM.102
321	CL1=(2*L+3)*(L-M+1)	YLM.103
326	CL2=(2*L+1)*(L+M+1)	YLM.104
333	CF1=SQRT(CL1/CL2)	YLM.105
340	CL1=L+M	YLM.106
342	CL2=L-M+1	YLM.107
344	CF2=SQRT(CL1/CL2)	YLM.108
	C	YLM.109
354	L2=L*(L+1)	YLM.110
357	N1=L2+2*L+M+3	YLM.111
362	N2=L2+M+1	YLM.112
363	N3=L2+M	YLM.113
364	N4=N1-2*M	YLM.114
	C	YLM.115
366	Y(N1)=CF1*(CTH*Y(N2)-CF2*STH*X*Y(N3))	YLM.116
422	Y(N4)=SIGN*CCNJG(Y(N1))	YLM.117

437	C	130 CONTINUE	YLM.118
444		140 CONTINUE	YLM.119
	C		YLM.120
444		IF(CCNJ.NE.(-1)) RETURN	YLM.121
446		LXX2=LMAX1**2	YLM.122
450		DO 150 I=1,LXX2	YLM.123
456		Y(I)=CCNJG(Y(I))	YLM.124
460		150 CONTINUE	YLM.125
461		PETLRN	YLM.126
	C		YLM.127
462		400 CONTINUE	YLM.128
462		PRINT 800, VECTOR	YLM.129
474		500 CONTINUE	YLM.130
474		PRINT 802, LMAX	YLM.131
	C		YLM.132
	C		YLM.133
		800 FORMAT(/// 10X 35HSPHERICAL HARMONIC ROUTINE *** VECTOR=(3E16.8YLM.134	YLM.134
		* , 1H) ///)	YLM.135
		802 FORMAT(/// 10X 45HSUBROUTINE.YLM *** L-VALUE IS TOO LARGE (L=	YLM.136
		2 , 17,*)*)	YLM.137
507		RETURN	YLM.138
507		END	YLM.139
			YLM.140

SUBPROGRAM LENGTH
000773

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

50	-	000034	100	-	000070	110	-	000111	140	-	000445
400	-	000463	500	-	000475	800	-	000527	802	-	000540

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS

B	-	000741	BCP	-	000732	CFT	-	000756	CF1	-	000622
CF2	-	000624	CL1	-	000764	CL2	-	000765	CPH	-	000747
CTH	-	000745	FAC	-	000626	FACT	-	000724	I	-	000751
IND	-	000740	L	-	000752	LL	-	000735	LMAX1	-	000736
LMX	-	000737	LXX2	-	000772	LX2	-	000750	L1	-	000754
L2	-	000753	M	-	000757	M2	-	000763	MP	-	000762
NA	-	000761	N1	-	000766	N2	-	000767	N3	-	000770
N4	-	000771	P	-	000604	PI	-	000734	PI4	-	000735
RHC	-	000742	PR	-	000743	SIGN	-	000760	SPH	-	000746
STH	-	000744	VECTOR	-	000617	X	-	000725	XX	-	000730

START OF CONSTANTS
000512

START OF TEMPORARIES
000551

START OF INDIRECTS
000577

	PROGRAM READIN	READIN.3
C		READIN.4
	COMMON/BLCKKA/A(10,2),B(10,2),D(10,3),RA(10,2),RB(10,2),ALAT,SPACEA,2	A.2
	2,BASIS(2),NPASIS	B.2
	COMMON/BLCKKP/T(7,8C,7),ILYR(10),LSYM(10)	B.2
	COMPLEX T	C.2
	COMMON/BLCKKC/V1(3),V2(3,2),V3(3,12),VX,VY,VZ,SGK(12),CGK(12),	C.2
	2 YIN(49),YCLT(49,12),XKLEKU,AREA2	C.2
	COMPLEX YIN,YCLT	C.4
	COMMON/BLCKKD/NLCW,NMUM,NDELTA,ARG,NRGY,LAYER,NUMMAX,NCIF,IST(6,3)	D.2
	2 ,LR	D.3
	COMMON/BLCKKE/LMAX,LTCP,LMAX1,LMAX2,LMAX3,LMAX4,LX	E.2
	2 ,ICNE,ITWC,ITHREE	E.2
	COMMON/BLCKKF/ITHT,IPHI,THTT,DIST,CSTH(12),VXX(2,12),KNUP,JP(12),	F.2
	2 JN(12),IQ(12),KQ(12)	F.2
	COMMON/BLCKKG/ESIG,SIG2,XK,DAMP,ZNORM,POTL,GAMA,SIGMA,LCWEN,NARG	G.2
	2 ,LAMBDA	G.2
	REAL LAMBDA	G.4
	INTEGER SIGMA	G.5
	COMPLEX ZNORM	G.5
	COMMON/BLCKKH/KX(7401),C(7532),LL(7532),YINT(169)	H.2
	COMPLEX YINT	H.2
	COMMON/BLCKKJ/MAXX(5),RMAG(9000),XPMAG(9000),MAXF	J.2
	COMMON/BLCKKK/BRB(36,2),TAU(36,36,2),TT(10,12),KTYP(10)	K.2
	COMPLEX BRB,TAU,TT	K.2
	COMMON/BLCKKL/ARG,F(169),XO(5),PATH1(10),PATH2(10,12)	L.2
	COMPLEX ARG,F	L.2
	COMMON/BLCKKM/RATIC,XCON,KSTAR	M.2
	COMPLEX KSTAR,XCON	M.2
	COMMON/BLCKKN/E(51),EN(51),S(12,51)	N.2
	COMMON/BLCKKX/TIME,ITEM,MULT,ISCALE,IMAX,ITYPE,LSUM,ILCCF,IPWR	X.2
	INTEGER TIME	X.2
	COMMON/BLCKKP/W(10),ATWT(10),TEMP,SURF,BULK,DIA(10),CCLT(10,12)	DEM.2
	COMMON/CONST/PI,PI4,PI42,CUN1,CCN2,CCN3,CON4,CCNSTT,IM,CNST	CCNST.2
	COMPLEX IM	CCNST.2
C		READIN.21
C		READIN.22
C	READ IN INSTRUCTIONS CONTAINED IN FOLLOWING SUBROUTINES *****	READIN.23
C		READIN.24
	READ EC8,CA,TE	READIN.25
12	READ ECC, F1,H2,H3,H4,H5,H6,H7,H8	READIN.26
36	PRINT EC1	READIN.27
42	PRINT EC2, H1,H2,H3,H4,H5,H6,H7,H8,CA,TE	READIN.28
C		READIN.29
C	MULT=0 *** SINGLE SCATTERING , NO PRINTCUT	READIN.30
C	MULT=1 *** SINGLE SCATTERING , PLOT SINGLE	READIN.31
C	MULT=2 *** DOUBLE SCATTERING , PLOT DOUBLE	READIN.32
C	MULT=3 *** DOUBLE SCATTERING , PLOT SINGLE+DOUBLE+COMBINED	READIN.33
C	MULT=4 *** DOUBLE SCATTERING , PLOT SINGLE+DOUBLE+COMBINED, (SAME	READIN.34
C	MULT=5 *** DOUBLE SCATTERING , NO PRINTCUT	READIN.35
C		READIN.36
72	READ P03, MULT,MAXR,LSUM,TIME	READIN.37
C		READIN.38
C	READ SELF ENERGY PARAMETERS	READIN.39
C		READIN.40
106	READ EC6, SIGMA,PCTL,GAMA	READIN.41
120	IF(SIGMA.EQ.2) LAMBDA=GAMA	READIN.42
C		READIN.43

	C	READ MATRIX INVERSION INSTRUCTIONS	READIN.44
	C		READIN.45
124	C	READ 804,NDIF	READIN.46
	C		READIN.47
	C	READ ATOMIC SCATTERING FACTORS *****	READIN.48
	C		READIN.49
132	C	CALL ATOMIC	READIN.50
133		LMAX1=LMAX+1	READIN.51
135		LMAX2=2*LMAX+1	READIN.52
136		LMAX3=LMAX1**2	READIN.53
137		LMAX4=LMAX3**2	READIN.54
141		CALL STACK	READIN.55
142		I=ITEM	READIN.56
143		ITEM=2	READIN.57
144		IF(I*TYPE.EQ.0) STOP	READIN.58
150		ITEM=I	READIN.59
152	110	CONTINUE	READIN.60
	C		READIN.61
	C	READ LATTICE PARAMETERS BY SUBPLANES *****	READIN.62
	C		READIN.63
152	C	CALL LATTICE	READIN.64
153	120	CONTINUE	READIN.65
	C		READIN.66
	C	READ BEAM COORDINATES,ENERGY RANGE AND CAMPING PARAMETERS *****	READIN.67
	C		READIN.68
153	C	PRINT RC1	READIN.69
157		PRINT RC2, F1,F2,F3,H4,H5,H6,H7,H8,CA,TE	READIN.70
207		CALL BEAM	READIN.71
210		CALL CDDYE	READIN.72
211	140	CONTINUE	READIN.73
	C		READIN.74
	C	ESTABLISH LOW ENERGY (NLOW) AND K-VALUE *****	READIN.75
	C		READIN.76
211	C	CALL ENERGY(NLOW,NRCY)	READIN.77
213		IF(NRCY.GT.0) GO TO 145	READIN.78
216		IF(NLOW.GT.200) STOP	READIN.79
222		NLOW=NLOW+2	READIN.80
224		GO TO 140	READIN.81
	C		READIN.82
224	145	CONTINUE	READIN.83
224		XKZERC=1./XK	READIN.84
226		XKZ=XK	READIN.85
	C		READIN.86
	C	CLEAR ENERGY AND INTENSITY ARRAYS *****	READIN.87
	C		READIN.88
226		NRG=NLOW-DELTA	READIN.89
231		DO 150 J=1,51	READIN.90
245		FN(J)=0.0	READIN.91
246		NRG=NRG+DELTA	READIN.92
247		E(J)=NRG	READIN.93
250		S(1,J)=0.0	READIN.94
251		S(2,J)=0.0	READIN.95
251		S(3,J)=0.0	READIN.96
252		S(4,J)=0.0	READIN.97
252	150	CONTINUE	READIN.98
	C		READIN.99
	C	SET INCIDENT K-VECTOR DIRECTION *****	READIN.100
	C		READIN.101

```

255      THETA=ITHT*CCN3
257      PHI=IPHI*CCN3
261      ST=SIN(THETA)
262      VX=ST*CCS(PHI)
265      VY=ST*SIN(PHI)
270      VZ=CCS(THETA)
272      THTT=1./VZ
274      V1(1)=VX*XKZ
276      V1(2)=VY*XKZ
277      V1(3)=VZ*XKZ

      C
      C      SPHERICAL HARMONICS FOR INCIDENT BEAM *****
      C
300      LCX=2*LMAX
302      CALL YLM(V1(1),V1(2),V1(3),YIN,LOX,+)
306      ARG=ALCW-NCDELTA
310      IF(ANLW.LE.51) GO TO 160
313      ISCALE=2
313      IMAX=101
315      160 CONTINUE
315      IF(TIME.NE.0) CALL SECOND(TIME1)
320      CALL TABLE
321      IF(TIME.NE.0) CALL SECOND(TIME2)
324      IF(TIME.NE.0) PRINT 834, TIME1, TIME2
325      AREA2=CCNSTT/(A(1,1)*B(1,2)-A(1,2)*B(1,1))*2
341      DIST=ABS(C(5,3)-C(8,3))*0.5

      C
      C
      800 FORMAT( 8A10 )
      801 FORMAT( 1H1 )
      802 FORMAT( / 10X 6H***** 8A10, 14X 2A10, // )
      803 FORMAT( 415 )
      804 FORMAT( 15 )
      806 FORMAT( 5X I2, 2F12.4 )
      808 FORMAT( 2A10 )
      834 FORMAT( / 10X*TIME FOR EXECUTION OF TABLE * F8.3,* TO *F8.3 )

      C
      C
345      END

```

READIN.102
 READIN.103
 READIN.104
 READIN.105
 READIN.106
 READIN.107
 READIN.108
 READIN.109
 READIN.110
 READIN.111
 READIN.112
 READIN.113
 READIN.114
 READIN.115
 READIN.116
 READIN.117
 READIN.118
 READIN.119
 READIN.120
 READIN.121
 READIN.122
 READIN.123
 READIN.124
 READIN.125
 READIN.126
 READIN.127
 READIN.128
 READIN.129
 READIN.130
 READIN.131
 READIN.132
 READIN.133
 READIN.134
 READIN.135
 READIN.136
 READIN.137
 READIN.138
 READIN.139
 READIN.140

PROGRAM LENGTH INCLUDING I/O BUFFERS
 000463

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

110	-	000152	120	-	000154	140	-	000212	145	-	000225
160	-	000316	800	-	000363	801	-	000365	802	-	000367
803	-	000375	804	-	000377	806	-	000401	808	-	000404
834	-	000406									

BLOCK NAMES AND LENGTHS

BLOCKA	-	000162	BLOCKB	-	004324	BLOCKC	-	002542	BLOCKD	-	000055
BLOCKE	-	000012	BLOCKF	-	000121	BLOCKG	-	000014	BLOCKH	-	042613
BLOCKJ	-	043254	BLOCKK	-	012712	BLOCKL	-	000737	BLOCKM	-	000005
BLOCKN	-	001212	BLOCKX	-	000011	BLOCKB	-	000221	CCNST	-	000012

		SUBROUTINE ATOMIC	ATCMIC.2
C			ATCMIC.3
C		ITYPE IS NUMBER OF DIFFERENT ATOMIC TYPES *****	ATCMIC.4
C			ATCMIC.5
		COMMON/BLCKKB/T(2,80,7),ILYR(10),LSYM(10)	B.2
		COMPLX T	B.3
		COMMON/BLCKKE/LMAX,LTCP,LMAX1,LMAX2,LMAX3,LMAX4,LX	E.2
		2 ,ICKE,ITWO,ITHREF	E.3
		COMMON/BLCKKG/ESIG,SIG2,XK,DAMP,ZNORM,POTL,GAMA,SIGMA,LGKEN,NNRG	G.2
		2 ,LAMBDA	G.3
		REAL LAMBDA	G.4
		INTEGER SIGMA	G.5
		COMPLEX ZNORM	G.6
		COMMON/BLCKKX/TIME,ITEM,MULT,ISCALE,IMAX,ITYPE,LSUP,ILCCP,IPWR	X.2
		INTEGER TIME	X.3
		COMMON/BLKDEH/W(10),ATWT(10),TEMP,SURF,BULK,DIN(10),DCLT(10,12)	DEB.2
			ATCMIC.11
		READ FCC, ITYPE	ATCMIC.12
7		IF(ITYPE.EQ.C) GO TO 200	ATCMIC.13
10		READ EC1, LMAX	ATCMIC.14
	C		ATCMIC.15
	C		ATCMIC.16
16		DC 100 I=1,ITYPE	ATCMIC.17
20		READ EC2, F1,F2,H3,F4,H5,H6	ATCMIC.18
27		READ ROH, ATWT(I)	ATCMIC.19
46		READ EC3, LTCP,LGKEN,NNRG	ATCMIC.20
60		IF(LMAX.GE.LTCP) LMAX=LTCP-1	ATCMIC.21
	C		ATCMIC.22
65		DC 90 L=1,LTCP	ATCMIC.23
67		READ EC4, (T(I,J,L), J=1,NNRG)	ATCMIC.24
105	50	CONTINUE	ATCMIC.25
110		PRINT EC5, LMAX,I,F1,H2,H3,H4,H5,H6	ATCMIC.26
133	100	CONTINUE	ATCMIC.27
136	200	CONTINUE	ATCMIC.28
	C		ATCMIC.29
	C		ATCMIC.30
		800 FORMAT(12)	ATCMIC.31
		801 FORMAT(15)	ATCMIC.32
		EC2 FORMAT(5A10)	ATCMIC.33
		803 FORMAT(315)	ATCMIC.34
		804 FORMAT(3E16.8)	ATCMIC.35
		EC5 FORMAT(// 10X *1-FACTORS READ IN TO L=* 13, * FOR ATOMIC TYPE *	ATCMIC.36
		2 12, *.... * 5A10)	ATCMIC.37
		808 FORMAT(F10.5)	ATCMIC.38
	C		ATCMIC.39
136		RETURN	ATCMIC.40
137		END	ATCMIC.41

SUBPRCPAM LENGTH
000211

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

200	-	000137	FCC	-	000145	801	-	000147	802	-	000151
803	-	000153	EC4	-	000155	805	-	000160	EC8	-	000172

```

SUBROUTINE STACK
C
C      THIS SUBROUTINE DOES INTEGRAL OVER 3 SPHERICAL HARMONICS FOR WHICH
C      (L1,M1) AND (L2,M2) ARE INDICES OF TWO OF THEM.
C      LL(NUM) DELIVERS COMBINED INDICES FOR (L3,M3), FOR WHICH THE INTEG
C      IS NOT EQUAL TO ZERO. (L3,M3) ARE INDICES OF COMPLEX CONJUGATE FUS
C
C      COMMON/BLCKCK/NLCW,NUM,NDELTA,ARG,NRGY,LAYER,NUPMAX,ACIF,IST(6,6)
2      ,LR
C      COMMON/BLCKCK/LMAX,LTCP,LMAX1,LMAX2,LMAX3,LMAX4,LX
2      ,ICNE,ITWC,ITFREE
C      COMMON/BLCKCK/KX(2401),C(7532),LL(7532),YINT(169)
C      COMPLEX YINT
C
C      NUM=0
2      ISTEP=0
C
C      LCM ARRAY G(IN,OUT) IS STACKED BACKWARDS IN ORDER
C      TO PERFORM MATRIX MULTIPLICATION EFFICIENTLY.
C      NCTF ... INCOMING BEAM INDEX VARIES MOST RAPIDLY FOR G(IN,OUT) ***
C
C      SUM OVER (LCUT) *****
3      DO 100 LL1=1,LMAX1
4      MX1=2*LL1-1
5      K1=-LL1
6      L3=LL1-1
10     DO 100 MM1=1,MX1
11     M3=MM1+K1
C
C      SUM OVER (LIN) *****
12     DO 100 LL2=1,LMAX1
14     MX2=2*LL2-1
15     K2=-LL2
16     L2=LL2-1
20     DO 100 MM2=1,MX2
21     M2=MM2+K2
C
C      M1=M2-M3
24     ISTEP=ISTEP+1
26     MY=IABS(M1)+1
27     LCW=IABS(L2-L3)+1
32     LI=L2+L3+1
34     KKK=C
C
C      SUM OVER (LAMBEA) *****
36     DO 90 LL3=LCW,LHI,2
38     IF(LL3.LT.MM) GO TO 90
40     KKK=KKK+1
42     ALM=ALM+1
43     LI=LI-1
C
44     C(NUM)=CDEFT(LL1,L3,L2,M1,M3,M2)
52     LL(NUM)=LI*(LI+1)+M1+1

```

STACK.2
 STACK.3
 STACK.4
 STACK.5
 STACK.6
 STACK.7
 STACK.8
 STACK.9
 D.2
 D.3
 E.2
 E.3
 H.2
 H.3
 STACK.13
 STACK.14
 STACK.15
 STACK.16
 STACK.17
 STACK.18
 STACK.19
 STACK.20
 STACK.21
 STACK.22
 STACK.23
 STACK.24
 STACK.25
 STACK.26
 STACK.27
 STACK.28
 STACK.29
 STACK.30
 STACK.31
 STACK.32
 STACK.33
 STACK.34
 STACK.35
 STACK.36
 STACK.37
 STACK.38
 STACK.39
 STACK.40
 STACK.41
 STACK.42
 STACK.43
 STACK.44
 STACK.45
 STACK.46
 STACK.47
 STACK.48
 STACK.49
 STACK.50
 STACK.51
 STACK.52
 STACK.53
 STACK.54
 STACK.55
 STACK.56

57	90	CONTINUE	STACK.57
62		KX(ISTEP)=KKK	STACK.58
	C		STACK.59
64	100	CONTINUE	STACK.60
76		NUMMAX=NUM	STACK.61
	C		STACK.62
77		PRINT 800,LMAX,ISTEP,L1,M1,L2,M2,L3,M3,NUMMAX	STACK.63
	C		STACK.64
	C		STACK.65
		800 FORMAT (/ 12X*SUBROUTINE STACK FORMS INTERMEDIATE ARRAYS (LL) AND	STACK.66
		2(C),* / 10X*L=* I3,* LMAX4=* I5,* (L1,M1,L2,M2,L3,M3)=(*	STACK.67
		3 6I3,*) NUMMAX=* I5)	STACK.68
	C		STACK.69
125		RETURN	STACK.70
126		END	STACK.71

SUBPROGRAM LENGTH
000211

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

90 - 000060 800 - 000133

BLOCK NAMES AND LENGTHS

BLOCKD - 000055 BLOCKE - 000012 BLOCKH - 042613

VARIABLE ASSIGNMENTS

C	-	004541S03	IST	-	000010S01	ISTEP	-	000165	KKK	-	000206
KX	-	000000S03	K1	-	000170	K2	-	000176	LHI	-	000205
LL	-	023315S03	LL1	-	000166	LL2	-	000174	LL3	-	000207
LMAX	-	000000S02	LMAX1	-	000002S02	L0W	-	000204	L1	-	000210
L2	-	000177	L3	-	000171	MM	-	000203	MM1	-	000172
MM2	-	000200	MX1	-	000167	MX2	-	000175	M1	-	000202
M2	-	000201	M3	-	000173	NUM	-	000164	NUMMAX	-	000006S01
YINT	-	042671S03									

START OF CONSTANTS

000131

START OF TEMPORARIES

000154

START OF INDIRECTS

000162

UNUSED COMPILER SPACE

124500

	FUNCTION CCEFT(L1,L2,L3,M1,M2,M3)	CCEFT.2
C		CCEFT.3
	DIMENSION X(6),Y(6)	CCEFT.4
C		CCEFT.5
	X(1)=L1	CCEFT.6
11	Y(1)=L1	CCEFT.7
12	X(2)=L2	CCEFT.8
13	Y(2)=L2	CCEFT.9
14	X(3)=L3	CCEFT.10
15	Y(3)=L3	CCEFT.11
16	X(4)=M1	CCEFT.12
20	X(5)=M2	CCEFT.13
21	X(6)=M3	CCEFT.14
22	Y(4)=C.C	CCEFT.15
23	Y(5)=C.C	CCEFT.16
23	Y(6)=C.C	CCEFT.17
C		CCEFT.18
24	CFT=SQRT((2*L1+1)*(2*L2+1)/(12.5663706*(2*L3+1)))	CCEFT.19
43	CCEFT=CFT*CLEB(X)*CLEB(Y)	CCEFT.20
C		CCEFT.21
C		CCEFT.22
52	RETURN	CCEFT.23
52	END	CCEFT.24

```

FLNCTCN CLEE(X)
DIMENSION X(6)
DIMENSION FACT(101)
DATA EPS/10.E-5/
DATA(FACT(1),I=1,30)/
X 1.CCCCC00000000CE+00, 1.C00CC0CCCC0CCCE+00,
X 1.41421356237341E+00, 2.44968974278371E+00,
X 4.8989794855645CE+00, 1.095445115C1C8CE+C1,
X 2.643261573CC08FE+C1, 7.09925373572240E+01,
X 2.C0C7984C63830CE+C2, 6.023552191C488E+02,
X 1.9C494054396762E+C3, 6.31797435852595E+03,
X 2.18861C51811534E+C4, 7.89114744508457E+04,
X 2.552557C1280214E+C5, 1.1435355C586437E+06,
X 4.57414362745827E+C6, 1.88596773062634E+07,
X 8.C0148342E54857E+C7, 3.48776576634565E+08,
X 1.55977626862895E+09, 7.14775241818756E+09,
X 3.352112C0E37367E+10, 1.60785623545420E+11,
X 7.87495471322957E+11, 3.93842735661453E+12,
X 2.C0821175442435E+13, 1.04345745809C42E+14,
X 5.52166553667054E+14, 2.97351CC4601142E+15/
DATA(FACT(1),I=31,60)/
X 1.62865852714862E+15, 9.06758640678641E+16,
X 5.129628C268C115E+17, 2.94674655533533E+18,
X 1.71823397428617E+18, 1.01652052779C88E+20,
X 6.C9512554674351E+20, 3.70955324645991E+21,
X 2.2669687743C413E+22, 1.4282115417528E+23,
X 9.C328C29C521680E+23, 5.78381552143564E+24,
X 3.74834112341403E+25, 2.45795164845C47E+26,
X 1.63042067417555E+27, 1.0937194378131C5E+28,
X 7.417561362C652E+28, 5.0855C136672575E+29,
X 3.5233286596539C0E+20, 2.466337C8975722E+31,
X 1.74395368C95878E+32, 1.24543918088317E+33,
X 8.86898963429102E+33, 5.53825915977C85E+34,
X 4.60461962425951E+35, 3.5632172884641E+36,
X 2.64645567711136E+37, 2.013129E8911719E+38,
X 1.53315404681445E+39, 1.17763756875137E+40/
DATA(FACT(1),I=61,90)/
X 5.12194446165793F+40, 7.12446539315476E+41,
X 5.AC981044778243E+42, 4.452649C0411166E+43,
X 3.5621192C328933E+44, 2.8718723147084CE+45,
X 2.23312C05775011E+46, 1.9097411C595541E+47,
X 1.57481285548711E+48, 1.308137807824C9E+49,
X 1.C5446661300462E+50, 9.2221396C291451E+50,
X 7.82524494C32043E+51, 6.858522C2780514E+52,
X 5.7512194719355E+53, 4.98087751415C82E+54,
X 4.3622284666521E+55, 3.d1028991055721E+56,
X 3.36155693214947E+57, 2.99C1C165C577C17E+58,
X 2.67524584926275E+59, 2.40772216423314E+60,
X 2.18C28515C36688E+61, 1.98533430460C97E+62,
X 1.8205C546126182E+63, 1.67842310348487E+64,
X 1.565C0555274135E+65, 1.45181172964C34E+66,
X 1.3619201234C039E+67, 1.26483287475282E+68/
DATA(FACT(1),I=91,101)/
X 1.1218959485C43C4E+69, 1.162756C05520414E+70,
X 1.11527638C7302E+71, 1.07553355177E91E+72,
X 1.04276850576775E+73, 1.015365C1749511E+74,
X 9.9583C27411C813E+74, 9.807559C7643175E+75,
X 5.70952175C118277E+76, 9.66554943789C34E+77,

```

CLE8.2
CLE8.3
CLE8.4
CLE8.5
CLE8.6
CLE8.7
CLE8.8
CLE8.9
CLE8.10
CLE8.11
CLE8.12
CLE8.13
CLE8.14
CLE8.15
CLE8.16
CLE8.17
CLE8.18
CLE8.19
CLE8.20
CLE8.21
CLE8.22
CLE8.23
CLE8.24
CLE8.25
CLE8.26
CLE8.27
CLE8.28
CLE8.29
CLE8.30
CLE8.31
CLE8.32
CLE8.33
CLE8.34
CLE8.35
CLE8.36
CLE8.37
CLE8.38
CLE8.39
CLE8.40
CLE8.41
CLE8.42
CLE8.43
CLE8.44
CLE8.45
CLE8.46
CLE8.47
CLE8.48
CLE8.49
CLE8.50
CLE8.51
CLE8.52
CLE8.53
CLE8.54
CLE8.55
CLE8.56
CLE8.57
CLE8.58
CLE8.59

	X 9.66054943779846E+7d/	CLEB.50
	CLEB=C.	CLEB.51
2	NA=2.*X(1)+EPS	CLEB.62
5	NB=2.*X(2)+EPS	CLEB.63
10	NC=2.*X(3)+EPS	CLEB.64
12	ND=SIGN(ABS(2.*X(4))+EPS,X(4))	CLEB.55
20	NE=SIGN(ABS(2.*X(5))+EPS,X(5))	CLEB.65
26	NF=-SIGN(ABS(2.*X(6))+EPS,X(6))	CLEB.57
33	IF(ABS(X(4)+X(5)-X(6)).GT.EPS) RETURN	CLEB.68
43	L3=X(2)+X(5)+EPS	CLEB.69
46	L8=X(1)+X(4)+EPS	CLEB.70
51	L10=X(3)-X(6)+EPS	CLEB.71
54	IF(ABS(X(2)+X(3)-FLCAT(L3)).GT.EPS) RETURN	CLEB.72
64	IF(ABS(X(1)+X(4)-FLCAT(L8)).GT.EPS) RETURN	CLEB.73
73	IF(ABS(X(3)-X(6)-FLCAT(L10)).GT.EPS) RETURN	CLEB.74
103	LC=L3+L8+L10	CLEB.75
105	IF(LC.GT.99) GO TO 2	CLEB.76
111	IF(ABS(NC).GT.NA.OR.	CLEB.77
	X IAPS(NE).GT.NB.OR.	CLEB.78
	X IAPS(NF).GT.NC) RETURN	CLEB.79
130	L1=LO-NC	CLEB.80
132	L2=L8-NC	CLEB.81
134	L9=L3-NE	CLEB.82
136	L11=L10-NF	CLEB.83
140	L6=LO-NB	CLEB.84
142	L7=LC-NA	CLEB.85
143	L4=L6-L2	CLEB.86
145	L5=L7-L3	CLEB.87
147	L20=NC+1	CLEB.88
151	KMIN=MAX(0,-L4,-L5)	CLEB.89
157	KMAX=MIN(L1,L2,L3)	CLEB.90
165	IF(KMAX.LT.KMIN) RETURN	CLEB.91
172	CC1=FACT(L1+1)*FACT(L6+1)*FACT(L7+1)*FACT(L8+1)	CLEB.92
	X *FACT(L2+1)*FACT(L3+1)	CLEB.93
	X *FACT(L9+1)*FACT(L10+1)*FACT(L11+1)/FACT(LO+2)	CLEB.94
	X *FACT(L20+1)/FACT(L20)	CLEB.95
215	SGN=1.	CLEB.96
216	DO 1 K=KMIN,KMAX	CLEB.97
230	L12=L1-K+1	CLEB.98
232	L13=L2-K+1	CLEB.99
234	L14=L3-K+1	CLEB.100
235	L15=L4+K+1	CLEB.101
236	L16=L5+K+1	CLEB.102
240	DEN=FACT(K+1)*FACT(L12)*FACT(L13)*FACT(L14)*FACT(L15)*FACT(L16)	CLEB.103
247	CLEB=CLEB+SGN*CC1/DEN**2	CLEB.104
252	SGN=-SGN	CLEB.105
253	1 CONTINUE	CLEB.106
263	IF(MOD(KMIN,2).NE.0) CLEB=-CLEB	CLEB.107
267	RETURN	CLEB.108
271	2 ARFF=C.	CLEB.109
272	CLEB=C./ARFF	CLEB.110
274	RETURN	CLEB.111
274	END	CLEB.112

SUBPROGRAM LENGTH
000565

```

C      SLBRCLTINE LATTICE
C
COMMON/BLCKKA/A(10,2),B(10,2),O(10,2),RA(10,2),RB(10,2),ALAT,SPACE
2,BASIS(2),NBASIS
COMMON/BLCKKB/T(2,8C,7),ILYK(10),LSYM(10)
COMPLEX T
COMMON/BLCKKD/NLCK,NLUM,NDELTA,ARG,NRCY,LAYFR,NUMMAX,NCTF,IST(6,6)
2,LF
COMMON/BLCKKE/BEE(3C,2),TAU(3C,36,2),TT(10,12),KTYP(10)
COMPLEX BEE,TAU,TT
COMMON/BLCKKL/ARG,H(169),XD(9),PATH1(10),PATH2(10,12)
COMPLEX ARG,H
COMMON/BLCKKX/TIME,ITEM,MULT,ISCALE,IMAX,ITYPE,LSUP,ILCCF,IPWR
INTEGER TIME
COMMON/CONST/PI,PI4,F142,CUN1,CCN2,CCN3,CCN4,CCNSTT,IM,CNST
COMPLEX IM
C
DIMENSION AX(2),BX(2),SHIFT(3),MM(4)
DIMENSION Q(27,8)
INTEGER TYPE,FACE
DATA(Q(1),I=1,27) / .5,0.,0.,.5,.5,.5,-.5,0.,.5,0.,.5,C.,.5,0.,.5,
20.,.5,1C*0.C/
DATA(Q(1),I=28,54) / .35355339,0.,0.,C.,1.,.70710678,0.,
2 0.,.35355339,0.,.35355339,0.,.35355339,0.,.35355339,0.,.35355339,
3 C.,.5,C.,.5,0.,.5,C.,.5,0.,.5/
DATA(Q(1),I=55,81) / .57735026,0.,0.,.35355339,.61237244,.35355339,
2 -.61237244,0.,.35355339,.70710678,0.,.35355339,.70710678,
3 0.,.35355339,.70710678,0.,0.,.20412415,.40824830,C.,.20412415,
4 .40824830,C.,.20412415,.40824830,C./
DATA(Q(1),I=82,108) / .5,0.,0.,0.,1.,1.,0.,0.,.5,0.,.5,C.,.5,0.,
2 .5,C.,.5,C.,.5,0.,.5,0.,.5,0.,.5/
DATA(Q(1),I=109,135) / .70710678,0.,C.,.70710678,.5,.70710678,-.5,
2 C.,.70710678,0.,.70710678,0.,.70710678,0.,.70710678,
3 1C*0.C/
DATA(Q(1),I=136,162) / .28867513,C.,0.,.70710678,1.2247449,.70710678,
2 -.1.2247449,1C*0.C.,.81649658,1.63299316,C.,.81649658,1.63299316,
3 0.,.81649658,1.63299316,0./
DATA(Q(1),I=163,189) / 1., 0.0, 0.0, .5,.866025404,.5,-.866025404,
2 0.,.5,C.,.5,C.,.5,C.,.5,0.,.5,0.,.288675135,0.,.288675135,0.,
3 .288675135,C.,.288675135,0.,.288675135/
DATA(Q(1),I=190,216) / 1.0,.866025404,.5,.866025404,1.F,1.732050808,
2 C.C,C.C,.866025404,0.0,.866025404,C.C,.866025404,C.C,.866025404,
3 C.C,.866025404,C.C,-.5,0.0,-.5,0.0,-.5,0.0,-.5,C.C,-.5/
C
C      READ IN LATTICE SPECIFICATIONS.
C
READ SCC, TYPE,FACE,ALAT,HEX
IF(HEX.EQ.C.C) HEX=ALAT
READ HLC, LAYER
JJ=C
26 IF(TYPE.EQ.3HFCC) GO TO 100
20 IF(TYPE.EQ.3HFCC) GO TO 110
32 IF(TYPE.EQ.3HFCC) GO TO 120
33 IF(TYPE.EQ.3HGR1) GO TO 130
C
59 CONTINUE

```

LATTICE.2
 LATTICE.3
 LATTICE.4
 A.2
 A.3
 B.2
 B.3
 D.2
 D.3
 K.2
 K.3
 L.2
 L.3
 X.2
 X.3
 CCNST.2
 CCNST.3
 LATTICE.12
 LATTICE.13
 LATTICE.14
 LATTICE.15
 LATTICE.16
 LATTICE.17
 LATTICE.18
 LATTICE.19
 LATTICE.20
 LATTICE.21
 LATTICE.22
 LATTICE.23
 LATTICE.24
 LATTICE.25
 LATTICE.26
 LATTICE.27
 LATTICE.28
 LATTICE.29
 LATTICE.30
 LATTICE.31
 LATTICE.32
 LATTICE.33
 LATTICE.34
 LATTICE.35
 LATTICE.36
 LATTICE.37
 LATTICE.38
 LATTICE.39
 LATTICE.40
 LATTICE.41
 LATTICE.42
 LATTICE.43
 LATTICE.44
 LATTICE.45
 LATTICE.46
 LATTICE.47
 LATTICE.48
 LATTICE.49
 LATTICE.50
 LATTICE.51
 LATTICE.52

35	PRINT 820, TYPE,FACE	LATTICE.53
45	STOP	LATTICE.54
	C	LATTICE.55
	C	LATTICE.56
	C	LATTICE.57
47	100 CONTINUE	LATTICE.58
47	IF(FACE.EQ.100.OR.FACE.EQ.010.OR.FACE.EQ.001) JJ=1	LATTICE.59
63	IF(FACE.EQ.110.OR.FACE.EQ.101.OR.FACE.EQ.011) JJ=2	LATTICE.60
76	IF(FACE.EQ.111) JJ=3	LATTICE.61
101	GO TO 160	LATTICE.62
	C	LATTICE.63
	C	LATTICE.64
	C	LATTICE.65
102	110 CONTINUE	LATTICE.66
102	IF(FACE.EQ.100.OR.FACE.EQ.010.OR.FACE.EQ.001) JJ=4	LATTICE.67
116	IF(FACE.EQ.110.OR.FACE.EQ.101.OR.FACE.EQ.011) JJ=5	LATTICE.68
131	IF(FACE.EQ.111) JJ=6	LATTICE.69
124	GO TO 160	LATTICE.70
	C	LATTICE.71
	C	LATTICE.72
	C	LATTICE.73
135	120 CONTINUE	LATTICE.74
135	JJ=7	LATTICE.75
136	GO TO 160	LATTICE.76
	C	LATTICE.77
137	130 CONTINUE	LATTICE.78
137	JJ=8	LATTICE.79
	C	LATTICE.80
	C	LATTICE.81
	C	LATTICE.82
	C	LATTICE.83
140	150 CONTINUE	LATTICE.84
140	IF(JJ.EQ.0) GO TO 99	LATTICE.85
152	SPACE=Q(1,JJ)	LATTICE.86
153	ZSP=SPACE*FX	LATTICE.87
154	BASIS(1)=Q(2,JJ)*ALAT	LATTICE.88
156	BASIS(2)=Q(3,JJ)*ALAT	LATTICE.89
157	AX(1)=Q(4,JJ)*ALAT	LATTICE.90
160	AX(2)=Q(5,JJ)*ALAT	LATTICE.91
161	BX(1)=Q(6,JJ)*ALAT	LATTICE.92
162	BX(2)=Q(7,JJ)*ALAT	LATTICE.93
163	NBASIS=1	LATTICE.94
164	IF(BASIS(1).EQ.0.0.AND.BASIS(2).EQ.0.0) NBASIS=C	LATTICE.95
173	K=7	LATTICE.96
	C	LATTICE.97
174	DO 170 J=1,2	LATTICE.98
176	DO 170 I=1,10	LATTICE.99
214	K=K+1	LATTICE.100
215	D(I,J)=C(K,JJ)*ALAT	LATTICE.101
216	A(I,J)=AX(J)	LATTICE.102
216	B(I,J)=BX(J)	LATTICE.103
220	170 CONTINUE	LATTICE.104
	C	LATTICE.105
	C	LATTICE.106
224	SP=-ZSP	LATTICE.107
225	DO 172 I=1,10	LATTICE.108
235	ILYR(I)=1	LATTICE.109
236	SP=SP+ZSP	LATTICE.110

237	D(I,3)=SP	LATTICE.111
237	KTP(I)=1	LATTICE.112
240	LSYM(I)=1	LATTICE.113
240	172 CCNTINUE	LATTICE.114
C		LATTICE.115
C	ALTER THE LATTICE GEOMETRY LAYER BY LAYER	LATTICE.116
C		LATTICE.117
242	LR=0	LATTICE.118
243	174 CCNTINUE	LATTICE.119
243	READ 812,KT,ITP,MM,SHIFT	LATTICE.120
257	IF(KT.EQ.0) GO TO 176	LATTICE.121
265	LR=LR+1	LATTICE.122
266	ILYR(LR)=ITP	LATTICE.123
267	KTP(LR)=KT	LATTICE.124
270	A(LR,1)=MM(1)*AX(1) + MM(2)*BX(1)	LATTICE.125
275	A(LR,2)=MM(1)*AX(2) + MM(2)*BX(2)	LATTICE.126
301	B(LR,1)=MM(3)*AX(1) + MM(4)*BX(1)	LATTICE.127
306	B(LR,2)=MM(3)*AX(2) + MM(4)*BX(2)	LATTICE.128
313	C(LR,1)=D(LR,1)+SHIFT(1)*ALAT	LATTICE.129
316	D(LR,2)=D(LR,2)+SHIFT(2)*ALAT	LATTICE.130
320	C(LR,2)=D(LR,3)-SHIFT(3)*ZSP	LATTICE.131
323	IF(LR.GT.1) XD(LR-1)=D(LR,3)-D(LR-1,3)	LATTICE.132
330	GO TO 174	LATTICE.133
C		LATTICE.134
331	176 CCNTINUE	LATTICE.135
C		LATTICE.136
331	LR1=LR+1	LATTICE.137
333	IF(LR1.LT.2) LR1=2	LATTICE.138
336	DC 180 I=LR1,10	LATTICE.139
345	D(I,3)=D(I-1,3)+ZSP	LATTICE.140
346	XD(I-1)=D(I,3)-D(I-1,3)	LATTICE.141
347	180 CCNTINUE	LATTICE.142
C		LATTICE.143
C		LATTICE.144
C		LATTICE.145
C		LATTICE.146
C	CALCULATE RECIPROCAL LATTICE NET.	LATTICE.147
C		LATTICE.148
351	DC 190 J=1,LAYER	LATTICE.149
361	AREA=A(J,1)*B(J,2)-A(J,2)*B(J,1)	LATTICE.150
363	XX=2.*PI/AREA	LATTICE.151
364	RA(J,1)= B(J,2)*XX	LATTICE.152
365	PA(J,2)=-B(J,1)*XX	LATTICE.153
366	RB(J,1)=-A(J,2)*XX	LATTICE.154
367	RB(J,2)= A(J,1)*XX	LATTICE.155
370	190 CCNTINUE	LATTICE.156
C		LATTICE.157
372	IF(JJ.EQ.1) PRINT 801	LATTICE.158
377	IF(JJ.EQ.2) PRINT 802	LATTICE.159
405	IF(JJ.EQ.3) PRINT 803	LATTICE.160
413	IF(JJ.EQ.4) PRINT 804	LATTICE.161
421	IF(JJ.EQ.5) PRINT 805	LATTICE.162
427	IF(JJ.EQ.6) PRINT 806	LATTICE.163
425	IF(JJ.EQ.7) PRINT 807	LATTICE.164
443	PRINT 830, ALAT,ZSP	LATTICE.165
453	PRINT 840, BASIS	LATTICE.166
C		LATTICE.167
461	DC 200 I=1,LAYER	LATTICE.168

```

463      PRINT 850, I, KTYP(I), ILYK(I), (C(I, J), J=1, 3), A(I, 1), A(I, 2), B(I, 1),
2      B(I, 2)
522      ZCC CONTINUE
525      PRINT 850
530      PRINT 850, (J, RA(J, 1), RA(J, 2), RB(J, 1), RB(J, 2), J=1, LAYER )

C
C
800 FORMAT( A3, 1X13, 3X 2F10.4 )
801 FORMAT( //10X 5+***** *FCC (100)-SURFACE* )
802 FORMAT( //10X 5+***** *FCC (110)-SURFACE* )
803 FORMAT( //10X 5+***** *FCC (111)-SURFACE* )
804 FORMAT( //10X 5+***** *BCC (100)-SURFACE* )
805 FORMAT( //10X 5+***** *BCC (110)-SURFACE* )
806 FORMAT( //10X 5+***** *BCC (111)-SURFACE* )
807 FORMAT( //10X 5+***** *HEX. CLOSE PACKED LATTICE. * )
810 FORMAT( F )
812 FORMAT( 615, 3F10.4 )
820 FORMAT( // 10X*UNRECOGNIZABLE CRYSTAL TYPE * A3, 1X13 )
830 FORMAT( // 10X*LATTICE CONSTANT IS * F9.5, * ANGSTROMS, INTERL
2YEN SPACING IN BULK IS * F9.5, * ANGSTROMS* / )
840 FORMAT( // 10X*BASIS VECTOR IN SURFACE PLANE IS (* F9.5, *, * F9.5,
2 *)* )
850 FORMAT( 10X*( * 12, *) CORE(* 12, *) SYN(* 12, *) C=( * 3F10.5,
2 *) A=( * 2F10.5, *) B=( * 2F10.5, *)* )
858 FORMAT( //10X*RECIPROCAL LATTICE, SURFACE NET VECTORS* / )
860 FORMAT( 35X*( * 12, *) KA=( * 2F10.5, *) RE=( * 2F10.5, *)* )

C
557      RETLRA
560      END

```

SUPPRCCRAM LENGTH
0C1360

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

99	-	000036	100	-	000050	110	-	000103	120	-	000136
130	-	000140	160	-	000141	174	-	000244	176	-	000332
800	-	000600	801	-	000604	802	-	000612	803	-	000620
804	-	000626	805	-	000634	806	-	000642	807	-	000650
810	-	000657	812	-	000661	820	-	000664	830	-	000673
840	-	000707	850	-	000717	858	-	000733	860	-	000742

BLOCK NAMES AND LENGTHS

BLOCKA	-	000163	BLOCKB	-	004324	BLOCKD	-	000055	BLOCKK	-	012712
BLOCKL	-	000737	BLOCKX	-	000011	CONST	-	000013			

VARIABLE ASSIGNMENTS

A	-	000005001	ALAT	-	000156501	AREA	-	001356	ARG	-	000000505
AX	-	000777	B	-	000024501	BASIS	-	000160501	BRB	-	000000504
BX	-	001007	C	-	000005001	FACE	-	001343	F	-	000002505
HEX	-	001344	I	-	001351	ILYK	-	004300502	IM	-	000010507
IST	-	000010503	ITP	-	001354	J	-	001350	JJ	-	001345
K	-	001347	KY	-	001353	KTYF	-	012700504	LAYER	-	000003503
LR	-	000054503	LRI	-	001355	LSYM	-	004312502	MM	-	001006
NBASIS	-	000162501	PATH1	-	000135505	PATH2	-	000547505	PI	-	000000507

```

SUBROUTINE BEAM
COMMON/HLOCKA/A(10,2),B(10,2),C(10,2),RA(10,2),RB(10,2),ALAT,SPACE
2,BASIS(2),ABASIS
COMMON/BLOCKB/T(2,80,7),ILYR(10),LSYN(10)
COMPLEX T
COMMON/BLOCKC/NLW,ANUM,NDELTA,ARG,NRGY,LAYER,NUMMAX,ACIF,IST(5,4)
2,LR
COMMON/BLOCKD/LMAX,LTCP,LMAX1,LMAX2,LMAX3,LMAX4,LX
2,ICNE,ITWC,ITRFE
COMMON/BLOCKE/ROB(36,2),TAU(26,26,2),TT(10,12),KTYP(10)
COMPLEX ROB,TAL,TT
COMMON/HLOCKF/ITHT,IPHI,THTT,DIST,CSIF(12),VXX(2,12),KNLM,JM(12),
2 JN(12),IQ(12),KC(12)
DIMENSION AN(2,4)

C
C      READ IN BEAM COORDINATES *****
C
C      READ ROB, ITHT, IPHI
C
C      READ IN ENERGY RANGE AND DAMPING LENGTH AT 30 E.V. *****
C
11  C      READ ROB, NLW,ANUM,NDELTA
C
C
C      KNLM IS TOTAL NUMBER OF BEAMS TO BE COMPUTED.
C
23  KNLM=0
24  100 CONTINUE
24  READ ROB, ITP,NN
24  IF(ITP.EQ.0) GO TO 150
24  DO 110 I=1,LAYER
24  K=I
24  KT=KTYP(I)
24  IF(ILYR(K).EQ.ITP) GO TO 120
24  110 CONTINUE
24  PRINT 812, ITP
24  READ ROB, ITP,NN
24  IF(ITP.NE.0) STOP
24  GO TO 150
C
74  120 CONTINUE
C
C
74  DO 140 I=1,6
74  IF(I.EQ.1.AND.ITP.EC.1) GO TO 130
104 IF(AN(1,I).EC.0.AND.NN(2,I).EC.0) GO TO 140
114 130 CONTINUE
120 N=AN(1,I)
120 N=AN(2,I)
121 KNLM=KNLM+1
123 IF(KNLM.GT.12) GO TO 140
132 VXX(1,KNLM)=N*RA(K,1) + N*RB(K,1)
137 VXX(2,KNLM)=N*RA(K,2) + N*RB(K,2)
142 JM(KNLM)=M
144 JN(KNLM)=N
145 IQ(KNLM)=ITP

```

BEAM.2
 BEAM.3
 A.3
 B.2
 B.7
 D.2
 D.3
 E.2
 E.3
 K.2
 K.3
 F.2
 F.3
 BEAM.10
 BEAM.11
 BEAM.12
 BEAM.13
 BEAM.14
 BEAM.15
 BEAM.16
 BEAM.17
 BEAM.18
 BEAM.19
 BEAM.20
 BEAM.21
 BEAM.22
 BEAM.23
 BEAM.24
 BEAM.25
 BEAM.26
 BEAM.27
 BEAM.28
 BEAM.29
 BEAM.30
 BEAM.31
 BEAM.32
 BEAM.33
 BEAM.34
 BEAM.35
 BEAM.36
 BEAM.37
 BEAM.38
 BEAM.39
 BEAM.40
 BEAM.41
 BEAM.42
 BEAM.43
 BEAM.44
 BEAM.45
 BEAM.46
 BEAM.47
 BEAM.48
 BEAM.49
 BEAM.50
 BEAM.51
 BEAM.52
 BEAM.53

```

146      KC(KALM)=KT
150      140 CONTINUE
      C
152      IF(KNUP.GT.12) KALM=12
156      GO TO 100
157      150 CONTINUE
      C
      C
157      NHIGH=NLOW+(KNUM-1)*NDELTA
164      PRINT AC+, ITH,IPH
172      PRINT ECS, NLCK,NHIGH
203      PRINT B10,LMAX
      C
      C
      800 FORMAT( 2I5 )
      802 FORMAT( 3I5 )
      804 FORMAT( // 10X*INCIDENT BEAM ANGLES ARE THETA=* I4,* PHI=* I4,
      2 / 42X 15H***** )
      806 FORMAT( // 10X *INCIDENT ELECTRON ENERGY FROM * I3, * TC * I3,
      2 * E.V. * )
      808 FORMAT( 15,6(1X 2I2) )
      810 FORMAT( / 10X *MAXIMUM L-VALUE IS * I2 )
      812 FORMAT( /// 10X*ICN CGRE TYPE * I2,* ENCOUNTERED IN SUBROUTINE BEABFAM,
      2*, BUT PHASE SHIFTS NOT SUPPLIED.* / )
      C
211      RETLRN
212      END

```

BEAM.54
BEAM.55
BEAM.56
BEAM.57
BEAM.58
BEAM.59
BEAM.60
BEAM.61
BEAM.62
BEAM.63
BEAM.64
BEAM.65
BEAM.66
BEAM.67
BEAM.68
BEAM.69
BEAM.70
BEAM.71
BEAM.72
BEAM.73
BEAM.74
BEAM.75
BEAM.76
BEAM.77
BEAM.78
BEAM.79
BEAM.80

SUBPROGRAM LENGTH
000342

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

100	-	000025	110	-	000052	120	-	000075	130	-	000115
140	-	000151	150	-	000160	800	-	000221	802	-	000223
804	-	000225	806	-	000237	808	-	000250	810	-	000253
812	-	000250									

BLOCK NAMES AND LENGTHS

BLOCKA	-	000163	BLOCKB	-	004324	BLOCKD	-	000055	BLOCKE	-	000012
BLOCKK	-	012712	BLOCKF	-	000131						

VARIABLE ASSIGNMENTS

A	-	CCCC00S01	B	-	000024S01	BASIS	-	000160S01	BBB	-	CC0000S05
CSTM	-	CCCC04S06	C	-	000050S01	I	-	000334	ILYR	-	CC4300S02
IPHI	-	CCCC01S06	IQ	-	000101S06	IST	-	000010S03	ITH	-	CC0000S06
ITP	-	CC0333	JM	-	000051S06	JN	-	000065S06	K	-	CC0335
KNUM	-	CCCC00S06	KC	-	000115S06	KT	-	000336	KTYP	-	012700S05
LAYER	-	CCCC00S03	LMAX	-	CC0000S04	LSYM	-	004312S02	M	-	CC0337
N	-	000340	NDELTA	-	000002S03	NHIGH	-	000341	NLOW	-	000000S03
NN	-	000317	NNLM	-	CC0000S03	RA	-	000106S01	RB	-	CC0132S01
T	-	CC0000S02	TAL	-	000220S05	TT	-	012320S05	VXX	-	CC0000S06

START OF CONSTANTS
000215

	SUBROUTINE DEBYE	DEBYE.2
C	COMMON/BLCKCB/TI2,BC,7),ILYR(10),LSYM(10)	DEBYE.3
	COMMON/BLCKCB/NLW,ANUM,NDELTA,ARG,NRGY,LAYER,NUMPAX,NCIF,IST(6,6)D.2	8.2
	COMMON/BLCKCB/W(10),ATWT(10),TEMP,SURF,BULK,CIN(10),DCLT(10,12)	8.3
	COMMON/BLCKCB/LA	0.3
	COMMON/BLCKCB/W(10),ATWT(10),TEMP,SURF,BULK,CIN(10),DCLT(10,12)	DEBYE.7
	DIMENSION X(200),Y(200)	DEBYE.8
C	CONST=72.04073	DEBYE.9
2	NUMBER=200	DEBYE.10
C		DEBYE.11
4	READ 800, TEMP	DEBYE.12
11	READ 802, SURF,BULK	DEBYE.13
C		DEBYE.14
21	ENTRY DEBYE1	DEBYE.15
C		DEBYE.16
26	IF(TEMP.LT.1.0) GO TO 150	DEBYE.17
37	SRF=SURF/TEMP	DEBYE.18
33	BLK=BULK/TEMP	DEBYE.19
35	LMN=0	DEBYE.20
35	DEL=SRF/NUMBER	DEBYE.21
40	100 CONTINUE	DEBYE.22
C		DEBYE.23
40	X(1)=C.C	DEBYE.24
41	Y(1)=1.C	DEBYE.25
42	LMN=LMN+1	DEBYE.26
44	DO 110 I=2,NUMBER	DEBYE.27
46	X(I)=X(I-1)+CEL	DEBYE.28
51	Y(I)=X(I)/(EXP(X(I))-1.0)	DEBYE.29
56	110 CONTINUE	DEBYE.30
C		DEBYE.31
60	CALL CUBINT(X,Y,NUMBER,ANS)	DEBYE.32
C		DEBYE.33
63	IF(LMN.GE.2) GO TO 120	DEBYE.34
66	FSURF=CONST*(C.25+ANS/SRF)/SURF	DEBYE.35
72	CEL=PLK/NUMBER	DEBYE.36
75	GO TO 100	DEBYE.37
C		DEBYE.38
75	120 CONTINUE	DEBYE.39
75	FBULK=CONST*(C.25+ANS/BLK)/BULK	DEBYE.40
C		DEBYE.41
C	CALCULATE DEBYE-WALLER CORRECTION FOR EACH LAYER (W(I)) *****	DEBYE.42
C		DEBYE.43
101	W(1)=FSURF/ATWT(1)	DEBYE.44
C		DEBYE.45
104	DO 130 I=2,LAYER	DEBYE.46
C		DEBYE.47
C	FOR SINGLE ATOMIC TYPE... NN=1.	DEBYE.48
C		DEBYE.49
112	NN=1	DEBYE.50
114	W(I)=FBULK/ATWT(NN)	DEBYE.51
114	130 CONTINUE	DEBYE.52
116	140 CONTINUE	DEBYE.53
C		DEBYE.54
116	PRINT #04, TEMP,SURF,BULK	DEBYE.55
130	RETURN	DEBYE.56
131	150 CONTINUE	DEBYE.57
131	DO 150 I=1,10	DEBYE.58
136	W(I)=0.0	DEBYE.59
137	160 CONTINUE	DEBYE.60
140	GO TO 140	DEBYE.61
C		DEBYE.62
C		DEBYE.63
	#03 FORMAT(F10.5)	DEBYE.64
	#02 FORMAT(2F10.5)	DEBYE.65
	#04 FORMAT(// 10X *LATTICE TEMPERATURE=* F5.0, /10X*DEBYE TEMPERATURE	DEBYE.66
	25 *F5.0, * AND* F5.0)	DEBYE.67
C		DEBYE.68
140	END	DEBYE.69

```

SLPRTIME CUBINT( X, Y, A, ANS)
DISPENSE X(S00), Y(S00)
IGC = 1
C1 = (Y(1)-Y(1))/((X(2)-X(1)))
C2 = (Y(1)-Y(2))/((X(2)-X(1)))
C3 = (C2-C1)/(X(1)-X(1))
N1 = N-1
C1 = 1-A1
C2 = (Y(1)-Y(1))/((X(1)-X(1)))
C3 = (C2-C1)/(X(1)-X(1))
C3 = (C2-C1)/(X(1)-X(1))
A = Y(1)-X(1)*C1-X(1)*C2-X(1)*C3
B = C1-C2*(X(1)-X(1))/X(1)-X(1)*C2*(X(1)-X(1))-X(1)*C3*(X(1)-X(1))
C = C2-C3*(X(1)-X(1))-X(1)*C3
C = C3
IF( A .EQ. 4) GO TO 6
GO TO (2, 3, 4), IGC
2 CALL CRATE( A, B, C, O.O, O.O, X(1), X(1), ANS)
IGC = 1
IF( I .EQ. A-4) IGC = 3
GO TO 5
3 CALL CRATE( A, B, C, O.O, O.O, X(1), X(1), SLP)
ANS = ANS+SLP
IF( I .EQ. A-4) IGC = 3
GO TO 5
4 CALL CRATE( A, B, C, O.O, O.O, X(1), X(1), SLP)
ANS = ANS+SLP
5 CENTIALE
C01 = C12
C12 = C23
C02 = C13
CONTINUE
RETURN
6 CALL CRATE( A, B, C, O.O, O.O, X(1), X(1), ANS)
RETURN
END

```

SUBPROGRAM LENGTH
000314

FLUCTUATION ASSIGNMENTS

STATEMENT ASSIGNMENTS
Z - C001C2 2 - 00012b 4 - 000124 5 - C00175
6 - C00204

BLOCK NAMES AND LENGTHS

VARIABLE		ASSIGNMENTS		C		C		C	
A	-	000307	H	-	000310	C	-	000311	C
CC1	-	000277	CC2	-	000271	CC3	-	000308	CC2
CC3	-	000305	D3	-	000304	I	-	000302	IGC
N1	-	000702	SLP	-	000313				

START OF CCASTANIS

CCC285

START CF TEMPERATURES
CCQ236

START CF INACTIVES
000766

UNUSEC COMPILER SPACE
124300

	SUBROUTINE GRATE1 A, B, C, D, E, F, M1, M2, ANS)	CUMINT.39
	DIMENSION G(2), Z(2)	CLUBIA1.40
14	Z(1) = M1	CUMINT.41
16	Z(2) = M2	CUMINT.42
20	OC1 = 1-M2	CUMINT.43
21	M1 = F*Z(1)	CUMINT.44
22	M2 = F*Z(1)*OC1/2.C	CLUBIA1.45
23	M3 = F*Z(1)*OC1/3.C	CUMINT.46
27	M4 = F*Z(1)*OC1/4.C	CUMINT.47
32	M5 = F*Z(1)*OC1/5.C	CLUBIA1.48
34	M6 = F*Z(1)*OC1/6.C	CUMINT.49
37	1 CL1 = M1+M2+M3+M4+M5+M6	CLUBIA1.50
42	ANS = G(2)-CL1	CUMINT.51
44	RETURN	CUMINT.52
45	END	CLUBIA1.53

	SUBPOLINE TABLE	TABLE.2
C		TABLE.2
C	A TABLE OF SPHERICAL HARMONICS (STORE) IS STORED IN THE LARGE ARR	TABLE.4
C	FOR EACH LATTICE POINT CONSISTENT WITH THE GEOMETRY .	TABLE.5
C	THE PARAMETER (LAYER) DETERMINES ELECTRON PATH LENGTH CUTOFF *****	TABLE.6
C		TABLE.7
	COMMON/BLCKCA/A(10,2),B(10,2),D(10,3),RA(10,2),RB(10,2),ALAT,SPACEA,2	A.3
2	,BASIS(2),NBASIS	C.2
	COMMON/BLCKCB/V1(3),V2(3,12),V3(3,12),VX,VY,VZ,SQK(12),CCK(12),	C.3
2	YIN(45),YOUT(49,12),XKZERO,AREA?	C.4
	COMPLEX YIN,YCUT	D.2
	COMMON/BLCKCD/NLOW,NLUM,NDELTA,NPG,NRGY,LAYER,NUMMAX,NLIF,IST(6,6)D.2	D.3
2	,LR	E.2
	COMMON/BLCKCE/LMAX,LTCP,LMAX1,LMAX2,LMAX3,LMAX4,LX	E.3
2	,ICNE,ITWC,ITREE	H.2
	COMMON/BLCKCH/KX(2401),C(7532),LL(7532),YINT(169)	H.3
	COMPLEX YINT	J.2
	COMMON/BLCKCJ/MAXX(51),RMAG(9000),XMAG(9000),MAXR	X.2
	COMMON/BLCKCX/TIME,ITEM,MULT,ISCALE,IMAX,ITYPE,LSUM,ILCCP,IPWR	X.3
	INTEGER TIME	LARGE.2
	LARGE C.QM(1296),RVEC(30000),C.C(49000)	DIM.2
C		DIM.3
C	DIMENSIONS OF KX,C,AND LL.	DIM.4
	NM3=2401 \$ NM1=7532 \$ NM2=7532	DIM.5
C	DIMENSIONS OF RMAG AND XMAG	DIM.6
4	NM4=9000 \$ NM5=9000	DIM.7
C	DIMENSIONS OF RVEC AND G-ARRAY IN LARGE CORE MEMORY.	DIM.8
5	NM6=30000 \$ NM7=49000	DIM.9
C		TABLE.17
C		TABLE.18
10	LINT=LMAX2-1	TABLE.19
11	MMAX=200	TABLE.20
12	RMAX=2.*C(LAYER,3)+ 0.05	TABLE.21
16	IF(MAXR.NE.C) RMAX=MAXR	TABLE.22
20	LYR=LAYER	TABLE.23
21	IF(MULT.LE.1) LYR=1	TABLE.24
25	PRINT 800, RMAX	TABLE.25
C		TABLE.26
33	LCC=0	TABLE.27
34	INDEX=0	TABLE.28
34	LCCC=C	TABLE.29
C		TABLE.30
C	INDEX=(NU1-1)*LAYER+NU2	TABLE.31
C	SUPMATION IS FASTEST OVER LAYER OF INITIAL SCATTERING *****	TABLE.32
C		TABLE.33
35	DC 200 NU2=1,LYR	TABLE.34
37	DC 200 NU1=1,LYR	TABLE.35
40	IF(NU1.EQ.NU2.AND.NU1.NE.1) GO TO 200	TABLE.36
47	INDEX=INDEX+1	TABLE.37
	920. FORMAT(5X,LAYERS *13,*,*13,* INDEX=* 14)	TABLE.38
C		TABLE.39
52	A1=A(NU1,1)	TABLE.40
54	A2=A(NU1,2)	TABLE.41
56	B1=B(NU1,1)	TABLE.42
57	B2=B(NU1,2)	TABLE.43
61	RR1=D(NU1,1)-C(NU2,1)	TABLE.44
63	RR2=D(NU1,2)-C(NU2,2)	TABLE.45

65	R2 = D(NU1,3) - D(NU2,3)	TABLE.46
67	RSC3 = R2**2	TABLE.47
70	RX = RMAX - C(NU1,3) - C(NU2,3)	TABLE.48
	C	TABLE.49
	C	TABLE.50
	C	TABLE.51
	C	TABLE.52
	C	TABLE.53
72	IF(LSCM.EQ.?) RX=RMAX	TABLE.54
	C	TABLE.55
75	R2=R2**2	TABLE.56
	C	TABLE.57
	C	TABLE.58
	C	TABLE.59
	C	TABLE.60
77	DO 180 M1=1,MMAX	TABLE.61
100	M=M1-1	TABLE.62
101	ICCM=C	TABLE.63
	C	TABLE.64
102	100 CONTINUE	TABLE.65
	C	TABLE.66
	C	TABLE.67
	C	TABLE.68
102	RHS=C.C	TABLE.69
103	DO 120 N1=1,MMAX	TABLE.70
105	N=N1-1	TABLE.71
107	R1=M*A1+N*B1+RPI	TABLE.72
114	R2=M*A2+N*B2+RP2	TABLE.73
121	RSC=R1**2+R2**2+RSC3	TABLE.74
124	IPASS=0	TABLE.75
124	IF(NBASIS.EQ.0) GO TO 103	TABLE.76
126	R4=R1+NBASIS(1)	TABLE.77
127	R5=R2+NBASIS(2)	TABLE.78
131	RBS=R4**2+R5**2+RSC3	TABLE.79
135	IPASS=1	TABLE.80
136	105 CONTINUE	TABLE.81
	C	TABLE.82
136	IF(N.EQ.0) RMM=RSC	TABLE.83
141	IF(N.EQ.0.AND.NBASIS.NE.0) GO TO 102	TABLE.84
147	IF(RSC.LT.C.C1) GO TO 120	TABLE.85
151	102 CONTINUE	TABLE.86
151	IF(RSC.LE.RX2) GO TO 110	TABLE.87
154	IF(NBASIS.EQ.0) GO TO 130	TABLE.88
155	IF(RBS.LE.RX2) IPASS=2	TABLE.89
160	IF(IPASS.EQ.?) GO TO 110	TABLE.90
162	IF(RSC.LE.RMM) GO TO 120	TABLE.91
164	GO TO 130	TABLE.92
	C	TABLE.93
165	110 CONTINUE	TABLE.94
165	IF(IPASS.EQ.1.AND.RSC.LT.0.01) GO TO 119	TABLE.95
176	IF(IPASS.EQ.2.AND.RBS.LT.0.01) GO TO 120	TABLE.96
206	IF(IPASS.EQ.?) AND.RBS.GT.RX2) GO TO 120	TABLE.97
215	LCC=LCC+1	TABLE.98
216	RMAG(LCC)=SQRT(RSC)	TABLE.99
227	IF(IPASS.EQ.?) RMAG(LCC)=SQRT(RBS)	TABLE.100
230	XMAG(LCC)=V1(1)*R1+V1(2)*R2+V1(3)*R3	TABLE.101
236	IF(IPASS.EQ.?) XMAG(LCC)=V1(1)*R4+V1(2)*R5+V1(3)*R3	TABLE.102
246	LCCC=LCC+1	TABLE.103

250	RVEC(LCCC)=R1	TABLE.104
252	IF(IPASS.EQ.2) RVEC(LOCO)=R4	TABLE.105
256	LCCC=LCCO+1	TABLE.106
257	RVEC(LCCC)=R2	TABLE.107
262	IF(IPASS.EQ.2) RVEC(LOCO)=R5	TABLE.108
266	LCCC=LCCO+1	TABLE.109
267	RVEC(LCCC)=R3	TABLE.110
271	LCL=LCCC-2	TABLE.111
910	FORMAT(1CX>(* 2I4,*) X=* F9.4,* Y=* F9.4,* Z=* F9.4)	TABLE.112
273	ICCM=1	TABLE.113
274	119 CONTINUE	TABLE.114
274	IPASS=IPASS+1	TABLE.115
276	IF(IPASS.EQ.2) GO TO 110	TABLE.116
277	120 CONTINUE	TABLE.117
C		TABLE.118
302	130 CONTINUE	TABLE.119
C		TABLE.120
C	SLM OVER (N.LT.ZERO) *****	TABLE.121
C		TABLE.122
302	DC 150 N1=1,MMAX	TABLE.123
304	N=-N1	TABLE.124
305	R1=M*A1+N*B1+RR1	TABLE.125
312	R2=M*A2+N*B2+RR2	TABLE.126
317	RSQ=P1**2+R2**2+RSQ3	TABLE.127
322	IPASS=0	TABLE.128
323	IF(NBASIS.EQ.0) GO TO 135	TABLE.129
325	R4=P1+R4SIS(1)	TABLE.130
326	R5=R2+R4SIS(2)	TABLE.131
330	RBS=R4**2+R5**2+RSQ3	TABLE.132
334	IPASS=1	TABLE.133
335	GO TO 130	TABLE.134
336	135 CONTINUE	TABLE.135
336	IF(RSQ.LT.0.01) GO TO 150	TABLE.136
341	136 CONTINUE	TABLE.137
341	IF(RSQ.LE.RX2) GO TO 140	TABLE.138
344	IF(NBASIS.EQ.0) GO TO 160	TABLE.139
345	IF(RBS.LE.RX2) IPASS=2	TABLE.140
350	IF(IPASS.EQ.2) GO TO 140	TABLE.141
352	IF(RSQ.LE.RMM) GO TO 130	TABLE.142
354	GO TO 160	TABLE.143
C		TABLE.144
355	140 CONTINUE	TABLE.145
355	IF(IPASS.EQ.2.AND.RBS.GT.RX2) GO TO 150	TABLE.146
366	LCC=LCC+1	TABLE.147
367	RMAG(LCC)=SQRT(RSQ)	TABLE.148
373	XMAG(LCC)=V1(1)*R1+V1(2)*R2+V1(3)*R3	TABLE.149
401	IF(IPASS.EQ.2) RMAG(LOC)=SQRT(RBS)	TABLE.150
407	IF(IPASS.EQ.2) XMAG(LOC)=V1(1)*R4+V1(2)*R5+V1(3)*R3	TABLE.151
417	LCCC=LCC+1	TABLE.152
421	RVEC(LCCC)=R1	TABLE.153
423	IF(IPASS.EQ.2) RVEC(LOCO)=R4	TABLE.154
427	LCCC=LCC+1	TABLE.155
430	RVEC(LCCC)=R2	TABLE.156
433	IF(IPASS.EQ.2) RVEC(LOCO)=R5	TABLE.157
437	LCCC=LCC+1	TABLE.158
440	RVEC(LCCC)=R3	TABLE.159
442	LCL=LCCC-2	TABLE.160
443	ICCM=1	TABLE.161

445	149 CONTINUE	TABLE.162
445	IPASS=IPASS+1	TABLE.163
447	IF(IPASS.EQ.2) GO TO 140	TABLE.164
450	150 CONTINUE	TABLE.165
C		TABLE.166
453	160 CONTINUE	TABLE.167
C		TABLE.168
452	IF(M.LE.0) GO TO 170	TABLE.169
C		TABLE.170
C	INCLUDE NEGATIVE M-VALUES IN THE SUMMATION *****	TABLE.171
C		TABLE.172
455	M=-M	TABLE.173
455	GO TO 100	TABLE.174
C		TABLE.175
456	170 CONTINUE	TABLE.176
456	IF(1CCM.EQ.0) GO TO 190	TABLE.177
457	180 CONTINUE	TABLE.178
C		TABLE.179
C	SUMMATION OVER LAYER (NU2) IS COMPLETED *****	TABLE.180
C		TABLE.181
462	190 CONTINUE	TABLE.182
462	MAXX(INDEX)=LCC	TABLE.183
464	NUX=NL1	TABLE.184
465	NLY=NL2	TABLE.185
467	PRINT 810,NUX,NLY,INDEX,MAXX(INDEX)	TABLE.186
504	200 CONTINUE	TABLE.187
511	LC=3*LCC	TABLE.188
512	NL=LAYER*(LAYER+1)*LMAX4	TABLE.189
516	PRINT 820, NUMMAX,NM1,NUMMAX,NM2,LMAX4,NM3,LCC,NM4,LCC,NM5,LC,NM6,	TABLE.190
	2 NL,NM7	TABLE.191
556	PRINT 830	TABLE.192
	900 FORMAT(2CX*X=* F9.4,* Y=*F9.4,* Z=* F9.4)	TABLE.193
562	IF(NUMMAX.GT.NM1.OR.NUMMAX.GT.NM2) ITEM=5	TABLE.194
573	IF(LCC.GT.NM4.OR.LCC.GT.NM5) ITEM=5	TABLE.195
604	IF(LMAX4.GT.NM3.OR.LC.GT.NM6) ITEM=5	TABLE.196
C		TABLE.197
C		TABLE.198
	800 FORMAT(// 10X*SUBROUTINE TABLE SETS PROPAGATOR CUTOFF LENGTH AT	TABLE.199
	2MAX=* F8.3)	TABLE.200
	810 FORMAT(10X*FINAL LAYERS OF SUMMATION ARE (* 12,*,*12,*) MAXX(*	TABLE.201
	2 13,*)=* 15,* = TOTAL NUMBER OF ATOMIC SITES WITHIN CUTOFF. *)	TABLE.202
	820 FORMAT(/ 10X*ARRAY DIMENSIONS REQUIRED (AND SUPPLIED) BY PROGRAM*	TABLE.203
	2 /27X* C=* 16,* (* 16, *)*	TABLE.204
	3 /27X* LL=* 16,* (* 16, *)*	TABLE.205
	4 /27X* KX=* 16,* (* 16, *)*	TABLE.206
	5 /27X* RMAG=* 16,* (* 16, *)*	TABLE.207
	6 /27X* XMAG=* 16,* (* 16, *)*	TABLE.208
	7 /27X* PVEC=* 16,* (* 16, *)*	TABLE.209
	8 /27X*G(COMPLEX)* 16,* (* 16, *)*	TABLE.210
	830 FORMAT(//)	TABLE.211
C		TABLE.212
616	RETURN	TABLE.213
617	END	TABLE.214

SLAPPROGRAM LENGTH
001072

	PROGRAM LCCP	LCCP.3
C	COMMON/BLCKKA/A(10,2),B(10,2),D(10,3),RA(1C,2),RB(1C,2),ALAT,SPACE	LOCP.4
	2,BASIS(2),NPBASIS	A.2
	COMMON/BLCKKB/T(2,8C,7),ILYR(1C),LSYM(1C)	A.3
	COMPLEX T	B.2
	COMMON/BLCKKC/V1(3),V2(3,12),V3(3,12),VX,VY,VZ,SGK(12),CCK(12),	B.3
	2 YIN(49),YCLT(49,12),XKZERJ,AKEA2	C.2
	COMPLEX YIN,YCUT	C.3
	COMMON/BLCKKD/NLCW,NNUM,NDELTA,ARG,NRGY,LAYER,NLMPAX,NCIF,IST(6,6)	C.4
	2 ,LR	D.2
	COMMON/BLCKKE/LMAX,LTUP,LMAX1,LMAX2,LMAX3,LMAX4,LX	D.3
	2 ,ICNE,ITWC,ITTHREE	E.2
	COMMON/BLCKKF/ITHT,IPHI,THTT,CIST,CSTH(12),VXX(2,12),KAUF,JP(12),	E.3
	2 JN(12),IC(12),KC(12)	F.2
	COMMON/BLCKKG/ESIG,SIG2,XK,DAMP,ZNORM,PCTL,GAMA,SIGMA,LCKEN,NARG	F.3
	2 ,LAMPCA	G.2
	REAL LAMPCA	G.3
	INTEGER SIGMA	G.4
	COMPLEX ZNORM	G.5
	COMMON/BLCKKH/KX(2401),C(7532),LL(7532),YINT(169)	G.6
	COMPLEX YINT	H.2
	COMMON/BLCKKJ/MAXX(51),RMAG(9000),XMAG(9000),MAXP	H.3
	COMMON/BLCKKK/BBB(36,2),TAU(36,36,2),TT(10,12),KTYP(1C)	J.2
	COMPLEX BBB,TAL,TT	K.2
	COMMON/BLCKKL/ARG,H(169),XD(9),PATH1(10),PATH2(10,12)	K.3
	COMPLEX ARG,H	L.2
	COMMON/BLCKKM/RATIO,XCON,KSTAR	L.3
	COMPLEX KSTAR,XCON	M.2
	COMMON/BLCKKN/E(51),EN(51),S(12,51)	M.3
	COMMON/BLCKKC/PX,PY,PZ	N.2
	COMMON/BLCKKX/TIME,ITEM,MULT,ISCALE,IMAX,ITYPE,LSUM,ILCCP,IPWR	Q.2
	INTEGER TIME	X.2
	COMMON/BLCKDEE/W(10),ATWT(10),TEMP,SURF,BULK,CIN(1C),DCLT(10,12)	X.3
	COMMON/CCNST/PI,PI4,PI42,CUN1,CCN2,CCN3,CCN4,CCNSTT,IM,CNST	DEB.2
	COMPLEX IM	CCNST.?
	DIMENSION TIM(20),JE(12)	CCNST.2
	LARGE C.QM(1255),RVFC(30000),C.G(49000)	LCCP.23
		LARGE.2
C		DIM.2
C	DIMENSIONS OF KX,C,AND LL.	DIM.3
	NM3=2401 & NM1=7532 & NM2=7532	DIM.4
C	DIMENSIONS OF RMAG AND XMAG	DIM.5
5	NM4=9000 & NM5=9000	DIM.6
C	DIMENSIONS OF PVFC AND G-ARRAY IN LARGE CORE MEMORY.	DIM.7
6	NM6=30000 & NM7=49000	DIM.8
C		DIM.9
C		LCCP.26
13	ICNE=LAYER*(LAYER-1)*LMAX4	LCCP.27
16	ITWC=ICNE+LAYER*LMAX4	LCCP.28
17	ITTHREE=ITWC+LAYER*LMAX4	LCCP.29
20	LCX=2*LMAX	LOOP.30
C		LOOP.31
C	SWEEP OVER ENERGY RANGE AND CALCULATE INTENSITIES *****	LCCP.32
C		LCCP.33
22	ILCCP=1	LCCP.34
23	DO 15C J=1,NALM	LOOP.35
25	IF(TIME.NE.C) CALL SECOND(TIME3)	LOOP.36
27	ARG=ARG+NCFLTA	LCCP.37

```

31      CALL ENERGY(NRG,NRGY)
32      IF(NRGY.EQ.C) GO TO 130
33      EN(J)=ESIG
34      RATIC=XK*XKZEHQ
35      KSTAR=CMPLX(XK,CAMP)
36      XCCN=-IM*CCN2*PI42*KSTAR
37      V1(1)=XK*VX
38      V1(2)=XK*VY
39      V1(3)=XK*VZ
40      FTR=AREA2/V1(3)**2
41
42      C
43      PRINT #15
44      PRINT #30
45      CP=1./CAMP
46      PRINT #20, NRG,NRGY
47      PRINT #22, ESIG,SIG2
48      PRINT #24, XK,CAMP
49      PRINT #26, ZACRM
50      PRINT #28, CP
51
52      C
53      DAMP IS THE DAMPING FACTOR ... MULTIFLY BY PATH LENGTH *****
54      IF V2 IS A PROPAGATING BEAM, CALCULATE ITS Z-COMPONENT *****
55
56      C
57      DO 115 JBEAM=1,KNUM
58      JB(JBEAM)=1
59      V2(1,JBEAM)=V1(1)+VXX(1,JBEAM)
60      V2(2,JBEAM)=V1(2)+VXX(2,JBEAM)
61      SQK(JBEAM)=V2(1,JBEAM)**2+V2(2,JBEAM)**2
62      QCK(JBEAM)=SQRT(SQK(JBEAM))
63      IF(QCK(JBEAM).GE.XK) JB(JBEAM)=0
64      IF(JB(JBEAM).EQ.0) GO TO 112
65      XKP=SQRT(XK**2-SQK(JBEAM))
66      V2(3,JBEAM)=-XKP
67      CSTD(JBEAM)=XK/XKP
68
69      C
70      OBTAIN INCIDENT AND OUTGOING DAMPING ARRAYS *****
71
72      C
73      DO 100 II=1,LAYER
74      IF(JBEAM.EQ.1) PATH1(II)=(D(II,3)+DIST)*THET
75      IF(JBEAM.EQ.1) DIN(II)=EXP(-PATH1(II)*CAMP)
76      PATH2(II,JBEAM)=(D(II,3)+DIST)*CSTD(JBEAM)
77      DCUT(II,JBEAM)=EXP(-PATH2(II,JBEAM)*CAMP)
78      100 CONTINUE
79
80      C
81      CSTD(JBEAM) IS INVERSE OF COSINE FOR OUTGOING BEAM *****
82      COMPLETE SPHERICAL HARMONICS FOR OUTGOING BEAMS *****
83
84      C
85      CALL YLM(V2(1,JBEAM),V2(2,JBEAM),V2(3,JBEAM),YINT,LCX,-1)
86
87      C
88      DO 105 L=1,LMAX3
89      YCLT(L,JBEAM)=YINT(L)
90      105 CONTINUE
91
92      C
93      DO 110 L=1,3
94      V3(L,JBEAM)=V1(L)-V2(L,JBEAM)
95      110 CONTINUE
96      GO TO 115
97
98      C

```

LCCP.38
 LCCP.39
 LCCP.40
 LCCP.41
 LCCP.42
 LCCP.43
 LCCP.44
 LCCP.45
 LCCP.46
 LCCP.47
 LCCP.48
 LCCP.49
 LCCP.50
 LCCP.51
 LCCP.52
 LCCP.53
 LCCP.54
 LCCP.55
 LCCP.56
 LCCP.57
 LCCP.58
 LCCP.59
 LCCP.60
 LCCP.61
 LCCP.62
 LCCP.63
 LCCP.64
 LCCP.65
 LCCP.66
 LCCP.67
 LCCP.68
 LCCP.69
 LCCP.70
 LCCP.71
 LCCP.72
 LCCP.73
 LCCP.74
 LCCP.75
 LCCP.76
 LCCP.77
 LCCP.78
 LCCP.79
 LCCP.80
 LCCP.81
 LCCP.82
 LCCP.83
 LCCP.84
 LCCP.85
 LCCP.86
 LCCP.87
 LCCP.88
 LCCP.89
 LCCP.90
 LCCP.91
 LCCP.92
 LCCP.93
 LCCP.94
 LCCP.95

276	112	CONTINUE	LCCP.96
303		V2(3,JBEAM)=0.0	LCCP.97
303		V2(1,JBEAM)=0.0	LOOP.98
304		V2(2,JBEAM)=0.0	LCCP.99
304		V2(3,JBEAM)=0.0	LCCP.100
305		CSTH(JBEAM)=0.0	LCCP.101
306		DO 113 L=1,LAYER	LCCP.102
315		DELTA(L,JBEAM)=0.0	LOOP.103
316		PATH2(L,JBEAM)=0.0	LCCP.104
316	113	CONTINUE	LCCP.105
317		DO 114 L=1,LMAX3	LOOP.106
327		YCLT(L,JBEAM)=(0.0,0.0)	LCCP.107
331	114	CONTINUE	LOOP.108
	C		LCCP.109
332	115	CONTINUE	LCCP.110
335		PRINT #12	LCCP.111
340		PRINT #13, V1	LCCP.112
346		DO 118 JBEAM=1,KALM	LOOP.113
350		PRINT #14, IC(JBEAM), JM(JBEAM), JN(JBEAM), (V2(I,JBEAM), I=1,3),	LCCP.114
		2 (V3(I,JBEAM), I=1,3)	LCCP.115
374	118	CONTINUE	LCCP.116
	C		LCCP.117
	C		LOOP.118
377		CALL FILLB	LOOP.119
400		IF(MULT.LE.1) GO TO 120	LCCP.120
403		IF(TIME.NE.C) CALL SECOND(TIME4)	LCCP.121
405		CALL GREEN(1)	LCCP.122
407		CALL PERTB	LOOP.123
410		IF(TIME.NE.C) CALL SECOND(TIME5)	LCCP.124
412		CALL INV	LCCP.125
414		IF(TIME.NE.C) CALL SECOND(TIME6)	LCCP.126
417	120	CONTINUE	LCCP.127
	C		LCCP.128
417		LM=1	LCCP.129
420		CALL KINEM	LCCP.130
421		CALL TSLM(LM,FTR,J)	LOOP.131
	C		LCCP.132
424		IF(MULT.LE.1) GO TO 128	LCCP.133
	C		LCCP.134
	C	CCMPUTE INTERPLANE PROPAGATORS. STACK HORIZONTALLY IN G(INDEX).	LCCP.135
	C		LOOP.136
427		INDEX=1	LCCP.137
427		IBEGIN=-LMAX4	LCCP.138
431		DO 125 I2=1,LAYER	LOOP.139
433		DO 125 I1=1,LAYER	LCCP.140
434		IF(I1.EQ.I2) GO TO 125	LCCP.141
436		IBEGIN=IBEGIN+LMAX4	LCCP.142
437		INDEX=INDEX+1	LCCP.143
441		CALL GREEN(INDEX,IBEGIN)	LOOP.144
443	125	CONTINUE	LCCP.145
450		IF(TIME.NE.C) CALL SECOND(TIME7)	LCCP.146
	C		LCCP.147
	C	GSTACK SETS UP MATRICES FOR DOUBLE DIFFRACTION.	LCCP.148
	C		LOOP.149
452		LM=2	LCCP.150
453		CALL GSTACK	LCCP.151
454		CALL TSLM(LM,FTR,J)	LOOP.152
457		IF(TIME.NE.C) CALL SECOND(TIME8)	LCCP.153

```

C
C      ITERATE TO ORDER NCIF.
C
462      KLM=0
463      NCF=NCIF-2
465      IF(NCF.LE.0) GO TO 128
466      DO 127 I=1,NCF
47C      CALL SCATT
471      CALL TSL4(I+2,FTP,J)
475      IF(TIME.NE.C) CALL SECOND(TIM(I))
5C1      KLM=I
503      127 CONTINUE
C
505      128 CONTINUE
505      IF(TIME.EQ.C) GO TO 130
506      PRINT 822, TIME3, TIME4, TIME5, TIME6, TIME7, TIME8
526      IF(KLM.EQ.0) GO TO 130
527      PRINT 834, (TIM(IJK), IJK=1,KLM)
536      130 CONTINUE
536      140 CONTINUE
536      PRINT 815
542      150 CONTINUE
C
545      PRINT 806
550      DO 150 I=1,KNUM
552      PRINT 804, IC(I), JN(I), JN(I)
566      PRINT 802, (E(IJ), S(I,J), J=1,ANUM)
607      150 CONTINUE
C
C
802 FORMAT( / 10X 4(*E=* F4.0,*--* E16.8,5X) )
804 FORMAT( // 10X*TYPE(*I2,*) BEAM(*I2,*,*I2,*) ENERGY-VS-INTEN-
2SITY* / 10X 7***** )
806 FORMAT( 15, 6(1X 2I2) )
812 FORMAT( / )
814 FORMAT( / 10X*ENERGY(*I2,*)= F4.0,* E.V. SINGLE=*E14.6,* DCUBLC
2LE=*E14.6,* TRIPLE=*E14.6, /40X 14H ***** 10X
3 14H ***** 12X 14H ***** / )
816 FORMAT( 2X 17(10H* * * * * ) )
818 FORMAT( / 10X*PROPAGATION VECTORS IN CRYSTAL V1=* 3F9.5 )
819 FORMAT( 15X*TYPE(*I2,*) BEAM(*I2,*,* I2,*) V2=* 3F9.5,
2 * V3=* 3F9.5 )
820 FORMAT( / 10X*INCIDENT BEAM ENERGY IS * I3,* E.V. $$$$ NRGY=*
2 I4 )
612 822 FORMAT( 10X*RENORMALIZED QUASIPARTICLE ENERGY E=(*LOCP.199
2 F9.4,*,* F9.4, *) E.V. * )
824 FORMAT( 10X*COMPLEX K-VECTOR KSTAR=(*LOCP.201
2 F8.5,*,* F8.5,*) INVERSE ANGSTROMS * )
826 FORMAT( 10X*CLASIPARTICLE RENORMALIZATION CONSTANT ZNORM=(*LOCP.203
2 F8.5, *,* F8.5, *) * )
828 FORMAT( 10X*INELASTIC DAMPING LENGTH LAMDA=(*LOCP.205
2 F8.4, * ANGSTROMS * )
830 FORMAT( 1H1 )
832 FORMAT( / 10X*TIME REQUIREMENTS FOR EXECUTION OF ENERGY LOOP*
2 / 12X*BEGIN LUCP AT * F8.3,
3 / 12X*TO END OF FILLB * F8.3,
4 / 12X*TO END OF SUBPL * F8.3,
LCCP.211
7 / 12X*TO END OF INV * F8.3,
5 / 12X*TO END OF GREEN * F8.3,
6 / 12X*DOUBLE DIFFRACTION* F8.3 )
834 FORMAT( 12X* NEXT ITERATION* F8.3 )
C
612      END
LCCP.212
LCCP.213
LCCP.214
LCCP.215
LCCP.216
LCCP.217
LCCP.218

```

```

SUBROUTINE FILLB
C
C FILLB FILLS THE B-ARRAYS, WHICH ARE OBTAINED FROM THE ATOMIC
C T-ARRAYS AND THE DEBYE WALLER FACTORS. PROVISION IS MADE FOR
C DIFFERENT SURFACE AND BULK DEBYE-WALLER FACTORS.
C B AND T-ARRAYS ARE BOTH DIAGONAL, IMPLYING SPHERICAL SYMMETRY*****
C
COMMON/BLCKKB/T(2,9C,7),ILYR(10),LSYM(1C)
COMMON/BLCKCK/NLCW,NNUM,NDELTA,ARG,NFGY,LAYER,ALCMAX,NCIF,IST(6,6)
2 ,LR
COMMON/BLCKCK/LMAX,LTOP,LMAX1,LMAX2,LMAX3,LMAX4,LX
2 ,ICNF,ITWC,ITREE
COMMON/BLCKCK/ESIG,SIG2,XK,DAMP,ZNORM,POTL,GAMA,SIGMA,LCHEN,NARG
2 ,LAMBDA
REAL LAMBDA
INTEGER SIGMA
COMPLEX ZNORM
COMMON/BLCKCK/RBB(36,2),TAU(36,36,2),TT(10,12),KTYP(1C)
COMPLEX RBB,TAL,TT
COMMON/PLCKCX/TIME,ITEM,MULT,ISCALE,IMAX,ITYPE,LSUP,ILCCP,IPWR
INTEGER TIME
COMMON/BLKDEE/W(10),ATWT(10),TEMP,SURF,BULK,DIN(10),CCLT(10,12)
DIMENSION CP(21,21),BCF(21),XINT(20)
COMPLEX TX(11),BGC
C
C LDM=21
2 SMALL=1.0E-10
C
C CALCULATE ARRAY CP IF ILOOP=1 *****
C
3 IF(ILCCP.NE.1) GO TO .20
6 LLL=LDM-2
7 LLD=LDM**2
11 DO 100 L=1,LLD
16 CP(L)=0.0
17 100 CONTINUE
C
20 CP(1,1)=1.0
21 CP(2,2)=1.0
C
22 DO 110 L=1,LLL
27 LL1=L+1
27 LL2=L+2
31 F1=1./LL1
32 CP(1,LL2)=-L*F1*CP(1,L)
C
40 DO 110 N=2,LL2
60 CP(N,LL2)=F1*((2*L+1)*CP(N-1,LL1)-L*CP(N,L))
65 110 CONTINUE
71 ILCCP=1
C
72 DO 115 L=1,LDM
C
73 SUM=0.0
74 DO 112 N=1,L

```

FILLB.2
 FILLB.3
 FILLB.4
 FILLB.5
 FILLB.6
 FILLB.7
 FILLB.8
 FILLB.9
 B.2
 B.3
 C.2
 D.3
 E.2
 E.3
 G.2
 G.3
 G.4
 G.5
 G.6
 K.2
 K.3
 X.2
 X.3
 DEB.2
 FILLB.17
 FILLB.18
 FILLB.19
 FILLB.20
 FILLB.21
 FILLB.22
 FILLB.23
 FILLB.24
 FILLB.25
 FILLB.26
 FILLB.27
 FILLB.28
 FILLB.29
 FILLB.30
 FILLB.31
 FILLB.32
 FILLB.33
 FILLB.34
 FILLB.35
 FILLB.36
 FILLB.37
 FILLB.38
 FILLB.39
 FILLB.40
 FILLB.41
 FILLB.42
 FILLB.43
 FILLB.44
 FILLB.45
 FILLB.46
 FILLB.47
 FILLB.48
 FILLB.49
 FILLB.50

103	SUM=SUM+CP(N,L)	FILL8.51
104	112 CONTINUE	FILL8.52
106	115 CONTINUE	FILL8.53
C		FILL8.54
110	120 CONTINUE	FILL8.55
C		FILL8.56
110	DC 220 II=1,LAYER	FILL8.57
112	KT=KTYP(II)	FILL8.58
113	IF(II.EC.1) CC TO 124	FILL8.59
115	III=II-1	FILL8.60
C		FILL8.61
C	IF ION CORE TYPE (KT) HAS PREVIOUSLY BEEN CALCULATED,	FILL8.62
C	GO TO THE END OF THE LOOP.	FILL8.63
C		FILL8.64
116	DC 122 J=1,III	FILL8.65
123	IF(KT.EC.KTYP(J)) GO TO 220	FILL8.66
124	122 CONTINUE	FILL8.67
126	124 CONTINUE	FILL8.68
C		FILL8.69
126	DC 130 L=1,LTCP	FILL8.70
140	TX(L)=T(KT,ARGY,L)	FILL8.71
142	130 CONTINUE	FILL8.72
144	140 CONTINUE	FILL8.73
C		FILL8.74
C	SET DEBYE-WALLER FACTOR ... BETA. STACK EXPONENTIAL COEFFICIENTS	FILL8.75
C		FILL8.76
144	BETA=2.*N(II)*XK**2	FILL8.77
C		FILL8.78
147	XINT(1)=1.0	FILL8.79
151	DC 150 I=1,19	FILL8.80
153	XINT(I+1)=XINT(I)*BETA/I	FILL8.81
156	IF(ABS(XINT(I+1)).GT.SMALL) GO TO 150	FILL8.82
163	LARGE=I+1	FILL8.83
164	GO TO 140	FILL8.84
164	150 CONTINUE	FILL8.85
166	160 CONTINUE	FILL8.86
C		FILL8.87
C		FILL8.88
C	COMPUTE FOURIER COEFFICIENTS BCF *****	FILL8.89
C		FILL8.90
166	XFCN=EXP(-BETA)	FILL8.91
172	DC 190 L=1,LTCP	FILL8.92
173	TCTAL=0.0	FILL8.93
173	ISIGN=-1	FILL8.94
C		FILL8.95
175	DC 180 N=1,L	FILL8.96
176	ISIGN=-ISIGN	FILL8.97
177	IF(ABS(CP(N,L)).LE.C.00001) GO TO 180	FILL8.98
205	LSTART=2	FILL8.99
206	IF(ISIGN.GT.0) LSTART=1	FILL8.100
211	SUM=0.0	FILL8.101
C		FILL8.102
212	DC 170 M=LSTART,LARGE,2	FILL8.103
223	SUM=SUM+XINT(M)/(M+N-1)	FILL8.104
227	170 CONTINUE	FILL8.105
C		FILL8.106
230	TCTAL=TCTAL+CP(N,L)*SUM	FILL8.107
235	180 CONTINUE	FILL8.108

240	C	LBCF=L	FILLB.109
240		BCF(L)=TOTAL*XPEN	FILLB.110
243		IF (ABS(TOTAL).LE.SMALL) GO TO 200	FILLB.111
240	190	CONTINUE	FILLB.112
250	200	CONTINUE	FILLB.113
	C		FILLB.114
	C	LBCF IS THE MAXIMUM INDEX FOR BCF.GE.SMALL *****	FILLB.115
	C		FILLB.116
250		LLLL=C	FILLB.117
	C		FILLB.118
251		DO 210 L=1,LTOP	FILLB.119
253		M=2*L-1	FILLB.120
255		CALL PCX(L,BCF,IX,LTOP,LBCF,BCG)	FILLB.121
	C		FILLB.122
261		DO 210 N=1,M	FILLB.123
266		LLLL=LLLL+1	FILLB.124
267		PBB(LLLLL,KT)=BCG	FILLB.125
275	210	CONTINUE	FILLB.126
	C		FILLB.127
300		PRINT 900, II,KT	FILLB.128
	900	FORMAT(/ 10X*FILLB LAYER* 13,* KTOP=* 13)	FILLB.129
310	220	CONTINUE	FILLB.130
	C		FILLB.131
	C		FILLB.132
313		RETURN	FILLB.133
313		END	FILLB.134
			FILLB.135
			FILLB.136

```

      SLOPOTIME MCXILL,BCF,TR,LTCP,LBCF,X)
C
      DIMENSION RCF(1)
      C=PLP X,Y,Z,TR(1)
C
      X=CC,C,C,C
12  GC '10 L1=1,LTCP      FILL0,137
14  Y=TR(L1)              FILL0,138
17  LCV=IABS(L1-L1)+1     FILL0,139
22  L1=L1-L1-1            FILL0,140
22  IF(L1-1,CF,LBCF) L1=LBCF FILL0,141
26  IF(L1-1,LT,LCV) GC TC 110 FILL0,142
C
71  GC 100 L2=LCV,LMI     FILL0,143
76  Z=Y*CF(L2)            FILL0,144
67  X=X*Z                  FILL0,145
46  100 CONTINUE          FILL0,146
52  110 CONTINUE          FILL0,147
C
C
C
C
55  RETLNA                 FILL0,148
55  END                    FILL0,149

```

SLBPCCFAD LENGTH
CCOLCE

FLACTICA ASSIGNMENTS

STATEFAM1 ASSIGNMENTS
110 - CCCC53

BLACK MAPES AND LENGTHS

VARIABLE ASSIGNMENTS

LM1	-	000103	LCM	-	000102	LI	-	000101	L2	-	000104
Y	-	000075	Z	-	000077						

START CF CCASTATS
0000C

START OF TEMPERATURES
000000

START CF INDIRECTS
CCC07:

UNLSEC CCHPTLER SPACE
1247CC

```

C          SUBROUTINE KINEP                      KINEMP=.3
C          CCMPOW=PLCCKC/V1(13),V2(13,12),V3(13,12),VV,VV,V2,SOH(12),CCR(12), KINEM=.3
C          2 V1N(6),VCUT(14,12),XKZENU,AREA2 C.2
C          CCMPLX YIN,VCUT C.3
C          CCMPOW=BLCCCK/MLCN,ANUM,NUM(14,ANG,NPGV,LAYER,NUPP2X,NCIF,1ST46),D.2
C          2 ,LP O.3
C          CCMPOW=BLCCCK/LPAX,1TCP,LPX1,LPAK2,LPAK3,LPAK4,LX E.2
C          2 ,AS,1TMC,1TPOEE F.2
C          CCMPCN=CLCCFC/1TPT,IPM1,TMTT,C1ST,C5T(121),XXT(12,12),KNUP,JM(12), F.3
C          2 JN(12),IC(12),K2(12) F.3
C          CCMPLXCLCCN/PPR(13,2),TAU(10,7,2),TT(10,12),KTYPE(1C) N.2
C          CCMPLX L DEF,TAL,TT N.3
C          CCMPCN=BLCCCK/MI(10),AT=TT(10),TEPP,SUNP,SUNK,DINI(1C),DELTT(10,12) D.2
C          CCMPLX X KINEMP=.0
C          C KINEMP=.11
C          DC 1IC JBEAP=1,KRNP KINEMP=.12
C          DC 1IC I=1,LAYER KINEMP=.13
C          KJ=NTV(1) KINEMP=.14
C          C KINEMP=.5 KINEMP=.5
C          X=(C,C,C,O) KINEMP=.6
C          C KINEMP=.17
C          11 DC 1CC L=1,LPAK7 KINEMP=.19
C          74 X=X=RPPL,KJ=VCUT(L,JBEAN+OVIN(L) KINEMP=.20
C          37 :CC CANTIALE KINEMP=.21
C          C KINEMP=.22 KINEMP=.22
C          TT(L,JBEAN+X)*JIN(1)*DOUT(I,JBEAP) KINEMP=.24
C          92 1IC CANTIALE KINEMP=.24
C          C KINEMP=.25 KINEMP=.25
C          C KINEMP=.27 KINEMP=.27
C          57 WTLAS KINEMP=.28
C          SAC KINEMP=.28

```

	SUBROUTINE GSTACK	GSTACK.2
C	COMMON/ELCCKC/NLCH, NNUM, NDELTA, ARG, NPGY, LAYER, NUMMAX, NCIF, IST(6,6)	GSTACK.3
2	,LR	D.2
	COMMON/BLCCCK/LMAX, LTGP, LMAX1, LMAX2, LMAX3, LMAX4, LX	D.3
2	, ICNE, ITWC, ITHRE	E.2
	COMMON/BLCCCK/BBB(36,2), TAU(36,36,2), TT(10,12), KTFP(10)	E.3
	COMPLEX BBB, TAU, TT	K.2
	COMMON/BLKDEB/W(10), ATWT(10), TEMP, SURF, BULK, CIN(10), CCLT(10,12)	K.3
	COMPLEX Z, ZERC, UNIT	DEB.2
	LARGE C.QM(1256), RVEC(30000), C.C(490CC)	GSTACK.8
C		LARGE.2
C	DIMENSIONS OF KX, C, AND LL.	DIM.2
	NM3=2401 \$ NM1=7532 \$ NM2=7532	DIM.3
C	DIMENSIONS OF RMAG AND XMAG	DIM.4
4	NM4=9000 \$ NM5=9000	DIM.5
C	DIMENSIONS OF RVEC AND G-ARRAY IN LARGE CORE MEMORY.	DIM.6
5	NM6=30000 \$ NM7=49000	DIM.7
C		DIM.8
C		DIM.9
C	MULTIPLY G*TAU AND STORE IN G(ICNE). DAMPING NOT INCLUDED.	GSTACK.11
C		GSTACK.12
10	UNIT=CMPLX(1.C,0.0)	GSTACK.13
12	ZERC=(0.0,0.C)	GSTACK.14
15	IBEGIN=-LMAX4	GSTACK.15
C		GSTACK.16
16	DO 130 I2=1, LAYER	GSTACK.17
20	DO 130 I1=1, LAYER	GSTACK.18
21	IF(I1.EQ.I2) GO TO 130	GSTACK.19
24	KK=KTFP(I1)	GSTACK.20
25	IBEGIN=IBEGIN+LMAX4	GSTACK.21
C		GSTACK.22
27	L2=ICNE	GSTACK.23
30	LS=IBEGIN-LMAX3	GSTACK.24
32	DO 110 LCUT=1, LMAX3	GSTACK.25
33	LS=LS+LMAX3	GSTACK.26
34	DO 110 LIN=1, LMAX3	GSTACK.27
36	LG=LS	GSTACK.28
37	Z=ZERC	GSTACK.29
C		GSTACK.30
42	DO 100 L=1, LMAX3	GSTACK.31
53	LG=LG+1	GSTACK.32
54	Z=Z+G(LG)*TAU(L, LIN, KK)	GSTACK.33
66	100 CONTINUE	GSTACK.34
67	L2=L2+1	GSTACK.35
70	G(L2)=Z	GSTACK.36
74	110 CONTINUE	GSTACK.37
C		GSTACK.38
C	STACK G(ICNE) BACK IN G(1).	GSTACK.39
C		GSTACK.40
101	DO 120 I=1, LMAX4	GSTACK.41
111	G(IBEGIN+I)=G(ICNE+I)	GSTACK.42
114	120 CONTINUE	GSTACK.43
116	130 CONTINUE	GSTACK.44
C		GSTACK.45
C	INITIALIZE G(ITWC)=0.	GSTACK.46
C		GSTACK.47
123	LMAX=LAYER*LMAX4	GSTACK.48
		GSTACK.49

125	II=ITWC	GSTACK.50
127	DC 140 J=1,JMAX	GSTACK.51
133	II=II+1	GSTACK.52
134	G(II)=ZERO	GSTACK.53
140	140 CONTINUE	GSTACK.54
141	LR=LAYER-1	GSTACK.55
142	IB=-LMAX4	GSTACK.56
144	IA=ICNE-LMAX4	GSTACK.57
145	IC=ITWC-LMAX4	GSTACK.58
	C	GSTACK.59
	C	GSTACK.60
	C	GSTACK.61
	C	GSTACK.62
	C	GSTACK.63
147	DC 180 I2=1,LAYER	GSTACK.64
151	IU=ID+LMAX4	GSTACK.65
	C	GSTACK.66
152	DC 160 I1=1,LAYER	GSTACK.67
154	IF(I1.EQ.I2) GO TO 160	GSTACK.68
157	F=CIN(I1)	GSTACK.69
160	IB=IB+LMAX4	GSTACK.70
	C	GSTACK.71
162	DC 150 L=1,LMAX4	GSTACK.72
173	G(IC+L)=G(IC+L)+G(IB+L)*F	GSTACK.73
202	150 CONTINUE	GSTACK.74
203	160 CONTINUE	GSTACK.75
	C	GSTACK.76
	C	GSTACK.77
	C	GSTACK.78
	C	GSTACK.79
206	IA=IA+LMAX4	GSTACK.80
207	CALL EXCX(IA+1,ID+1)	GSTACK.81
	C	GSTACK.82
	C	GSTACK.83
	C	GSTACK.84
214	F=CIN(I2)	GSTACK.85
216	ICL=IC	GSTACK.86
220	IAL=IA	GSTACK.87
222	DC 170 L=1,LMAX3	GSTACK.88
223	DC 170 LP=1,LMAX3	GSTACK.89
234	ICL=ICL+1	GSTACK.90
235	IAL=IAL+1	GSTACK.91
236	G(ICL)=G(IAL)	GSTACK.92
243	IF(L.EQ.LP) G(ICL)=G(ICL)+UNIT*F	GSTACK.93
253	170 CONTINUE	GSTACK.94
257	180 CONTINUE	GSTACK.95
	C	GSTACK.96
	C	GSTACK.97
261	RETURN	GSTACK.98
262	END	GSTACK.99

SUBPROGRAM LENGTH
000346

FUNCTION ASSIGNMENTS

```

SUBROUTINE SCATT
COMMON/BLCKCD/NLOW,ANUM,NDELTA,ARG,NRGY,LAYER,NUMMAX,NEIF,IST(6,6)
2 ,LR
COMMON/BLCKCE/LMAX,LTCP,LMAX1,LMAX2,LMAX3,LMAX4,LX
2 ,ICNE,ITW,C,ITHREE
COMPLEX ZERO
LARGE C,CM(1256),RVEC(30000),C,G(49000)
C
C DIMENSIONS OF KX,C,AND LL.
NM3=2401 $ NM1=7532 $ NM2=7532
C DIMENSIONS OF RMAG AND XMAG
4 NM4=9000 $ NM5=9000
C DIMENSIONS OF RVEC AND G-ARRAY IN LARGE CORE MEMORY.
5 NM6=30000 $ NM7=49000
C
C
10 ZERO=(0.0,0.0)
C
C INITIALIZE WORKING SPACE G(IGAMMA)=C.
C
12 IT1=ITHREE+1
14 IT2=IT1+LAYER*LMAX4
17 DO 100 I=IT1,IT2
25 G(I)=ZERO
30 100 CONTINUE
C
C INCREASE ORDER OF PERTURBATION EXPANSION FOR T-MATRIX.
C G(IGAMMA)=G(IBETA)*G(IALPHA). G(IGAMMA) IS STACKED HORIZONTALLY.
C
31 IBETA=-LMAX4
32 IGAMMA=-LMAX4+ITHREE
34 DO 110 I2=1,LAYER
36 IGAMMA=IGAMMA+LMAX4
37 DO 110 I1=1,LAYER
C
41 IF(I1.EQ.I2) GO TO 110
42 IBETA=IBETA+LMAX4
44 IALPHA=IBETA+(I1-1)*LMAX4
50 CALL ARITX(IBETA+1,2,IALPHA+1,IGAMMA+1,1,1)
60 110 CONTINUE
C
C TRANSFER G(IGAMMA) TO G(IALPHA)...STACK VERTICALLY.
C ADD CONTRIBUTION FROM G(IALPHA) TO G(ICELTA).
C
65 DO 120 I2=1,LAYER
66 II=(I2-1)*LMAX4+1
71 IGAM=ITHREE+II
72 IALPH=ICNE+II
74 ICEL=ITWO+II
C
76 CALL EXCX(IALPH,IGAM)
100 CALL ARITX(ICEL,1,IALPH,ICEL,0,0)
104 120 CONTINUE
C
C G(ICELTA) IS THE ITERATED PART OF THE SCATTERING MATRIX.
C IT IS TO BE MULTIPLIED BY TAU*DCUT*G(IDELTA) *** IN TSUM.
C
C
107 RETURN
107 END

```

SCATT.2
 D.2
 D.3
 E.2
 E.3
 SCATT.5
 LARGE.2
 DIM.2
 DIM.3
 DIM.4
 DIM.5
 DIM.6
 DIM.7
 DIM.8
 DIM.9
 SCATT.8
 SCATT.9
 SCATT.10
 SCATT.11
 SCATT.12
 SCATT.13
 SCATT.14
 SCATT.15
 SCATT.16
 SCATT.17
 SCATT.18
 SCATT.19
 SCATT.20
 SCATT.21
 SCATT.22
 SCATT.23
 SCATT.24
 SCATT.25
 SCATT.26
 SCATT.27
 SCATT.28
 SCATT.29
 SCATT.30
 SCATT.31
 SCATT.32
 SCATT.33
 SCATT.34
 SCATT.35
 SCATT.36
 SCATT.37
 SCATT.38
 SCATT.39
 SCATT.40
 SCATT.41
 SCATT.42
 SCATT.43
 SCATT.44
 SCATT.45
 SCATT.46
 SCATT.47
 SCATT.48
 SCATT.49
 SCATT.50
 SCATT.51
 SCATT.52

	SUBROUTINE TSLM(ITR,FTR,JINDEX)	TSUM.2
C	CCMMCN/BLCKCK/A(1C,2),B(1C,2),C(10,3),RA(10,2),RB(1C,2),ALAT,SPACEA.2	TSUM.3
	2,BASIS(2),NPBASIS	A.3
	CCMMCN/BLCKCK/V1(3),V2(3,12),V3(3,12),VX,VY,VZ,SQK(12),CCK(12),	C.3
	2 YIN(49),YCLT(49,12),XKZERO,AREA2	C.3
	CCMPLEX YIN,YCUT	C.4
	CCMMCN/BLCKCK/NLCK,NKUM,NDELTA,MPG,MPGY,LAYER,NUMMAX,NCIF,IST(5,6)	D.2
	2 ,LR	D.3
	CCMMCN/BLCKCK/LMAX,LTCP,LMAX1,LMAX2,LMAX3,LMAX4,LX	E.2
	2 ,ICNE,ITWC,ITHPEE	E.3
	CCMMCN/BLCKCK/ITHT,IPHI,THTT,DIST,CSTH(12),VXX(2,12),KNLF,JM(12),	F.2
	2 JN(12),IC(12),KC(12)	F.3
	CCMMCN/BLCKCK/RHB(36,2),TAU(36,36,2),TT(10,12),KTYF(10)	K.2
	CCMPLEX RPE,TAL,TT	K.3
	CCMMCN/BLCKCK/E(51),EN(51),S(12,51)	N.2
	CCMMCN/BLCKCK/W(10),ATWT(10),TEMP,SURF,BULK,CIN(1C),DCLT(10,12)	OFB.2
	CCMPLEX Z,ZERO,CPX,TERM,SM1,SM(12)	TSUM.12
	DIMENSION NBEAM(4)	TSUM.13
	DATA(NBEAM(I),I=1,4)/00,01,10,11/	TSUM.14
	LARGE C.JM(1296),RVEC(30000),C.G(49000)	LARGE.2
C		DIM.2
C	DIMENSIONS OF KX,C,AND LL.	DIM.3
	NM3=2401 \$ NM1=7532 \$ NM2=7532	DIM.4
C	DIMENSIONS OF RMAG AND XMAG	DIM.5
	NM4=9000 \$ NM5=9000	DIM.6
C	DIMENSIONS OF RVEC AND G-ARRAY IN LARGE CORE MEMORY.	DIM.7
11	NM6=30000 \$ NM7=49000	DIM.8
C		DIM.9
13	ZERO=(0.0,0.0)	TSUM.17
16	IF(ITR.EQ.1) GO TO 125	TSUM.18
17	IHGIN=ITWC-LMAX4	TSUM.19
C		TSUM.20
21	DO 120 I2=1,LAYER	TSUM.21
24	IHGIN=IHGIN+LMAX4	TSUM.22
25	KK=KTYF(I2)	TSUM.23
C		TSUM.24
27	DO 50 I=1,KNUM	TSUM.25
34	SM(I)=ZERO	TSUM.26
36	90 CCNTINLE	TSUM.27
C		TSUM.28
37	DO 110 LCUT=1,LMAX3	TSUM.29
42	IC=IHGIN	TSUM.30
43	DO 110 LIN=1,LMAX3	TSUM.31
45	TERM=ZERO	TSUM.32
C		TSUM.33
47	DO 100 L=1,LMAX3	TSUM.34
61	IC=IC+1	TSUM.35
62	TERM=TERM+TAL(LCUT,L,KK)*G(IC)	TSUM.36
73	100 CCNTINLE	TSUM.37
C		TSUM.38
74	CPX=TERM*YIN(LIN)	TSUM.39
104	DO 110 I=1,KNUM	TSUM.40
117	SM(I)=SM(I)+CPX*YCLT(LCUT,I)	TSUM.41
127	110 CCNTINLE	TSUM.42
C		TSUM.43
137	DO 120 I=1,KNUM	TSUM.44
150	TT(12,I)=SM(I)*DCLT(12,I)	TSUM.45

154	120 CONTINUE	TSUM.46
C		TSUM.47
C	SUM TT-VALLES OVER SURFACE LAYERS.	TSUM.48
C		TSUM.49
162	125 CONTINUE	TSUM.50
162	GASP=C.C	TSUM.51
163	PRINT 800, ITR	TSUM.52
171	CC 140 JEEAM=1,KALM	TSUM.53
175	SM1=ZERO	TSUM.54
177	PRINT 803,JM(JBEAM),JN(JBEAM)	TSUM.55
211	DC 130 I2=1,LAYER	TSUM.56
215	Y=ZERO	TSUM.57
217	DC 128 INC=1,3	TSUM.58
230	Y=Y+V*(INC,JBEAM)*C(I2,IND)	TSUM.59
232	128 CONTINUE	TSUM.60
234	Z=CMPLX(COS(Y),SIN(Y))	TSUM.61
246	TERM=Z*TT(I2,JBEAM)	TSUM.62
255	SM1=SM1+TERM	TSUM.63
261	TMG=CABS(TT(I2,JBEAM))	TSUM.64
265	IF(TMG.GT.2.C) GASP=100.0	TSUM.65
274	PRINT 802, I ,TMG,TT(I2,JBEAM),TERM,SM1	TSUM.66
317	130 CONTINUE	TSUM.67
C		TSUM.68
322	S(JBEAM,JINDX)=FTR*CABS(SM1)**2	TSUM.69
331	SLCG=ALCG10(S(JBEAM,JINDX))	TSUM.70
336	PRINT 804,E(JINDX),S(JBEAM,JINDX),SLCG	TSUM.71
361	140 CONTINUE	TSUM.72
364	IF(GASP.LT.0.0001) RETURN	TSUM.73
367	PRINT 803	TSUM.74
372	STOP	TSUM.75
C		TSUM.76
C		TSUM.77
800	FORMAT(/// 10X*ITERATION OF T-MATRIX TO ORDER* I3 /)	TSUM.78
802	FORMAT(10X*LAYER=*I3,* T=/*F11.6,* / *2F10.6,* TERM=*2F10.6,	TSUM.79
	2 * SUM=*2F10.6)	TSUM.80
804	FORMAT(/10X*ENERGY=* F4.0,* E.V. INTENSITY=* E14.6,* LCGARITSUM.81	
	2THM=* F10.5,/ 42X 11H*****)	TSUM.82
806	FORMAT(/ 3X 212,* BEAM*)	TSUM.83
808	FORMAT(// 10X*PROGRAM STOPPED DUE TO NON-CONVERGENCE OF LAYER SCATSUM.84	
	2TTERING AMPLITUDE*)	TSUM.85
C		TSUM.86
377	RETURN	TSUM.87
400	END	TSUM.88

	SUPRCUTINE INV	INV.2
C		INV.3
C	INVERTS MATRIX (I-BEB*GSP) AND CALCULATES TAU-MATRIX.	INV.4
C		INV.5
	CCPMCN/BLCCCK/V1(3),V2(3,12),V3(3,12),VX,VY,VZ,SGK(12),CCK(12),	C.2
2	YIN(49),YCLT(49,12),XKZEK,AREA2	C.3
	CCMPLEX YIN,YCLT	C.4
	CCPMCN/BLCCCK/LMAX,LTUP,LMAX1,LMAX2,LMAX3,LMAX4,LX	E.2
2	,ICNE,ITAC,ITHREE	E.3
	CCPMCN/BLCCCK/ITHT,IPHI,THTT,DIST,CSTH(12),VXX(2,12),KNUP,JM(12),	F.2
2	JN(12),IC(12),KQ(12)	F.3
	CCPMCN/BLCCCK/BBB(36,2),TAU(36,36,2),TT(10,12),KTYP(10)	K.2
	CCMPLEX BBB,TAL,TT	K.3
	CCPMCN/BLCCCK/TIME,ITEM,MULT,ISCALE,IMAX,ITYPE,LSUP,TLCCP,IPWR	X.2
	INTEGER TIME	X.3
	CCPMCN/CONST/PI,P14,P142,CON1,CCN2,CCN3,CON4,CCNSTT,IM,CNST	CCNST.2
	CCMPLEX IM	CCNST.3
	CCMPLEX X,L,TERM,P(12)	INV.12
	DIMENSION PX(12)	INV.13
	LARGE C,CM(1256),RVEL(30000),C.C(49000)	LARGE.2
	LARGE Q(20000)	INV.15
C		DIP.2
C	DIMENSIONS OF KX,C,AND LL.	DIP.3
	NM2=2401 \$ NM1=7532 \$ NM2=7532	DIM.4
C	DIMENSIONS OF RMAG AND XPAG	DIP.5
4	AP4=9000 \$ AP5=9000	DIM.6
C	DIMENSIONS OF RVEC AND G-ARRAY IN LARGE CORE MEMORY.	DIP.7
5	NM6=30000 \$ NM7=49000	DIP.8
C		DIP.9
C		INV.17
10	DC 112 KK=1,ITYPE	INV.18
11	INDEX=C	INV.19
11	IX=0	INV.20
12	DC 100 I=1,LMAX3	INV.21
14	DC 100 J=1,LMAX3	INV.22
27	X=BBB(J,KK)	INV.23
31	IX=IX+1	INV.24
32	INDEX=INDEX+2	INV.25
34	L=(C,C,C,C)	INV.26
36	IF(I.EC,J) U=CMPLX(1.0,0.0)	INV.27
42	U=U-X*C(IX)	INV.28
54	Q(INDEX)=AIMAG(U)	INV.29
60	Q(INDEX-1)=REAL(U)	INV.30
62	100 CCNTINLE	INV.31
C		INV.32
C	Q IS NOW STACKED ROWWISE (AS IS G)	INV.33
C	INVERT G-MATRIX	INV.34
C		INV.35
66	CALL CINV(LMAX3)	INV.36
C		INV.37
C	OUTPUT IN ARRAY Q IS STACKED COLUMNWISE	INV.38
C	CALCULATE TAL-ARRAY ... TAU=Q*BBB	INV.39
C		INV.40
70	INDEX=C	INV.41
71	DC 110 I=1,LMAX3	INV.42
73	DC 110 J=1,LMAX3	INV.43
110	INDEX=INDEX+2	INV.44
112	TERM=CMPLX(C(INDEX-1),Q(INDEX))	INV.45
120	TAL(J,I,KK)=BBB(J,KK)*TERM	INV.46
125	110 CCNTINLE	INV.47
132	I=C	INV.48
133	CALL TEST(I,KK)	INV.49
134	112 CCNTINLE	INV.50
C		INV.51
C		INV.52
137	RETURN	INV.53
137	END	INV.54

	SUBROUTINE CINV(NCM)	INV.55
C		INV.56
C	COMPLEX MATRIX INVERSION ROUTINE.	INV.57
C	TOTAL LARGE CORE LENGTH IS(5*NDIM*NDIM). DIMENSION IPIVCT(NDIM).	INV.58
C	THE COMPLEX MATRIX C(NDIM,NDIM) OCCUPIES THE FIRST (2*NCM*NDIM)	INV.59
C	LOCATIONS IN EXTENDED CORE. THE INVERTED MATRIX IS DELIVERED	INV.60
C	IN THE SAME LOCATION. THE REMAINING LENGTH REQUIRED IN	INV.61
C	EXTENDED CORE IS FOR INTERMEDIATE WORKING ARRAYS.	INV.62
C		INV.63
	LARGE C(49000),RVEC(30000),C.C(49000)	LARGE.2
	LARGE C(20000)	INV.65
	COMMON/MTX/NDIM,NSC,N1,N2,N3,N4,N5,IPIVOT(50)	INV.66
	INTEGER PLUS,TIMES	INV.67
	DATA PLUS,TIMES/1,2/	INV.68
C		DIM.2
C	DIMENSIONS OF KX,C,AND LL.	DIM.3
	NM3=2401 \$ NM1=7532 \$ NM2=7532	DIM.4
C	DIMENSIONS OF RMAG AND XMAG	DIM.5
5	NM4=9000 \$ NM5=9000	DIM.6
C	DIMENSIONS OF RVEC AND G-ARRAY IN LARGE CORE MEMORY.	DIM.7
6	NM6=30000 \$ NM7=49000	DIM.8
C		DIM.9
C		INV.70
12	NDIM=NCM	INV.71
12	NSC=NDIM*NDIM	INV.72
14	N1=C	INV.73
14	N2=NSC	INV.74
15	N3=N2+NSC	INV.75
16	N4=N3+NSC	INV.76
17	N5=N4+NSC	INV.77
C		INV.78
C	STORE REAL AND IMAGINARY PARTS OF (Q) IN BLOCKS 3 AND 4.	INV.79
C		INV.80
21	DO 100 I=1,NSC	INV.81
27	J2=2*I	INV.82
30	J1=J2-1	INV.83
31	Q(N3+I)=C(J1)	INV.84
34	Q(N4+I)=C(J2)	INV.85
36	100 CONTINUE	INV.86
C		INV.87
C	PERFORM INVERSION ON BLOCKS 3 AND 4.	INV.88
C	AT FINAL STEP, BLOCKS 3 AND 4 ARE THE REAL AND IMAGINARY PARTS	INV.89
C	OF THE INVERTED COMPLEX MATRIX (Q).	INV.90
C		INV.91
C		INV.92
40	CALL EXCH(N3,N3)	INV.93
42	CALL EXCH(N4,N4)	INV.94
44	CALL EXCH(N2,N3)	INV.95
46	CALL INVERT(N2)	INV.96
50	CALL ARITH(N2,TIMES,N4,N1,+1.C)	INV.97
54	CALL EXCH(N4,N4)	INV.98
56	CALL ARITH(N4,TIMES,N1,N2,+1.C)	INV.99
62	CALL ARITH(N3,PLUS,N2,N3)	INV.100
65	CALL INVERT(N3)	INV.101
67	CALL EXCH(N1,N1)	INV.102
71	CALL ARITH(N1,TIMES,N3,N4,-1.C)	INV.103
C		INV.104
C		INV.105
	STORE INVERTED COMPLEX ARRAY COLUMNWISE IN (C).	INV.106
C		INV.107
75	DO 110 I=1,NSC	INV.108
105	J2=2*I	INV.109
106	J1=J2-1	INV.110
107	Q(J1)=C(N3+I)	INV.111
112	Q(J2)=C(N4+I)	INV.112
114	110 CONTINUE	INV.113
C		INV.114
C		INV.115
116	RETURN	INV.116
116	END	INV.117
		INV.118

	SLPROTINE INVERT(MM)	INV.119
C		INV.120
C	REAL MATRIX INVERSION IS PERFORMED ON THE MATRIX STORED IN	INV.121
C	EXTENDED CORE BLOCK BEGINNING AT LOCATION (MM+1).	INV.122
C	INVERTED MATRIX IS RETURNED IN SAME LOCATION.	INV.123
C		INV.124
	LARGE C(4000),RVEC(30000),C.G(49000)	LARGE.2
	LARGE C(20000)	INV.125
	COMMON/NTRX/NDIM,NSQ,N1,N2,N3,N4,N5,PIVOT(50)	INV.127
C		DIM.2
C	DIMENSIONS OF KX,C,AND LL.	DIM.3
	NM3=2401 \$ NM1=7532 \$ NM2=7532	DIM.4
C	DIMENSIONS OF RMAG AND XMAG	DIM.5
5	NM4=9000 \$ NM5=9000	DIM.6
C	DIMENSIONS OF RVEC AND G-ARRAY IN LARGE CORE MEMORY.	DIM.7
6	NM6=30000 \$ NM7=49000	DIM.8
C		DIM.9
C		INV.125
11	SMALL=1.E-15	INV.130
12	KK=0	INV.131
C		INV.132
C	INITIALIZE THE UNIT MATRIX STARTING AT LOCATION (N5+1)	INV.133
C		INV.134
13	DO 100 I=1,NDIM	INV.135
14	DO 100 J=1,NDIM	INV.136
20	KK=KK+1	INV.137
21	G(N5+KK)=C.C	INV.138
24	IF(I.EQ.J) G(N5+KK)=1.0	INV.139
30	100 CONTINUE	INV.140
C		INV.141
C	PERFORM INVERSION USING GAUSS-JORDAN METHOD.	INV.142
C		INV.143
34	JJ=0	INV.144
34	KK=0	INV.145
35	DO 160 I=1,NDIM	INV.146
37	PIVOT=C.0	INV.147
C		INV.148
40	DO 110 J=1,NDIM	INV.149
45	JJ=JJ+1	INV.150
46	IF(ABS(C(MM+JJ)).LE.PIVOT) GO TO 110	INV.151
53	PIV=C(MM+JJ)	INV.152
55	PIVCT=ABS(PIV)	INV.153
56	IPIV=J	INV.154
57	110 CONTINUE	INV.155
60	IF(PIVCT.LE.SMALL) GO TO 200	INV.156
63	PIVCT=1./PIV	INV.157
C		INV.158
64	DO 120 J=1,NDIM	INV.159
72	KK=KK+1	INV.160
73	M1=MM+KK	INV.161
74	M2=N5+KK	INV.162
75	C(M1)=C(M1)*PIVCT	INV.163
102	C(M2)=C(M2)*PIVCT	INV.164
104	120 CONTINUE	INV.165
C		INV.166
106	MM=JJ-NDIM	INV.167
107	LL=C	INV.168
C		INV.169

110	DC 150 K=1,NCIM	INV.170
112	IF(K.NE.1) GC TO 130	INV.171
114	LL=LL+NCIM	INV.172
115	GC TO 150	INV.173
C		INV.174
116	130 CONTINUE	INV.175
116	AAA=C(MM+LL+IPIV)	INV.176
C		INV.177
122	DC 140 J=1,NCIM	INV.178
134	LL=LL+1	INV.179
135	M1=MM+LL	INV.180
136	M2=N5+LL	INV.181
137	Q(M1)=C(M1)-AAA*C(MM+PN+J)	INV.182
145	Q(M2)=C(M2)-AAA*Q(N5+PN+J)	INV.183
151	140 CONTINUE	INV.184
154	150 CONTINUE	INV.185
157	IFIVCT(I)=IPIV	INV.186
161	160 CONTINUE	INV.187
C		INV.188
C	RESTACK BLOCK 5 IN BLOCK MM.	INV.189
C		INV.190
163	L=C	INV.191
164	DC 170 I=1,NCIM	INV.192
165	K=(IFIVCT(I)-1)*NCIM	INV.193
171	DC 170 J=1,NCIM	INV.194
174	L=L+1	INV.195
175	K=K+1	INV.196
176	Q(MM+K)=Q(N5+L)	INV.197
202	170 CONTINUE	INV.198
C		INV.199
206	RETURN	INV.200
C		INV.201
C		INV.202
207	200 CONTINUE	INV.203
207	PRINT ECO,I	INV.204
216	800 FORMAT(// 10X*DETERMINANT=0	INV.205
	STOP	INV.206
C		INV.207
220	END	INV.208
		INV.209

	SLBRCLTINE EXCH(M1,M2)	INV.210
C		INV.211
C	EXCHANGES STORAGE FROM COLUMNWISE TO ROWWISE AND VICE-VERSA.	INV.212
C		INV.213
	LARGE C.QM(1296),RVEC(30000),C.C(49000)	LARGE.2
	LARGE C(20000)	INV.215
	COMMON/MTRX/NCIM,NSQ,N1,N2,N3,N4,N5,IPIVOT(50)	INV.216
C		DIM.2
C	DIMENSIONS OF KX,C,AND LL.	DIM.3
	NM3=2401 \$ NM1=7532 \$ NM2=7532	DIM.4
C	DIMENSIONS OF RMAG AND XMAG	DIM.5
6	NM4=9000 \$ NM5=9000	DIM.6
C	DIMENSIONS OF RVEC AND G-ARRAY IN LARGE CORE MEMORY.	DIM.7
10	NM6=30000 \$ NM7=49000	DIM.8
C		DIM.9
C		INV.218
12	L=M2	INV.219
13	DO 100 I=1,NCIM	INV.220
15	K=I-NCIM	INV.221
16	DO 100 J=1,NCIM	INV.222
23	K=K+NCIM	INV.223
24	L=L+1	INV.224
26	Q(N5+K)=Q(L)	INV.225
32	100 CONTINUE	INV.226
C		INV.227
36	DO 110 I=1,NSQ	INV.228
44	Q(M1+I)=Q(N5+I)	INV.229
45	110 CONTINUE	INV.230
C		INV.231
C		INV.232
50	RETURN	INV.233
50	END	INV.234
		INV.235

C	SLBPOLINE ARITH(M1,OP,M2,M3,SIGN)	INV.236
C	PERFORMS A*CP*B=C, WHERE CP=(PLUS,TIMES).	INV.237
C	FOR OP=TIMES, (A) IS ROWWISE, (B) IS COLUMNWISE, AND	INV.238
C	(C) IS DELIVERED AS A COLUMNWISE ARRAY.	INV.239
C		INV.240
	LARGE C.QM(1296),RVEC(30000),C.G(49CCC)	INV.241
	LARGE C(20000)	LANGE.2
	COMMON/MTRX/NDIM,NSC,N1,N2,N3,N4,N5,IPIVOT(50)	INV.242
	INTEGER CP	INV.243
	REAL IFIVCT	INV.244
C		INV.245
C	DIMENSIONS OF KX,C,AND LL.	DIM.2
C	NM3=2401 \$ NM1=7532 \$ NM2=7532	DIM.3
C	DIMENSIONS OF RMAG AND XMAG	DIM.4
11	NM4=9CCC \$ NM5=9CCC	DIM.5
C	DIMENSIONS OF RVEC AND G-ARRAY IN LARGE CORE MEMORY.	DIM.6
13	NM6=3CCC \$ NM7=4CCC	DIM.7
C		DIM.8
C		DIM.9
15	IF(CP.EQ.2) GO TO 110	INV.248
17	IF(CP.NE.1) STOP	INV.249
C		INV.250
C	MATRIX ADDITION PACKAGE.	INV.251
C		INV.252
22	DO 100 I=1,NSC	INV.253
32	Q(M2+I)=Q(M1+I)+Q(M2+I)	INV.254
35	100 CONTINUE	INV.255
37	RETURN	INV.256
C		INV.257
C	MATRIX MULTIPLICATION PACKAGE.	INV.258
C		INV.259
37	110 CONTINUE	INV.260
37	II=-NDIM	INV.261
41	DO 130 I=1,NDIM	INV.262
42	II=II+NDIM	INV.263
43	SMALLIN(IPIVCT,Q(M2+II+1),NDIM)	INV.264
52	JJ=-NDIM	INV.265
C		INV.266
53	DO 130 J=1,NDIM	INV.267
54	JJ=JJ+NDIM	INV.268
55	II=II+J	INV.269
56	X=C.C	INV.270
C		INV.271
60	DO 120 K=1,NDIM	INV.272
65	I3=JJ+K	INV.273
66	X=X+IPIVCT(K)*C(M1+I3)	INV.274
73	120 CONTINUE	INV.275
74	C(M2+II)=SIGN*X	INV.276
102	130 CONTINUE	INV.277
C		INV.278
C		INV.279
107	RETURN	INV.280
107	END	INV.281
		INV.282

SLBPROGRAM LENGTH

	SLEROUTINE TEST(ITER, KK)	TEST.2
C		TEST.3
C	TEST AND PRINT SUPPED TAU-VALUES.	TEST.4
C		TEST.5
	COMMON/BLCKKC/V1(3), V2(3,12), V3(3,12), VX, VY, VZ, SOK(12), CCK(12),	C.2
2	YIN(45), YCLT(45,12), XKZERO, AREA2	C.3
	CCOMPLEX YIN, YCUT	C.4
	COMMON/BLCKKE/LPAX, LTCP, LPAX1, LPAX2, LPAX3, LPAX4, LX	E.2
2	, ICNE, ITWC, ITHREE	E.3
	COMMON/BLCKKF/ITHT, IPF1, THTT, DIST, CSTH(12), VXX(2,12), KNUF, JM(12),	F.2
2	JN(12), IQ(12), KC(12)	F.3
	COMMON/BLCKKK/BBB(36,2), TAU(36,36,2), TT(10,12), KTYF(10)	K.2
	CCOMPLEX EBE, TAL, TT	K.3
	COMMON/CONST/PI, PI4, PI42, CUN1, CCN2, CCN3, CCN4, CCNSTT, IP, CNST	CONST.2
	CCOMPLEX IP	CCNST.3
	CCOMPLEX TERM, PP, P(12)	TEST.11
	DIMENSION PX(12)	TEST.12
C		TEST.13
C		TEST.14
	DO 100 I=1, KNUF	TEST.15
11	P(I)=(C.0,0.0)	TEST.16
13	100 CONTINUE	TEST.17
C		TEST.18
14	DO 110 LIN=1, LPAX3	TEST.19
17	DO 110 LCUT=1, LPAX3	TEST.20
20	TERM=TAU(LCUT, LIN, KK)*YIN(LIN)	TEST.21
33	DO 110 J=1, KNUF	TEST.22
45	P(J)=P(J)+TERM*YCLT(LCUT, J)	TEST.23
55	110 CONTINUE	TEST.24
C		TEST.25
64	DO 120 I=1, KNUF	TEST.26
65	PX(I)=CNST*CAPS(P(I))**2	TEST.27
74	120 CONTINUE	TEST.28
C		TEST.29
76	PRINT 800, ITER	TEST.30
103	PRINT #10, (KK, IQ(I), JM(I), JN(I), PX(I), I=1, KNUF)	TEST.31
C		TEST.32
C		TEST.33
	800 FORMAT(/ 10X*NUMBER OF ITERATIONS=, I4)	TEST.34
	810 FORMAT(2CX*CCRE(*12,*) SYP(*12,*) TAU(*12,*,* 12, *)**	TEST.35
	2 E16.8)	TEST.36
C		TEST.37
136	RETURN	TEST.38
136	END	TEST.39

	SLBRCLTINE PERTB	PERT.2
C	PERFORMS PERTURBATION EXPANSION TO CERTAIN TAU.	PERT.3
C		PERT.4
C	COMMON/BLCKCK/V1(3),V2(3,12),V3(3,12),VX,VY,VZ,SQK(12),CCK(12),	PERT.5
2	YIN(49),YCLT(49,12),XKZ=RU,AREA2	C.2
	COMPLEX YIN,YCLT	C.3
	COMMON/BLCKCK/LMAX,LTCP,LMAX1,LMAX2,LMAX3,LMAX4,LX	C.4
2	,ICNE,ITWC,ITHREE	E.2
	COMMON/BLCKCK/BBB(36,2),TAU(36,36,2),TT(10,12),KTYPI(10)	E.3
	COMPLEX BBB,TAL,TT	K.2
	COMMON/BLCKCKX/TIME,ITEM,MULT,ISCALE,IMAX,ITYPE,LSUP,ILCLP,IPWR	K.3
	INTEGER TIME	X.2
	COMMON/CONST/PI,F14,P142,CON1,CON2,CON3,CON4,CCNSTT,IM,CAST	X.3
	COMPLEX IM	CONST.2
	COMPLEX P,PP	CCNST.3
	DIMENSION IL(20)	PERT.11
	LARGE C,JM(1296),RVEC(30000),C.C(49000)	PERT.12
C		LARGE.2
C	DIMENSIONS OF KX,C,AND LL.	DIM.2
	NM3=2401 \$ NM1=7532 \$ NM2=7532	DIM.3
C	DIMENSIONS OF RMAG AND XMAG	DIM.4
4	NM4=9000 \$ NM5=9000	DIM.5
C	DIMENSIONS OF RVEC AND G-ARRAY IN LARGE CORE MEMORY.	DIM.6
5	NM6=30000 \$ NM7=49000	DIM.7
C		DIM.8
C		DIM.9
10	IPWR=20	PERT.15
10	SMALL=C.0001	PERT.16
C		PERT.17
12	DC 17C KK=1,ITYPE	PERT.18
14	JK=LMAX4	PERT.19
15	KL=0	PERT.20
C		PERT.21
16	DC 11C J=1,LMAX3	PERT.22
17	PP=BRP(J,KK)	PERT.23
24	DC 11C K=1,LMAX3	PERT.24
31	JK=JK+1	PERT.25
32	KL=KL+1	PERT.26
33	G(JK)=PP*G(KL)	PERT.27
44	110 CCNTINUE	PERT.28
C		PERT.29
47	DC 12C I=1,IPWR	PERT.30
54	IL(1)=(I+1)*LMAX4+1	PERT.31
56	120 CCNTINUE	PERT.32
C		PERT.33
57	CALL EXCX(IL(1),LMAX4+1)	PERT.34
C		PERT.35
63	DC 13C I=2,IPWR	PERT.36
65	CALL ARITX(LMAX4+1,2,IL(I-1),IL(I),2,C)	PERT.37
73	130 CCNTINUE	PERT.38
C		PERT.39
C	STACK UNIT MATRIX IN G(2).	PERT.40
C		PERT.41
76	JK=LMAX4	PERT.42
77	DC 14C I=1,LMAX3	PERT.43
101	DC 140 J=1,LMAX3	PERT.44
111	JK=JK+1	PERT.45
		PERT.46

112	G(JK)=(0.C,0.0)	PERT.47
116	IF(I.EQ.J) G(JK)=C*PLX(1.0,0.0)	PERT.48
123	140 CCNTINUE	PERT.49
C		PERT.50
C	PERFORM MATRIX MULTIPLICATIONS UP TO (G*T)**IPWR.	PERT.51
C		PERT.52
127	IF(KK.EQ.1) PRINT 800	PERT.53
134	DC 150 I=1,IPWR	PERT.54
136	CALL ARITX(LMAX4+1,1,IL(I),LMAX4+1,0,C)	PERT.55
145	JK=LMAX4	PERT.56
147	DC 145 J=1,LMAX2	PERT.57
150	PP=BBB(J,KK)	PERT.58
155	DC 145 K=1,LMAX3	PERT.59
167	JK=JK+1	PERT.60
170	TAL(K,J,KK)=G(JK)*PP	PERT.61
177	145 CCNTINUE	PERT.62
C		PERT.63
C	TEST FOR CONVERGENCE	PERT.64
C		PERT.65
203	P=(0.C,C.C)	PERT.66
205	DC 146 LIN=1,LMAX3	PERT.67
207	DC 146 LCUT=1,LMAX3	PERT.68
222	P=P+TAL(LCUT,LIN,KK)*YIN(LIN)*YCUT(LCUT,1)	PERT.69
235	146 CCNTINUE	PERT.70
241	PX=CONST*ABS(P)**2	PERT.71
244	IF(I.LE.-) GC TO 148	PERT.72
C		PERT.73
247	IF(ABS(PX-STORE).LE.SMALL) GC TO 160	PERT.74
253	148 CCNTINUE	PERT.75
253	STORE=PX	PERT.76
255	150 CCNTINUE	PERT.77
257	160 CCNTINUE	PERT.78
C		PERT.79
257	CALL TEST(I,KK)	PERT.80
261	170 CCNTINUE	PERT.81
C		PERT.82
C		PERT.83
264	800 FORMAT(/ 10X*OBTAIN SUBPLANE PREPAGATOR BY ITERATION*)	PERT.84
264	RETURN	PERT.85
264	END	PERT.86

	SUBROUTINE EXCX(M1,M2)	EXCH.2
C		EXCH.3
C	EXCHANGES STORAGE FROM COLUMNWISE TO ROWWISE AND VICE-VERSA	EXCH.4
C	REPLACES MATRIX STARTING AT G(M2) BY ONE STARTING AT G(M1).	EXCH.5
C		EXCH.6
	COMMON/BLOCKE/LMAX,LTUP,LMAX1,LMAX2,LMAX3,LMAX4,LX	E.2
2	,ICNE,ITWC,ITHEE	E.3
	LARGE C.QM(1256),RVEC(30000),C.G(49000)	LARGE.2
C		DIM.2
C	DIMENSIONS OF KX,C,AND LL.	DIM.3
	NM3=2401 \$ NM1=7532 \$ NM2=7532	DIM.4
C	DIMENSIONS OF RMAG AND XMAG	DIM.5
•6	NM4=9000 \$ NM5=9000	DIM.6
C	DIMENSIONS OF RVEC AND G-ARRAY IN LARGE CORE MEMORY.	DIM.7
10	NM6=30000 \$ NM7=49000	DIM.8
C		DIM.9
C		EXCH.10
12	LG=M2-1	EXCH.11
14	DO 100 I=1,LMAX2	EXCH.12
15	K=I-LMAX3	EXCH.13
16	DO 100 J=1,LMAX3	EXCH.14
23	K=K+LMAX3	EXCH.15
24	LG=LG+1	EXCH.16
26	GM(K)=G(LG)	EXCH.17
33	100 CONTINUE	EXCH.18
C		EXCH.19
36	DO 110 I=1,LMAX4	EXCH.20
44	G(M1+I-1)=GM(I)	EXCH.21
47	110 CONTINUE	EXCH.22
C		EXCH.23
C		EXCH.24
51	RETURN	EXCH.25
51	END	EXCH.26

```

SUBROUTINE ARITH(M1,OP,M2,M3,CUT,ACC)
C
C   MATRIX ADDITION AND MULTIPLICATION ROUTINE.
C   CP=1 FOR ADDITION. LP=2 FOR MULTIPLICATION.
C   G(M1) MUST BE HORIZONTAL AND G(M2) MUST BE VERTICAL.
C   IF CUT=1, OUTPUT IS HORIZONTAL. IF CUT=2, OUTPUT IS VERTICAL.
C   ACC=0 FOR G(M3) TO BE A SIMPLE PRODUCT.
C   IF ACC=1, G(M3) IS A CUMULATIVE PRODUCT.
C
COMMON/BLCKE/LMAX,LTOP,LMAX1,LMAX2,LMAX3,LMAX4,LX
2  ,ICKE,ITWC,ITREE
INTEGER OP,CLT,ACC
COMPLEX X,CR(45)
LARGE C,C2(1296),RVEC(30000),C,C(49000)
C
C   DIMENSIONS OF KX,C,AND LL.
C   NM3=2401 & NM1=7532 & NM2=7532
C   DIMENSIONS OF RMAG AND XMAG
C   NM4=9000 & NM5=9000
12  C   DIMENSIONS OF RVEC AND G-ARRAY IN LARGE CORE MEMORY.
14  C   NM6=30000 & NM7=49000
C
C
16  LMAX5=2*LMAX3
17  J1=M1-1
21  J2=M2-1
22  J3=M3-1
23  N1=M1
24  N2=M2
24  N3=M3
25  IF(OP.EQ.2) GO TO 110
27  IF(OP.NE.1) STOP
C
C   MATRIX ADDITION PACKAGE.
C
32  DO 100 I=1,LMAX4
45  G(J3+I)=G(J1+I)+G(J2+I)
53  100 CONTINUE
54  RETURN
C
C   MATRIX MULTIPLICATION PACKAGE
C
55  110 CONTINUE
C
55  IF(CUT.EQ.2) GO TO 115
57  N2=N1
60  J1=J2
61  IF(CUT.NE.1) STOP
65  115 CONTINUE
C
65  I1=-LMAX3
67  DO 120 I=1,LMAX3
70  I1=I1+LMAX3
71  SMALLIN(CP,G(N2+I1),LMAX5)
100  JJ=-LMAX3
C
101  DO 130 J=1,LMAX3
102  JJ=JJ+LMAX3
C
C
103  I1=I1+J
104  X=(C,C,C,0)
C
107  DO 120 K=1,LMAX3
116  I2=JJ+K
117  X=X+CR(K)*G(J1+I2)
131  120 CONTINUE
132  IF(ACC.EQ.C) G(J3+I1)=X
145  IF(ACC.EQ.1) G(J3+I1)=G(J3+I1)+X
157  130 CONTINUE
C
C
144  RETURN
144  END

```

ARITH.2
 ARITH.3
 ARITH.4
 ARITH.5
 ARITH.6
 ARITH.7
 ARITH.8
 ARITH.9
 ARITH.10
 E.2
 F.3
 ARITH.12
 ARITH.13
 LARGE.3
 DIM.2
 DIM.3
 DIM.4
 DIM.5
 DIM.6
 DIM.7
 DIM.8
 DIM.9
 ARITH.16
 ARITH.17
 ARITH.18
 ARITH.19
 ARITH.20
 ARITH.21
 ARITH.22
 ARITH.23
 ARITH.24
 ARITH.25
 ARITH.26
 ARITH.27
 ARITH.28
 ARITH.29
 ARITH.30
 ARITH.31
 ARITH.32
 ARITH.33
 ARITH.34
 ARITH.35
 ARITH.36
 ARITH.37
 ARITH.38
 ARITH.39
 ARITH.40
 ARITH.41
 ARITH.42
 ARITH.43
 ARITH.44
 ARITH.45
 ARITH.46
 ARITH.47
 ARITH.48
 ARITH.49
 ARITH.50
 ARITH.51

 ARITH.52
 ARITH.53
 ARITH.54
 ARITH.55
 ARITH.56
 ARITH.57
 ARITH.58
 ARITH.59
 ARITH.60
 ARITH.61
 ARITH.62
 ARITH.63
 ARITH.64
 ARITH.65

```

SUBROUTINE GREEN(INDEX,IBEGIN)
C
C CCMON/PUFF/LCG,NRC,KKK
C CCMON/BLCKC/NLCW,NNUM,NDELTA,ARG,NRGY,LAYER,NUMMAX,NCIF,IST(6,6)
2  ,LR
C CCMON/BLCKE/LMAX,LTUP,LMAX1,LMAX2,LMAX3,LMAX4,LX
2  ,ICNE,ITWC,ITHREE
C CCMON/BLCKH/KX(2401),C(7532),LL(7532),YINT(169)
C CPLEX YINT
C CCMON/BLCKJ/MAXX(51),RMAG( 9000),XMAG( 9000),MAXR
C CCMON/BLCKK/BRB(36,2),TAU(36,36,2),TT(10,12),KTYP(10)
C CPLEX REB,TAL,TT
C CCMON/BLCKL/ARG,H(169),XD(9),FATH1(10),PATH2(10,12)
C CPLEX ARG,H
C CCMON/BLCKM/RATIO,XCCN,KSTAR
C CPLEX KSTAR,XCCN
C CCMON/BLCKN/PX,PY,PZ
C CPLEX XX,TERM
LARGE CQM(1296),RVEC(30000),C.G(49000)
C
C DIMENSIONS OF KX,C,AND LL.
NM3=2401 $ NM1=7532 $ NM2=7532
C DIMENSIONS OF RMAG AND XMAG
NM4=9000 $ NM5=9000
C DIMENSIONS OF RVEC AND G-ARRAY IN LARGE CORE MEMORY.
NM6=30000 $ NM7=49000
C
C
12 IC=IBEGIN
13 IF(INDEX.EQ.1) ID=0
16 DO 50 I=1,LMAX4
23 IV=ID+I
24 G(IV)=(C.0,C.0)
32 50 CONTINUE
C
33 MIN=1
34 IF(INDEX.GT.1) MIN=MAXX(INDEX-1)+1
37 MAX=MAXX(INDEX)
40 IF(MIN.GT.MAX) GO TO 130
44 LCCC=(MIN-1)*2-2
46 LINT=LMAX2-1
C
50 DO 120 LCC=MIN,MAX
51 LCCC=LCCC+3
52 R1=RVEC(LCCC)
54 R2=RVEC(LCCC+1)
55 R3=RVEC(LCCC+2)
57 CALL YLM(R1,R2,R3,YINT,LINT,-1)
C
63 ARG=KSTAR*RMAG(LCC)
70 XX=XCCN*CFXP( CMPLX(0.0,XMAG(LCC)*RATIO) )
C
105 CALL FANKFL
C
106 INCX=IC
107 NCX=0

```

GREEN.2
 GREEN.3
 GREEN.4
 D.2
 D.3
 E.2
 E.3
 H.2
 H.3
 J.2
 K.2
 K.3
 L.2
 L.3
 M.2
 M.3
 Q.2
 GREEN.14
 LARGE.2
 DIM.2
 DIM.3
 DIM.4
 DIM.5
 DIM.6
 DIM.7
 DIM.8
 DIM.9
 GREEN.17
 GREEN.18
 GREEN.19
 GREEN.20
 GREEN.21
 GREEN.22
 GREEN.23
 GREEN.24
 GREEN.25
 GREEN.26
 GREEN.27
 GREEN.28
 GREEN.29
 GREEN.30
 GREEN.31
 GREEN.32
 GREEN.33
 GREEN.34
 GREEN.35
 GREEN.36
 GREEN.37
 GREEN.38
 GREEN.39
 GREEN.40
 GREEN.41
 GREEN.42
 GREEN.43
 GREEN.44
 GREEN.45
 GREEN.46
 GREEN.47

110	DO 11C L=1,LMAX4	GREEN.48
114	JJ=KX(L)	GREEN.49
115	TERM=(C.C,0.C)	GREEN.50
120	INCX=INCX+1	GREEN.51
	C	GREEN.52
121	CC 10C I=1,JJ	GREEN.53
126	NCX=NCX+1	GREEN.54
127	LLL=LL(NCX)	GREEN.55
	C	GREEN.56
131	TERM=TERM+YINT(LLL)*H(LLL)*C(NCX)	GREEN.57
146	10C CONTINUE	GREEN.58
147	G(INCX)=G(INCX)+TERM*XX	GREEN.59
161	11C CONTINUE	GREEN.60
163	IF(INCX.GT.NUMMAX) PRINT 800, NCX	GREEN.61
175	120 CONTINUE	GREEN.62
200	13C CONTINUE	GREEN.63
200	RETURN	GREEN.64
	C	GREEN.65
	800 FORMAT(20X 44P***** INDEX OF ARRAY EXCEEDS NUMMAX (	GREEN.66
	2 15, 13H) *****)	GREEN.67
	81C FORMAT(// 10X 10H***** * END OF FILE TAPE 2 INDEX=* 14,	GREEN.68
	2 * LCC=* 15)	GREEN.69
	820 FORMAT(// 10X 10H***** * PARITY ERROR TAPE 2 INDEX=* 14,	GREEN.70
	2 * LCC=* 15)	GREEN.71
	C	GREEN.72
	C	GREEN.73
201	END	GREEN.74

	SUBROUTINE HANKEL	HANKEL.2
C	COMMON/BLCKCF/LMAX, LTOP, LMAX1, LMAX2, LMAX3, LMAX4, LX	HANKEL.3
2	ICNE, ITHC, ITHREE	E.2
	COMMON/BLCKCL/ARG, F(169), XD(9), PATH1(10), PATH2(10,12)	E.2
	CCPLEX ARG, H	L.2
	COMMON/CGAST/PI, P14, P142, CUN1, CCN2, CCN3, CON4, CONSTT, IM, CAST	L.2
	CCPLEX IM	CONST.2
	CCPLEX HF(31), IX, FFF, XPON, ZINV	CCNST.3
C		HANKEL.7
C		HANKEL.8
	MAX=LMAX2-2	HANKEL.9
3	XPON=CEXP(IM*ARG)	HANKEL.10
15	ZINV=1./ARG	HANKEL.11
C		HANKEL.12
23	HH(1)=-IM*ZINV*XPON	HANKEL.13
34	HF(2)=-ZINV*(1.+IM*ZINV)*XPON	HANKEL.14
C		HANKEL.15
55	DO 100 I=1, MAX	HANKEL.16
64	HF(1+2)=(I+1+1)*ZINV*HH(I+1)-HH(I)	HANKEL.17
100	100 CONTINUE	HANKEL.18
C		HANKEL.19
101	K=C	HANKEL.20
102	IX=IM	HANKEL.21
C		HANKEL.22
105	DO 110 I=1, LMAX2	HANKEL.23
106	NPI=2*I-1	HANKEL.24
107	IX=-IX*IM	HANKEL.25
116	FFF=IX*HF(I)	HANKEL.26
C		HANKEL.27
125	DO 130 J=1, NBR	HANKEL.28
126	N=N+1	HANKEL.29
127	H(N)=FFF	HANKEL.30
130	130 CONTINUE	HANKEL.31
C		HANKEL.32
141	RETURN	HANKEL.33
14	END	HANKEL.34
		HANKEL.35
		HANKEL.36

***** DATA DECK LISTING *****

PUNCH=00

7 DECEMBER 1972

ALUMINUM (PERTURBATION EXPANSION) PENDRY PHASE SHIFTS *****

5 30 1 MULT-MAXR-LSUM-TIME

SIGMA=00

5 NDIF

1 ITYPE=NUMBER OF ATOMIC TYPES CONSIDERED

3 LMAX

ALUMINUM *****

26.9815 ATOMIC WEIGHT OF ALUMINUM

6 40 71 LTOP,LOWEN AND NNRG

5.8536364E-01	-5.98783025E-01	5.69197909E-01	-6.21556115E-01	5.51888103E-01
-6.42409151E-01	6.33666305E-01	-6.61365386E-01	5.14715739E-01	-6.78462968E-01
4.95321422E-01	-6.93752629E-01	4.75726765E-01	-7.06972431E-01	4.55914424E-01
-7.18610708E-01	4.36000773E-01	-7.28899886E-01	4.16091926E-01	-7.37837210E-01
3.96284099E-01	-7.45483758E-01	3.76834837E-01	-7.51674909E-01	3.57577060E-01
-7.56837265E-01	7.38567087E-01	-7.61020432E-01	3.19865652E-01	-7.64275189E-01
3.01488133E-01	-7.66653055E-01	2.83675748E-01	-7.68859745E-01	2.66202636E-01
-7.68779901E-01	2.49093718E-01	-7.68851214E-01	2.3237142E-01	-7.68311396E-01
2.16853903E-01	-7.67197993E-01	2.00243446E-01	-7.65520786E-01	1.84828090E-01
-7.6333314E-01	1.69818442E-01	-7.60725536E-01	1.55223344E-01	-7.57708025E-01
1.41049953E-01	-7.54310137E-01	1.27424002E-01	-7.50578809E-01	1.14169335E-01
-7.46431859E-01	1.01288386E-01	-7.42070043E-01	8.87857016E-02	-7.37453733E-01
7.66629788E-02	-7.32602872E-01	6.49706038E-02	-7.2754477E-01	5.36269647E-02
-7.22959004E-01	4.26209167E-02	-7.16822637E-01	3.19518185E-02	-7.11229621E-01
2.16145571E-02	-7.05494252E-01	1.16605716E-02	-6.99658753E-01	2.00875512E-03
-6.93651318E-01	-7.34366784E-03	-6.87572898E-01	-1.63997563E-02	-6.81404171E-01
-2.51428368E-02	-4.75155533E-01	-3.36012900E-02	-6.68840630E-01	-4.17723033E-02
-6.62465236E-01	-4.96793459E-02	-6.56037507E-01	-5.73240508E-02	-6.49565373E-01
-6.47162040E-02	-6.43056536E-01	-7.18268846E-02	-6.36554601E-01	-7.87029565E-02
-6.29968471E-01	-8.53480775E-02	-6.23394220E-01	-9.17659964E-02	-6.16807745E-01
-9.79405459E-02	-4.10214756E-01	-1.03917248E-01	-6.03657304E-01	-1.09668079E-01
-5.97041483E-01	-1.15216667E-01	-5.90461884E-01	-1.20566686E-01	-5.83892948E-01
-1.25721955E-01	-5.77338966E-01	-1.30672447E-01	-5.70870401E-01	-1.35443404E-01
-5.64302110E-01	-1.40037013E-01	-5.57917572E-01	-1.44457179E-01	-5.51360147E-01
-1.48707425E-01	-5.44933067E-01	-1.52780225E-01	-5.38546271E-01	-1.56696024E-01
-5.32192770E-01	-1.60458064E-01	-5.25875157E-01	-1.64049592E-01	-5.19595926E-01
-1.67533855E-01	-5.13357469E-01	-1.70854095E-01	-5.07142882E-01	-1.74033551E-01
-5.01011960E-01	-1.77075454E-01	-4.94909202E-01	-1.79983025E-01	-4.88855810E-01
-1.82759471E-01	-4.82853691E-01	0.	0.	0.
-1.23098214E-01	-9.14831520E-03	-8.31401068E-02	-6.08086767E-03	-6.42315045E-02
-3.70740220E-03	-4.62866446E-02	-1.96551081E-03	-2.92350701E-02	-7.99958752E-04
-1.23004349E-02	-1.61484916E-04	2.69561566E-03	-7.07404911E-06	1.75433186E-02
-3.05460278E-04	2.15886077E-02	-1.00925475E-03	4.48744175E-02	-2.07529984E-03
5.74475328E-02	-3.46415598E-03	6.93697163E-02	-5.14452710E-03	8.06305278E-02
-7.07659237E-03	9.12625321E-02	-9.22951614E-03	1.01296014E-01	-1.15737675E-02
1.10759262E-01	-1.40823892E-02	1.19611699E-01	-1.67113669E-02	1.27981043E-01
-1.94644002E-02	1.35888749E-01	-2.23223742E-02	1.43355081E-01	-2.52675485E-02
1.50399233E-01	-2.82834531E-02	1.57031461E-01	-3.13574814E-02	1.63281311E-01
-3.44622063E-02	1.69165060E-01	-3.76026553E-02	1.74608248E-01	-4.07607811E-02
1.79495738E-01	-4.39254015E-02	1.84816164E-01	-4.71113514E-02	1.89405124E-01
-5.02706026E-02	1.93677163E-01	-5.33962907E-02	1.97616740E-01	-5.64783407E-02
2.01325756E-01	-5.95084174E-02	2.04470181E-01	-6.23051303E-02	2.07487606E-01
-6.51181936E-02	2.10381562E-01	-6.79453896E-02	2.13145430E-01	-7.07845878E-02
2.15412454E-01	-7.36337424E-02	2.18431100E-01	-7.65570791E-02	2.20898881E-01
-7.94454353E-02	2.23221192E-01	-8.23139297E-02	2.25432951E-01	-8.51538587E-02
2.27450110E-01	-8.79626462E-02	2.29349353E-01	-9.07218009E-02	2.31132263E-01
-9.34531436E-02	2.32802846E-01	-9.61548372E-02	2.34344989E-01	-9.88251221E-02
2.35822465E-01	-1.01462346E-01	2.37167775E-01	-1.04022828E-01	2.38421607E-01

$t^l(e)$ for

$E=40,42,44,\dots$ eV.

Values are listed with
Real part followed
by Imaginary part
for each energy.

$l=0$

$l=1$

-1.06613223E-1	2.39587030E-01	-1.09142280E-01	2.40647631E-01	-1.11638827E-01
2.41464516E-01	-1.14101766E-01	2.42575142E-01	-1.16531113E-01	2.43412672E-01
-1.18909301E-01	2.44179532E-01	-1.21265486E-01	2.44878483E-01	-1.23588882E-01
2.45510630E-01	-1.25878752E-01	2.46074677E-01	-1.28127642E-01	2.46579696E-01
-1.30345052E-01	2.47027648E-01	-1.32533616E-01	2.47430445E-01	-1.34683210E-01
2.47759955E-01	-1.36802945E-01	2.48045146E-01	-1.38844517E-01	2.48282188E-01
-1.40934489E-01	2.48472708E-01	-1.42952488E-01	2.48478296E-01	-1.44938174E-01
2.48720507E-01	-1.46891234E-01	2.48780859E-01	-1.48871385E-01	2.48800835E-01
-1.50498369E-01	2.48781885E-01	-1.52551956E-01	2.48755423E-01	-1.54371941E-01
2.48432834E-01	-1.56158141E-01	0.	0.	
-5.50368616E-01	3.85711499E-01	-5.48328161E-01	-4.10641741E-01	-5.44099187E-01
-4.33299020E-01	-5.38119856E-01	-4.53597894E-01	-5.30781198E-01	-4.71568589E-01
-5.22429608E-01	-4.87249270E-01	-5.13330479E-01	-4.99224772E-01	-5.03895177E-01
-5.09422406E-01	-4.94255720E-01	-5.19087368E-01	-4.84555986E-01	-5.27068718E-01
-4.74808676E-01	-5.33818069E-01	-4.65323381E-01	-5.38516431E-01	-4.55985683E-01
-5.42536930E-01	-4.46825850E-01	-5.45935253E-01	-4.37870942E-01	-5.48727050E-01
-4.29144963E-01	-5.50937724E-01	-4.20778634E-01	-5.52194594E-01	-4.12635769E-01
-5.53130117E-01	-4.64721354E-01	-5.53755098E-01	-3.97039792E-01	-5.54083814E-01
-3.89594906E-01	-5.54124953E-01	-3.82487798E-01	-5.53650623E-01	-3.75567909E-01
-5.53038137E-01	-3.68853895E-01	-5.52291302E-01	-3.62316556E-01	-5.51413986E-01
-3.56013801E-01	-5.50410087E-01	-3.49910639E-01	-5.49225557E-01	-3.43985282E-01
-5.47952677E-01	-3.38235026E-01	-5.46593770E-01	-3.32657297E-01	-5.45151192E-01
-3.27249626E-01	-5.43627313E-01	-3.22033314E-01	-5.4191010E-01	-3.16970907E-01
-5.40280898E-01	-3.12059743E-01	-5.38528241E-01	-3.07297276E-01	-5.36724332E-01
-3.02481054E-01	-5.34870473E-01	-2.98228406E-01	-5.32935891E-01	-2.93907684E-01
-5.30970709E-01	-2.89716462E-01	-5.28375499E-01	-2.85652407E-01	-5.26950851E-01
-2.81713272E-01	-5.24897374E-01	-2.77915859E-01	-5.22787567E-01	-2.74229695E-01
-5.20464786E-01	-2.70652632E-01	-5.18529070E-01	-2.67182605E-01	-5.16380487E-01
-2.63817624E-01	-5.14219123E-01	-2.60573880E-01	-5.12020202E-01	-2.57422369E-01
-5.09421618E-01	-2.54361244E-01	-5.07623031E-01	-2.51388728E-01	-5.05424134E-01
-2.48503109E-01	-5.03224649E-01	-2.45718496E-01	-5.01003448E-01	-2.43010242E-01
-4.98791953E-01	-5.40375597E-01	-4.96589593E-01	-2.37873473E-01	-4.94395833E-01
-2.35322433E-01	-4.92210168E-01	-2.32914464E-01	-4.90015456E-01	-2.30568444E-01
-4.87836502E-01	-2.28282501E-01	-4.85672619E-01	-2.26055605E-01	-4.83523160E-01
-2.23886575E-01	-4.81387506E-01	-2.21784203E-01	-4.79253372E-01	-2.19732457E-01
-4.77137909E-01	-2.17730298E-01	-4.75040418E-01	-2.15776722E-01	-4.72960230E-01
-2.13870760E-01	-4.70896707E-01	-2.12011477E-01	-4.68849240E-01	-2.10197971E-01
-4.66917243E-01	-2.08429366E-01	-4.64900159E-01	-2.06744818E-01	-4.62797451E-01
-2.05023507E-01	-4.60808603E-01	0.	0.	
-1.04212514E-01	-9.34874912E-02	-1.13994733E-01	-1.1484453E-02	-1.23434929E-01
-1.38169786E-02	-1.32559472E-02	-1.63361340E-02	-1.41390434E-01	-1.90400633E-02
-1.49946336E-01	-2.19253783E-02	-1.58720331E-01	-2.51449793E-02	-1.66990184E-01
-2.84726032E-02	-1.74786300E-01	-3.18961485E-02	-1.82136114E-01	-3.54008043E-02
-1.49064528E-01	-1.89737193E-02	-1.95810877E-01	-4.27022175E-02	-2.02062656E-01
-4.64313754E-02	-2.07847770E-01	-5.01460158E-02	-2.13192191E-01	-5.38323250E-02
-2.18120165E-01	-5.74777373E-02	-2.22693541E-01	-6.10940611E-02	-2.26873400E-01
-6.46367384E-02	-2.30681289E-01	-6.80956637E-02	-2.34137490E-01	-7.14617432E-02
-2.37261112E-01	-7.47268173E-02	-2.40023050E-01	-7.78444209E-02	-2.42512993E-01
-8.08737745E-02	-2.44744338E-01	-8.37947095E-02	-2.46741511E-01	-8.66076235E-02
-2.48487008E-01	-8.93084402E-02	-2.49961304E-01	-9.18472002E-02	-2.51260374E-01
-9.42809127E-02	-2.52392917E-01	-9.66253342E-02	-2.53367188E-01	-9.88724962E-02
-2.54191031E-01	-1.01020687E-01	-2.54914288E-01	-1.03012750E-01	-2.55332226E-01
-1.04930791E-01	-2.55749697E-01	-1.06773985E-01	-2.56071331E-01	-1.08541621E-01
-2.56301546E-01	-1.10233105E-01	-2.56403706E-01	-1.11813944E-01	-2.56444299E-01
-1.13319732E-01	-2.56425938E-01	-1.14780197E-01	-2.56351131E-01	-1.16185121E-01
-2.56222287E-01	-1.17534332E-01	-2.56016126E-01	-1.18797430E-01	-2.55773958E-01
-1.20019786E-01	-2.55497195E-01	-1.21201322E-01	-2.55187201E-01	-1.22341985E-01
-2.54845296E-01	-1.23441745E-01	-2.54458134E-01	-1.24401784E-01	-2.54049259E-01
-1.25490371E-01	-2.53619472E-01	-1.26467494E-01	-2.53149554E-01	-1.27413155E-01
-2.52700263E-01	-1.28327368E-01	-2.52204486E-01	-1.29199251E-01	-2.51694914E-01
-1.30045234E-01	-2.51172044E-01	-1.30865333E-01	-2.50626304E-01	-1.31659570E-01
-2.50088351E-01	-1.32427978E-01	-2.49524239E-01	-1.33144278E-01	-2.48950935E-01
-1.33878014E-01	-2.48368778E-01	-1.34569217E-01	-2.47778106E-01	-1.35237924E-01
-2.47179248E-01	-1.35884179E-01	-2.46569692E-01	-1.36543505E-01	-2.45954073E-01
-1.37102758E-01	-2.45332632E-01	-1.37581981E-01	-2.44745610E-01	-1.38241218E-01
-2.44673242E-01	-1.38780520E-01	-2.43435759E-01	-1.39299938E-01	-2.42793388E-01

l=2

l=3

-1.51242563E-02	-1.95367911E-04	-1.73795412E-02	-2.64365301E-04	-1.97698604E-02
-3.50164799E-04	-2.22889964E-02	-4.55138643E-04	-2.49373543E-02	-5.81770230E-04
-2.76918563E-02	-7.32648642E-04	-3.06545308E-02	-9.1578878E-04	-3.36795279E-02
-1.12472355E-03	-3.67655282E-02	-1.36766242E-03	-3.99111986E-02	-1.64073936E-03
-4.31151908E-02	-1.94816060E-03	-4.64939251E-02	-2.303A5975E-03	-4.98663923E-02
-2.69785469E-03	-5.32342364E-02	-3.11921449E-03	-5.659A8389E-02	-3.58102443E-03
-5.99613496E-02	-4.08038395E-03	-6.34715077E-02	-4.64026557E-03	-6.69036336E-02
-5.23075764E-03	-7.02615216E-02	-5.85121592E-03	-7.35449610E-02	-6.50104273E-03
-7.67493254E-02	-7.17968194E-03	-7.99490359E-02	-7.89126645E-03	-8.30554139E-02
-8.62A60346E-03	-8.60915505E-02	-9.39096091E-03	-8.90642989E-02	-1.01776450E-02
-9.19442981E-02	-1.0879949E-02	-9.49011671E-02	-1.18455119E-02	-9.77285083E-02
-1.27144781E-02	-1.00450378E-01	-1.35322233E-02	-1.03070597E-01	-1.44801627E-02
-1.05492773E-01	-1.53737934E-02	-1.08050983E-01	-1.62851447E-02	-1.10401873E-01
-1.71899873E-02	-1.12649105E-01	-1.80957151E-02	-1.14796157E-01	-1.89978113E-02
-1.16446342E-01	-1.98948436E-02	-1.18910628E-01	-2.078A2791E-02	-1.20680168E-01
-2.16726784E-02	-1.224579A3E-01	-2.25467668E-02	-1.24146959E-01	-2.34093445E-02
-1.25749857E-01	-2.42592822E-02	-1.27263216E-01	-2.50970135E-02	-1.28698831E-01
-2.59132553E-02	-1.30059022E-01	-2.67191157E-02	-1.31346015E-01	-2.75097564E-02
-1.32561948E-01	-2.82843906E-02	-1.33596292E-01	-2.90365449E-02	-1.34770094E-01
-2.97740615E-02	-1.35785040E-01	-3.04963668E-02	-1.36742756E-01	-3.1202924E-02
-1.37444803E-01	-3.18932227E-02	-1.38478678E-01	-3.25508198E-02	-1.39267059E-01
-3.32126729E-02	-1.40011126E-01	-3.38514318E-02	-1.40712016E-01	-3.44757675E-02
-1.41170829E-01	-3.50853709E-02	-1.41976013E-01	-3.56711868E-02	-1.42547691E-01
-3.62490515E-02	-1.43086668E-01	-3.68127558E-02	-1.43593723E-01	-3.73641025E-02
-1.44069609E-01	-3.79029059E-02	-1.44505034E-01	-3.84232569E-02	-1.44915872E-01
-3.89335704E-02	-1.45302670E-01	-3.94337199E-02	-1.45665956E-01	-3.99235862E-02
-1.46006244E-01	-1.46403056E-02	-1.46324030E-01	-4.08750253E-02	-1.46619001E-01
-4.13103927E-02	-1.46894025E-01	-4.17780657E-02	-1.47147159E-01	-4.22149569E-02
-1.47379646E-01	-4.26409851E-02	0.	0.	
-1.78426423E-03	-2.71865443E-06	-2.12560430E-03	-3.95360888E-04	-2.53248495E-03
-5.74414023E-06	-1.00103931E-03	-8.24761311E-06	-3.52786732E-03	-1.16426690E-05
-4.10996165E-03	-1.61276075E-05	-4.71837097E-03	-2.16749273E-05	-5.39084744E-03
-2.88354083E-05	-6.12445639E-03	-3.79008068E-05	-6.91655036E-03	-4.91946381E-05
-7.76472952E-03	-6.30608257E-05	-8.66102835E-03	-7.97580099E-05	-9.61225639E-03
-9.98137156E-05	-1.06164362E-02	-1.23649140E-04	-1.16777467E-02	-1.51705821E-04
-1.27765049E-02	-1.84444780E-04	-1.39750039E-02	-2.23812437E-04	-1.51960659E-02
-2.68297282E-04	-1.64393135E-02	-3.18228870E-04	-1.77043803E-02	-3.73942806E-04
-1.89909106E-02	-4.35780616E-04	-2.02589854E-02	-5.02124873E-04	-2.15683774E-02
-5.76084009E-04	-2.29179197E-02	-6.58203828E-04	-2.43066803E-02	-7.49044440E-04
-2.57336576E-02	-8.49179814E-04	-2.72544127E-02	-9.63190896E-04	-2.87823899E-02
-1.08602099E-03	-3.03176987E-02	-1.21795469E-03	-3.18644271E-02	-1.35927935E-03
-3.34106429E-02	-1.51028498E-03	-3.49998496E-02	-1.67458097E-03	-3.65804465E-02
-1.84724056E-03	-3.81528814E-02	-2.02925760E-03	-3.97176427E-02	-2.22042830E-03
-4.12751631E-02	-2.42085102E-03	-4.28544913E-02	-2.63476031E-03	-4.44126120E-02
-2.85534888E-03	-4.59502821E-02	-3.08432689E-03	-4.74692137E-02	-3.32100932E-03
-4.89470776E-02	-3.56531562E-03	-5.04705655E-02	-3.82048000E-03	-5.19443741E-02
-4.08191734E-03	-5.33893830E-02	-4.34887311E-03	-5.48044283E-02	-4.62112103E-03
-5.61963056E-02	-4.89866256E-03	-5.75761224E-02	-5.18479442E-03	-5.89218671E-02
-5.47317415E-03	-6.02344413E-02	-5.76530801E-03	-6.15147081E-02	-6.06031299E-03
-6.27634940E-02	-6.35791634E-03	-6.39914457E-02	-6.65992858E-03	-6.51844214E-02
-6.96304365E-03	-6.63432727E-02	-7.26695736E-03	-6.74688182E-02	-7.57137714E-03
-6.85618455E-02	-7.87602153E-03	-6.96275345E-02	-8.18147347E-03	-7.06599314E-02
-8.48651523E-03	-7.16597923E-02	-8.79027194E-03	-7.26278462E-02	-9.09267963E-03
-7.35647971E-02	-9.39348475E-03	-7.44717486E-02	-9.69255615E-03	-7.53487163E-02
-9.98949361E-03	-7.61963395E-02	-1.02840707E-02	-7.701E2364E-02	-1.05760700E-02
-7.78060050E-02	-1.08652831E-02	-7.85692243E-02	-1.11575101E-02	-7.93554545E-02
-1.14345590E-02	-8.00152386E-02	-1.17142463E-02	-8.06901024E-02	-1.19903957E-02
-8.13575559E-02	-1.22628386E-02	0.	0.	

L=4

L=5

***** ALUMINUM (PERTURBATION EXPANSION) FENDRY PHASE SHIFTS *****

7 DECEMBER 1972

T-FACTORS READ IN TO L= 3 FOR ATOMIC TYPE 1.... ALUMINUM *****

SUBROUTINE STACK FORMS INTERMEDIATE ARRAYS (LL) AND (C).

L= 3 LMAX= 256 (L1,M1,L2,M2,L3,M3)=(4 0 3 3 2 1) NUMPAX= 498

*****FCC (100)-SURFACE

LATTICE CONSTANT IS 4.04000 ANGSTROMS, INTERLAYER SPACING IN BLK IS 2.02000 ANGSTROMS

BASIS VECTOR IN SURFACE PLANE IS (0.00000, 0.00000)

(1) CORE(1)	SYM(1)	D=(	0.00000	0.00000	0.00000)	A=(	2.02000	2.02000)	B=(	2.02000	-2.02000)
(2) CORE(1)	SYM(1)	C=(	2.02000	0.00000	2.02000)	A=(	2.02000	2.02000)	B=(	2.02000	-2.02000)
(3) CORE(1)	SYM(1)	D=(	0.00000	0.00000	4.04000)	A=(	2.02000	2.02000)	B=(	2.02000	-2.02000)

RECIPROCAL LATTICE, SURFACE NET VECTORS

(1)	RA=(	1.55524	1.55524)	RB=(	1.55524	-1.55524)
(2)	RA=(	1.55524	1.55524)	RB=(	1.55524	-1.55524)
(3)	RA=(	1.55524	1.55524)	RB=(	1.55524	-1.55524)

***** ALUMINUM (PERTURBATION EXPANSION) PENDRY PHASE SHIFTS *****

7 DECEMBER 1972

INCIDENT BEAM ANGLES ARE THETA= 0 PHI= 0

INCIDENT ELECTRON ENERGY FROM 40 TO 40 E.V.

MAXIMUM L-VALUE IS 3

LATTICE TEMPERATURE= 293
CRYSTAL TEMPERATURES = 426 AND 426

SUBROUTINE TABLE SETS PROPAGATOR CUTOFF LENGTH AT RPAX= 30.000

FINAL LAYERS OF SUMMATION ARE (1, 1)	MAXX(1)= 348 = TOTAL NUMBER OF ATOMIC SITES WITHIN CUTOFF.
FINAL LAYERS OF SUMMATION ARE (2, 1)	MAXX(2)= 648 = TOTAL NUMBER OF ATOMIC SITES WITHIN CUTOFF.
FINAL LAYERS OF SUMMATION ARE (3, 1)	MAXX(3)= 897 = TOTAL NUMBER OF ATOMIC SITES WITHIN CUTOFF.
FINAL LAYERS OF SUMMATION ARE (1, 2)	MAXX(4)= 1197 = TOTAL NUMBER OF ATOMIC SITES WITHIN CUTOFF.
FINAL LAYERS OF SUMMATION ARE (3, 2)	MAXX(5)= 1413 = TOTAL NUMBER OF ATOMIC SITES WITHIN CUTOFF.
FINAL LAYERS OF SUMMATION ARE (1, 3)	MAXX(6)= 1662 = TOTAL NUMBER OF ATOMIC SITES WITHIN CUTOFF.
FINAL LAYERS OF SUMMATION ARE (2, 3)	MAXX(7)= 1878 = TOTAL NUMBER OF ATOMIC SITES WITHIN CUTOFF.

ARRAY DIMENSIONS REQUIRED (AND SUPPLIED) BY PROGRAM

C=	496	(7532)
LL=	498	(7532)
KX=	256	(2401)
RMAG=	1878	(9000)
XMAG=	1878	(9000)
RVEC=	5634	(30000)
G(COMPLEX)=	3072	(49000)

TIME FOR EXECUTION OF TABLE 2.374 TO 2.422

.....

```

INCIDENT BEAM ENERGY IS 40 E.V.      $$$$  ARGY= 6
RENORMALIZED QUASIPARTICLE ENERGY      E=( 49.8866, 4.0855) E.V.
COMPLEX K-VECTOR      KSTAR=( 3.62655, .14825) INVERSE ANGSTROMS
QUASIPARTICLE RENORMALIZATION CONSTANT  ZNORM=( .82761, -.25235)
INELASTIC DAMPING LENGTH      LAMBDA= 6.7455 ANGSTROMS

```

PROPAGATION VECTORS IN CRYSTAL

	V_x	V_y	V_z	
TYPE(1) BEAM(C, 0)	V1= 0.00000	0.00000	3.62655	$\leftarrow$ incident beam k_i
TYPE(1) BEAM(C, 0)	V2= 0.30000	0.00000	-3.62659	
TYPE(1) BEAM(C, 1)	V3= 1.55524	-1.55524	-2.8835C	
TYPE(1) BEAM(1, 0)	V4= 1.55524	1.55524	-2.8835C	
TYPE(1) BEAM(1, 0)	V5= 3.11049	0.00000	-1.66467	

V3= 0.00000	C.00000	7.25317
V3= -1.55524	1.55524	6.51008
V3= -1.55524	-1.55524	6.51008
V3= -3.11049	C.00000	5.49124

 $\sim k_i - k_r$

FILL LAYER 1 KTYF= 1

	V_x	V_y	V_z	
TYPE(1) BEAM(C, 0)	V1= 0.00000	0.00000	3.62655	$\leftarrow$ outgoing beams, k_r
TYPE(1) BEAM(C, 0)	V2= 0.30000	0.00000	-3.62659	
TYPE(1) BEAM(C, 1)	V3= 1.55524	-1.55524	-2.8835C	
TYPE(1) BEAM(1, 0)	V4= 1.55524	1.55524	-2.8835C	
TYPE(1) BEAM(1, 0)	V5= 3.11049	0.00000	-1.66467	

V3= 0.00000	C.00000	7.25317
V3= -1.55524	1.55524	6.51008
V3= -1.55524	-1.55524	6.51008
V3= -3.11049	C.00000	5.49124

 $\sim k_i - k_r$

```

FILLP      LAYER 1      KTYF= 1
CAPTAIN SUBPLANE PROPAGATOR BY ITERATION

```

```

NUMBER OF ITERATIONS= 8
CCRE( 1)  SYM( 1)  TAU( 0, 0)= 2.45357843E-C1
CCRE( 1)  SYM( 1)  TAU( 0, 1)= 1.09488631E-01
CCRE( 1)  SYM( 1)  TAU( 1, 0)= 1.09488631E-01
CCRE( 1)  SYM( 1)  TAU( 1, 1)= 4.3402518EE-C2

```

} τ obtained by iterative expansion.

```
NUMBER OF ITERATIONS= 0
CCRE( 1)  SYM( 1)  TAU( 0, 0)= 2.45346903E-01
CCRE( 1)  SYM( 1)  TAU( 0, 1)= 1.09495116E-01
CCRE( 1)  SYM( 1)  TAU( 1, 0)= 1.09495116E-01
CCRE( 1)  SYM( 1)  TAU( 1, 1)= 4.34094746E-02 }  $\tau$  obtained by matrix inversion.
```

ITERATION OF T-MATRIX TO ORDER 1

```

0 0 BEAP
  LAYER= 1 T=/ .146542/ -.047547 -.139036 TERM= -.047547 -.139036 SLN= -.047547 -.139036
  LAYER= 2 T=/ .080731/ -.026123 -.076368 TERM= .079357 .014829 SLN= .003610 -.124207
  LAYER= 3 T=/ .044354/ -.014352 -.041968 TERM= -.028539 .033593 SLN= .003271 -.050254

```

ENERGY= 40 E.V. INTENSITY= 4.021236E-C3 LOGARITHM= -2.39564

```

0 [ BEAM
  LAYER= 1 T= / .087518/ -.030972 -.081962 TFRM= -.030972 -.081962 SLM= -.030972 -.081962
  LAYER= 2 T= / .044563/ -.015753 -.041686 TFRM= -.045642 -.043452 SLM= -.040815 -.038459
  LAYER= 3 T= / .022665/ -.008012 -.021202 TFRM= -.016264 -.015681 SLM= -.024450 -.024181

```

ENERGY= 40 E.V. INTENSITY= 1.741977E-C3 LOGARITHM= -2.75396

```

1 C BEAM
  LAYER= 1 T=/.097618/ -.036572 -.081562 TERM= -.036572 -.081562 SUM= -.036572 -.081562
  LAYER= 2 T=/.044563/ -.015753 -.041686 TERM= -.044562 -.041686 SUM= -.040815 -.038495
  LAYER= 3 T=/.022665/ -.008012 -.021202 TERM= -.016264 -.015681 SUM= -.022445C -.054181

```

ENERGY= 40 E.V. INTENSITY= 1.7419735-C2 LOGARITHM= -2.75896

1 1 REAM

LAYER= 1 T=/ .023683/ .010645 -.021154 TERM= .010645 -.021154 SUM= .010645 -.021154
 LAYER= 2 T=/ .009805/ .004405 -.003758 TERM= -.008291 -.005234 SUM= .002355 -.026388
 LAYER= 3 T=/ .004055/ .001825 -.003526 TERM= -.002489 .003207 SUM= -.000130 -.023181
 ENERGY= 40 E.V. INTENSITY= 2.64533CE-04 LOGARITHM= -7.57686

ITERATION OF T-MATRIX TO ORDER 2

0 C BEAM
 LAYER= 1 T=/ .107227/ -.045228 -.097332 TERM= -.045228 -.097332 SUM= -.045228 -.097332
 LAYER= 2 T=/ .049185/ -.013846 -.047197 TERM= .047503 .011155 SUM= .002676 -.086173
 LAYER= 3 T=/ .030007/ -.003638 -.029472 TERM= -.022334 .020040 SUM= -.019658 -.066132
 ENERGY= 40 E.V. INTENSITY= 2.34670CE-03 LOGARITHM= -2.62954

0 1 BEAM
 LAYER= 1 T=/ .070775/ -.011252 -.069875 TERM= -.011252 -.069875 SUM= -.011252 -.069875
 LAYER= 2 T=/ .034556/ .003247 -.034504 TERM= -.021733 .026995 SUM= -.032586 -.042880
 LAYER= 3 T=/ .018295/ .000193 -.018294 TERM= .016906 -.006994 SUM= -.016080 -.049873
 ENERGY= 40 E.V. INTENSITY= 1.353755E-03 LOGARITHM= -2.86846

1 C BEAM
 LAYER= 1 T=/ .070775/ -.011252 -.069875 TERM= -.011252 -.069875 SUM= -.011252 -.069875
 LAYER= 2 T=/ .034556/ .003247 -.034504 TERM= -.021733 .026995 SUM= -.032586 -.042880
 LAYER= 3 T=/ .018295/ .000193 -.018294 TERM= .016906 -.006994 SUM= -.016080 -.049873
 ENERGY= 40 E.V. INTENSITY= 1.353755E-03 LOGARITHM= -2.86846

1 1 BEAM
 LAYER= 1 T=/ .052564/ .040966 -.033571 TERM= .040966 -.033571 SUM= .040966 -.033571
 LAYER= 2 T=/ .025146/ .019555 -.017351 TERM= -.015381 -.021144 SUM= .025585 -.054715
 LAYER= 3 T=/ .008204/ .005847 -.005755 TERM= -.006845 .004523 SUM= .018740 -.050192
 ENERGY= 40 E.V. INTENSITY= 1.415157E-03 LOGARITHM= -2.84920

ITERATION OF T-MATRIX TO ORDER 3

0 C BEAM
 LAYER= 1 T=/ .095884/ -.042094 -.090581 TERM= -.042094 -.090581 SUM= -.042094 -.090581
 LAYER= 2 T=/ .048017/ -.010831 -.046780 TERM= .046057 .013580 SUM= .003963 -.077002
 LAYER= 3 T=/ .033505/ -.007820 -.032580 TERM= -.023870 .023513 SUM= -.019507 -.053489
 ENERGY= 40 E.V. INTENSITY= 1.605858E-03 LOGARITHM= -2.79428

0 1 BEAM
 LAYER= 1 T=/ .064597/ -.008122 -.064084 TERM= -.008122 -.064084 SUM= -.008122 -.064084
 LAYER= 2 T=/ .033509/ .005812 -.033408 TERM= -.023268 .024666 SUM= -.031390 -.039418
 LAYER= 3 T=/ .020372/ -.000695 -.020360 TERM= .018458 -.006621 SUM= -.012932 -.048039

ENERGY= 40 E.V. INTENSITY= 1.220178E-C? LOGARITHM= -2.91358

1 C BEAM

LAYER= 1	T=	.C64597/	-.008122	-.064084	TERM=	-.008122	-.064084	SUM=	-.008122	-.064084
LAYER= 2	T=	.C33909/	.005812	-.C334C8	TERM=	-.023268	.024666	SUM=	-.031390	-.039418
LAYER= 3	T=	.02C372/	-.000695	-.C20360	TERM=	.018458	-.008621	SUM=	-.C12932	-.048C39

ENERGY= 40 E.V. INTENSITY= 1.220178E-C? LOGARITHM= -2.91358

1 L BEAM

LAYER= 1	T=	.C52141/	.043455	-.C28817	TERM=	.043455	-.028817	SUM=	.C43455	-.028817
LAYER= 2	T=	.026183/	.021160	-.C16256	TERM=	-.014135	-.022631	SUM=	.C25319	-.051448
LAYER= 3	T=	.C09C02/	.005954	-.006752	TERM=	-.007141	.005481	SUM=	.C22178	-.045967

ENERGY= 40 E.V. INTENSITY= 1.284203E-03 LOGARITHM= -2.89137

ITERATION OF T-MATRIX TO ORDER 4

0 C BEAM

LAYER= 1	T=	.100112/	-.042810	-.090497	TERM=	-.042810	-.090497	SUM=	-.042810	-.090497
LAYER= 2	T=	.047065/	-.010290	-.C45926	TERM=	-.045048	.013630	SUM=	.002238	-.C76867
LAYER= 3	T=	.033757/	-.C07758	-.032895	TERM=	-.024171	.023623	SUM=	-.C21933	-.053244

ENERGY= 40 E.V. INTENSITY= 1.634807E-0? LOGARITHM= -2.78653

0 L BEAM

LAYER= 1	T=	.064022/	-.008642	-.C63436	TERM=	-.008642	-.063436	SUM=	-.008642	-.063436
LAYER= 2	T=	.033587/	.006755	-.C32900	TERM=	-.023779	.023720	SUM=	-.C32420	-.039716
LAYER= 3	T=	.020521/	-.C00598	-.020512	TERM=	.C18636	-.C08591	SUM=	-.C13784	-.048307

ENERGY= 40 E.V. INTENSITY= 1.244122E-C? LOGARITHM= -2.60514

1 C BEAM

LAYER= 1	T=	.064022/	-.008642	-.C63436	TERM=	-.008642	-.063436	SUM=	-.008642	-.063436
LAYER= 2	T=	.033587/	.006755	-.C32900	TERM=	-.023779	.023720	SUM=	-.C32420	-.039716
LAYER= 3	T=	.020521/	-.C00598	-.020512	TERM=	.C18636	-.008591	SUM=	-.C13784	-.048307

ENERGY= 40 E.V. INTENSITY= 1.244122E-0? LOGARITHM= -2.60514

1 L BEAM

LAYER= 1	T=	.051255/	.043006	-.C27884	TERM=	.C43006	-.027884	SUM=	.043006	-.C27884
LAYER= 2	T=	.027468/	.022324	-.C15005	TERM=	-.C13773	-.023766	SUM=	.C29234	-.051650
LAYER= 3	T=	.006116/	.006055	-.C06814	TERM=	-.007252	.005523	SUM=	.C21581	-.046128

ENERGY= 40 E.V. INTENSITY= 1.287209E-03 LOGARITHM= -2.89033

ITERATION OF T-MATRIX TO ORDER 5

0 0 BEAM

LAYER=	1	T=	/	.C59846/	-.043225	-.C6CC05	TERM=	-.043225	-.09C005	SUM=	-.043225	-.090005
LAYER=	2	T=	/	.C46895/	-.010396	-.045732	TERM=	.C44532	.013441	SUM=	.001707	-.076564
LAYER=	3	T=	/	.C33814/	-.007325	-.C33011	TERM=	-.024494	.023311	SUM=	-.C22787	-.053252

ENERGY= 4C E.V. INTENSITY= 1.654065E-03 LOGARITHM= -2.78145

0 1 BEAM

LAYER=	1	T=	/	.C63421/	-.008834	-.C62802	TERM=	-.0C8834	-.062802	SUM=	-.008834	-.062802
LAYER=	2	T=	/	.033412/	.006698	-.032734	TERM=	-.023636	.023616	SUM=	-.C3247C	-.C35187
LAYER=	3	T=	/	.020539/	-.000219	-.C20538	TERM=	.018808	-.008253	SUM=	-.C13662	-.047439

ENERGY= 40 E.V. INTENSITY= 1.201531E-03 LOGARITHM= -2.92027

1 C BEAM

LAYER=	1	T=	/	.063421/	-.008834	-.C62802	TERM=	-.0C8834	-.062802	SUM=	-.008834	-.062802
LAYER=	2	T=	/	.C33412/	.006698	-.032734	TERM=	-.023636	.023616	SUM=	-.C3247C	-.C35187
LAYER=	3	T=	/	.020539/	-.000219	-.C20538	TERM=	.018808	-.008253	SUM=	-.013662	-.047439

ENERGY= 4C E.V. INTENSITY= 1.201531E-03 LOGARITHM= -2.92027

1 1 BEAM

LAYER=	1	T=	/	.040834/	.042934	-.C27217	TERM=	.042934	-.027217	SUM=	.042934	-.027217
LAYER=	2	T=	/	.C27217/	.022316	-.C13845	TERM=	-.013845	-.023744	SUM=	.C29294	-.C5C961
LAYER=	3	T=	/	.009282/	.006288	-.C06828	TERM=	-.0C7483	.005491	SUM=	.071816	-.045471

ENERGY= 40 E.V. INTENSITY= 1.253962E-03 LOGARITHM= -2.90172

TIME REQUIREMENTS FOR EXECUTION OF ENERGY LCCP
 BEGIN LCCP AT 2.423
 TC END OF FILL 2.431
 TC END OF SUBPL 4.166
 TC END OF INV 4.305
 TC END OF GREEN 10.742
 CCUBLE DIFFRACTION 10.685
 NEXT ITERATION 11.031
 NEXT ITERATION 11.176
 NEXT ITERATION 11.321

TYPE(1) BEAM(C, 0) ENERGY-VS-INTENSITY

E= 4C-- 1.654065E-03 E=

TYPE(1) BEAM(C, 1) ENERGY-VS-INTENSITY

E= 4C-- 1.201531E-03 E=

TYPE(1) BEAM(1, 0) ENERGY-VS-INTENSITY

E= 40-- 1.20153053E-03 E=

TYPE(1) BEAM(1, 1) ENERGY-VS-INTENSITY

E= 40-- 1.25396198E-03 E=

LEGAL NOTICE

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.

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