ORNL
MASTER COPY

OAK RIDGE NATIONAL LABORATORY
operated by
UNION CARBIDE CORPORATION
for the
U. S. ATOMIC ENERGY COMMISSION

ORNL - TM - 1425

56

OR GNU,

A FORTRAN PROGRAM FOR CALCULATING FREQUENCY-DISTRIBUTION FUNCTIONS OF CUBIC CRYSTALS

L. J. Raubenheimer
G. Gilat

NOTICE This document contains information of a preliminary nature and was prepared primarily for internal use at the Oak Ridge National Laboratory. It is subject to revision or correction and therefore does not represent a final report.

LEGAL NOTICE

This report was prepared as an account of Government sponsored work. Neither the United States, nor the Commission, nor any person acting on behalf of the Commission:

A. Makes any warranty or representation, expressed or implied, with respect to the accuracy, completeness, or usefulness of the information contained in this report, or that the use of any information, apparatus, method, or process disclosed in this report may not infringe privately owned rights; or

B. Assumes any liabilities with respect to the use of, or for damages resulting from the use of any information, apparatus, method, or process disclosed in this report.

As used in the above, "person acting on behalf of the Commission" includes any employee or contractor of the Commission, or employee of such contractor, to the extent that such employee or contractor of the Commission, or employee of such contractor prepares, disseminates, or provides access to, any information pursuant to his employment or contract with the Commission, or his employment with such contractor.

Errata and Miscellaneous Improvements to OR GNU

June 9, 1966

The following changes should be made to the symbolic program listing, pp. 26-39:

The force constants PX(18) and PX(19) for bcc only, have to be interchanged on cards 484 and 485, so that PX(18) appears on the former and PX(19) on the latter.

Card No. 304 should be altered to read

```
D3Q = 1.0E-4                                    304
```

This change gets rid of the small spurious spikes sometimes exhibited by $g(\nu)$ (see for example the curve for sodium, Ref. 1, Fig. 4). Unfortunately, this change influences the table of numbers on page 25 somewhat. The last three numbers are unchanged to nine decimal places, but other numbers agree to anywhere from three to eight places. The highest "negative frequency" channel contains 3·6256E+02 instead of 5·8620E+01.

Confusion may arise from the fact that array V in subroutine FREQ corresponds to E elsewhere, while PH in FREQ is the same as V in other subroutines; it is therefore suggested that, in subroutine FREQ, V and PH be replaced by E and V respectively. On cards 313 and 315, replace V by E; on cards 297, 315 and 348, replace PH by V. On card 291, replace PH(12) by E(12).

Four remaining Logical IF statements may, if necessary be replaced by their arithmetic counterparts as follows:

```
        IF(V(L)-XM)          21, 19, 19                590
        IF(AL(1)-AL(2))      521, 522, 522             599
522     IF(AL(2)-AL(3))      523, 527, 527             603
523     IF(AL(3)-AL(1))      524, 525, 525             604
```

The above corrections and suggested changes have been made to all card decks.

The term 1·OE-99 on card No. 219 may cause an underflow on some computers, in which case this term should simply be left out.

ORNL-TM-1425

Contract No. W-7405-eng-26

SOLID STATE DIVISION

OR GNU,

A FORTRAN PROGRAM FOR CALCULATING FREQUENCY-DISTRIBUTION FUNCTIONS OF CUBIC CRYSTALS

L. J. Raubenheimer and G. Gilat

JUNE 1966

OAK RIDGE NATIONAL LABORATORY
Oak Ridge, Tennessee
operated by
UNION CARBIDE CORPORATION
for the
U.S. ATOMIC ENERGY COMMISSION

CONTENTS

Page

ABSTRACT 1

INTRODUCTION 1

THEORY 2

Running Conditions 3

Statistical Weights 5

Computation Time 6

APPLICATIONS 8

EXTENSION TO NONCUBIC SYSTEMS 8

OTHER APPLICATIONS 10

INPUT DATA FOR THE PROGRAM 10

Data Cards 10

Special Tapes Required 14

DESCRIPTION OF SUBROUTINES 15

SOME NOTES ON FORTRAN 63 22

ADDED NOTE 24

EXAMPLE: SOME CRUDE RESULTS FOR SODIUM 25

SYMBOLIC PROGRAM LISTING 26

OR GNU,

A FORTRAN PROGRAM FOR CALCULATING FREQUENCY-DISTRIBUTION FUNCTIONS OF CUBIC CRYSTALS

L. J. Raubenheimer† and G. Gilat

ABSTRACT

This report describes a computer program which finds the frequency-distribution function of a centrosymmetric cubic solid for which a Born-von-Kármán force model is available. The FORTRAN language is used in order to facilitate modifications necessary for other force models, as well as other crystal structures. Detailed instructions for using the program are included, and the required card decks may be obtained from the authors.

INTRODUCTION

A procedure was developed recently[1] for finding the frequency distribution of a cubic solid to a high degree of accuracy. This report contains the listing of a computer program, written for this purpose in FORTRAN 63 for the CDC 1604-A computer. The program covers three cases: the (simple cubic) CsCl structure, and the fcc and bcc structures, both with one atom per primitive unit cell. The program uses Born-von-Kármán force models for all three cases, but these can be

†Guest scientist from the South African Atomic Energy Board, Pretoria, Republic of South Africa.

[1]G. Gilat and L. J. Raubenheimer, Phys. Rev. 144, 390 (1966). There is a mistake in Eq. (16). The factor $(2b^2)$ in the first term should be replaced by b^2.

replaced by other models if necessary. A brief discussion is included to indicate the alterations needed to make the program applicable to other cubic systems, such as the diamond structure which has a complex dynamical matrix. The possibility of extending the method to noncubic systems is also discussed. Indications are that the procedure may find useful applications in other fields such as the computation of the electron density of states, as derived from band structure calculations.

THEORY

For a complete treatment of the present method, the reader is referred to an article,[1] which is an extention of the earlier extrapolation method.[2] Briefly, a dynamical matrix is found and diagonalized at points on a "shifted" simple cubic mesh which uniformly fills the irreducible section of the first Brillouin zone (hereafter to be called the irreducible zone). Reciprocal space is assumed to be filled with small cubes, each of which has a mesh point at its center. Eigenfrequencies anywhere in a small cube are obtained by linearly extrapolating from the values found at the center point. Because of the assumption of linear extrapolation, one gets constant frequency surfaces which are represented by planes, parallel to each other and evenly spaced throughout the small cube. The number of modes in any frequency range is then proportional to the volume element (in q-space) falling between the corresponding constant frequency planes inside the small cube. The frequency distribution of the solid is obtained by adding together the contributions from all the different small cubes. In

---

[2]G. Gilat and G. Dolling, Phys. Letters 8, 304 (1964).

order for linear extrapolation to be adequate, the mesh of points at which diagonalizations are to be performed should be sufficiently fine.

A few points not treated extensively in Ref. 1 are discussed next.

Running Conditions

The irreducible zone for the (simple cubic) CsCl case is defined by the four planes

$$q_x = q_{x_{max}} = 0.5;\ q_z = 0;\ q_x = q_y;\ \text{and}\ q_y = q_z \quad , \qquad (1)$$

with the q_α in reduced units (i.e., units of $2\pi/a$). For fcc with one atom per primitive unit cell there are five planes:

$$q_x = q_{x_{max}} = 1.0;\ q_z = 0;\ q_x = q_y;\ q_y = q_z;\ \text{and}\ q_x + q_y + q_z = \frac{3}{2} \ , \qquad (2)$$

while for bcc we have four planes:

$$q_z = 0;\ q_x = q_y;\ q_y = q_z;\ \text{and}\ q_x + q_y = 1 \quad , \qquad (3)$$

the latter case also having $q_{x_{max}} = 1$.

We define a "mesh number," n (JIMAX in the program), as the number of slices into which the q_x-axis is initially divided, giving a distance 2b between mesh points:

$$2b = \frac{q_{x_{max}}}{n} \quad . \qquad (4)$$

(n equals KK, 4KK and 2KK, respectively, for simple cubic, fcc and bcc, where KK is a parameter read in from data card 6.) Mesh points are assumed to lie on a "shifted" mesh at the centers of small cubes, with coordinates:

$$q_\alpha = (m_\alpha - \frac{1}{2}).2b \ , \quad (\alpha = 1,2,3) \quad , \qquad (5)$$

where the m_α are running indices falling within appropriate limits. This shifted mesh avoids most of the degeneracies which complicate the extrapolation procedure.

Since extrapolation calculations tend to be less accurate close to the origin, we first treat only those cubes having running index m_x between certain limits, which we call IMIN and IMAX. These two numbers are obtained from data card 6, or if read in as zero, the following value(s) is(are) automatically assumed:

$$\begin{aligned} &\text{IMIN} = 3, \text{ and/or} \\ &\text{IMAX} = n \end{aligned} \tag{6}$$

Upon completing calculations for all cubes within these limits of m_x, the program subdivides a part of the section for which $m_x < \text{IMIN}$, by a mesh which is three times finer than that for the "outer" part, with new limits on m_x given by

$$\begin{aligned} &\text{IMAX}' = 3.(\text{IMIN} - 1) \\ &\text{IMIN}' = 4 \ , \end{aligned} \tag{7}$$

and a distance between mesh points:

$$2b' = 2b/3 \ .$$

This section is called the "intermediate" part; after going through it, the "inner" section is taken into account, having

$$\begin{aligned} &\text{IMAX}'' = 3.(\text{IMIN}' - 1) = 9 \ , \\ &\text{IMIN}'' = 1 \ , \end{aligned} \tag{8}$$

and the separation between mesh points becoming:

$$2b'' = 2b'/3 = 2b/9 \ .$$

If IMIN is read in as 1, then the entire irreducible zone is treated as one section, with no intermediate or inner part. (Notice that in this case IMAX' is automatically zero.)

Statistical Weights

It is of course impossible to fill the irreducible zone exactly with small cubes; as much as 5/6 of the volume of a "surface" cube may fall outside the boundaries of the irreducible zone, if the zone is to be filled completely. Such a cube is treated as if it falls completely inside the irreducible zone, except that all contributions from it are reduced by its statistical weight (which happens to be proportional to the fraction of the volume inside the zone).

In general, the weight, W_{q_c} (U in the program), given to a small cube by the present method is exactly the same as that which the center point of the cube would have in the well known root sampling method. In the latter method, for simple cubic or fcc, a point (q_x, q_y, q_z) is given a weight proportional to the number of ways in which $\pm q_1$, $\pm q_2$ and $\pm q_3$ can be permuted to give distinct combinations. For bcc it is the number of ways in which $\pm[q_x$ or $(1-q_x)]$, $\pm[q_y$ or $(1-q_y)]$ and $\pm[q_z$ or $(1-q_z)]$ can be permuted, whichever gives the lowest weight. Assuming that any point to be treated falls in the region $q_x > q_y > q_z > 0$, as is always the case in our irreducible zone, we now give some examples of weights obtained. (The asterisk (*) on the coordinate q_1 should be taken to indicate that for bcc only, q_1^* should be read as $[q_1$ or $(1-q_1)]$; of the two possible weights obtained in this way (for bcc), the lowest is the correct value.)

(a) $W(q_x^* > q_y > q_z > 0) \propto 2^3 \cdot 3! = 48$,

(b) $W(q_x^* > q_y > 0;\ q_z = 0) \propto 2^2 \cdot 3! = 24$,

(c) $W(q_x^* > q_y > q_z > 0) \propto 2^3 \cdot 3!/2! = 24$,

(d) $W(q_x^* = q_y > 0;\ q_z = 0) \propto 12$,

(e) $W(q_x^* = q_y = q_z > 0) \propto 8$,

(f) $W(q_x^* > 0;\ q_y = q_z = 0) \propto 6$ and

(g) $W(q_x^* = q_y = q_z = 0) \propto 1$.

When using a shifted mesh (Eq. (5)), none of the coordinates can be zero, hence cases b, d, f and g are avoided. Only a, c and e occur, with relative weights of

$$48:24:8 \text{ , or } 6:3:1 \text{ ;}$$

for this reason the weights used in the program are 6, 3 and 1.

Computation Time

The total number of diagonalizations as a function of KK and IMIN (see INPUT DATA, card 6) is given for the three cases, simple cubic, fcc and bcc, respectively, by:

$$\begin{aligned} N_{sc} &= \frac{1}{6}\,KK\,(KK+1)(KK+2) + \Delta N \\ N_{fcc} &= \frac{2}{3}\,KK\,(2.KK+1)(4.KK+1) + \Delta N \\ N_{bcc} &= \frac{1}{3}\,KK\,(KK+1)(KK+2) + \Delta N \end{aligned} \tag{9}$$

$$\begin{aligned} &\text{where} \quad \Delta N = \frac{1}{3}(IMIN-1)(13.IMIN^2 - 14.IMIN + 3) + 155 \text{ for } IMIN \neq 1 \\ &\text{and} \quad \Delta N = 0 \text{ for } IMIN = 1. \end{aligned} \tag{10}$$

From (10), $\Delta N = 207$ for IMIN = 3.

If a single mesh is used over the entire irreducible zone, as is the case for IMIN = 1, then the total number of diagonalizations is given by Eq. (9), with $\Delta N = 0$.

In deriving Eq. (10), one has to keep in mind that the intermediate and inner sections of the irreducible zone for fcc and bcc have the same form as that for simple cubic.

In Ref. 1, calculations for fcc and bcc are reported, both structures with one atom per primitive unit cell. Respective mesh numbers were 32 and 40 (or values of 8 and 20 for KK) and in both cases IMIN was 3. These numbers, respectively, required 3199 and 3287 matrix diagonalizations, which limited computation times to about 20 minutes.* Hence the CDC 1604-A computer, which has an effective memory cycle time of 4.5 microseconds, was able to treat about 2 2/3 mesh points per second. Some calculations have also been performed for the CsCl structure. Since the dynamical matrix in this case is 6 x 6, computing time is of course longer. Specifically, with the four-neighbor force model found for ordered beta-brass,[3] the computer used about one second per diagonalization. Preliminary calculations with mesh numbers up to 20 for this structure indicate that a value of closer to 30 will be necessary in order to get a frequency distribution function of the same quality as those reported in Ref. 1. This will require about 100 minutes of computer time. If one is purely interested in thermodynamic properties, rather than in the minor details of $g(\nu)\Delta\nu$ itself, then a considerably smaller mesh number may be adequate in all three cases, with corresponding saving of computer time.

---

[3]G. Gilat and G. Dolling, Phys. Rev. 138, A1053 (1965).

*See Added Note, p. 24.

In all calculations performed so far, the frequency ranges were divided into several thousands of channels. Again, this may not be necessary for most applications; it may be that even a hundred channels are sufficient. Of course, one then loses a lot of detail in $g(\nu)\Delta\nu$.

We should add that over eighty percent of all computation time is used for matrix diagonalization by Subroutine JACØBI; hence in order to make the procedure more efficient, this subroutine will have to be optimized.

APPLICATIONS

The present program can be applied directly to any one of three types of cubic materials, provided a Born-Von-Kármán model is available for that material. The three types are listed next, together with the maximum number of sets of neighbors at present allowed for by the program, in brackets:

(a) The simple cubic CsCl structure (4)

(b) fcc with one atom per primitive unit cell (8) and

(c) bcc with one atom per primitive unit cell (8).

The present program assumes the dynamical matrix to be real, so in order to extend the method to cases involving complex numbers, such as the diamond structure, subroutines JACOBI and GRAD have to be altered; JACØBI will then have to diagonalize a hermitian dynamical matrix while GRAD will have to perform multiplication of complex matrices.

EXTENSION TO NONCUBIC SYSTEMS

If systems other than cubic are to be considered, more elaborate changes will be necessary. The mesh of points filling the irreducible

zone will no longer be simple cubic and hence the volume enclosing each mesh point will be other than cubic. In general it will probably be a parallelepiped for which cross-section areas will have to be derived (see Ref. 1). Also, the weights of parallelepipeds at the surfaces of the irreducible zone will have to be looked at more carefully. In the cubic case any small cube which lies partly outside the irreducible zone has a weight proportional to the fraction of the cube falling inside the irreducible zone.

The parameter NSFB (see INPUT DATA, card 2), which has the value 1 for simple cubic, 2 for fcc or 3 for bcc, causes the three routines (see DESCRIPTION OF SUBROUTINES) GNU, CØNST and ELEM, to do different calculations for each of the three cases. In all three routines, transfer is affected by statements of the form

$$\text{GØ TØ } (N_1, N_2, N_3) \text{ NSFB} \quad .$$

In program GNU this leads to the setting up of running conditions over the appropriate irreducible zone, in CØNST some convenient constants are calculated for each structure type, while in ELEM, transfer is effected to the section calculating the elements of the appropriate dynamical matrix. In case a new type of structure is to be treated, one may either replace all of these sections referred to by NSFB, or add another possibility in each case, with all transfer control statements becoming:

$$\text{GØ TØ } (N_1, N_2, N_3, N_4) \text{ NSFB} \quad .$$

OTHER APPLICATIONS

In principle, this method should be applicable to the finding of other types of distribution functions. Provided one is able to set up an equation at any point in the Brillouin zone, the eigenvalues of which are to be sorted into histogram channels, the method can, in principle, be used to find the distribution of these eigenvalues. An example is the calculation of the density of states of electrons in a metal.

INPUT DATA FOR THE PROGRAM

All data are read in at the beginning of the main program, GNU. Comment cards have also been inserted in the main program, describing the meaning of the difference constants read in.†

Data Cards

1) TITLE: FØRMAT (10A8) - 80 characters (one full card) of Hollerith information, to be used as a title with the print-out and plot.

2) NSFB, NNN, NFC: FØRMAT (3I5). The number NSFB indicates the type of structure. It should have the value 1 for simple cubic, 2 for fcc or 3 for bcc. NNN is the dimension of the dynamical matrix, i.e. three times the number of atoms per primitive unit cell. NFC is the number of force constants to be read in (see data card 5).

3) DVA, XM, DQ: FØRMAT (3F10.0). DVA is the width, $\Delta\nu$ (in THz), of the frequency intervals which make up the $g(\nu)\Delta\nu$ histogram, i.e.,

---

†The listing of a sample data deck for Na is given at the end of this report, together with the frequency distribution obtained from these data. Results are extremely crude, but are intended only for checking the program.

the width of the channels into which XM (next paragraph) is to be divided. Its value depends entirely on the required resolution; it may be as large as, say, XM/100, but no smaller than XM/6950 in the present program. In Ref. 1, values of DVA ranging from XM/4000 to XM/5000 were used.

XM is a frequency, in THz (i.e., 10^{12} c/s), which must be slightly higher than the highest frequency expected to occur in computation.

DQ (δq_α in Ref. 1, Eq. (4)) is the increment in the q_α ($\alpha = 1,2,3$) to be used in determining all three components of $\text{grad}_q \nu$ numerically. A value of 0.0001 is suggested.

4) AMAS(1), AMAS(4), AMAS(7), AMAS(10): FØRMAT (4F10.0). Masses (in atomic units) of the atoms in the unit cell. If only one atom/unit cell, then only the first number is non-zero, etc.

5) (PX(I), I = 1, NFC): FØRMAT (4F20.6). The PX(I) are the force constants, four per card. The order in which they are read in, is determined as follows: In general the force constant matrix for an atom with coordinates $x > y > z > 0$, which belongs to the i-th set of neighbors, has the form:[4]

$$\begin{matrix} \alpha_1^{(i)} & \beta_3^{(i)} & \beta_2^{(i)} \\ \beta_3^{(i)} & \alpha_2^{(i)} & \beta_1^{(i)} \\ \beta_2^{(i)} & \beta_1^{(i)} & \alpha_3^{(i)} \end{matrix}$$

where, because of symmetry, some of these elements may be equal to each other, or zero. The elements should be supplied in the order

[4]G. L. Squires, "Inelastic Scattering of Neutrons in Solids and Liquids," International Atomic Energy Agency, Vienna (1963), Vol. II, p. 71.

$$\alpha_1^{(1)}, \alpha_2^{(1)}, \alpha_3^{(1)}, \beta_1^{(1)}, \beta_2^{(1)}, \beta_3^{(1)}, \alpha_1^{(2)}, \alpha_2^{(2)}, \ldots ,$$

including only non-zero and independent force constants. For instance for the bcc structure, the two closest neighbors are of types (111) and (200) respectively, and a two-neighbor force model calculation requires only four independent non-zero force constants, namely $\alpha_1^{(1)}$, $\beta_1^{(1)}$, $\alpha_1^{(2)}$ and $\alpha_2^{(2)}$. Up to four neighbors are allowed for in the simple cubic case, and up to eight in the other two cases. The number NFC (card 2) indicates how many force constants are to be read in.

6) NCTA, NCTC, NTAPE, NPLØT, KK, KG, IMIN, IMAX, NDIV, NPRINT: FØRMAT (10I5).

NCTA controls print-out of $g(\nu)\Delta\nu$.

NCTA < 0: the array is not printed out;

= 0: only the final array of $g(\nu)\Delta\nu$ is printed out;

> 0: intermediate arrays are also printed out.

Normally only the final array is needed, although the intermediate ones are useful for comparing the contributions from the three subsections of the irreducible zone (see: Running Conditions).

NCTC determines whether or not (part of) the array $g(\nu)\Delta\nu/\nu^2$ is to be found. If less than or equal to zero, the array is not obtained; otherwise the fraction (1/NCTC) of it is calculated and printed out, starting from zero frequency.

NTAPE has the same function as NCTA, except that it controls the writing of $g(\nu)\Delta\nu$ on special tape 7, which may then be saved for future use.

NPLØT, if greater than zero, signifies that a plot of $g(\nu)\Delta\nu$ is to be made; if equal to or less than zero, no plot will be made, in which case data card 7 is not to be supplied.

KK is a number which determines the "fineness" of the mesh which divides up the irreducible zone. q_x, which runs from 0 to 0.5 for simple cubic and from 0 to 1 for fcc and bcc, is to be divided up into KK, 4KK and 2KK parts respectively for the three cases. (This number of divisions corresponds to the "mesh number" of Eq. (4). The number of diagonalizations as a function of KK has been treated under the heading Computation Time.

KG determines whether, for each mesh point, the following information is to be printed out: the serial number of the point, its three running indices (the m_α of Eq. (5)), the statistical weight, U (W_{q_c} of Ref. 1, Eq. (20)), the three components of the reduced q-vector, and all the eigenfrequencies obtained at the point. If KG = 0, all these numbers are printed <u>and</u> punched on cards; if KG = 1, they are printed only; if KG = 2, they are punched only, while if KG > 2, they are neither punched nor printed out.

IMIN and IMAX are the minimum and maximum values of the running index, m_x (Eq. (5)), on the crude mesh. Either or both may be read in as zero, in which case the program sets IMIN = 3 and/or IMAX = the appropriate "mesh number" (Eq. (4)). Cubes having running index m_x < IMIN, are further subdivided into smaller cubes before extrapolation, as explained under the heading Running Conditions. If, however, IMIN is read in as unity, no such further subdivision is affected.

NDIV is no longer used. Simply leave it blank.

NPRINT controls the print-out of information pertaining to each extrapolation. Normally NPRINT = 0 or 1, in which case no information is printed out. The information printed for NPRINT $\geq$ 2 is: V(L) (the eigenfrequency), GRD ($=|\mathrm{grad}_q \nu_L|$), AL_1, AL_2, AL_3 (direction cosines of the gradient vector), DELW (thickness of slices into which the cube is subdivided), W_1 ($|W_1|$ is the distance from the cube center to the first corner; if $W_1 > 0$, we have the 6-sided case, while if $W_1 < 0$, we have the 4-sided case), VOLUME (correct volume of the cube), DIFF (the error made in finding the volume by adding all volume elements together) and NDEL (the number of frequency channels to which this extrapolation contributes).

7) PLØTW, PLØTH: FØRMAT (2F10.0): These parameters are the width and maximum height, in inches, of the plot. This card is only required if NPLØT (card 6) is greater than zero. Convenient parameters for the standard CALCOMP plotter are:

PLØTW = 12.0, PLØTH = 8.5.

Special Tapes Required

Plotting is done on tape No. 6. The only order referring to this tape number is the first executable statement of the main program:

CALL PLØTS (BUFFER, 2000, 6) .

If NTAPE (data card 6) is greater than or equal to zero, the distribution $g(\nu)\Delta\nu$ is also written on special tape 7, format (5E16.8). The final distribution of $g(\nu)\Delta\nu$ is preceded by TITLE (one line, format 10A8).

DESCRIPTION OF SUBROUTINES

Program GNU:

DIMENSION CTA(7000), CTC(7000), etc.

This is the main program. Depending on the value of parameter NSFB, it sets up running conditions and finds appropriate weights, as it fills up the irreducible zone with a (simple cubic) mesh of points. If required (see Running Conditions), it divides the zone into 3 parts--an outer, filled with a crude mesh, an intermediate, filled with a somewhat finer mesh, and an innermost filled with an even finer mesh. It calls the necessary subroutines for finding eigenvalues, for extrapolating, etc., and controls the printing and/or punching of most of the output data. $g(\nu)\Delta\nu$ is stored in array CTA and is normalized so that its sum equals 10^{10}, before it is printed out. In order to check for the occurrence of "negative" frequencies, $g(\nu)\Delta\nu$ is stored in CTA from location 51 upwards. (Negative frequencies can occur as a result of extrapolation at $q \sim 0$--see discussion in Ref. 1.) The first 50 channels, which should contain any contributions from these "negative" frequencies, are finally printed out, before and separate from the final $g(\nu)\Delta\nu$. If required, $g(\nu)\Delta\nu/\nu^2$ is computed (see INPUT DATA, card 6 - NCTC) and stored in array CTC. It is normalized so that the average of channels 11, 12, 13, and 14 is unity. (This is done purely because computational errors for the first few channels usually are large.)

Some of the arrays and variables used by GNU are:

Q(3) - the three Cartesian components of the q-vector in reduced units (i.e., units of $2\pi/a$).

GD1(12), GD2(12), GD3(12) - respectively contain the x, y and z components of the gradients corresponding to the different eigenvalues.

V(12) - the eigenfrequencies. (Not to be confused with array $\underset{\sim}{V}$ in Subroutine FREQ, which contains numbers proportional to the squares of the eigenfrequencies.)

U - the weighting factor for a small cube. It corresponds to W_{q_c} in Eq. (20) of Ref. 1. For cubic systems, U is directly proportional to the fraction of the volume of the small cube which falls inside the irreducible zone. In the program it is conveniently taken to be six times that fraction.

The first executable order of this program, i.e.,

CALL PLØTS (BUFFER, 2000, 6) ,

activates all plotting routines (see subroutine GNPLØT), "informing" the program that all plotting information has to be stored in array <u>BUFFER</u> (2000 locations) and then dumped out onto special tape 6. The array <u>BUFFER</u> is emptied every time it gets full and also at the end of the calculation. The CALCOMP plotter uses this tape as a control tape for plotting.

Subroutine CØNST

DIMENSIØN PX(60), AMAS(12)

The purpose of this subroutine is mainly to prevent unnecessary repetition of calculations by subroutine ELEM.

The array PX initially contains the force constants as read in. If, however, a particular force constant always appears with the same

numerical factor in the elements of the dynamical matrix, the factor is here incorporated into the force constant.

In the case of the CsCl structure, the array AMAS, which contains the atomic masses of the two atoms in the unit cell in locations 1 and 4, respectively, is altered, so that

$$\text{AMAS}(1) = \text{AMAS}(2) = \text{AMAS}(3) = [\text{AMAS}(1)]^{-\frac{1}{2}}$$

and

$$\text{AMAS}(4) = \text{AMAS}(5) = \text{AMAS}(6) = [\text{AMAS}(4)]^{-\frac{1}{2}},$$

where the values of AMAS(1) and (4) on the right hand side are those which were read in originally.

A few numerical constants, CPT, SC and ANN, are also determined for later use.

Subroutine FREQ (Q, GD1, GD2, GD3)

DIMENSIØN A(12,12), Q(3), PH(12), V(12), EV(12,12), AA(12,12), DA(12,12), GD1(12), GD2(12), GD3(12), etc.

This subroutine controls the routines ELEM, JACØBI and GRAD. Together these subroutines perform the calculations represented by Eqs. (1) through (8) of Ref. 1. For any q-vector (array Q(3)), subroutine FREQ calls the routine ELEM to set up dynamical matrix $\underline{A}$ ($\underline{D}^{o}(q)$ of Ref. 1). Subroutine JACØBI then diagonalizes $\underline{A}$, storing eigenvalues in $\underline{V}$ and eigenvectors in $\underline{EV}$. (These eigenvalues are proportional to the squares of the eigenfrequencies.) FREQ finds the eigenfrequencies themselves and stores them in array $\underline{PH}$. After adding a small amount, δq_α (DQ in the program), to the α-th component of the q-vector, the modified dynamical matrix $\underline{D}^{o}(\underline{q} + \hat{e}_\alpha\, \delta q_\alpha)$ is obtained by ELEM and stored in

array AA. The "change" in the dynamical matrix, $\underline{\Delta}^{\alpha}(q)$ (Ref. 1, Eq. (4)), is then found by FREQ and stored in array DA. Subroutine GRAD performs the operations of Eqs. (6) and (8) of Ref. 1, with matrix U of Ref. 1 equivalent to the array EV. This yields the α-th components of the gradients of all the eigenfrequencies. The procedure of finding a modified dynamical matrix, etc., is repeated for each of the three Cartesian directions so that finally the x-components of all gradients are in array GD1, the y-components in GD2 and the z-components in GD3.

Subroutine ELEM (Q,A)

DIMENSION Q(3), A(12,12), etc.

This routine finds the elments of the dynamical matrix $\underline{D}^{o}(\underline{q})$ and stores them in array A. The components of q should be in Q. N is the dimension of $\underline{D}^{o}(\underline{q})$. Depending on the value of code number NSFB (data card 6) the subroutine sets up the required matrix for any of the following three cases:

NSFB = 1: simple cubic (CsCl structure),

= 2: fcc (monatomic),

= 3: bcc (monatomic).

If another theory, rather than that of Born and von Kármán, is to be used, ELEM and part of CØNST will have to be replaced.

Subroutine JACOBI (A,B,E,N)

DIMENSION A(12,12), B(12,12), E(12).

This routine diagonalizes (real) matrix A, stores eigenvalues in E and eigenvectors as columns in B. The reason for introducing a few double precision constants is a precaution against the extremely rare

possibility that YSQ (statement 77) is zero within the range of single precision. This has happened only once in probably 500,000 calculations. Normally, the double precision constants are not used.

Subroutine GRAD (A,G)

DIMENSION EV(12,12), A(12,12), G(12), V(12), etc.

This subroutine calculates one Cartesian component of the gradient of each of the eigenfrequencies, V_j, and stores them in G. The formal variable G stands for GD1, GD2 or GD3, depending on which Cartesian component is calculated. The program performs the operations of Eq. (6) and (8), Ref. 1, with A of the subroutine equivalent to $\underline{\Delta}^{\alpha}(\underline{q})$ and EV equivalent to the eigenvector matrix U in Ref. 1.

Subroutine SWEEP (Q, GD1, GD2, GD3)

DIMENSION CTA(7000), etc.

Using the eigenfrequencies and gradients obtained at the center of a small cube, this subroutine extrapolates each eigenfrequency linearly throughout the cube, utilizing Eqs. (9) through (21) of Ref. 1. It finds contributions $U\Delta V$ to be added to the appropriate channels $(\nu, \nu + \Delta\nu)$ of the $g(\nu)\Delta\nu$ histogram (array CTA). ΔV is the volume element (in q-space) which falls between the appropriate frequency limits $(\nu, \nu + \Delta\nu)$ and U is the weighting factor of the small cube. [One eventually ends up with a $g(\nu)\Delta\nu$ which is normalized to $3N.U_{max}.V$, where N is the number of atoms per unit cell, U_{max} the weight of a general cube, i.e. one lying entirely inside the irreducible zone, and V the volume of

the zone. In program GNU, however, the distribution is normalized to 10^{10}.]

The subroutine always divides each cube into at least 160 slices, using intermediate array CTD.* The volume of each slice is taken to be its area times its thickness. This approximation turns out to be extremely accurate, reproducing the volume of the cube to within better than 10^{-6}. Nevertheless, the total number of frequencies from such extrapolation is normalized to equal exactly U times the volume of the cube, before being sorted into the channels of CTA. Subdivision of each cube is made such that an integral number of channels of CTD exactly corresponds to one channel of CTA.

An alternative procedure which is perhaps more elegant and does not utilize the approximation of Eq. (10), Ref. 1, is finding the volume elements V between any two frequencies by exact integration of the function S(w) (Ref. 1) between the appropriate limits. This may even save some computer time, since the use of intermediate finer frequency channels are not required. Also, normalizing of the contributions from a small cube is then no longer necessary since the sum of volume elements should be exactly equal to the volume of the cube.

Subroutine WRITE (ARRAY, NDIM, DELF, NA, NB, NFMT):
DIMENSION ARRAY (NDIM).

This routine is used for printing out both of arrays CTA (which contains $g(\nu)\Delta\nu$) and CTC (which holds $g(\nu)\Delta\nu/\nu^2$). The formal variables have the following meanings:

ARRAY - the set of numbers to be printed out.

---

*This procedure is no longer used. The current method is described in the next paragraph.

NDIM - dimension of ARRAY, part of which is to be printed out.

DELF - width of the frequency intervals (in THz or 10^{12} cps) into which the entire range of frequencies has been divided (same as DVA in GNU, and DF in GNPLØT).

NA,NB - the locations of the first and last numbers, in ARRAY, which are to be printed.

NFMT - a code number indicating which format is to be used in print-out:

NFMT = 1, FØRMAT(IX, F10.5, 7X, 10F10.0)

NFMT = 2, FØRMAT(IX, F10.5, 7X, 10E10.3)

The frequency is assumed to start from zero. On each line a frequency followed by ten numbers from ARRAY is printed. This frequency corresponds to that on the low side of the first channel represented in each row.

Subroutine GNPLØT (G, NDIM, FMAX, DF, PLØTW, PLØTH, TITLE):
DIMENSION G(NDIM), TITLE(10).

This routine makes a plot of $g(\nu)\Delta\nu$. The number of frequencies in each channel is represented by a single point and successive points are connected by straight lines. All the necessary scaling of numbers is done by this subroutine. The formal variables have the following meanings:

G - the array of numbers to be plotted (identical to CTA in GNU).

NDIM - dimension of the array G, part of which is to be plotted.

FMAX - the highest expected frequency (in THz), corresponding to XM, data card 3.

DF - the width of the frequency channels into which FMAX has been divided. It is equivalent to DVA, data card 3. NOTE: each frequency channel corresponds to a location in G.

PLØTH - width (in inches) of plot.

PLØTH - height (in inches) of plot. The largest number in G is scaled to this height.

TITLE - a 10-word array containing 80 characters of Hollerith information, read in as data by program GNU, and written below the plot by this subroutine.

NOTE: This subroutine has been written for the CDC 1604-A computer with CALCOMP plotter. Use has been made of the plotting subroutines available, such as PLØT, SYMBØL and NUMBER which draw lines, write Hollerith information and write numbers respectively.

SOME NOTES ON FORTRAN 63[5]

Although a detailed discussion of the particular version of FORTRAN is not intended, a few remarks are probably in order:

1) As is common for most versions of FORTRAN, all variables having names starting with I, J, K, L, M or N are treated as integers. However, mixed operations are allowed; that is, if on the right hand side of a FORTRAN statement, integers and floating point variables are mixed, everything is changed to floating point before execution. The final form of the result is determined by the type of variable being calculated.

---

[5]FORTRAN 63/Reference Manual, CONTROL DATA CORPORATION, 8100 34th Ave., S., Minneapolis 20, Minn. (Jan. 1964).

2) Both "logical" (two-branch) and "arithmetic" (three-branch) IF-statements are used. "Logical" IF-statements contain abbreviations such as .GT. (greater than), .GE. (greater than or equal to), .NE. (not equal to), etc. If the logical statement in brackets is true, transfer takes place to the first address, otherwise to the second one. Several logical IF's may be combined using .AND. or .ØR.

3) When using a standard tape unit for reading in, writing out or punching, the number of the particular tape unit is not required. Only when a non-standard tape is used, does one have to specify it by number. For instance, in statement 299 of the main program:

```
299 WRITE(7,9125) (CTA(J), J = 1,NA)  ,
```

writing takes place on special tape 7 according to format 9125.

4) CØMMØN statements with or without block identifiers may be used. It may be necessary, for some other versions of FORTRAN, to make a single CØMMØN statement for all subroutines.

5) The line printer used with the 1604-A at ORNL allows 120 characters per line, the first of which controls the printer in the usual way.

6) The computer allows a word length of eight Hollerith characters. The only places where more than six are being used, though, are with the reading in, printing out and writing (by the plotter) of array TITLE (10 locations) - see program GNU, formats 123 and 9123, and subroutine GNPLØT.

7) The 1604-A has a 32K core memory, hence for a smaller machine, dimensions of arrays may have to be reduced. Effective memory cycle time is 4.5 microseconds.

ADDED NOTE:

Computation time depends much more strongly on the number of channels into which the frequency range is subdivided, than was believed at first. If the number of channels (XM/DVA) is around 100, the total time required for extrapolating all three eigenfrequencies is about 1/4 of that required for diagonalization, or 20% of the total time used per cube, in the case of one atom per unit cell. Upon increasing the number of channels to 4000, extrapolation time becomes about 2 1/2 times the diagonalization time, or about 70% of the total time per cube. Hence computation time can be drastically cut by reducing the number of frequency channels.

EXAMPLE - SOME CRUDE RESULTS FOR SODIUM

INPUT DATA

```
CRUDE FREQ DISTR FOR SODIUM.  MESH NO. = 2*KK = 6 .
   3    3   13
   0.05      4.2        0.0001
 22.991
    1178.0                1320.0                472.0              104.0
     -38.0                  -0.4                -65.0               52.0
      -7.0                   3.0                 14.0               17.0
      33.0
   0    3    0    1    3    7    0    0    0    0
  12.0      8.5
```

LISTING OF OUTPUT ON SPECIAL TAPE 7

```
CRUDE FREQ DISTR FOR SODIUM.  MESH NO. = 2*KK = 6 .
  6.23688883E+04  2.50354609E+05  6.06351398E+05  1.18220212E+06  1.96164074E+06
  2.89313314E+06  4.12573376E+06  5.53273481E+06  7.23247484E+06  9.42577495E+06
  1.21938614E+07  1.50412758E+07  1.81606808E+07  2.13220185E+07  2.68620000E+07
  3.30204189E+07  5.38464414E+07  6.52112832E+07  7.01483320E+07  7.85173863E+07
  8.56799591E+07  9.20651184E+07  9.99928469E+07  1.05597737E+08  1.09821642E+08
  1.14794321E+08  1.12554104E+08  1.09346317E+08  1.23257006E+08  1.33808608E+08
  1.50451070E+08  1.70146369E+08  1.75751036E+08  1.71718456E+08  1.60370674E+08
  1.41974076E+08  1.28368968E+08  1.23113054E+08  1.21442961E+08  1.13656939E+08
  1.11196085E+08  1.21759753E+08  1.32673313E+08  1.32696468E+08  1.21086234E+08
  1.20049459E+08  1.25293357E+08  1.41357848E+08  1.61204512E+08  1.82117726E+08
  1.97917652E+08  2.05563178E+08  2.04580392E+08  1.99693083E+08  1.96009852E+08
  1.90260207E+08  1.79560917E+08  1.68182249E+08  1.67104290E+08  1.71185325E+08
  1.71498471E+08  1.69682590E+08  1.68297111E+08  1.68290751E+08  1.81372580E+08
  1.90959580E+08  1.99468019E+08  1.99047705E+08  2.20839963E+08  2.76864495E+08
  3.70516186E+08  4.78676680E+08  3.08357828E+08  2.57318441E+08  1.90475282E+08
  1.50205943E+08  8.66534406E+07  9.40471737E+06  1.07052708E+06  6.14663243E-02
             0               0               0               0
```

Additional Output:

Contents of the fifty "negative frequency" channels are printed out, all containing zeros except the highest one, (-0.05,0), which yields the number 5.8620E+01. The final $g(\nu)\Delta\nu$ (NCTA = 0) as well as the lower one-third of $g(\nu)\Delta\nu/\nu^2$ (NCTC = 3) is printed out on the standard output unit, but these are not reproduced here. A plot is also made (NPLOT = 1), but is not included here.

These results are useful for checking the program, since they require a total execution time of only 60 seconds on the 1604-A computer. To obtain a reasonable distribution for sodium, a much larger mesh number is needed (see Ref. 1).

SYMBOLIC PROGRAM LISTING

```
      PROGRAM GNU
C
      DIMENSION CTA(7000),CTC(7000),BUFFER(2000),TITLE(10)
      DIMENSION AL(3),Q(3),V(12),GD1(12),GD2(12),GD3(12),PX(60)
      DIMENSION AMAS(12)
      DIMENSION EV(12,12)
      COMMON/2/ANN
      COMMON/3/PI,CPT,PX,U,SC,NSFB,AMAS,DQ
      COMMON/4/EV,V
      COMMON/5/LA,CTA,DVA,AHALF,VOLUME,ASQ,NPRINT,XM
      COMMON/6/KM,KK,I,J,K
      COMMON/7/NNN
C
C  THE ONLY (DIMENSIONED) ARRAYS IN COMMON STATEMENTS ARE CTA(7000),
C  EV(12,12) AND V(12) ,  V IN GNU IS EQUIVALENT TO PH IN SUBR. FREQ
C
  101 FORMAT(1HK )
 7102 FORMAT(5X,22HFREQ, CHANNEL WIDTH =  ,F6.4 )
  105 FORMAT(1X,I4,3I3,2X,3HU =,F4.1,5X,3HQ =,3F9.5,5X,3HV =,6F9.5)
  106 FORMAT(1HK,10X, 22HARRAY OF G(NU)/(NU)**2  ///)
  107 FORMAT(1X,I5,3I3,F5.1,4X,6F9.5/8F9.5)
 2107 FORMAT(4F20.6)
  108 FORMAT(1H0,10X,18HATOMIC MASS(ES) =  ,4F10.4 )
 2109 FORMAT(16H0FORCE CONSTANTS  )
 1111 FORMAT(33HK  (NSFB, NNN, KK, IMIN, IMAX) =  ,10I4)
 1112 FORMAT(44H0    CYCLE NO., KK, NO. CUBES, IMIN, IMAX =  ,10I5)
  119 FORMAT(8F10.0)
 1120 FORMAT(20I5)
 2120 FORMAT(1H0, 10E11.4)
  123 FORMAT(10A8)
 8123 FORMAT(18HKFREQ(10**12 CPS), ,20X, 24HG(NU) HISTOGRAM CHANNELS //)
 9123 FORMAT(1HK,10X,10A8/1HK)
  125 FORMAT(10F8.0)
 9125 FORMAT(5E16.8)
 1002 FORMAT(1H1)
 1003 FORMAT(1H1/35HK      *NEGATIVE* FREQUENCY CHANNELS  /1HK)
C
C  ALL *LOGICAL IF* STATEMENTS HAVE BEEN REPLACED BY ARITHMETIC IF*S
C  SOME LOGICAL IFS HAVE BEEN LEFT IN AS COMMENTS, INDICATED BY C*
C
C  ACTIVATE PLOTTING ROUTINES FIRST
      CALL PLOTS(BUFFER,2000,6)
C
C  NDIMA AND NDIMC ARE THE DIMENSIONS OF CTA AND CTC
      NDIMA = 7000
      NDIMC = NDIMA
C
C READ 1ST DATA CARD
      READ 123, TITLE
C
C READ 2ND DATA CARD
      READ 1120, NSFB, NNN, NFC
C  NSFB = 1 FOR SIMPLE CUBE,  =2 FOR FCC  AND =3 FOR BCC ,
C  NNN = DIMENSION  ( =3 TIMES NO. OF ATOMS/UNIT CELL )
C  NFC = NO. OF FORCE CONSTANTS TO BE READ IN FURTHER ON .
C
C READ 3RD DATA CARD
      READ 119, DVA, XM, DQ
C  DVA = CHANNEL WIDTH OF HISTOGRAM (IN 10**12 C/S)
C  XM = MAX FREQ  (UNITS OF 10**12 C/S)
C  DQ = DELTA Q FOR COMPUTING GRAD(NU), USUALLY DQ = 0.0001
C
```

```
C READ 4TH DATA CARD
      READ 119,AMAS(1),AMAS(4),AMAS(7),AMAS(10)
C
C READ 5TH DATA CARD(S) = FORCE CONSTANTS
      READ 2107, (PX(I), I=1,NFC)
C
      L = NFC + 1
      DO 527 J = L,60
  527 PX(J) = 0.0
C
C READ 6TH DATA CARD
      READ 1120, NCTA,NCTC,NTAPE,NPLOT,KK,KG,IMIN,IMAX,NDIV,NPRINT
C  NDIV IS NOT USED ANY MORE
C
      IF(NPLOT)710,710,402
C
C READ 7TH DATA CARD = PLOT INFORMATION
C  READ IN PLOT WIDTH (OR LENGTH) AND MAX PLOT HEIGHT IN INCHES  .
  402 READ 119, PLOTW, PLOTH
C
  710 PRINT 9123, TITLE
      PRINT 108,AMAS(1),AMAS(4),AMAS(7),AMAS(10)
      PRINT 2109
      PRINT 2107, (PX(I), I=1,NFC)
C
  310 LA = XM/DVA + 51.0
C  WE START STORING G(NU) IN LOC. 51 OF CTA IN CASE OF *NEG* FREQS
      NS = 0
C  NS IS TO BE THE MESH POINT SEQUENCE NUMBER
C FIND SOME CONVENIENT CONSTANTS ETC. WITH SUBROUTINE CONST .
C
      CALL CONST
C
      DO 10 I = 1,NDIMA
   10 CTA(I) = 0.0
      DO 406 M = 1,NDIMC
  406 CTC(M) = 0.0
C
  311 GO TO (312,313,314) NSFB
  312 JIMAX = KK
      PP = 0.5
      GO TO 320
  313 JIMAX = 4*KK
      PP = 1.0
      GO TO 320
  314 JIMAX = 2*KK
      PP = 1.0
  320 CUBE = PP/JIMAX
      IF(IMAX)1320,1320,2320
 1320 IMAX = JIMAX
 2320 IF(IMIN)321,321,401
  321 IMIN = 3
  401 PRINT 1111,NSFB,NNN,KK,IMIN,IMAX
      PRINT 101
C
C  BRILLOUIN ZONE NOW BROKEN UP INTO THREE PARTS (KM = 1,3)
C
      DO 69 KM = 1,3
   11 AHALF = CUBE/2.0
      VOLUME = 8.0*AHALF**3
C**  AHALF IS HALF THE LENGTH OF A LITTLE CUBE EDGE  .
      ASQ = AHALF**2
C
      DO 350 I = IMIN,IMAX
      GO TO (864,865,866) NSFB
```

```
  864 JMAX = I
      GO TO 867
  865 JMAX = 3*KK+0.5001-ABSF(3*KK+0.5-I)
      KFCC = 6*KK + 1
      GO TO 867
  866 JMAX = KK+0.5001-ABSF(KK+0.5-I)
  867 DO 350 J=1,JMAX
  868 DO 350 K = 1,J
      GO TO (345,870,345) NSFB
  870 IF(I+J+K-KFCC)345,345,350
  345 Q(1) = (I-0.5)*CUBE
      Q(2) = (J-0.5)*CUBE
      Q(3) = (K-0.5)*CUBE
C  THE *LOGICAL IF* FORM OF THE STATEMENT(S) WHICH FOLLOW(S), IS
C*    IF(JMAX.NE.J.AND.J.NE.K) 349,346
      IF((JMAX-J)*(J-K))349,346,349
  346 IF(K-JMAX)348,347,347
  347 U=1.0
      GO TO 19
  348 U = 3.0
      GO TO 19
  349 U = 6.0
   19 NS = NS + 1
      CALL FREQ(C,GD1,GD2,GD3)
   31 IF(KG-1)34,34,29
   34 PRINT 105, NS,I,J,K,U,(Q(M),M=1,3),(V(M),M=1,NNN)
C  THE *LOGICAL IF* FORM OF THE STATEMENT(S) WHICH FOLLOW(S), IS
C* 29 IF(KG.EQ.0.OR.KG.EQ.2) 35,370
   29 IF(KG*(KG-2))370,35,370
   35 PUNCH 107, NS,I,J,K,U,(Q(M),M=1,3),(V(M),M=1,NNN)
C
  370 CALL SWEEP(Q,GD1,GD2,GD3)
  350 CONTINUE
      PRINT 101
C
      SA = 0.0
C  THE *LOGICAL IF* FORM OF THE STATEMENT(S) WHICH FOLLOW(S), IS
C*    IF(NCTA.GT.0.OR.NTAPE.GT.0.OR.IMIN.EQ.1) 47,68
      IF(NCTA)447,447,47
  447 IF(NTAPE)547,547,47
  547 IF(IMIN-1)68,47,68
   47 DO 56 I=1,LA
   56 SA = SA+CTA(I)
      RA = 1.0E+10/SA
      DO 57 J=1,LA
   57 CTA(J)=CTA(J)*RA
   68 PRINT 101
C  THE *LOGICAL IF* FORM OF THE STATEMENT(S) WHICH FOLLOW(S), IS
C*    IF(NCTA.GT.0.AND.IMIN.GT.1) 63,66
      IF(NCTA)66,66,163
  163 IF(IMIN-1)66,66,63
   63 CALL WRITE(CTA,NDIMA,DVA,51,LA,1)
      PRINT 101
   66 IF(NTAPE)167,167,67
   67 WRITE(7,9125) (CTA(J), J=51,LA)
  167 CONTINUE
      IF(IMIN-1)646,746,646
C  THE *LOGICAL IF* FORM OF THE STATEMENT(S) WHICH FOLLOW(S), IS
C*646 IF(NCTA.GT.0.OR.NTAPE.GT.0) 46,746
  646 IF(NCTA) 246,246,46
  246 IF(NTAPE) 746,746,46
   46 DO 146,J=1,LA
  146 CTA(J)=CTA(J)/RA
  746 PRINT 1112, KM,KK,NS,IMIN,IMAX
      PRINT 101
```

```
      KK = 3*KK
      CUBE = CUBE/3.0
      IMAX = (IMIN-1)*3
      GO TO (71,70,69) KM
   70 IMIN = 1
      GO TO 69
   71 IMIN = 4
   69 CONTINUE
C
  202 PRINT 1003
      PRINT 2120, (CTA(J), J=1,50)
  201 DO 298 I=51,LA
  298 CTA(I-50) = CTA(I)
      NA = LA-50
      NAA = NA + 1
      DO 4000 J = NAA, LA
 4000 CTA(J) = 0.0
      IF(NCTA)6298,3298,3298
 3298 PRINT 1002
      PRINT 9123, TITLE
      PRINT 7102, DVA
      PRINT 8123
      CALL WRITE(CTA,NDIMA,DVA,1,NA,1)
 6298 IF(NCTC)5298,4298,4298
 4298 PRINT 1002
      CON5 = (CTA(11)/110.25 + CTA(12)/132.25 + CTA(13)/156.25 +
     B CTA(14)/182.25)/(4.0*DVA**2) + 1.0E-99
C  CON5 NORMALIZES CTC SO THAT AVE. OF CHANNELS 11 THRU 14 IS UNITY .
      FRQ = -DVA/2.0
      NFRAC = NA/NCTC
      DO 61 I = 1,NFRAC
      FRQ = FRQ + DVA
   61 CTC(I) = CTA(I)/(CON5*FRQ**2)
      PRINT 106
      CALL WRITE(CTC,NDIMC,DVA,1,NFRAC,2)
 5298 IF(NPLOT)2298,2298,1298
 1298 CALL GNPLOT(CTA,NDIMA,XM,DVA,PLOTW,PLOTH,TITLE)
 2298 IF(NTAPE) 200,199,199
  199 WRITE(7,123) TITLE
  299 WRITE(7,9125) (CTA(J), J=1,NA)
      END FILE 7
  200 END
      SUBROUTINE CONST
C
C THIS SUBROUTINE SETS UP CONSTANTS, MULTIPLIES FORCE CONSTANTS
C BY CONVENIENT FACTORS, ETC., SO AS TO AVOID REPETITION IN
C SUBROUTINE ELEM .
C
      DIMENSION PX(60), AMAS(12)
      COMMON/2/ANN
      COMMON/3/PI,CPT,PX,U,SC,NSFB,AMAS,DQ
      COMMON/7/N
C
      HH=1.66043
      PI=3.14159265
      SC=39.4784176*HH
  307 ANN = 0.5/(DQ*SC)
C  ANN AND SC ARE CHANGED FURTHER DOWN FOR FCC AND BCC .
C
      GO TO (51,52,53) NSFB
C
C  SIMPLE CUBIC  (CESIUM CHLORIDE STRUCTURE)
   51 CPT=8.0*(PX(1)+PX(13)+2.0*PX(14))
  306 AMAS(1) = AMAS(2) = AMAS(3) = 1.0/SQRTF(AMAS(1))
      AMAS(4) = AMAS(5) = AMAS(6) = 1.0/SQRTF(AMAS(4))
```

```
C  IN THE CS=CL CASE, ARRAY AMAS CONTAINS 1/(SQRTS. OF MASSES) -
C  THESE ARE TAKEN INTO ACCOUNT BY SUBROUTINE ELEM .
      RETURN
C
C  FCC (MONATOMIC)
   52 CPT=2.0*(PX(4)+2.0*PX(5)+PX(25)+2.0*PX(26))
      ANN = ANN/AMAS(1)
      SC = SC*AMAS(1)
      DO 1 I = 1,3
    1 PX(I)=4.0*PX(I)
      PX(4)=2.0*PX(4)
      PX(5)=2.0*PX(5)
      DO 2 I=6,9
    2 PX(I)=8.0*PX(I)
      DO 3 I=10,16
    3 PX(I)=4.0*PX(I)
      DO 4 I=17,24
    4 PX(I)=8.0*PX(I)
      PX(25)=2.0*PX(25)
      PX(26)=2.0*PX(26)
      RETURN
C
C BCC (MONATOMIC)
   53 CPT=2.0*(PX(3)+2.0*(PX(4)+PX(6)+2.0*(PX(1)
     1+PX(5)+PX(8)+PX(12)+2.0*PX(9))))+2.0*(PX(14)
     2+2.0*(PX(15)+2.0*(PX(16)+PX(20)+PX(21)+PX(22)
     3+2.0*PX(17))))
      ANN = ANN/AMAS(1)
      SC = SC*AMAS(1)
   90 RETURN
      END
      SUBROUTINE FREQ(Q,GD1,GD2,GD3)
C
      DIMENSION Q(3),PH(12),A(12,12),B(12,12)
      DIMENSION EV(12,12),AA(12,12),DA(12,12),V(12),W(12),PX(60),QL(3)
      DIMENSION GD1(12),GD2(12),GD3(12)
      DIMENSION AMAS(12)
      COMMON/2/ANN
      COMMON/3/PI,CPT,PX,U,SC,NSFB,AMAS,DQ
      COMMON/4/EV,PH
      COMMON/7/N
C
   90 FAC=0.0
      IF(U-1.0)10,1004,10
 1004 D10=1.0E-4
      D20=-1.0E-4
      D30=0.0
      Q(1)=Q(1)+D10
      Q(2)=Q(2)+D20
      Q(3)=Q(3)+D30
      FAC=1.0
   10 CALLELEM(Q,A)
      DO 12 I=1,N
      DO 12 J=1,N
   12 B(I,J)=A(I,J)
   15 CALL JACOBI(B,EV,V,N)
      DO 16 I=1,N
   16 PH(I)=SQRTF(V(I)/SC)
   17 QL(1)=Q(1)+DQ
      QL(2) = Q(2)
      QL(3) = Q(3)
      CALL ELEM(QL,AA)
      DO 19 I=1,N
      DO 19 J=1,N
   19 DA(J,I)=DA(I,J)=AA(I,J)-A(I,J)
```

```
   22 CALL GRAD(DA,GD1)
      IF(Q(1)-Q(2))24,23,24
   24 QL(1) = Q(1)
      QL(2) = Q(2) + DQ
      QL(3) = Q(3)
      CALL ELEM(QL,AA)
      DO 29 I=1,N
      DO 29 J=1,N
   29 DA(J,I)=DA(I,J)=AA(I,J)-A(I,J)
   32 CALL GRAD(DA,GD2)
      IF(Q(2)-Q(3))26,25,26
   23 DO 33 J=1,N
   33 GD2(J)=GD1(J)
   26 QL(1) = Q(1)
      QL(2) = Q(2)
      QL(3) = Q(3) + DQ
      CALL ELEM(QL,AA)
      Q(1)=Q(1)-FAC*D1Q
      Q(2)=Q(2)-FAC*D2Q
      Q(3)=Q(3)-FAC*D3Q
      DO 34 I=1,N
      DO 34 J=1,N
   34 DA(J,I)=DA(I,J)=AA(I,J)-A(I,J)
   37 CALL GRAD(DA,GD3)
      DO 370 J=1,N
  370 PH(J)=PH(J)-(GD1(J)*D1Q+GD2(J)*D2Q+GD3(J)*D3Q)*FAC
      RETURN
   25 DO 40 J=1,N
   40 GD3(J)=GD2(J)
   38 RETURN
      END


      SUBROUTINE ELEM(Q,A)
C
      DIMENSION A(12,12),Q(3),C(12,3),S(12,3),PX(60)
      DIMENSION AMAS(12)
      DIMENSION EV(12,12), V(12)
      COMMON/3/PI,CPT,PX,U,SC,NSFB,AMAS,DQ
      COMMON/4/EV,V
      COMMON/7/N
C
      GO TO (1,2,3) NSFB
C
C  SIMPLE CUBIC (CESIUM CHLORIDE STRUCTURE)
C
    1 DO 510 I = 1,3
      S(1,I)=SINF(PI*Q(I))
      C(1,I)=COSF(PI*Q(I))
      S(2,I)=2.0*S(1,I)*C(1,I)
      C(2,I)=1.0-2.0*S(1,I)**2
      S(3,I)=S(2,I)*C(1,I)+S(1,I)*C(2,I)
  510 C(3,I)=C(2,I)*C(1,I)-S(1,I)*S(2,I)
      DO 526 I = 1,3
  517 IF(I-3)512,511,512
  511 J = 1
      GO TO 513
  512 J = I+1
  513 IF(I-1)515,514,515
  514 K=3
      GO TO 516
  515 K = I-1
  5160 A(I,I)=CPT+2.0*(PX(3)*(1.0-C(2,I))+PX(5)*(2.0-C(2,J)-C(2,K))
```

```
     A   +2.0*(PX(7)*(2.0-C(2,I)*(C(2,J)+C(2,K)))+PX(9 )*(1.0-C(2,J)
     B    *C(2,K))))
     0A(I+3,I+3)=CPT+2.0*(PX(4)*(1.0-C(2,I))+PX(6)*(2.0-C(2,J)-C(2,K))
     A   +2.0*(PX(8)*(2.0-C(2,I)*(C(2,J)+C(2,K)))+PX(10)*(1.0-C(2,J)
     B    *C(2,K))))
     0 A(I,I+3)=A(I+3,I)=-8.0*(C(1,J)*C(1,K)*(PX(1)*C(1,I)+PX(13)*
     A    C(3,I)))
      A(I,J)=A(J,I)=4.0*PX(11)*S(2,I)*S(2,J)
      A(I+3,J+3)=A(J+3,I+3)=4.0*PX(12)*S(2,I)*S(2,J)
     0 A(I,J+3)=A(I+3,J)=A(J,I+3)=A(J+3,I)=8.0*(S(1,I)*S(1,J)*(C(1,K)*
     A    PX(2)+PX(15)*C(3,K))+PX(16)*C(1,K)*(S(3,I)*S(1,J)+S(1,I)*
     B    S(3,J)))
  526 CONTINUE
      DO 527 I = 1,N
      DO 527 J=1,N
  527 A(I,J) = A(I,J)*AMAS(I)*AMAS(J)
  518 RETURN
C
C  FCC
C
    2 DO 125 I = 1,3
      CON1=PI*Q(I)
      S(1,I)=SINF(CON1)
      C(1,I)=COSF(CON1)
      S(2,I)=2.0*S(1,I)*C(1,I)
      C(2,I)=1.0-2.0*S(1,I)**2
      S(3,I)=S(2,I)*C(1,I)+S(1,I)*C(2,I)
      C(3,I)=C(2,I)*C(1,I)-S(1,I)*S(2,I)
      S(4,I)=2.0*S(2,I)*C(2,I)
  125 C(4,I)=1.0-2.0*S(2,I)**2
      DO 126 I = 1,N
  117 IF(I-3)112,111,112
  111 J = 1
      GO TO 113
  112 J = I+1
  113 IF(I-1)115,114,115
  114 K=3
      GO TO 116
  115 K = I - 1
  1160 A(I,I)=CPT+     PX(1)*(2.0-C(1,I)*(C(1,J)+C(1,K)))
     1  +  PX(2)*(1.0-C(1,J)*C(1,K))-     PX(4)*C(2,I)
     2-     PX(5)*(C(2,J)+C(2,K))+     PX(6)*(1.0-C(2,I)*C(1,J)*
     3C(1,K))+     PX(7)*(2.0-C(1,I)*(C(2,J)*C(1,K)+C(1,J)*
     4C(2,K)))+     PX(10)*(2.0-C(2,I)*(C(2,J)+C(2,K)))
     5+     PX(11)*(1.0-C(2,J)*C(2,K))
     6+     PX(13)*(2.0-C(3,I)*(C(1,J)+C(1,K)))
     7+     PX(14)*(2.0-C(1,I)*(C(3,J)+C(3,K)))
     8+     PX(15)*(2.0-C(3,J)*C(1,K)-C(3,K)*C(1,J))
     9+     PX(17)*(1.0-C(2,I)*C(2,J)*C(2,K))
     0  A0 = PX(19)*(2.0-C(3,I)*(C(2,J)*C(1,K)+C(1,J)*
     1C(2,K)))+
     2     PX(20)*(2.0-C(2,I)*(C(1,J)*C(3,K)+C(3,J)*C(1,K)))
     3+     PX(21)*(2.0-C(1,I)*(C(3,J)*C(2,K)+C(2,J)*
     4C(3,K)))-     PX(26)*(C(4,J)+C(4,K))-     PX(25)*
     5C(4,I)
      A(I,I)=A(I,I)+A0
     0 A(I,J)=A(J,I)=     PX(3)*S(1,I)*S(1,J)+     PX(9)
     1*C(1,K)*(S(2,I)*S(1,J)+S(1,I)*S(2,J))+
     2     PX(8)*S(1,I)*S(1,J)*C(2,K)+     PX(12)*S(2,I)*
     3S(2,J)+     PX(16)*(S(3,I)*S(1,J)+S(1,I)*S(3,J))+
     4     PX(18)*S(2,I)*S(2,J)*C(2,K)+
     5     PX(24)*C(1,K)*(S(3,I)*S(2,J)+S(2,I)*
     6S(3,J))+     PX(23)*C(2,K)*(S(1,I)*S(3,J)+S(3,I)*
     7S(1,J))+     PX(22)*C(3,K)*(S(1,I)*S(2,J)+S(2,I)*
     8S(1,J))
```

```
  126 CONTINUE
  118 RETURN
C
C  BCC
    3 DO 10 I = 1,3
      S(1,I)=SINF(PI*Q(I))
      C(1,I)=COSF(PI*Q(I))
      S(2,I)=2.0*S(1,I)*C(1,I)
      C(2,I)=1.0-2.0*S(1,I)**2
      S(3,I)=S(2,I)*C(1,I)+S(1,I)*C(2,I)
      C(3,I)=C(2,I)*C(1,I)-S(1,I)*S(2,I)
      S(4,I)=2.0*S(2,I)*C(2,I)
   10 C(4,I)=1.0-2.0*S(2,I)**2
      DO 26 I=1,3
   17 IF(I-3)12,11,12
   11 J=1
      GO TO 13
   12 J=I+1
   13 IF(I-1)15,14,15
   14 K=3
      GO TO 16
   15 K=I-1
   160 A(I,I)=CPT-8.0*((PX(1)*C(1,I)+PX(8)*C(3,I))*
     1C(1,J)*C(1,K)+PX(12)*C(2,I)*C(2,J)*C(2,K)+
     2PX(9)*C(1,I)*(C(3,J)*C(1,K)+C(1,J)*C(3,K)))
     3-4.0*(PX(5)*C(2,I)*(C(2,J)+C(2,K))+PX(6)*C(2,J)*
     4C(2,K))-2.0*(PX(3)*C(2,I)+PX(4)*(C(2,J)+C(2,K)))
     5-8.0*(PX(16)*C(1,I)*C(3,J)*C(3,K)+
     6PX(17)*C(3,I)*(C(1,J)*C(3,K)+C(3,J)*C(1,K)))
     7-2.0*(PX(14)*C(4,I)+(PX(15)+2.0*PX(21)*C(2,I))*
     8(C(4,J)+C(4,K))+2.0*(PX(20)*C(4,I)*(C(2,J)+C(2,K))
     9+PX(22)*(C(4,J)*C(2,K)+C(2,J)*C(4,K))))
     0 A(I,J)=A(J,I)=4.0*(PX(7)+2.0*PX(13)*C(2,K))
     1*S(2,I)*S(2,J)+4.0*PX(23)*(S(4,I)*S(2,J)+
     2S(2,I)*S(4,J))+8.0*(S(1,I)*S(1,J)*(PX(2)*
     3C(1,K)+PX(10)*C(3,K))+(PX(11)*C(1,K)+PX(19)
     4*C(3,K))*(S(3,I)*S(1,J)+S(1,I)*S(3,J))+PX(18)
     5*S(3,I)*S(3,J)*C(1,K))
   26 CONTINUE
   18 RETURN
      END


      SUBROUTINE JACOBI(A,B,E,N)
C
      DIMENSION A(12,12),B(12,12),E(12)
      EQUIVALENCE (Y,YA)
      TYPE DOUBLE  DENOM, YY, DD,YA
      S=0.0
      DO10I=1,N
   10 S=S+ABSF(A(I,I))
      TEST=S/N
      DO12I=1,N
      DO11J=1,N
   11 B(I,J)=0.0
   12 B(I,I)=1.0
      GOTO15
   25 DO13I=1,N
      II=I-1
      DO13J=1,II
      P=A(I,J)
      IF(ABSF(P)-AMAX)13,16,16
   16 Y=(A(J,J)-A(I,I))/2.0
```

```
      D=Y**2+P*A(J,I)
      IF(D) 18,18,77
   77 YSQ = Y+SQRTF(D)
      IF(YSQ)31,31,17
   31 DD = YA**2 + P*A(J,I)
      YY=DSQRT(DD)
      DENOM=Y+YY
      YSQ=DABS(DENOM)
      SIGN=P*A(I,J)
      IF(SIGN)32,17,17
   32 YSQ=-YSQ
   17 X=P/YSQ
      GOTO19
   18 X=P/Y
   19 U=SQRTF(1+X*X)
      C=1/U
      S=X*C
      DO20K=1,N
      Y=A(K,I)
      Z=A(K,J)
      A(K,I)=C*Y-S*Z
      A(K,J)=S*Y+C*Z
      Y=B(K,I)
      Z=B(K,J)
      B(K,I)=C*Y-S*Z
   20 B(K,J)=S*Y+C*Z
      DO 1 K=1,N
      Y=A(I,K)
      Z=A(J,K)
      A(I,K)=C*Y-S*Z
    1 A(J,K)=S*Y+C*Z
   13 CONTINUE
   15 S=0.0
      DO21I=1,N
      II=I-1
      DO21J=1,II
      IF(ABSF(A(I,J))-S)21,21,22
   22 S=ABSF(A(I,J))
   21 CONTINUE
   23 R=S/TEST
      AMAX=S/5.0
      IF(R-1.0E-6)24,24,25
   24 DO26I=1,N
   26 E(I)=A(I,I)
      RETURN
      END


      SUBROUTINE GRAD(A,G)
C
      DIMENSION EV(12,12),A(12,12),G(12),P(12,12),V(12)
      COMMON/2/ANN
      COMMON/4/EV,V
      COMMON/7/N
      DO10I=1,N
      Y=0.0
      DO11J=1,N
      X=0.0
      DO12K=1,N
   12 X=X+EV(K,I)*A(K,J)
   11 P(I,J)=X
      DO13K=1,N
   13 Y=Y+P(I,K)*EV(K,I)
```

```
   10 G(I) = Y*ANN/V(I)
      RETURN
      END


      SUBROUTINE SWEEP(Q,GD1,GD2,GD3)
C
      COMMON/3/PI,CPT,PX,U,SC,NSFB,AMAS,DQ
      COMMON/4/EV,V
      COMMON/5/LA,CTA,DVA,AHALF,VOLUME,ASQ,NPRINT,XM
      COMMON/6/KM,KQ,IZ,JZ,KZ
      COMMON/7/NNN
      DIMENSION CTA(7000),AL(3),V(12),GD1(12),GD2(12),GD3(12)
      DIMENSION PX(60),Q(3)
      DIMENSION AMAS(12)
      DIMENSION EV(12,12)
C
 1000 FORMAT(5X, 7F11.6, 2E15.5, I6)
 1002 FORMAT(26HKV(L),GT,MAX FREQ  V(L) =  ,E15.6/ 1WK)
C
      DO 20 L = 1,NNN
      IF (V(L).GT.XM) 19, 21
   19 PRINT 1002,V(L)
      GO TO 20
   21 GRD = SQRTF(GD1(L)**2+GD2(L)**2+GD3(L)**2)
      DELW=DVA/GRD
      AL(1)=ABSF(GD1(L))/GRD
      AL(2)=ABSF(GD2(L))/GRD
      AL(3)=ABSF(GD3(L))/GRD
      ALMN=1.0/(AL(1)*AL(2)*AL(3))
      IF(AL(1).GE.AL(2)) 522,521
  521 AP=AL(1)
      AL(1)=AL(2)
      AL(2)=AP
  522 IF(AL(2).GE.AL(3)) 527, 523
  523 IF(AL(3).GE.AL(1))525,524
C
  524 AP=AL(2)
      AL(2)=AL(3)
      AL(3)=AP
      GO TO 527
  525 AP=AL(3)
      AL(3)=AL(2)
      AL(2)=AL(1)
      AL(1)=AP
  527 CONTINUE
      W1=(-AL(1)+AL(2)+AL(3))*AHALF
      W2=(AL(1)-AL(2)+AL(3))*AHALF
      W3=(AL(1)+AL(2)-AL(3))*AHALF
      W4=(AL(1)+AL(2)+AL(3))*AHALF
C
C  CONSTANTS NEEDED
      ULMN = U*ALMN
      CON = 0.5*ULMN
      WSQ = W1**2
      AL23 = 4.0*AL(2)*AL(3)*ASQ
      B3 = CON*W4
      A3 = B3*W4
      C3 = ULMN/6.0
      B42 = ULMN*AHALF*AL(3)
      A42 = 2.0*B42*AHALF*(AL(1)+AL(2))
      A5 = CON*(2.0*AL23 - WSQ)
      B5 = CON*W1
      C5 = -C3
      A6 = ULMN*(AL23 - WSQ)
```

```
      A41=4.0*U*ASQ/AL(I)
      SUM = 0.0
C
      DO 900 JJJ = 1,7
      GO TO (751,752,753,754,755,756,757) JJJ
  751 A = A3
      B = B3
      C = C3
      WN = -W4
      WX = -W3
      FMAX = V(L) + WN*GRD
      NN2 = FMAX/DVA + 1.0
      AN2 = NN2 - 1
      N2 = NN2 + 50
      NMIN = N2
      BW = (FMAX - AN2*DVA)/GRD
      GO TO 770
C
  752 A = A42
      B = B42
      C = 0.0
      WX = -W2
      GO TO 770
C
  753 A = A5
      B = B5
      C = C5
      WX = -ABSF(W1)
      GO TO 770
C
  754 IF(W1) 4754,900,6754
 4754 A = A41
      B = 0.0
      C = 0.0
      WX = -W1
      GO TO 770
C
 6754 A = A6
      B = 0.0
      C = -ULMN/3.0
      WX = W1
      GO TO 770
C
  755 A = A5
      B = -B5
      C = C5
      WX = W2
      GO TO 770
C
  756 A = A42
      B = -B42
      C = 0.0
      WX = W3
      GO TO 770
C
  757 A = A3
      B = -B3
      C = C3
      WX = W4
C
  770 FMIN = FMAX
      FMAX = V(L) + WX*GRD
      N1 = N2
      NN2 = FMAX/DVA + 1.0
      AN2 = NN2 - 1
```

```
      N2 = NN2+50
      AW = DELW - BW
      BW = (FMAX - AN2*DVA)/GRD
      IF(N2-N1) 850,850,780
  780 NFRST = N1 + 1
      NLAST = N2 - 1
      W = WN + AW
C
      DELN = AW*(A + B*(WN+W) + C*(WN**2+WN*W+W**2))
      CTA(N1) = DELN + CTA(N1)
      SUM = SUM+DELN
C
C
      DO 800 KN = NFRST,NLAST
      WW = W + DELW
      DELN = DELW*(A + B*(W+WW) + C*(W**2+W*WW+WW**2))
      CTA(KN) = DELN + CTA(KN)
      SUM = SUM + DELN
  800 W = W + DELW
C
      WW = W + BW
      DELN = BW*(A + B*(W+WW) + C*(W**2+W*WW+WW**2))
      CTA(N2) = DELN + CTA(N2)
      SUM = SUM + DELN
      W = WW
      GO TO 900
C
  850 CW = WX - WN
      DELN = CW*(A + B*(WN+WX) + C*(WN**2+WN*WX+WX**2))
      CTA(N1) = DELN + CTA(N1)
      SUM = SUM + DELN
  900 WN = WX
C
      GO TO (20, 30) NPRINT
   30 SUM = SUM/L
      NDEL = N2 - NMIN + 1
 7654 DIFF = VOLUME - SUM
      PRINT 1000, V(L),GRD,AL(1),AL(2),AL(3),DELW,W1,VOLUME,DIFF,NDEL
C  THE VOLUME PRINTED HERE IS THE CORRECT VALUE . DIFF=CORRECT-SUMMED
C
   20 CONTINUE
      RETURN
      END


      SUBROUTINE WRITE (ARRAY,NDIM,DELF,NA,NB,NFMT)
C
      DIMENSION ARRAY(NDIM)
  101 FORMAT(1X,F10.5,7X,10F10.0)
  102 FORMAT(1X,F10.5,7X,10E10.3)
C
      NLINES = (NB-NA+9)/10
      N2 = NA-1
      FINCR=10*DELF
      FRQ = -FINCR
      DO 5 J=1,NLINES
      N1=N2+1
      N2=N1+9
      FRQ=FRQ+FINCR
      GO TO (1,2) NFMT
    1 PRINT 101, FRQ, (ARRAY(K), K=N1,N2)
      GO TO 5
    2 PRINT 102, FRQ,  (ARRAY(K), K=N1,N2)
    5 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE GNPLOT(G,NDIM,FMAX,DF,PLOTW,PLOTH,TITLE)
C
      DIMENSION G(NDIM),TITLE(10)
C
C     COMPUTE NECESSARY CONVERSION CONSTANTS ETC.
      NPOINTS = FMAX/DF
      XFACTOR = PLOTW/FMAX
C     YFACTOR TO BE COMPUTED LATER WHEN GMAX HAS BEEN FOUND
      NFMAX = FMAX
      DX = DF*XFACTOR
      YMARK = PLOTH/5.0
C
C     CALIBRATE Y-AXIS IN ARBRITARY UNITS (PLOT HEIGHT = 5.0)
      Y = - YMARK + 1.5
      DO 10 M = 1,6
      MY = M - 1
      Y = Y + YMARK
      CALL PLOT (0.0, Y, 3)
      CALL PLOT (-0.1, Y, 2)
   10 CALL NUMBER (-0.24, Y - 0.07, 0.14, MY,  0.0, 2H11)
      CALL SYMBOL(-1.0,PLOTH-2.5,0.2,23HG(NU) (ARBITRARY UNITS) ,90.,23)
C
C     DRAW Y-AXIS
      CALL PLOT (0.0, PLOTH + 1.8, 3)
      CALL PLOT ( 0.0, 1.5, 2)
C
C     WRITE OUT TITLE
C
      CALL SYMBOL(0.0,0.5,0.14,TITLE,0.0,80)
C
      IF (NFMAX - 5) 131, 131, 132
  131 INT = 1
      GO TO 135
  132 IF (NFMAX - 19) 133, 133, 134
  133 INT = 2
      GO TO 135
  134 INT = 5
  135 XMARK = XFACTOR*INT
      X = -XMARK
      MARK = -INT
      MAX = NFMAX + 1
      DO 100 N = 1,MAX,INT
      X = X + XMARK
      MARK = MARK + INT
      CALL PLOT (X,1.5, 3)
      CALL PLOT (X, 1.4, 2)
      IF ( MARK - 9) 141, 141, 142
  141 DWRITE = 0.16
      GO TO 100
  142 DWRITE = 0.1
  100 CALL NUMBER (X - DWRITE, 1.2, 0.14, MARK, 0.0, 2H12)
      CALL SYMBOL(PLOTW-3.16,0.8,0.2,14HFREQ (10  CPS) , 0.0,14)
      CALL SYMBOL (PLOTW - 1.8, 0.95, 0.1, 2H12, 0.0, 2)
C     DRAW X-AXIS
      CALL PLOT (PLOTW, 1.5, 3)
      CALL PLOT (0.0, 1.5, 2)
C
```

```
C     FIND GMAX AND SCALE ALL G-VALUES WHEN PLOTTING
      GMAX = 0.0
      DO 200 K = 1, NPOINTS
      IF (GMAX - G(K)) 199, 199, 200
  199 GMAX = G(K)
  200 CONTINUE
C
      YFACTOR = PLOTH/GMAX
      X = -DX/2.0
      CALL PLOT(0.0, 1.5, 3)
      DO 300 L = 1,NPOINTS
      X = X + DX
      Y = G(L)*YFACTOR + 1.5
  300 CALL PLOT(X,Y,2)
C     END OF GRAPH CALL
      CALL PLOT( PLOTW + 5.0, 0.0, -3)
      RETURN
      END
```

INTERNAL DISTRIBUTION

1. T. O. Baldwin
2. J. H. Barrett
3. D. S. Billington
4. M. A. Breazeale
5. J. H. Crawford
6. H. L. Davis
7. J. S. Faulkner
8. W. B. Gauster
9. G. Gilat
10. D. K. Holmes
11. R. B. Murray
12. R. M. Nicklow
13. U. Opik
14-73. L. J. Raubenheimer
74. M. T. Robinson
75. H. G. Smith
76. M. K. Wilkinson
77. E. O. Wollan
78-79. Central Research Library
80. Document Reference Section
81-85. Laboratory Records Department
86. Laboratory Records, ORNL R.C.
87. ORNL Patent Office
88-102. Division of Technical Information Extension
103. Research and Development Division, ORO

EXTERNAL DISTRIBUTION

104. A. D. B. Woods, Chalk River Nuclear Laboratories, Chalk River, Ontario, Canada
105. R. A. Cowley, Chalk River Nuclear Laboratories, Chalk River, Ontario, Canada
106. Bernard Mozer, Brookhaven National Laboratory, Upton, L. I., New York
107. J. A. Krumhansl, Cornell University, Ithaca, New York
108. G. O. Jones, Queen Mary College, London, England
109. Lothar Meyer, University of Chicago, Chicago, Illinois
110. G. K. Horton, Rutgers University, New Brunswick, New Jersey
111. G. Caglioti, CNEN al C.C.R. Euratom, Ispra, Italy
112. I. Pelah, Soreq Nuclear Establishment, Yavne, Israel
113. W. Cochran, University of Edinburgh, Edinburgh, Scotland
114. P. K. Iyengar, Atomic Energy Establishment Trombay, Bombay, India
115. John S. Brown, Rutgers University, New Brunswick, New Jersey
116. B. N. Brockhouse, McMaster University, Hamilton, Ontario, Canada
117. Walter Kley, Euratom, Ispra, Italy
118. G. L. Squires, Cambridge University, Cambridge, England
119. M. F. Collins, AERE, Harwell, Didcot, Berks., England
120. D. Cribier, C.E.N. de Saclay, Gif-sur-Yvette, France
121. R. Stedman, AB Atomenergi, Studsvik, Sweden
122. J. L. Yarnell, Los Alamos Scientific Laboratory, Los Alamos, New Mexico
123. R. O. Simmons, University of Illinois, Urbana, Illinois
124. G. Dolling, Chalk River Nuclear Laboratories, Chalk River, Ontario, Canada

125. Frank Herman, Lockheed Research Labs., Palo Alto, California
126. R. L. Jacobs, University of Chicago, Chicago, Illinois
127. M. Doyama, Argonne National Laboratory, Argonne, Illinois
128. A. E. Dixon, McMaster University, Hamilton, Ontario, Canada
129. S. K. Joshi, University of California, Riverside, California
130. S. K. Sinha, Iowa State University, Ames, Iowa
131. C. C. Lin, University of Oklahoma, Norman, Oklahoma
132. J. C. Phillips, University of Chicago, Chicago, Illinois
133. Volker Heine, University of Chicago, Chicago, Illinois
134. H. Ehrenreich, Harvard University, Cambridge, Massachusetts
135. A. R. Mackintosh, Iowa State University, Ames, Iowa
136. N. D. Lang, Harvard University, Cambridge, Massachusetts
137. L. F. Mattheis, Bell Telephone Laboratories, Murray Hill, New Jersey
138. A. J. Freeman, Massachusetts Institute of Technology, Cambridge, Massachusetts
139. H. B. Rosenstock, U. S. Naval Research Laboratory, Washington, D. C.
140. F. W. de Wette, University of Texas, Austin, Texas
141. R. M. J. Cotterill, Argonne National Laboratory, Argonne, Illinois
142. J. R. Hardy, AERE, Harwell, Didcot, Berks., England
143. W. B. Daniels, Princeton University, Princeton, New Jersey
144. R. E. Schmunk, U.S. AEC Idaho Operations Office, Idaho Falls, Idaho
145. A. J. Leadbetter, Bristol University, Bristol, England
146. E. R. Cowley, University of Edinburgh, Edinburgh, Scotland
147. Robert Shamu, Argonne National Laboratory, Argonne, Illinois
148. P. Egelstaff, AERE, Harwell, Didcot, Berks., England
149. Eric Svensson, McMaster University, Hamilton, Ontario, Canada
150. M. Rowe, McMaster University, Hamilton, Ontario, Canada
151. Y. N. Lwin, Western Illinois University, Macomb, Illinois
152. J. P. Hugo, South African Atomic Energy Board, Private Bag 256, Pretoria, Tvl, South Africa
153. J. R. Clem, University of Maryland, College Park, Maryland
154. Edward Metzbower, Latrobe Hall, Johns Hopkins Univ., Baltimore, Maryland
155. Elisha Cohen, Department of Physics, Johns Hopkins University, Baltimore, Maryland
156. D. L. McDonald, Materials Physics Section, A.A.E.C. Research Establishment, Private Mail Bag, Sutherland, N.S.W., Australia