NEW APPROACHES TO DIRECT METHODS IN X-RAY CRYSTALLOGRAPHY

by

Stephen Robert Brown

A thesis submitted to the University of Glasgow for the degree of Doctor of Philosophy in the Faculty of Science.

Chemistry Department November 1987

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

To Karen

ACKNOWLEDGMENTS

The three years spent working on the research documented in this thesis has been a stimulating and enjoyable period for me. This has largely been due to some of the people I have come into contact with during this time and I would like to take this opportunity to pay tribute to them.

In particular I would like to express my sincere thanks to my supervisor, Chris Gilmore, for the encouragement, advice and ideas he has given me in the course of this project. I am indebted also to Andy Freer, Kelvin Tyler and Paul Mallinson for their assistance with theoretical and technical matters. In addition I am grateful to my other friends in the Chemistry Department, most notably Fiona Hayes, David Adam and, latterly, Colin Bannister for their help and cooperation on various topics.

Finally, thanks are due to the S.E.R.C. for funding this research and to the University of Glasgow who, together with the S.E.R.C., paid expenses for me to make four trips to LURE, Paris.

CONTENTS

SUMMARY

This thesis is concerned with the use of direct methods to solve crystal structures from X-ray data. Particular interest is centred upon why certain structures should be difficult to determine and what steps can be taken to remove or overcome such problems. Throughout the thesis use is made of the direct methods computer program MITHRIL. Many of the new techniques discussed in this thesis have now been incorporated into the program.

The first chapter contains an introduction to direct methods - it lays the groundwork for the techniques used elsewhere in the thesis. As well as explaining the theory of direct methods the chapter also includes a discussion of why direct methods fail in certain cases.

Direct methods are based on a few simple phase relationships. From the point of view of solving the phase problem it is obviously vital that these relationships are accurate. Chapter 2 is concerned with making phase relationships more reliable. The chapter is in two parts: the first part deals with schemes based on the $(\sin^2\theta/\lambda^2)$ values of reflections participating in phase relationships while the second part investigates quartet reliability in relation to the frequency of three new cross terms. Although the $(\sin^2\theta/\overset{2}{\lambda})$ based schemes are very simple they are shown to be highly effective and extremely useful. Rather disappointingly no link could be established between quartet reliability and the occurrence of the new cross terms.

Chapter 3 discusses the implementation of a symbolic addition module, LSAM, into the MITHRIL system. In its original form LSAM was among the first complete systems of

computer programs devised for the automatic solution of crystal structures. MITHRIL now contains a substantially upgraded form of LSAM, however, the method remains essentially unaltered. A drawback of LSAM is that it deals only with centrosymmetric structures but, even so, it still provides an alternative to the multisolution method already present in MITHRIL. It is shown that the introduction of LSAM has been very successful and, in particular, that it is often highly advantageous to use symbolic addition when dealing with heavy atom structures.

The results and conclusions of using MITHRIL to attempt to solve all twenty five structures on the database of structures resistant to direct methods compiled by Sheldrick are given in Chapter 4. Detailed accounts of the strategies and tactics employed on each structure are given and the results are summarised in tabular form. Conclusions are drawn on the substituent parts of MITHRIL that performed well and those that did not. MITHRIL has already been improved in the light of the experience gained in this chapter and two new options, REVIEW and MISS have been added to the program. REVIEW is a module which enables the best phase sets produced by the program to be identified more readily. The MISS command, located in the normalisation module, adds missing weak reflections to incomplete data sets and assigns structure factors to them in accordance with their ($\sin\theta/\lambda$) values. In addition, advice is provided on what steps to take when confronted with a difficult structure.

The final step in a direct methods analysis is the calculation and interpretation of an E-map. When dealing with difficult structures the E-maps are often complex and difficult to interpret. Chapter 5 is concerned with the use of colour computer graphics routines to aid the interpretation. Two separate modules, MITHGRAPH and CONTOUR, have been written for use in conjunction with MITHRIL.

MITHGRAPH performs a series of display and edit routines as well as allowing the user to manipulate molecular fragments in various ways. As the name suggests, CONTOUR, displays contoured sections of an E-map or weighted Fourier map so allowing direct observation of the three dimensional map. It is shown how the contouring routine was of great value in the determination of two of the difficult structures from the database mentioned above. The final chapter also discusses SUBMIT, a program which runs on a colour graphics workstation - it incorporates both MITHGRAPH and CONTOUR modules. SUBMIT can perform Fourier recycling of known fragments, E-map and weighted Fourier map calculations as well as allowing direct observation of previously stored E-maps. Instructions for using MITHGRAPH, CONTOUR and SUBMIT are included in the chapter.

CHAPTER1

INTRODUCTION TO DIRECT METHODS

INTRODUCTION TO DIRECT METHODS

1.1 THE PHASE PROBLEM

An X-ray diffraction experiment gives, amongst other things, information on:

(1) The shape and size of the unit cell (from the geometry of diffraction).

(2) The symmetry within the crystal (from the symmetry of the diffraction pattern plus the possible presence of systematically absent reflections and the physical properties of the crystal)

(3) The position of the atoms within the unit cell.

The latter information is obviously the most important and practical, but whereas the results in (1) and (2) come immediately from data which is, in general, experimentally measurable, it is a far from trivial exercise to extract the atomic positions from X-ray data.

The intensity of a reflection of index $\underline{h}$, $I_{\underline{h}}$, depends on the positions of the atoms in the unit cell. If the atomic positions are known then the intensities can be calculated. However it is the inverse problem that needs to be solved. That is, to find the atomic positions given the intensities.

To start with we define the structure factor as follows:

$$F_{\underline{h}} = \left|F_{\underline{h}}\right| \exp(i\phi_{\underline{h}}) \qquad (1.1)$$

where $\left|F_{\underline{h}}\right|$ is the structure amplitude and $\phi_{\underline{h}}$ is its phase.

If the observed amplitude is corrected for geometrical factors and scaled, then the structure amplitude can be found from the experimental data as:

$$\left|F_{\underline{h}}\right| = (I_{\underline{h}})^{1/2} \qquad (1.2)$$

The atomic arrangement in the unit cell can be found if the electron density within it is known. At the resolution of X-ray scattering experiments the atomic centres are at the peaks of roughly spherical blobs of electron density.

No X-ray lens is available to form this image but it is possible to calculate the electron density at any point in the unit cell using the equation:

$$\rho(\underline{x}) = \frac{1}{V} \sum_{\underline{h}} F_{\underline{h}} \cos[2\pi \underline{h}.\underline{x} - \phi_{\underline{h}}] \qquad (1.3)$$

where $\underline{x}$ is the vector notation for a point in the unit cell and V is the volume of the unit cell. The summation is over all the observed reflections.

The only unknowns on the right hand side of this equation are the phases, so the problem of determining the crystal structure is essentially that of determining the phases: this constitutes the phase problem in X-ray crystallography. In order to reconstruct the electron density function with the information obtained from X-ray diffraction, the phase of each of the structure factors involved must be deduced.

Note that in the centrosymmetric case the phases are restricted to values of either 0 or π. This leads to the use of 'signs' in structure determination: a sign of '+' corresponds to a phase angle of 0 whereas a sign of '-' is equivalent to an angle of π.

Several methods have been developed to overcome the phase problem: the most frequently used being vector (or Patterson) methods and direct methods. Added to this, a lot of interest has recently been focussed upon applying the maximum entropy method of image reconstruction to the crystallographic phase problem[1,2].

In vector methods the Patterson function:

$$P(\underline{x}) = \frac{1}{V} \sum_{\underline{h}} \left|F_{\underline{h}}\right|^2 \exp[-2\pi i \underline{h}.\underline{x}] \qquad (1.4)$$

is used. The maxima in the Patterson function are used to find the positions of one or more heavy atoms, if present, or to locate a part of the structure with known geometry. From this partly solved structure the rest can be found by making use of various refinement techniques.

Patterson methods, however, cannot be used when all the atoms of a structure are of roughly the same size and there is no known fragment. Direct methods are not limited in this way and is undoubtedly the more powerful of the two methods. This is borne out by the fact that direct methods are used to solve the majority of structures today.

Readers interested in the history of direct methods are referred to two comprehensive reviews written by Woolfson[3] and Viterbo[4] respectively.

1.2 DIRECT METHODS

Methods which attempt to predict the phases of the structure factors directly by mathematical means are known as "direct methods". Direct methods can, in most cases, reduce the phase problem to an objective procedure in which any decisions are of a purely mathematical nature. As a consequence direct methods are now almost exclusively applied by the use of computer programs. Indeed it is true to say that the development of direct methods has been catalysed by the great advances made in the field of computer technology over the last thirty years.

Direct methods can be highly automated and solve most structures routinely. However, it is worth pointing out that in the case of difficult structures, human judgement and interpretation can often improve the methods.

The first quantitative relations between the phases of the structure factors were given by Harker & Kasper[5]. By application of Cauchy-Schwartz inequality relationships to the structure factor equation they proved (for centrosymmetric structures) that when $|F_{\underline{h}}|$ and $|F_{2\underline{h}}|$ are both strong the sign of $|F_{2\underline{h}}|$ is likely to be positive.

These methods were used to solve a few simple crystal structures. However, as the complexity of the crystal increases, it becomes necessary to bring in probability methods as inequalities alone impose restrictions only on phases whose intensities are very large. Karle and Hauptman[6,7] developed the concept of probable relationships between phases of reflections - it was largely because of this work that they were awarded the Nobel Prize for Chemistry in 1985.

An algebraic basis for direct methods can be seen in the Sayre equation[8]:

$$F_{\underline{h}} = \frac{1}{V}\frac{f}{g}\sum_{\underline{k}} F_{\underline{k}}F_{\underline{h}-\underline{k}} \qquad (1.5)$$

where f is the scattering factor for the real atom, g is the common scattering factor of the squared atom and V is the unit cell volume. This equation, however, makes the assumption that all of the atoms are equal. In order to make use of the fact that the atoms are discrete it is necessary to introduce normalised structure factors.

1.3 NORMALISED STRUCTURE FACTOR, E

It can be seen from Sayre's equation (1.5) that structure factors with a large diffraction angle θ, play a less important role than structure factors with a small θ, because $F_{\underline{h}}$ decreases when θ increases. Therefore in direct methods the normalised structure factor, $E_{\underline{h}}$, is used rather than $F_{\underline{h}}$. The E's are structure factors corresponding to the scattering which would be obtained from point atoms at rest, normalised to $\langle |E_{\underline{h}}|^2 \rangle = 1$. Since for point atoms there is no decrease in diffracted intensity with increasing scattering angle, expressions using $E_{\underline{h}}$'s are independent of θ. When formulated in terms of $E_{\underline{h}}$'s equation (1.5) becomes:

$$E_{\underline{h}} = N^{1/2}\langle E_{\underline{k}}E_{\underline{h}-\underline{k}}\rangle_{\underline{k}} \qquad (1.6)$$

where N is the number of atoms in the unit cell and the average is over all measured intensities $\underline{k}$.

Structure factors are normalised using the equation:

$$E_{\underline{h}}^2 = k \left|F_{\underline{h}}\right|^2 / \varepsilon \sum_j f_j^2 \exp(-2B\sin^2\theta/\lambda^2) \qquad (1.7)$$

where

k - scale factor

ε - factor for point group symmetry

f_j - scattering factor for atom j

B - temperature factor

λ - wavelength of radiation used

An important point to note is that the phase of a reflection is not altered in any way in going from $F_{\underline{h}}$ to $E_{\underline{h}}$.

The computation of E-magnitudes is a highly important process since it is the starting point for all direct methods calculations. All of the procedures which follow normalisation of the structure factors are directly influenced by the E-magnitudes. Factors which affect the actual values of the $E_{\underline{h}}$'s should, therefore, be closely considered.

As well as the normalised structure factor, $E_{\underline{h}}$, direct methods also make use of the unitary structure factor, $U_{\underline{h}}$, where $U_{\underline{h}}$ lies in the range zero to one. Unitary structure factors can be computed from E-magnitudes using the equation:

$$\left|U_{\underline{h}}\right| = N^{(-1/2)} \left|E_{\underline{h}}\right| \qquad (1.8)$$

Note that since this equation involves N, the number of atoms in the unit cell, the unitary structure factor contains information on structure complexity.

1.3.1 TEMPERATURE FACTOR, B AND SCALE FACTOR, K

Atoms in crystals vibrate about their rest points. The extent of the vibration is influenced greatly by the temperature. Thermal vibration causes the electron cloud surrounding an atom to spread out over a larger volume. This means that the scattering power of the real atom diminishes more rapidly than in the ideal, stationary model. The change in scattering power is given by:

$$f = f_o \exp(-B\sin^2\theta/\lambda^2) \qquad (1.9)$$

where f is the ideal and f_o the observed scattering factor for a point atom. The temperature factor, B, is related to the mean square amplitude ($\bar{U}^2$) of atomic vibration by:

$$B = 8\ ^2\bar{U}^2 \qquad (1.10)$$

It can be shown that the average observed intensity in a region around a particular value of θ is of the form:

$$\langle I\rangle_\theta = \sum_{j=1}^{N} f_j^2 \exp(-2B\sin^2\theta/\lambda^2) \qquad (1.11)$$

In order to obtain E values using equation (1.7) it is necessary to know the values of B and K. This is achieved by first transforming (1.11) as follows:

$$\log\left[\frac{\langle I\rangle_\theta}{\sum_{j=1}^{N} f_j^2}\right] = \log K - (2B\sin^2\theta/\lambda^2) \qquad (1.12)$$

The left hand side of (1.12) plotted against $(\sin^2\theta/\lambda^2)$ constitutes what is known as a Wilson plot and should be an approximate straight line.

1.4 ORIGIN DEFINITION

To define a crystal structure completely the atomic positions must be specified with respect to a fixed frame of reference. This involves setting up a system of axes around an origin. Normally there is a choice of permissible origin positions since space group symmetry usually determines the direction of the axes but not their absolute positions. Moving the origin changes the phases of the structure factors (but not their magnitudes), so the selection of a particular origin imposes a corresponding pattern of relative phases on the diffraction pattern.

The structure factor is defined in terms of atomic coordinates as:

$$F_{\underline{h}} = \sum_j f_j \exp(2\pi i \underline{h}.\underline{x}_j) \qquad (1.13)$$

If the origin is moved a distance Δx this expression changes to:

$$F'_{\underline{h}} = \sum_j f_j \exp 2\pi i \underline{h}.(\underline{x}_j - \Delta\underline{x}) \qquad (1.14)$$

$$= F_{\underline{h}} \exp(-2\pi i \underline{h}.\Delta\underline{x}) \qquad (1.15)$$

The phase of $F_{\underline{h}}$ is therefore changed by $-2\pi\underline{h}.\Delta\underline{x}$ when the origin is moved a distance $\Delta\underline{x}$.

For example, in space group $P\bar{1}$ the origin is conventionally chosen at one of the eight centres of symmetry in the unit cell differing in coordinates x, y and z by 0.5 or 0.0 fractions. If, for example the origin is moved from (0,0,0) to (0.5,0,0) then the phases of all reflections of the form (hkl) with h odd will change by an odd multiple of π (a reverse of sign), whereas those with h even will change by an even multiple of π (retain the same sign). This means reflections can be put into categories according to whether or not their signs depend upon origin position in the x, y or z direction.

1.5 STRUCTURE INVARIANTS

A structure invariant is a quantity, the value of which is uniquely defined by the structure and remains unchanged when the origin of the unit cell is shifted. A simple example of a structure invariant is the intensity of a diffracted beam, $I_{\underline{h}}$ (or $F^2_{\underline{h}}$). However, as shown above the structure factor, $F_{\underline{h}}$ (or $E_{\underline{h}}$) is not itself a structure invariant, otherwise it could be measured and the phase problem would not exist.

The value of structure invariants lies in the fact that they do not depend on choice of origin but only on atomic positions. One of the most crucial results in the use of direct methods is the fact that the product $E_{\underline{h}}E_{\underline{k}}E_{-\underline{h}-\underline{k}}$ is an invariant.

$$E'_{\underline{h}}E'_{\underline{k}}E'_{-\underline{h}-\underline{k}} = E_{\underline{h}}\exp 2\pi i\underline{h}\cdot\Delta\underline{x}E_{\underline{k}}\exp 2\pi i\underline{k}\cdot\Delta\underline{x}E_{-\underline{h}-\underline{k}}\exp 2\pi i(-\underline{h}-\underline{k})\cdot\Delta\underline{x}$$

$$= E_{\underline{h}}E_{\underline{k}}E_{-\underline{h}-\underline{k}}$$

Since the moduli of the structure factors are invariants themselves, the sum of the three phases:

$$\phi_{\underline{h}} + \phi_{\underline{k}} + \phi_{-\underline{h}-\underline{k}} \qquad (1.16)$$

is also an invariant. In general, if the sum of the reflection indices equals zero the corresponding product of the structure factors is a structure invariant. As well as the triple phase structure invariant two other invariants are of particular use in direct methods:

$$\underline{\text{Quartet}}: \quad \phi_{-\underline{h}} + \phi_{\underline{k}} + \phi_{\underline{l}} + \phi_{\underline{m}} \qquad (1.17)$$

where $\underline{h}+\underline{k}+\underline{l}+\underline{m} = 0$ and

$$\underline{\text{Quintet}}: \quad \phi_{-\underline{h}} + \phi_{\underline{k}} + \phi_{\underline{l}} + \phi_{\underline{m}} + \phi_{\underline{n}} \qquad (1.18)$$

where $\underline{h}+\underline{k}+\underline{l}+\underline{m}+\underline{n} = 0$

1.6 STRUCTURE SEMINVARIANTS

Seminvariants originate from using the restrictions imposed by space group symmetry when defining the origin. Many linear combinations of phases, in addition to the universal structure invariants previously discussed, will remain unchanged in value when the origin is shifted only in

the restricted ways allowed by space group symmetry. For example, in the space group $P2_1$ the linear combination of the three phases:

$$\phi_{h_1k_1l_1} + \phi_{h_2k_2l_2} + \phi_{h_3k_3l_3} \qquad (1.19)$$

is a seminvariant only if $(k_1+k_2+k_3)$ equals zero, and $(h_1+h_2+h_3)$ and $(l_1+l_2+l_3)$ are even integers.

Of particular use in direct methods are structure seminvariants which involve only one phase - these are called $\sum_1$ relationships. They are used in the phase determination process and also in the calculation of some figures of merit. $\sum_1$'s are derived from structure invariants and seminvariants by making use of space group specific relationships between reflections. In space group $P\bar{1}$, for example the relationship:

$$S(E_{2\underline{h}}) \approx S(|E_{\underline{h}}|^2 - 1) \qquad (1.20)$$

holds. It can be extended to include space group symmetry as in $P2_1$:

$$S(E_{2h},0,E_{2l}) \approx \frac{1}{N^{1/2}} \langle (-1)^k (|Ehkl|^2 - 1) \rangle_k \qquad (1.21)$$

Another example of a $\sum_1$ relationship occurs in space group $P2_12_12_1$ where if a reflection (hkl) has a phase angle of α then the reflection with indices $(h\bar{k}l)$ will have a phase angle of $\pi-\alpha$. If two such reflections are involved in a triple phase invariant then it is possible to obtain a phase value for the remaining reflection. To illustrate this principle suppose that in $P2_12_12_1$ the reflections (0 3 4) and (0 $\bar{3}$ 4) are involved in an invariant such that:

$$\phi_{034} + \phi_{0\bar{3}4} + \phi_{00\bar{8}} + \pi \approx 0$$

and that:

$$\phi_{034} = \alpha$$

By space group symmetry the phase of reflection (0 $\bar{3}$ 4) is $\pi-\alpha$ and substitution into the equation above yields a value of 0 for the phase of reflection (0 0 $\bar{8}$). It is often possible to obtain phase indications by making use of relationships such as that just illustrated.

Note that the invariants are a subset of the seminvariants and that there are far more seminvariants than invariants.

The occurrence of structure invariants and seminvariants lends itself to the rule that the reflections used to define the origin must not combine together to give a structure invariant or seminvariant. This means that reflections used in origin definition must be linearly independent.

1.7 ENANTIOMORPH DEFINITION

In addition to fixing the origin it is also necessary, for non-centrosymmetric structures, to define the enantiomorph. If a phase has a value of ϕ for a given structure, it will have a value of $-\phi$ for the enantiomorphic structure. The signs of all structure invariants and seminvariants will be similarly reversed. Therefore, once the origin has been defined, the sign of an invariant (or seminvariant) which differs from 0 or π can be arbitrarily assigned. Having done this all invariants and seminvariants

become uniquely defined. For efficient enantiomorph definition this invariant should involve large E-magnitudes.

In theory the enantiomorph can be defined by assigning a sign to a particular structure invariant. In practice, however, it is usually sufficient to define the enantiomorph by assigning a sign to the phase of a single reflection. This reflection should participate in invariants whose phase sums are neither 0 or Π.

1.8 THE NEIGHBOURHOOD PRINCIPLE

The phase sum of a structure invariant is uniquely determined by the E-magnitudes. It has become clear that for fixed enantiomorph, the phase sum is more sensitive to one or more small sets of E-magnitudes than to the great bulk of remaining E-magnitudes. These sets are called neighbourhoods. The phase sum is most sensitive to the E-magnitudes of the first neighbourhood, followed by those of the second neighbourhood, then those of the third and so on. The neighbourhoods are nested so, for example, the second neighbourhood will contain all the E-magnitudes of the first neighbourhood as well as some additional ones.

Provided the E-magnitudes of the neighbourhoods are known, an estimate of the phase sum can be obtained from its conditional probability distribution. This estimate is particularly good in the favourable case where the variance of the distribution is small.

1.9 PHASE RELATIONSHIPS

The values of all invariants and seminvariants are uniquely defined by the structure; however, for the majority of them, it is not possible to predict their values. Fortunately there are classes of structure invariants and seminvariants, for which it is possible to get a reliable estimate of the phase sum. These relationships, known as phase relationships, play a key role in the use of direct methods to solve the phase problem.

There now follows a brief description of some of the more important phase relationships including the use of probability to estimate their values.

1.9.1 TRIPLETS

Triplets are three phase structure invariants of the form:

$$\Phi_3 = \phi_{\underline{h}} + \phi_{\underline{k}} + \phi_{\underline{l}} \qquad (1.22)$$

where the sum of the indices $\underline{h}$, $\underline{k}$ and $\underline{l}$ is zero.

For centrosymmetric structures this phase sum can have values of 0 or Π, commonly referred to as "signs" of - or + respectively. In 1952 Sayre[8] stated that when the E-magnitudes involved are large, the following relationship holds:

$$S(E_{\underline{h}}) = S(E_{\underline{k}} E_{\underline{l}}) \qquad (1.23)$$

where S means "sign of". Cochran & Woolfson[9] derived, under the assumption that atoms were randomly distributed within the unit cell, that the probability of:

$$S(E_{\underline{h}})S(E_{\underline{k}})S(E_{\underline{l}}) = +1 \qquad (1.24)$$

is given by

$$P = 1/2 + 1/2\tanh(\aleph/2) \qquad (1.25)$$

where

$$\aleph = 2\sigma_3\sigma_2^{-3/2}\left|E_{\underline{h}}E_{\underline{k}}E_{\underline{l}}\right|$$

and

$$\sigma_n = \sum_{j=1}^{N} Z_j$$

If there are several indications to the sign of $S(E_{\underline{h}})$ then (1.23) becomes:

$$S(E_{\underline{h}}) = S(\sum E_{\underline{k}} \cdot E_{\underline{l}}) \qquad (1.26)$$

with associated probability:

$$P = 1/2 + 1/2\tanh(\sum \aleph/2) \qquad (1.27)$$

For the non-centrosymmetric case Cochran[10] derived the corresponding probability distribution:

$$P(\phi_3|\aleph) = \frac{\exp(\aleph\cos(\phi_3))}{2\pi I_o(\aleph)} \qquad (1.28)$$

where I_o is a modified zero order Bessel function of the first kind.

Graphs of (1.28) are shown in Figure 1.1 for $\aleph$ values of 2.316 and 0.731. These distributions always have a maximum at 0^o and a minimum at 180^o. It is clear that the larger the value of the product $E_{\underline{h}}E_{\underline{k}}E_{\underline{l}}$, the smaller is the variance of the distribution.

The first neighbourhood of a triplet is made up of the E-magnitudes of the principal terms ($|E_{\underline{h}}|$, $|E_{\underline{k}}|$ and $|E_{\underline{l}}|$) while the second neighbourhood is formed by consideration of the quintet:

$$\Phi_5 = \phi_{\underline{h}} + \phi_{\underline{k}} + \phi_{\underline{l}} + \phi_{\underline{p}} + \phi_{-\underline{p}} \qquad (1.29)$$

where $|E_{\underline{p}}|$ is large. The six unique cross terms $|E_{\underline{h}+\underline{p}}|$, $|E_{\underline{h}-\underline{p}}|$, $|E_{\underline{k}+\underline{p}}|$, $|E_{\underline{k}-\underline{p}}|$, $|E_{\underline{l}+\underline{p}}|$ and $|E_{\underline{l}-\underline{p}}|$ define the second neighbourhood of the triplet. The second neighbourhood is used in several formulae for the estimation of quartet reliabilty - this is discussed in Chapter 2.

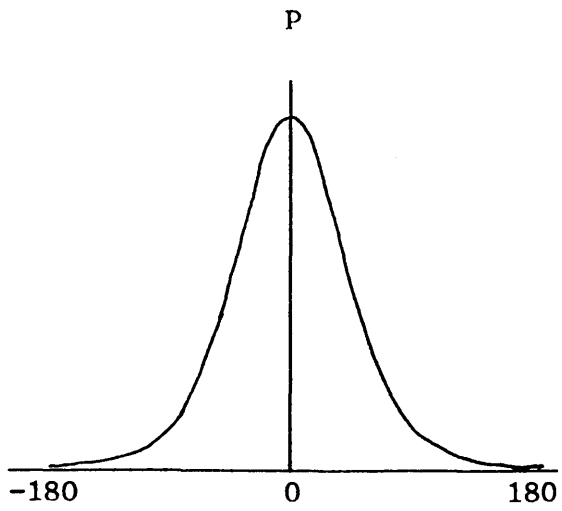
1.9.2 QUARTETS

The quartet phase relationship is of the form:

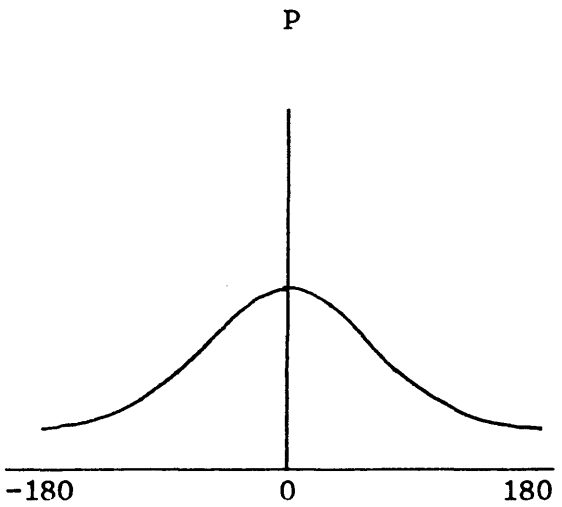
$$\Phi_4 = \phi_{\underline{h}} + \phi_{\underline{k}} + \phi_{\underline{l}} + \phi_{\underline{m}} \qquad (1.30)$$

where $\underline{h}+\underline{k}+\underline{l}+\underline{m} = 0$.

This relationship is extremely important in solving the phase problem in space groups where there is no



χ=2.316



χ= 0.731

Figure1.1 Probability distribution for triplet phase relationships

translational symmetry.

The first neighbourhood consists of $|E_{\underline{h}}|$, $|E_{\underline{k}}|$, $|E_{\underline{l}}|$ and $|E_{\underline{m}}|$, whilst the second neighbourhood adds the cross terms $|E_{\underline{h}+\underline{k}}|$, $|E_{\underline{h}+\underline{l}}|$ and $|E_{\underline{k}+\underline{l}}|$. The addition of the third neighbourhood is accomplished by introducing an arbitrary vector $\underline{p}$ and its associated vector $\underline{q}$ such that:

$$\underline{h} + \underline{k} + \underline{p} + \underline{q} = 0$$

It is necessary that both $E_{\underline{p}}$ and $E_{\underline{q}}$ are large. This has produced a second quartet invariant:

$$\Phi_{\underline{pq}} = \phi_{\underline{h}} + \phi_{\underline{k}} + \phi_{\underline{p}} + \phi_{\underline{q}} \qquad (1.31)$$

Indirectly, a third quartet invariant has been defined:

$$\Phi_{\underline{lm}} = \phi_{\underline{l}} + \phi_{\underline{m}} + \phi_{-\underline{p}} + \phi_{-\underline{q}} \qquad (1.32)$$

The quartet (1.31) has a second neighbourhood comprising of $|E_{\underline{h}}|$, $|E_{\underline{k}}|$, $|E_{\underline{p}}|$, $|E_{\underline{q}}|$, $|E_{\underline{h}+\underline{k}}|$, $|E_{\underline{k}+\underline{p}}|$ and $|E_{\underline{h}+\underline{p}}|$ while quartet (1.32) has a second neighbourhood made up of $|E_{\underline{l}}|$, $|E_{\underline{m}}|$, $|E_{\underline{p}}|$, $|E_{\underline{q}}|$, $|E_{\underline{l}+\underline{m}}|$, $|E_{\underline{m}-\underline{p}}|$ and $|E_{-\underline{p}+\underline{q}}|$. An identity exists between (1.30), (1.31) and (1.32) such that:

$$\Phi_4 + \Phi_{\underline{pq}} + \Phi_{\underline{lm}} = 0 \qquad (1.33)$$

so that Φ_4 can be estimated not only by the 7 E-magnitudes comprising its second neighbourhood, but also by the two invariants (1.31) and (1.32). A total of 21 E-magnitudes are now involved in the estimation of Φ_4, of which only 13 are unique. These 13 E-magnitudes define the third neighbourhood. The three neighbourhoods are illustrated in Figure 1.2. The three quartets (1.30), (1.31) and (1.32) define a trio.

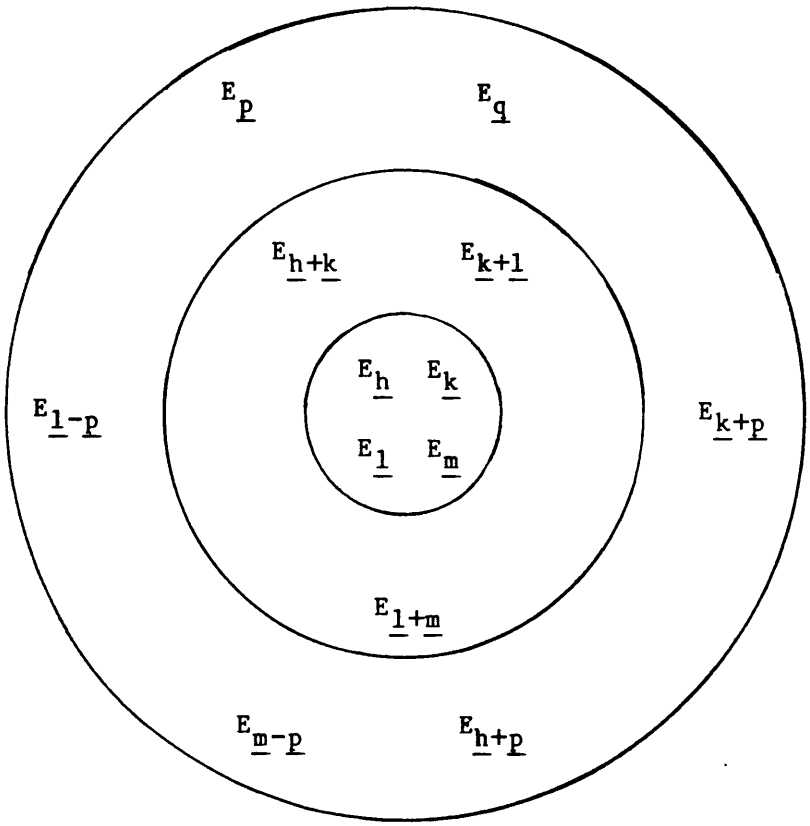


Figure 1.2 Quartet Neighbourhoods

The value of Φ_4 can be reliably estimated when the product $|E_{\underline{h}}E_{\underline{k}}E_{\underline{l}}E_{\underline{m}}|$ is large, and is dependent on the E-magnitudes of the cross terms $|E_{\underline{h}+\underline{k}}|$, $|E_{\underline{h}+\underline{l}}|$ and $|E_{\underline{k}+\underline{l}}|$. Three types of quartet can be distinguished: that for which $\cos(\Phi_4)$ is estimated to be zero (a positive quartet), one in which $\cos(\Phi_4)$ is estimated to be -1 (a negative quartet), and the enantiomorph sensitive relationships between these two extremes.

It has been shown by Schenk[11] that when the magnitudes of the 3 cross term E's are large, Φ_4 probably equals zero. Such positive quartets do contain useful information but are not widely used because they are highly correlated with the triplets. More important in direct methods, however is the case where the E's of the cross terms are all small. These are the so called "negative quartets". In this case ϕ_4 probably has a value around π(Hauptman[12]). Negative quartets can be used both as a figure of merit NQEST[13] and in an active mode to generate new phases[14]. Hauptman[15] has also shown that phase sums of $\mp\pi/2$ can be predicted from intermediate values of the cross terms. Figure 1.3 illustrates the associated probability distributions for the separate cases where the cross terms have large, small and, lastly, intermediate values.

1.9.3 QUINTETS

Quintet invariants take the form:

$$\Phi_5 = \phi_{\underline{h}} + \phi_{\underline{k}} + \phi_{\underline{l}} + \phi_{\underline{m}} + \phi_{\underline{n}} \qquad (1.34)$$

where:

$$\underline{h} + \underline{k} + \underline{l} + \underline{m} + \underline{n} = 0$$

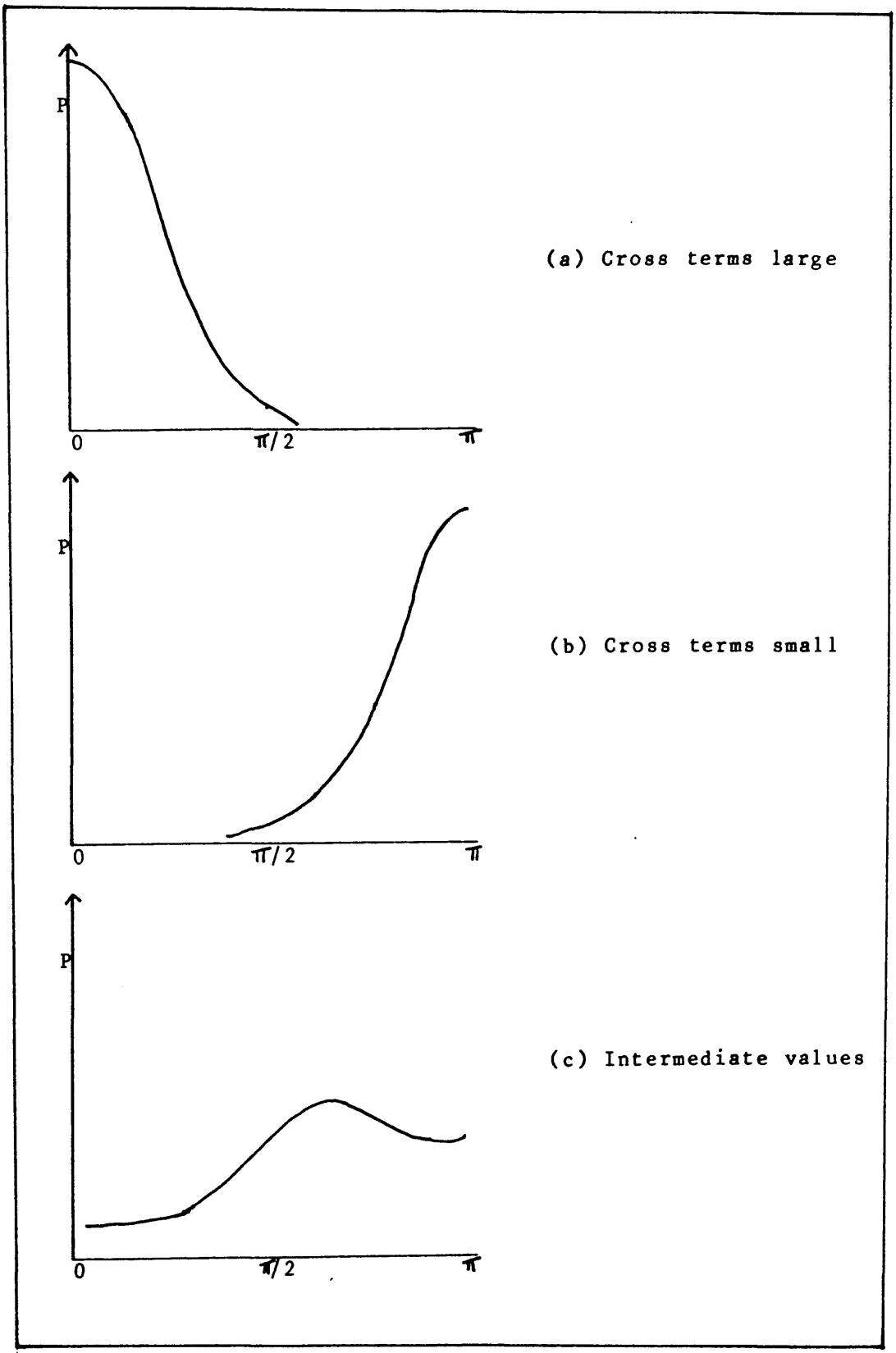


Figure 1.3 Probability distributions for the three types of quartet invariants

The first neighbourhood consists of the five principal terms $E_{\underline{h}}$, $E_{\underline{k}}$, $E_{\underline{l}}$, $E_{\underline{m}}$ and $E_{\underline{n}}$ whilst the second neighbourhood consists of the ten unique cross terms. As yet there are no completely reliable quintet distributions. However, quintets can be valuable in situations where negative invariants are needed but where quartets are difficult to generate - this arises because the second neighbourhood of a quintet involves 10 cross terms not all of which need to be small, in contrast to the equivalent quartet formula which requires three small cross terms. Probability distributions for negative quintets have been proposed by Giacovazzo[16], Hauptman and Fortier[17] and also by Bricogne[2] whose distribution is derived by maximum entropy methods.

1.10 TWO APPROACHES TO DIRECT METHODS

Direct methods computer programs can be split into two distinct categories: those such as MULTAN[18], SHELX[19], GENTAN[20] and MITHRIL[21] (in its original form) based on the multisolution principle[22] and those such as SIMPEL[23] based on the symbolic addition procedure[24,25]. To begin with the two techniques are identical in the respect that both involve normalisation of structure factors, generation of invariants and production of a starting set of phases to be used to find new phases. It is in the processes of initial phasing and phase extension that the techniques differ markedly.

1.11 THE STARTING SET

In order to solve the phase problem it is necessary to define a starting set of reflections that will be used to generate new phases. This set consists of the origin and (where relevant) enantiomorph defining reflections, the $\sum_1$ determined phases together with a small number of reflections of unknown phase which are to be used in a phase permutation or symbolic addition method.

The convergence procedure of Germain, Main & Woolfson[26] is used to find the starting set in such a way that leads to strong phase development and multiple indications. Convergence mapping is not a robust procedure and small changes in the E-magnitudes can change the map quite noticeably. Originally convergence mapping used triplet phase relationships only but Freer and Gilmore[14] have shown that the process can be extended to include other invariants and seminvariants.

1.12 INITIAL PHASING

The simplest form of initial phasing is the phase permutation method in which a set of phases, given ranges of values are permuted. The method is enhanced using magic integer sequences[27]. Alternatives to this within the multisolution class of methods include random phasing techniques which can be used either with linear equations and least squares[28] or with the tangent formula. The permutation method will be considered in more detail as it is the standard

and most frequently used procedure.

To make use of the derived triplet (and, where applicable, quartet and quintet) phase relationships, a number of reflections with known phases are required in order to obtain new phase information. However, only the phases of the origin and enantiomorph reflections are known explicitly and it is necessary to add a few reflections with unknown phases to the starting set. In the multisolution process, the unknown reflections are, each in turn, represented by a series of 4 different numerical values. These values are assigned by use of magic integer theory[29] which is described in the following section.

The phases of these variable reflections are permuted until the correct sequence is found. This means, for example, if there are n variable reflections in the starting set then 4^n phase sets would be produced. It is therefore desirable to keep the number of starting set reflections to a minimum, since the computing time needed to generate a single phase set is not negligible.

1.13 MAGIC INTEGER PHASE PERMUTATION

Magic integer sequences have extended the power of the phase permutation method of initial phasing and are used in most direct methods computer programs. The concept of magic integers and their use was introduced by White & Woolfson[30]. For a sequence of n integers m_1, m_2, m_3 ....etc, n phases are represented by the equations:

$$\phi_i = m_i X \qquad (1.35)$$

These equations are approximately satisfied by some value of X in the range $0 \leqslant X \leqslant 2\pi$. The nature and size of the errors involved has been investigated by Main[31] who has also described a recipe for deriving a magic integer sequence which minimises the r.m.s. error in the represented phases. The equations (1.35) are used to assign phase values to the starting set reflections found by the convergence procedure by assigning a value to the variable X. Since the phases of these reflections are unknown, several sets of values are tried by giving X a suitable number of different values - the values being equally spaced between 0 and 2π. The enantiomorph may be defined by exploring only half of the n-dimensional space i.e. $0 \leqslant X \leqslant \pi$.

Only phases whose values are unrestricted by space group symmetry are represented by the magic integers. Restricted phases switch between their two allowed values in the phase permutations.

For each value X there corresponds a set of n phases. An efficient magic integer sequence ensures that these phase sets are as different from each other as possible.

Magic integer sequences suitable for phase permutation are shown in the table below. They are based on the Fibonacci sequence 1,1,2,3,5,8,..... etc. whose numbers form the differences between adjacent numbers of the sequences. Also shown are the number of permutations appropriate to the sequence and the r.m.s error of the phases represented.

n	Perms	Error	Sequence
1	4	26	1
2	12	29	2 3
3	20	37	3 4 5
4	32	42	5 7 8 9
5	50	45	8 11 13 14 15
6	80	47	13 18 21 23 24 25
7	128	48	21 29 34 37 39 40 41
8	206	49	34 47 55 60 63 65 66 67
9	332	50	55 76 89 97 102 105 107 108 109
10	536	51	89 123 144 157 165 170 173 175 176 177

1.14 THE TANGENT FORMULA: PHASE EXTENSION AND REFINEMENT

The processes of phase extension and refinement are quite separate but since both are performed by the tangent formula they are normally considered together.

The most useful phase determining formula is the triplet phase relationship. Employing this relationship it is possible, by starting with a limited number of phases, to expand to a number of phases large enough to be able to calculate an electron density function for the structure.

Frequently there will be several indications for a phase $\phi_{\underline{h}}$. It is possible to obtain the combined probable value of $\phi_{\underline{h}}$ by means of the tangent formula of Hauptman & Karle[32] which makes the assumption that all of the triplets involved are independent.

$$\tan\phi_{\underline{h}} = \frac{\sum_{\underline{k}} |E_{\underline{k}} \cdot E_{\underline{1}}| \sin(\phi_{\underline{k}} + \phi_{\underline{1}})}{\sum_{\underline{k}} |E_{\underline{k}} \cdot E_{\underline{1}}| \cos(\phi_{\underline{k}} + \phi_{\underline{1}})} \qquad (1.36)$$

A set of phases can be refined by an iterative process using the tangent formula. Refinement will continue until there are negligible shifts in the phase values from one cycle to the next. As will be explained later the fact that the tangent formula expands as well as refines phases is one of the fundamental weaknesses of direct methods.

The tangent formula, however, is rarely used in its original form: a variety of modifications have been made in order to take advantage of new theories and developments.

Application of triplet phase relationships frequently leads to several estimates for the value of a single phase. Those triplets which involve large E-magnitudes contribute more strongly to the estimate than others. Reflections at the bottom of the convergence map, therefore participate greatly in the early stages of phase expansion. This information is applied to the tangent formula by means of a weighting scheme. Reflections at the bottom of the convergence map have a weight of approximately 1, while those further up the map are downweighted.

$$\tan\phi_{\underline{h}} = \frac{\sum_{\underline{k}} W_{\underline{k}} W_{\underline{1}} |E_{\underline{k}} E_{\underline{1}}| \sin(\phi_{\underline{k}} + \phi_{\underline{1}})}{\sum_{\underline{k}} W_{\underline{k}} W_{\underline{1}} |E_{\underline{k}} E_{\underline{1}}| \cos(\phi_{\underline{k}} + \phi_{\underline{1}})} \qquad (1.37)$$

$W_{\underline{h}}$, the weight associated with $\phi_{\underline{h}}$, is inversely proportional to the variance of the phase. This weighting scheme plays an important role in the early stages of phase determination, but all the weights quickly become equal to unity and so have little effect during the later stages.

Hull and Irwin[33] have modified the above tangent formula using statistical weights. This scheme is useful in cases of pseudo-symmetry, over consistent phase sets, symmorphic space groups and heavy atom cases.

In its original form, the tangent formula used triplets only. Now, of course, there are many other types of relationships available from which phase information can be extracted and the tangent formula can be readily extended to include other relationships[34]. The above formula uses only triplets but quartet and quintet terms can easily be added to it.

Another modification of the tangent formula[35] deals with the case where an oriented group of atoms has been used in setting up the phase relationships.

1.15 SYMBOLIC ADDITION

The most important formula for the solution of structures by symbolic addition is the triple product sign relationship (1.24). The variance of this relationship is calculated using the Cochran & Woolfson formula (1.25).

The corresponding sign relationship for a positive quartet is:

$$\phi_{\underline{h}} + \phi_{\underline{k}} + \phi_{\underline{l}} + \phi_{\underline{m}} \approx 0 \qquad (1.38)$$

where, the sum of the reflection indices is zero.

Using these formulae it can be shown that, in the case of a triplet, if the signs of two structure factors are known then the probable sign of the third is obtained. For quartets it is necessary to have three known signs to obtain a fourth. This is the underlying principle of the symbolic addition method.

In the symbolic addition procedure a starting set of unknown reflections are represented by symbols. By making use of triple (or quartet) sign relationships which involve starting set reflections or origin reflections, combinations of signs and symbols can be generated to represent other reflections. A trivial example of this is as follows. Suppose that the 3 phases ϕ_1, ϕ_6 and ϕ_9 are involved in a triplet phase relationship such that:

$$\phi_1 + \phi_6 + \phi_9 = 0 \qquad (1.39)$$

and that 2 of the phases have been determined as $\phi_6 = a+\Pi$ and $\phi_9 = c$. For the equation (1.39) to hold ϕ_1 must equal $-(a+c+\Pi)$.

It is then usually possible to match different symbol representations of the same reflection and thereby generate relationships between the symbols. These relationships are solved so that appropriate signs are assigned to each reflection. When this point has been reached, the final step in the procedure - calculation and interpretation of an electron density function, can then take place.

The symbolic addition procedure basically consists of

four parts:

(a) Allocate symbols to a small number of starting set reflections.

(b) Use sign relationships involving these reflections and the origin defining reflections to generate symbol indications for new reflections which can then also be used.

(c) Match up all the different symbol indications for each reflection to form a series of simultaneous equations and then solve these equations to find the appropriate sign for each symbol.

(d) Go back to the indications in (b) and substitute in the symbol signs to allocate signs to the unknown phases.

Symbolic addition can lead to a single solution but sometimes one ends with several symbols to which no definite values can be assigned. If there are n such symbols then there are 2^n possible sets of signs for centrosymmetric structures and 4^n for non-centrosymmetric structures. It is often advantageous to apply symbolic addition in such a way that a multiplicity of solutions is generated by design rather than accidentally. This means there is some insurance against the possibility of wrong indications of relationships between the sign symbols.

1.16 DISADVANTAGE OF SYMBOLIC ADDITION

The symbolic addition method sometimes comes up against the problem of the same phase being indicated by different sets of symbols and yet being unable to combine the phase information[36].

For example, if two sign indications were (a+b) and (c+d) then it is wrong to write a new phase indication as 1/2(a+b+c+d) because of the 2Π ambiguity. If $a=b=\Pi/2$ and $c=d=\Pi/2$ then both indications equal Π but the combination would indicate an incorrect phase of 0.

For this reason there are advantages of introducing phases rather than symbols at an early stage as it means that it is always possible to combine all indications for a new phase by means of the tangent formula. Tangent refinement also offered the considerable advantage of being easier to program and automate than symbolic addition. This led to the introduction of MULTAN (multiple tangent formula method) and other multisolution methods and caused symbolic addition programs, such as LSAM, to become largely superseded.

1.17 FIGURES OF MERIT

Phase sets are assessed, prior to calculation of E-maps by figures of merit. Among the most commonly used figures of merit are ABSFOM PSIZERO, RESID, NQEST and NQINT. These figures of merit are computed for each phase set. In addition a sixth combined figure of merit is usually produced in order

to provide an overall picture of the individual quantities. It should be noted that other figures of merit exist which are also widely used in Direct Methods computer programs[37].

1.17.1 ABSFOM

ABSFOM[38,39,40] is a measure of the internal consistency among the sigma-2 relationships and is calculated using the formula

$$\text{ABSFOM} = \frac{\sum_{\underline{h}} (\alpha_{\underline{h}} - \alpha_{R\underline{h}})}{\sum_{\underline{h}} (\alpha_{E\underline{h}} - \alpha_{R\underline{h}})} \qquad (1.40)$$

where:

$$\alpha_h = \left| \sum_k \chi_{hk} \cos(\phi_k + \phi_{h-k}) \right|$$

$$\chi_{hk} = 2\sigma_3\sigma_2^{-3/2} \left| E_{\underline{h}} E_{\underline{k}} E_{\underline{h}-\underline{k}} \right| \qquad (1.41)$$

$$\alpha_n = \sum_{j=1}^{N} Z_j$$

$$\alpha_{E\underline{h}}^2 = \sum_{k_r} \chi_{hk}^2 + \sum_{k_r} \sum_{k_{r'}} \chi_{hk}\chi_{hk'} \frac{I_1(\chi_{hk}) I_1(\chi_{hk'})}{I_0(\chi_{hk}) I_0(\chi_{hk'})}$$

where I_0 and I_1 are modified Bessel functions of the zero and first orders respectively and

$$\alpha_{R\underline{h}} = (\sum_r \kappa_r^2)^{1/2}$$

$\alpha_{R\underline{h}}$ is the value expected from random phases and $\alpha_{E\underline{h}}$ is an estimated value of $\alpha_{\underline{h}}$. Therefore ABSFOM is zero for random phases and 1 if $\alpha_{\underline{h}}$ is equal to its expectation value. The correct set of signs should correspond to one of the higher values of ABSFOM, typically in the range 0.9 to 1.2.

ABSFOM is the least reliable of the figures of merit: it becomes too large when the phases are over-consistent and too small when they are under-consistent. ABSFOM is closely correlated with the tangent refinement method.

1.17.2 PSI-ZERO

PSI-ZERO is defined[41] as

$$\text{PSIZERO} = \frac{\sum_{\underline{h}} \left| \sum_{\underline{k}} E_{\underline{k}} E_{\underline{h}-\underline{k}} \right|}{\sum_{\underline{h}} \left(\sum_{\underline{k}} \left| E_{\underline{k}} E_{\underline{h}-\underline{k}} \right|^2 \right)^{1/2}} \qquad (1.42)$$

where the $E_{\underline{h}}$ values involved are either very small or zero, that is to say PSIZERO uses the "weak" reflections. PSIZERO should have a low value, typically below 1.0, for the correct set of signs. PSIZERO is independent of the phasing

process and it is also very sensitive to atomic positions.

It sometimes happens that PSIZERO is poor in a phase set which otherwise has good figures of merit. This often indicates a correct fragment in the wrong position.

1.17.3 RESID

RESID[25] can be calculated as follows:

$$\text{RESID} = \frac{\sum_{\underline{h}} \left| \alpha_{\underline{h}} - (\alpha_{\underline{h}})_{est} \right|}{\sum_{\underline{h}} \alpha_{\underline{h}}} \qquad (1.43)$$

This is a residual between the calculated and estimated α's. The correct set of phases should correspond to a low value of RESID, often less than 20.0% RESID is closely related to the crystallographic R-factor.

1.17.4 NQEST, NQINT

NQEST[13,42,43] uses negative quartets. It is calculated as follows:

$$\text{NQEST} = \sum_{h,k,l,m} W_{hklm} \cos(\phi_{\underline{h}} + \phi_{\underline{k}} + \phi_{\underline{l}} + \phi_{\underline{m}}) / \sum_{h,k,l,m} W_{hklm} \qquad (1.44)$$

The summation is over all the negative quartets and where, for centrosymmetric structures weights are defined as $W_{hklm} = |1-2P^{+}|$ (P^{+} is the probability). For non-centrosymmetric

structures $W_{hklm} = 1/\sigma^2$ where σ^2 is the variance of the quartet probability distribution.

NQINT is the corresponding negative quintet figure of merit. The quartet terms in the NQEST formula are replaced by quintet terms and the weights are modified by the discriminant, Δ[17], such that $W_{hklm} = -\Delta|(1-2P^+)|$ for the centrosymmetric case.

NQEST and NQINT can range from -1.0 to +1.0 with the most negative values likeliest to correspond to the correct set of signs. Note that NQEST and NQINT use information contained in the small E magnitudes - PSIZERO uses similar information in a different way and can be regarded as independent. These two figures of merit also offer the advantage of being, to a good approximation, independent of the phasing process.

1.17.5 COMBINED FIGURE OF MERIT

The five figures of merit are combined together to give a single figure of merit as shown overleaf.

$$CFOM = W_1\frac{(ABSFOM-ABSFOM_{min})}{(ABSFOM_{max}-ABSFOM_{min})} + W_2\frac{(PSIZERO_{max}-PSIZERO)}{(PSIZERO_{max}-PSIZERO_{min})} +$$

$$W_3\frac{(RESID_{max}-RESID)}{(RESID_{max}-RESID_{min})} + W_4\frac{(NQEST_{max}-NQEST)}{(NQEST_{max}-NQEST_{min})} +$$

$$W_5\frac{(NQINT_{max}-NQINT)}{(NQINT_{max}-NQINT_{min})} \qquad (1.45)$$

where W_1, W_2, W_3, W_4 and W_5 are weights which are usually chosen as unity. They may be changed to give more weighting to any one of the figures of merit. CFOM should be as large as possible and has a maximum value of $W_1+W_2+W_3+W_4+W_5$. If a figure of merit is not used then the corresponding weight is assigned a value of zero.

CFOM is used to rank the solutions before they are passed to the E-map routine.

1.18 CALCULATION OF E-MAPS

The final stage of a direct methods analysis is the calculation of one or more E-maps. It is important to have a high level of user interaction in the interpretation of E-maps, particularly when there are many maps to search. It is often possible to reject incorrect solutions on the basis of peak heights, peak distribution, peak positions or fragmentation patterns prior to making a full interpretation.

Maps are normally calculated in an order determined by the figures of merit.

Although it is the last step in the structure determination process, the interpretation of E-maps is by no means trivial. As is discussed briefly in the following section and in more detail in Chapter5 it can sometimes be a difficult and time-consuming task, often being the difference between success and failure. The maps are often very complex especially in structures which have many atoms in the asymmetric unit. The interpretation of E-maps can often be enhanced by the use of computer graphics facilities and a section of this thesis is concerned with this subject.

1.19 FAILURES OF DIRECT METHODS

Over recent years direct methods computer programs have become more and more successful in solving the phase problem and are now used to solve the majority of structures. Occasionally, though, the programs fail and no E-maps can be found from which a recognisable fragment can be recycled to derive the complete structure. Apart from the trivial case where the X-ray data is of poor quality, the reasons for failure are difficult to interpret. There are, however, some limitations and weaknesses of direct methods which account for a sizeable portion of these failures.

The first of these is that there can be problems associated with the choice of a reliable starting set. The idea, when choosing a starting set, is to use a number of reflections small enough to give a manageable number of phase permutations yet large enough to lead to strong development of

new phase information. Starting from a small number of assigned phase values, it is possible, by use of phase relationships and the tangent formula, to calculate enough new phases to give a recognisable Fourier representation of the structure. If any of the starting set reflections are involved in bad (i.e. incorrect) phase relationships then this whole process breaks down. This problem occurs less frequently when higher invariants are used.

Somewhat related to the problem just mentioned is the difficulty associated with deciding which phase relationships are most reliable. Most direct methods programs are based on the use of the triplet phase relationship which can be expressed as:

$$\phi_{\underline{h}} + \phi_{\underline{k}} + \phi_{\underline{l}} \approx 0$$

The probability is given by the Cochran & Woolfson formula (1.25) under the assumption that the atomic positions are random variables. This is obviously not the case since atoms are subject to chemical rules governing their distances to other atoms and their interatomic angles. Therefore, (1.25) does not reflect the true probability.

Another limitation of direct methods is the tangent formula itself. For some structures even the correct phases are unstable under the process of tangent formula refinement. This instability in the formula arises from the assumption that all phase relationships are independent - this is not the case. Once again higher invariants, with their better probability relationships improve this somewhat. Probably the best course of action to take when this situation occurs is to use symbolic addition as it does not refine the phases and so neatly avoids the problem.

As direct methods become more successful, they are

being applied to bigger and more complex molecules. A common problem with such structures is recognition of the solution when it has been made: the E-maps are crowded and difficult to interpret. As this thesis will attempt to illustrate, this problem can be greatly reduced by using high quality, interactive colour graphics facilities.

1.20 MITHRIL - AN INTEGRATED DIRECT METHODS COMPUTER PROGRAM

Many important theoretical developments in direct methods took place in the 1970's. Many workers were involved in producing new ideas and concepts which were greatly extending the power of the method. There was, however, little or no overlap in the areas of research undertaken. The computer program MITHRIL was written by Gilmore[21] to incorporate all the new developments in direct methods in a single program. MITHRIL was also written in such a way as to take full advantage of the increasingly powerful computers widely used in crystallographic laboratories.

Built around a heavily modified MULTAN80[44] system, MITHRIL also incorporates features from all of the following direct methods programs: MAGEX[45], YZARC[28], SIMPEL[46], SHELXTL[47], RANTAN[48] and QTAN[49]. On top of this new options such as quintet invariants, the third neighbourhood quartet formula and the active use of higher invariants were added to MITHRIL. The program can be run either as a menu-driven real time interactive package or else in batch mode.

The program is written in modular form, each module has its own menu and user options. A flowchart of the program modules is shown in Figure 1.4.

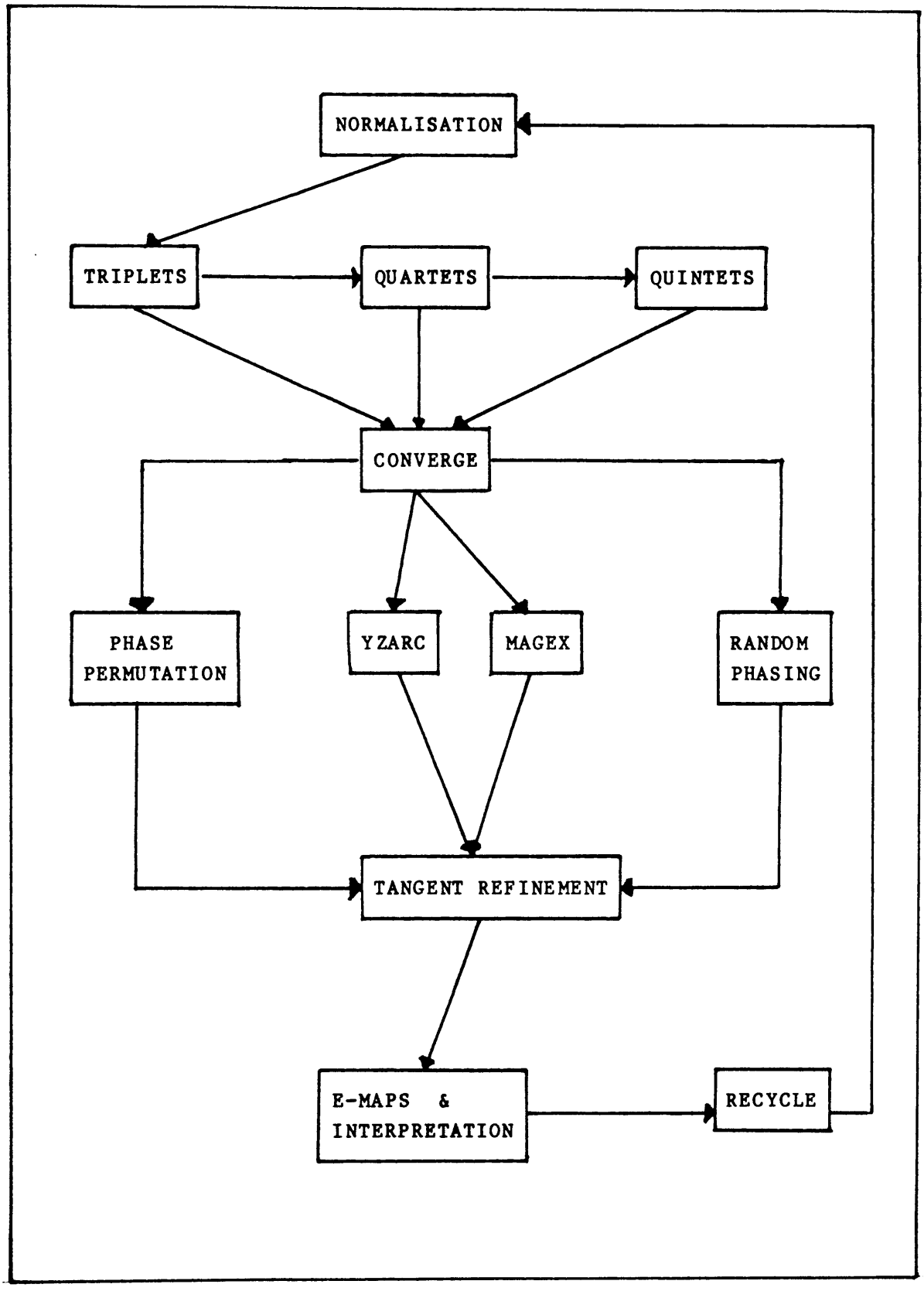


Figure14 Flowchart of the MITHRIL package

MITHRIL is an efficient and powerful computer program for solving crystal structures from X-ray data. It is widely in use in crystallographic laboratories and solves most structures routinely. In the case of difficult structures which do not solve using default options, MITHRIL has an extensive range of alternative options which can be tried. Indeed it is only the most obstinate of structures that cannot be solved using MITHRIL.

REFERENCES

1. R.Narayan, R.Nityananda & G.V.Vani (1983). Contributions to Crystallography, Edited by C.N.R.Rao, Indian Academy of Sciences, 51-69

2 G. Bricogne (1984). Acta Cryst., A40, 410-445

3. M.M.Woolson (1987). Acta Cryst., A43, 593 -612

4. D.Viterbo (1986). God.Jugosl.cent.kristalogr., 21, 149-165

5. J.S.Kasper, C.M.Lucht & D.Harker (1951). Acta Cryst. 3,436

6. H.Hauptman & J.Karle (1953). Solution of the Phase Problem. I. The Centrosymmetric Crystal. Pittsburgh: Polycrystal Book Service.

7. H.Hauptman & J.Karle (1956). Acta Cryst., 6, 131

8. D.Sayre (1952). Acta Cryst.,5,60

9. W.Cochran & M.M.Woolfson (1955). Acta Cryst., 8, 1

10. W.Cochran (1955). Acta Cryst., 8, 473-478

11. H.Schenk (1973). Acta Cryst., A29, 77-82

12. H.Hauptman (1974). Acta Cryst., A30, 472-476

13. G.T.DeTitta, J.W.Edmonds, D.A.Langs & H.Hauptman (1975)., Acta Cryst.,A31,472

14. A.A.Freer & C.J.Gilmore (1980). Acta Cryst., A36, 470-475

15. H.Hauptman (1975). Acta Cryst., A31, 680-687

16. C.Giacovazzo (1977). Acta Cryst., A33, 944-948

17. H.Hauptman & S.Fortier (1977). Acta Cryst., A33, 575-580

18. P.Main (1985). Crystallographic Computing 3, Edited by G.M.Sheldrick, C.Kruger & R.Goddard, Oxford, Clarendon Press 206-215

19. G.M.Sheldrick (1985). Crystallographic Computing 3, Edited by G.M.Sheldrick, C.Kruger & R.Goddard, Oxford, Clarendon Press 184-189

20. S.R.Hall (1985). J. Appl. Cryst., 18, 263-264

21. C.J.Gilmore (1984). J. Appl. Cryst.,17, 42-46

22. M.M.Woolfson (1984). Methods and Applications in Crystallographic Computing, Edited by S.R.Hall & T.Ashida. Oxford, Clarendon Press 106 -119

23. H.Schenk & C.T.Kiers (1985). Crystallographic Computing 3, Edited by G.M.Sheldrick, C.Kruger & R.Goddard, Oxford, Clarendon Press 200-205

24. J.Gillis (1948). Acta Cryst., 1, 174-179

25. J.Karle & I.L.Karle (1966). Acta Cryst.,21,849

26. G.Germain, P.Main & M.M.Woolfson (1970). Acta Cryst., B26, 274

27. P.Main (1977). Acta Cryst., A33, 750-757

28. R.Bagio, M.M.Woolfson, J.P.Declerq & G.Germain (1978). Acta Cryst., A34, 883-892

29. P.Main (1978). Acta Cryst., A34, 31

30. P.S.White & M.M.Woolfson (1975). Acta Cryst., A31, 53

31. P.Main (1977). Acta Cryst., A33, 750-757

32. H.Hauptman & J.Karle (1956). Acta Cryst., 9, 635

33. S.E.Hull & M.J.Irwin (1978). Acta Cryst., A33, 575

34. N.van der Putten & H.Schenk (1979). Acta Cryst., A35, 381

35. P.Main (1976). Crystallographic Computing Techniques, Edited by F.R.Ahmed, Copenhagen, Munksgaard 97-105

36. G.Germain, P.Main & M.M.Woolfson (1974). Direct Methods in Crystallography (Lecture Notes). NATO Advanced Study Institute.

37. G.Cascarano, C.Giacovazzo & D.Viterbo (1987). Acta Cryst., A43,22

38. J.P.Declerq, G.Germain & M.M.Woolfson (1979). Acta Cryst., A35,622

39. M.F.C.Ladd & R.A.Palmer (1980) in Theory and Practice of Direct Methods in Crystallography, Edited by M.F.C.Ladd & R.A.Palmer, Plenum Press, 93-119

40. G.Germain, P.Main & M.M.Woolfson (1971). Acta Cryst., A27, 368

41. W.Cochran & A.S.Douglas (1957). Proc. R. Soc London Ser.A. 227,486

42. H.Schenck (1974). Acta Cryst.,A30,477

43. C.J.Gilmore (1977). Acta Cryst.,A33,712

44. P.Main, S.J.Fiske, G.Germain, S.E.Hull, J.P.Declerq, L.Lessinger & M.M.Woolfson (1980). MULTAN-80. A System of Computer Programs for the Automatic Solution of Crystal Structures from X-ray Diffraction Data. Univ of York, England and Louvain, Belgium

45. S.E.Hull, D.Viterbo, M.M.Woolfson & Z.Shao-Hui (1981). Acta Cryst., A37, 566-572

46. O.Overbeek & H.Schenk (1978). in Computing in Crystallography, Delft University Press, 108-112

47. G.M.Sheldrick (1981). SHELXTL. A System of Computer Programs for the Solution and Refinement of Crystal Structures, Nicolet Instruments, USA

48. Yao Jia-Xing (1981). Acta Cryst., A37, 642-644

49. D.A.Langs & H.Hauptman (1978). in Computing in Crystallography, Delft University Press, 113-118

CHAPTER2

TOWARDS MORE RELIABLE PHASE RELATIONSHIPS

2. TOWARDS MORE RELIABLE PHASE RELATIONSHIPS

2.1 INTRODUCTION

Direct methods have traditionally been based upon the simple triple phase structure invariant:

$$\phi_{\underline{h}} + \phi_{\underline{k}} + \phi_{-\underline{h}-\underline{k}} \approx 0 \qquad (2.1)$$

which can also be expressed in the form:

$$\cos(\phi_{\underline{h}} + \phi_{\underline{k}} + \phi_{-\underline{h}-\underline{k}}) \approx +1 \qquad (2.2)$$

From the point of view of solving the phase problem it is obviously important that this formula is reliable. Its validity, however, is statistical and substantial deviations from zero in (2.1) can occur. Unreliable triplets lead to wrong phases being assigned in the phase expansion process. In difficult structures the occurrence of such relationships may be very frequent. Therefore, any additional formulae which can give an estimate of triplet reliability are of great value. Several such formulae are in use in direct methods computer programs:

a) TPROD and MDKS[1].

b) The use of quadrupoles[2].

c) The P_6 formula[3,4].

d) The use of fourth order Karle-Hauptman determinants[5,6,7].

e) The P_{10} formula[8].

f) The L.E. formula[9,10].

These formulae are quite similar. They all estimate triplet cosines via their second neighbourhoods; they have one or more empirically determined scale factors, and they are all very approximate. Under favourable circumstances the severely anomalous triplets having cosines less than 0.0 can be identified and removed from the phasing procedure. A disadvantage common to all these methods is that they are very expensive in terms of computing time.

An alternative to these formulae is a technique introduced by Main[11] which uses structural information and group scattering factors to estimate triplets. These formulae are fast to compute.

A very serious drawback of all the above methods is that none of them takes into account the reasons why a triplet should be unreliable in the first place. In effect they all treat the symptoms rather than look for a cure.

More recently the use of other phase relationships such as quartets and, to a much lesser extent, quintets have extended the power of direct methods. Once again, of course, it is vital that these relationships should be reliable. There are no such schemes, as those described above for triplets, available for quartet phase relationships. They were thought to be unnecessary since calculation of the third neighbourhood should, in theory, weed out the unreliable quartets. In practice, however, not all of the bad quartets get excluded from the phasing process.

The chapter is split into two sections: the first

deals with the reliability of phase relationships on the basis of the $(\sin^2\theta/\lambda^2)$ values of their participating reflections while the second is concerned with the relationship between quartet reliabilty and the presence of three new cross terms.

PART I TECHNIQUES BASED ON $(\sin^2\theta/\lambda^2)$ VALUES

2.2 UNRELIABLE PHASE RELATIONSHIPS

Direct Methods use the normalised structure factor, E, which can be expressed as a function of the quantity

$$\frac{1}{\exp(-2B\sin^2\theta/\lambda^2)} \qquad (2.3)$$

where:

B - temperature factor
θ - Bragg angle of reflection
λ - wavelength of radiation used

It is apparent from equation (2.3) above that any error in the temperature factor will be magnified by large values of $(\sin^2\theta/\lambda^2)$. A direct consequence of this is that reflections with high $(\sin^2\theta/\lambda^2)$ values will have unreliable structure factors. As will be shown later the reliability of a triplet or quartet phase relationship is directly

proportional to the E-magnitudes of the participating reflections. This means that if there is an error in the E-magnitude of any reflection participating in a phase relationship then the reliability of that relationship will be in error also. In this manner an unjustified importance can be attached to certain phase relationships which involve reflections with large ($\sin^2\theta/\lambda^2$) values.

If in a triplet phase relationship one or more of the reflections involved has an inaccurate structure factor (and therefore an incorrect phase) the relationship obviously becomes invalid and could well cause great problems later on in the phasing process, especially if the product of the three E-magnitudes involved is large.

2.3 TWO METHODS OF COUNTERACTING UNRELIABLE PHASE RELATIONSHIPS

In an attempt to counteract the effect of unreliable reflections involved in triplet relationships two quite separate approaches were taken, and have subsequently been coded into the MITHRIL package. The first is a simple cut-off technique while the second is a more sophisticated weighting scheme.

2.3.1 CUT-OFF TECHNIQUE FOR TRIPLETS

This is the cruder of the two approaches but, as will be shown later, it has nonetheless proved itself to be an

effective technique. The method simply consists of removing any triplets which involve two or three reflections with high $(\sin^2\theta/\lambda^2)$ values.

The first step in this method is to compute $(\sin^2\theta/\lambda^2)$ values for all of the reflections . The user then enters a cut-off value, call it X (where $0 \leq X \leq 1.0$). Any triplets involving two or three reflections which have $(\sin^2\theta/\lambda^2)$ values greater than X times the maximum value are immediately removed.

2.3.2 DOWNWEIGHT TRIPLETS

Triplet relationships in direct methods computer programs are used according to their reliability which can be defined in terms of a definite quantity, kappa. The reliability is of the form:

$$\kappa = 2\sigma_3\sigma_2^{-3/2}\left|E_{\underline{h}}E_{\underline{k}}E_{-\underline{h}-\underline{k}}\right| \qquad (2.4)$$

where

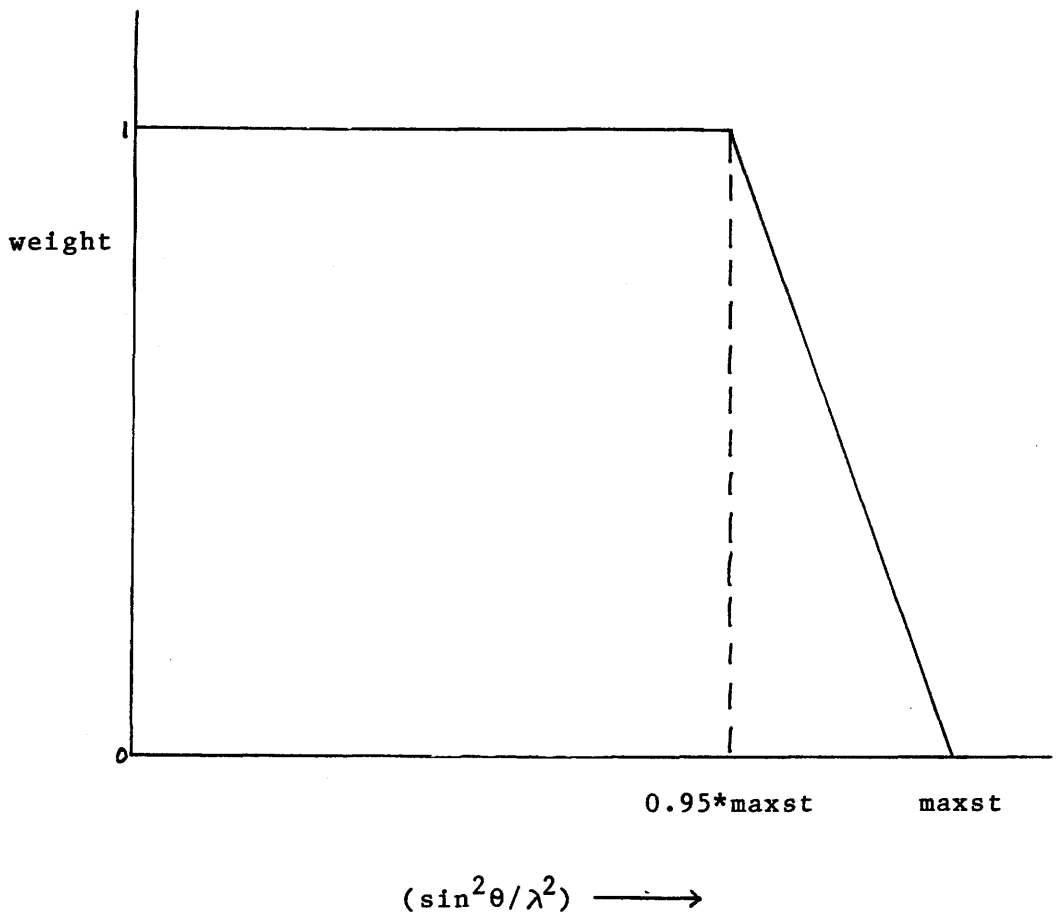
$$\sigma_n = \sum_{j=1}^{N} Z_j^n$$

Z_j - atomic number of the j^{th} atom in the unit cell
N - number of atoms in the unit cell

Each reflection is assigned a weight, W, according to its $(\sin^2\theta/\lambda^2)$ value. The expression now becomes.

$$\varkappa = 2\sigma_3\sigma_2^{-3/2} W_{\underline{h}} W_{\underline{k}} W_{\underline{h}-\underline{k}} \left| E_{\underline{h}} E_{\underline{k}} E_{-\underline{h}-\underline{k}} \right| \qquad (2.5)$$

Many types of weighting scheme were tried. The one eventually used is of the form



where

maxst - maximum value of $(\sin^2\theta/\lambda^2)$

So it is only those reflections which have $(\sin^2\theta/\lambda^2)$

values greater than 0.95 times the maximum value which are downweighted - all other reflections are assigned weights of unity. In this way unreliable triplets are substantially downweighted and become of far less importance in the phasing procedure.

2.4 WEIGHTING SCHEME FOR QUARTETS

Quartet phase relationships in direct methods computer programs are also used according to their reliability which is calculated in an analagous fashion to (2.4) above. An identical weighting scheme to the one just described for triplets has been formulated for quartets.

2.5 LOW ANGLE REFLECTIONS

Reflections at low angle are sometimes unreliable due to absorption and extinction errors. It was thought that triplet and quartet phase relationships would become more reliable if these reflections were either rejected by means of a cut-off scheme as above or by using another weighting scheme.

Several different methods using both these approaches were experimented with but none was successful and the idea was abandoned. It may well be the case, however, that further investigation of this area will prove fruitful. It should be remembered that low angle reflections are normally involved in a lot of triplets and other phase relationships. Care must be

taken, therefore, when excluding such reflections to ensure that the phasing process does not break down.

2.6 USING THE CUT-OFF TECHNIQUE AND WEIGHTING SCHEME IN MITHRIL

Both the weighting scheme and the cut-off technique have been integrated into the direct methods program MITHRIL[12]. As with other programs, MITHRIL requires structure factors to be normalised before generating structure invariants.

The menu in the triplet module in MITHRIL has been modified so that it is now of the form:

```
THE FOLLOWING COMMANDS ARE CURRENTLY AVAILABLE:
 TRIPLET    ( PARAMETERS: (1) NO. OF REFLECTIONS TO BE USED  )
            (             (2) CUT-OFF FOR (SIN(T)/L)**2      )
 WEIGHT     ( WEIGHT E'S ACCORDING TO (SIN(T)/L)**2 VALUES   )
            ( 1/0 = PRINT/DON'T PRINT TABLE OF WEIGHTED E'S  )
 L.E.       ( USE LINEAR EQUATIONS TO ESTIMATE THE TRIPLETS  )
            ( PARAMETERS: (1) MAX. NO OF MISSING TERMS   (0) )
            (             (2) MINIMUM DIAGONAL TERM      (1) )
            (             (3) MAX TERMS FOR EQNS 1-3   (0.5) )
            (             (4) MINIMUM NO. OF EQUATIONS   (5) )
 LIST       (PRINT TRIPLETS-PLEASE USE THIS ONLY IF ESSENTIAL)
 MDKS       ( INVOKES THE MDKS FORMULA TO ESTIMATE TRIPLETS  )
            ( ONLY PARAMETER IS MIN NO OF CONTRIBUTORS TO  D )
            (*** BOTH MDKS AND L.E. OPTIONS ARE VERY SLOW ***)
In addition the commands: TITLE,END,MENU,LEVEL,NOPRINT,PRINT
                          DEFAULT,HARD,VERY HARD,MODEL,SHOW,X
may be issued at any time.
Only the first 4 characters of any command are required.
<CR> or a call to another module terminates current input.
```

As can be seen, the cut-off value is entered as the second parameter to the TRIPLET command in the triplet module

of MITHRIL. For example, if a user was to issue the command "TRIPLET 400,0.85", the program would first of all generate triplet phase relationships for the top 400 reflections and then reject any containing two or more reflections with $(\sin^2\theta/\lambda^2)$ values greater than $0.85*(\sin^2\theta/\lambda^2)_{max}$.

The weighting scheme is invoked in the triplet module by means of the WEIGHT command. If a '1' is entered as a parameter to this command, a table of reflections with their weighted E-magnitudes is printed. The weighting scheme for quartets is utilised in exactly the same fashion: a WEIGHT command has also been added to the menu in the quartet module.

2.7 RESULTS

The cut-off technique and weighting schemes were tested on several structures from the database of structures resistant to direct methods compiled by Sheldrick. The table below summarises the relevant crystallographic details of the seven structures selected for analysis.

Structure	Spacegroup	Formula	Z	Ref.
AZET	$Pca2_1$	$C_{21}H_{16}ClNO$	8	13
PGE2	P1	$C_{20}H_{32}O_5$	1	14
GOLDMAN2	Cc	$C_{28}H_{16}$	8	15
BED	I4	$C_{26}H_{26}N_4O_4$	8	16
NO55	Fdd2	$C_{20}H_{24}N_4$	16	17
APAPA	$P4_12_12$	$C_{30}H_{37}N_{15}O_{16}P_2\cdot 6H_2O$	8	18
TVAL	P1	$C_{54}H_{90}N_6O_{18}$	2	19,20

Rather than going into the details of how many incorrect triplets or quartets were rooted out by either method, it was decided to concentrate on their overall effect i.e. whether or not these methods were helpful in solving difficult structures.

The tables below list the figures of merit for the phase set which yielded the best E-map (i.e. most atoms) in each case, together with the number of atoms found. Table 2.1 summarises the results obtained without using either the weighting scheme or the cut-off technique while Table 2.2 contains the results obtained by using one of these methods.

Table 2.1 : No weights or cut-off

Structure	ABSFOM	PSIZERO	RESID	NQEST	No of atoms	Rank
AZET	1.84	3.23	49.03	-0.07	16/48	1
PGE2	1.33	-	25.31	-0.09	12/25	1
GOLDMAN2	1.15	2.10	28.27	-0.05	17/56	2
BED	1.12	1.19	22.97	-0.16	19/38	2
NO55	-	-	-	-	0/24	-
APAPA	1.07	1.16	26.75	-0.09	26/63	1
TVAL	1.25	-	22.83	-	80/156	1

Table 2.2: With weights or cut-off

Structure	ABSFOM	PSIZERO	RESID	NQEST	No of atoms	Rank
AZET	1.58	2.71	37.87	-0.06+	31/48	1
PGE2	1.33	-	25.32	-0.08	15/25	1
GOLDMAN2*	1.14	0.86	18.84	0.01	56/56	1
BED	1.14	1.18	22.69	-0.17	21/38	2
NO55	0.66	1.70	28.29	-0.03	21/24	1
APAPA	1.09	1.10	25.12	-0.05+	20/63	1
TVAL*	1.28	-	23.14	-	80/156	1

* signifies cut-off technique used rather than weighting scheme

+ signifies quartet weighting scheme used also

Note that no figures of merit are listed for NO55 in Table 2.1 because no appropriate solution could be found. In both tables "rank" refers to the position of the CFOM value for the phase set used with respect to all the other phase sets produced.

Note that use of a weighting scheme or cut-off parameter is the only difference between tables 2.1 and 2.2: the runs were otherwise identical.

2.8 STRUCTURE DETERMINATION OF MORNON

2.8.1 BACKGROUND

MORNON is a polypeptide structure of biological interest containing 138 non-hydrogen atoms in its asymmetric unit. X-ray data for the structure was collected at the Pasteur Institute, Paris where, despite exhaustive trials using various direct methods techniques, the structure could not be determined. At Glasgow problems were encountered because of the sheer physical size of the structure and also because of the very large data set. Another difficulty was that the peptide sequence which accompanied the X-ray data was later found to be incorrect. Once these problems were overcome, however, the structure was determined relatively easily using a big version of MITHRIL. The solution was found by taking advantage of the cut-off technique already described in this chapter.

Some crystallographic data for MORNON is given below:

Orthorhombic
a = 64.8 ($\mathring{A}$) b = 9.33 c = 39.9
Space group: $C222_1$
Formula: $C_{91}N_{23}O_{24}H_{158}$; Z = 8
A total of 13271 reflections in the data set

The correct peptide sequence for the structure is:

Ac.Aib.Aib.Ala.Aib.Aib.Gln.Aib.Aib.Aib.Gln.
Leu.Aib.Pro.Val.Aib.Val.Aib.Val.Gly.Gln.Try

where:

Ac = Acetyl
Aib = di-methyl Alanine
Ala = Alanine
Gln = Glutamine
Val = Valine
Gly = Glycine
Leu = Leucine
Pro = Proline
Try = Tryptophan

2.8.2 STRUCTURE DETERMINATION

Using a standard version of MITHRIL it was found that when the MORNON data set was expanded over a full hemisphere, the store capacity of the program was exceeded and no useful information could be obtained. In an attempt to get round this problem trials were carried out, without success, using an edited data set which had all reflections with $(\sin\theta/\lambda)$ values greater than 0.45 removed. An obvious disadvantage of using the reduced data set was that any information contained in the excluded reflections could not be made use of. It was only when a specially expanded version of MITHRIL was used that the structure was found. The bigger version of the program could deal with up to 100000 expanded reflections, as compared to only 24000 in the original, and so coped easily with the full MORNON data set.

Normalisation of the structure factors using a K-curve gave a reasonable distribution of the E-magnitudes even though there were a few large values. Using default parameters throughout the rest of the program, several problems became

apparent. It was noticed that while a lot of triplets were produced, most had low reliability values and so relatively few were used in the convergence procedure. The low reliability values obtained for MORNON are a direct consequence of the large number of atoms in the unit cell of the structure: triplet reliabilty varies inversely with $N^{1/2}$ while quartet reliability varies inversely with N, where N is the number of atoms. The convergence map itself looked moderately good, however, a few undetermined phases showed up in tangent refinement. The figures of merit produced for the phase sets were not expected to be very good since a large number (612) of reflections were used to generate triplets but, even allowing for this, they were still very poor.

The convergence procedure was stabilised by introducing quartets as well as increasing the number of reflections used to generate triplets. Very few negative quartets were produced for this structure and it was found necessary to use a large number of reflections for their generation also. The statistically weighted tangent formula was preferred to the standard formula as it brought about an improvement in the figures of merit as well as removing the problem of undetermined phases. However, even though the figures of merit were better they were still fairly poor and no solution was found.

A further noticeable improvement in the figures of merit was observed when the cut-off technique was employed and, in fact, this proved to be the key step in solving the structure. In the successful run: normalisation was by means of a K-curve; triplets were generated for the top 800 reflections and those which fell below the cut-off value of 0.70 were rejected; quartets were produced for the top 400 reflections; there were 11 reflections in the starting set; and phase refinement and expansion were carried out using the statistically weighted tangent formula. The best phase set

out of the 640 produced yielded a 52 atom fragment. Successive cycles of weighted Fourier synthesis located all but nine of the 138 non-hydrogen atoms. Problems were initially encountered in interpreting the maps but these were removed when the 5 atom ring of the proline unit appeared: this provided a 'foothold' for the structure. Refinement of the structure was being carried out at the Pasteur Institute and had not been completed at the time of writing. The R-factor had been reduced to 0.12 but difficulties had arisen with solvent molecules present in the crystal. An ORTEP[21] diagram of MORNON is given in Figure 2.1 while Table 2.3 contains the coordinates of the non-hydrogen atoms in the structure.

2.9 DISCUSSION

It is apparent that use of both the weighting and the cut-off techniques have been very successful. Examination of the results shows that for 5 of the 7 test structures considered in section 2.7 it was possible to find more atoms using one of the two schemes than without, in one structure it made no difference and only in one case was the solution worse. Another signficant point in favour of the weighting scheme and the cut-off technique is that, unlike the formulae discussed in the introduction, both are extremely fast: the time taken to generate triplets or quartets is not noticeably increased when either method is used.

The most spectacular improvements caused by using these techniques were observed for GOLDMAN2 and NO55. In GOLDMAN2, use of a cut-off value meant it was possible to locate all 56 atoms in the asymmetric unit - this compares

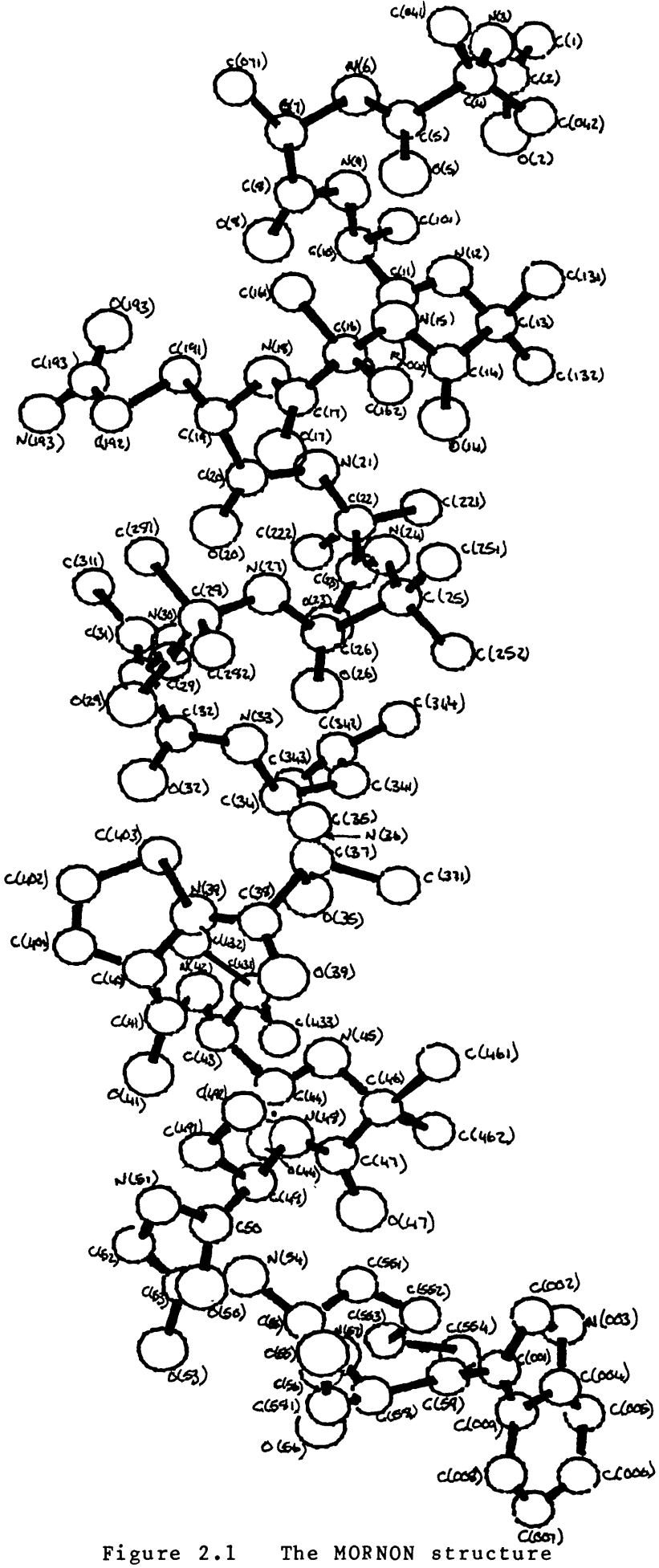


Figure 2.1 The MORNON structure

TABLE 2.3 Coordinates and isotropic temperature factors for MORNON

Atom	x/a	y/b	z/c	U(iso)
C(001)	-.1991(4)	-.1629(29)	.2218(7)	.063
C(1)	.2356(4)	-.6143(32)	.1153(7)	.089
C(002)	-.1840(4)	-.2554(38)	.2145(8)	.094
C(2)	.2231(4)	-.5105(34)	.1353(7)	.097
C(4)	.2217(3)	-.3453(29)	.1819(7)	.073
C(004)	-.2131(4)	-.3445(32)	.1965(7)	.081
C(5)	.2092(3)	-.2291(26)	.1602(6)	.047
C(005)	-.2270(4)	-.4322(36)	.1777(8)	.120
C(006)	-.2475(4)	-.4025(33)	.1826(7)	.107
C(007)	-.2536(4)	-.2691(39)	.1965(6)	.080
C(7)	.2073(3)	-.0482(29)	.1193(6)	.071
C(008)	-.2373(4)	-.1867(33)	.2106(6)	.079
C(8)	.1875(4)	-.0878(31)	.1032(7)	.088
C(009)	-.2163(4)	-.2169(29)	.2110(6)	.058
C(10)	.1702(3)	-.2820(30)	.0657(6)	.056
C(11)	.1525(4)	-.3162(30)	.0936(7)	.058
C(13)	.1421(3)	-.4280(27)	.1479(6)	.056
C(14)	.1300(4)	-.2943(34)	.1598(7)	.079
C(16)	.1348(3)	-.0393(27)	.1791(6)	.053
C(17)	.1214(3)	.0255(25)	.1545(6)	.056
C(19)	.1159(4)	.1085(27)	.0989(8)	.075
C(20)	.0947(3)	.0368(30)	.0917(6)	.061
C(22)	.0765(4)	-.2124(40)	.0821(7)	.087
C(23)	.0616(4)	-.1579(26)	.1113(7)	.075
C(25)	.0545(3)	-.1425(27)	.1691(6)	.057
C(26)	.0445(3)	.0220(25)	.1661(5)	.042
C(28)	.0526(4)	.2766(29)	.1645(6)	.069
C(29)	.0380(3)	.2988(26)	.1350(7)	.060
C(31)	.0305(3)	.2744(31)	.0775(6)	.061
C(32)	.0098(4)	.2003(32)	.0748(6)	.066
C(34)	-.0134(3)	.0122(28)	.0942(7)	.069
C(35)	-.0239(3)	.0297(28)	.1247(6)	.070
C(37)	-.0281(4)	.1010(33)	.1803(8)	.109
C(38)	-.0472(4)	.2066(38)	.1720(7)	.082
C(40)	-.0610(4)	.4175(34)	.1486(8)	.098
C(041)	.2077(3)	-.4265(28)	.2048(6)	.071
C(41)	-.0792(4)	.3558(31)	.1339(7)	.097
C(042)	.2370(4)	-.2587(32)	.2007(7)	.081
C(43)	-.0918(5)	.1830(42)	.0923(8)	.105
C(44)	-.1095(5)	.1044(36)	.1169(7)	.108
C(46)	-.1141(4)	-.0371(36)	.1680(9)	.118
C(47)	-.1265(4)	.0880(27)	.1816(6)	.066
C(49)	-.1281(4)	.3091(31)	.2074(7)	.073
C(50)	-.1437(5)	.3665(35)	.1832(8)	.103
C(52)	-.1553(4)	.4476(37)	.1284(9)	.141
C(53)	-.1718(4)	.3470(43)	.1271(7)	.125
C(55)	-.1897(4)	.0862(37)	.1275(7)	.107
C(56)	-.2015(4)	.1162(32)	.1615(8)	.096
C(58)	-.2020(4)	.1114(32)	.2191(7)	.086
C(59)	-.1992(4)	.2399(38)	.2405(8)	.088
C(071)	.2234(4)	.0023(37)	.0877(8)	.108
C(101)	.1762(4)	-.4147(30)	.0472(6)	.077
C(131)	.1554(4)	-.4776(34)	.1771(7)	.099
C(132)	.1272(4)	-.5303(30)	.1360(7)	.083
C(161)	.1552(3)	.0682(29)	.1819(6)	.068

Table 2.3 (cont)

C(162)	.1242(4)	-.0676(29)	.2128(6)	.075
C(191)	.1298(3)	.1238(30)	.0655(6)	.059
C(192)	.1176(5)	.2462(50)	.0437(8)	.127
C(193)	.1289(4)	.2365(35)	.0077(7)	.075
C(221)	.0805(3)	-.3513(25)	.0875(7)	.064
C(222)	.0663(4)	-.1701(32)	.0497(7)	.078
C(251)	.0354(4)	-.2442(33)	.1713(7)	.079
C(252)	.0666(4)	-.1727(42)	.1995(7)	.116
C(281)	.0734(3)	.3655(27)	.1558(7)	.059
C(282)	.0440(4)	.3108(34)	.1990(7)	.084
C(311)	.0427(5)	.2306(41)	.0488(7)	.098
C(312)	.0601(3)	.3113(26)	.0445(5)	.049
C(341)	-.0109(4)	-.1578(33)	.0851(7)	.083
C(342)	-.0024(5)	-.1909(36)	.0532(9)	.129
C(343)	.0064(5)	-.3352(31)	.0541(7)	.101
C(344)	-.0161(6)	-.1431(63)	.0200(10)	.290
C(371)	-.0130(5)	.1521(40)	.2100(6)	.111
C(372)	-.0377(4)	-.0518(40)	.1958(9)	.150
C(401)	-.0528(5)	.5347(36)	.1317(9)	.136
C(402)	-.0318(5)	.5311(42)	.1405(10)	.162
C(403)	-.0250(4)	.3589(39)	.1420(8)	.126
C(431)	-.0814(5)	.0644(45)	.0694(8)	.162
C(432)	-.0648(6)	.1470(34)	.0416(10)	.186
C(433)	-.1008(7)	-.0146(50)	.0560(9)	.218
C(461)	-.1262(4)	-.1727(37)	.1549(9)	.119
C(462)	-.0968(5)	-.1021(36)	.2022(10)	.137
C(482)	-.0996(6)	.3814(40)	.2430(11)	.191
C(483)	-.1273(5)	.5546(35)	.2356(11)	.175
C(491)	-.1144(4)	.4316(32)	.2175(8)	.104
C(521)	-.1417(6)	.5047(53)	.0839(11)	.192
C(551)	-.1804(3)	-.0413(30)	.1239(7)	.091
C(552)	-.1944(5)	-.1805(57)	.1222(9)	.172
C(553)	-.2082(5)	-.1615(43)	.0853(9)	.165
C(554)	-.2155(5)	-.3242(54)	.0899(9)	.160
C(581)	-.1954(4)	-.0147(31)	.2415(7)	.081
N(003)	-.1918(3)	-.3567(24)	.1974(6)	.084
N(3)	.2332(3)	-.4347(22)	.1561(5)	.064
N(6)	.2182(3)	-.1742(22)	.1355(5)	.062
N(9)	.1883(3)	-.2354(24)	.0852(5)	.060
N(12)	.1582(3)	-.3813(21)	.1217(5)	.061
N(15)	.1443(3)	-.1731(23)	.1659(5)	.055
N(18)	.1282(2)	.0320(20)	.1235(4)	.046
N(21)	.0958(3)	-.1210(22)	.0896(5)	.055
N(24)	.0676(2)	-.1667(20)	.1414(4)	.042
N(27)	.0586(3)	.1282(21)	.1620(5)	.058
N(30)	.0417(3)	.2389(22)	.1050(5)	.054
N(33)	.0056(3)	.0846(21)	.0945(5)	.078
N(36)	-.0171(3)	.0648(24)	.1503(5)	.074
N(39)	-.0446(3)	.3135(27)	.1532(6)	.095
N(42)	-.0749(3)	.2450(34)	.1111(6)	.100
N(45)	-.0986(3)	.0296(25)	.1422(5)	.080
N(48)	-.1173(3)	.1998(25)	.1901(5)	.072
N(51)	-.1393(3)	.3919(27)	.1525(6)	.092
N(54)	-.1717(3)	.2074(30)	.1264(6)	.106
N(57)	-.1922(3)	.1235(27)	.1878(5)	.084
N(193)	.1183(3)	.3077(28)	-.0168(6)	.093
O(2)	.2045(2)	-.5056(21)	.1314(4)	.083
O(5)	.1917(2)	-.2031(19)	.1716(4)	.066

Table 2.3 (cont)

O(8)	.1725(2)	-.0232(19)	.1031(4)	.074
O(11)	.1340(2)	-.2853(22)	.0857(4)	.070
O(14)	.1118(2)	-.2789(20)	.1587(4)	.070
O(17)	.1043(2)	.0786(17)	.1611(4)	.057
O(20)	.0788(2)	.1016(17)	.0852(4)	.065
O(23)	.0433(2)	-.1038(19)	.1021(4)	.070
O(26)	.0262(2)	.0459(15)	.1683(4)	.061
O(29)	.0252(2)	.3930(19)	.1404(4)	.079
O(32)	-.0059(2)	.2584(21)	.0618(5)	.087
O(35)	-.0450(2)	.0122(19)	.1211(5)	.087
O(38)	-.0649(3)	.1863(25)	.1856(5)	.098
O(41)	-.0979(3)	.4002(22)	.1335(5)	.111
O(44)	-.1277(3)	.1155(24)	.1115(5)	.098
O(47)	-.1445(2)	.0542(19)	.1888(5)	.094
O(50)	-.1618(3)	.4151(21)	.1961(5)	.095
O(53)	-.1929(3)	.3951(27)	.1249(5)	.140
O(56)	-.2217(2)	.1129(22)	.1555(5)	.095
O(59)	-.1801(3)	.2597(23)	.2494(6)	.120
O(193)	.1452(3)	.1780(26)	.0028(6)	.117

very favourably to the 17 atom fragment found using the ordinary triplets. In the case of NO55, the weighting scheme was the difference between solving the structure and not solving it. These two examples, on their own, are enough to justify the use of both techniques in the MITHRIL package.

In many cases it has been found that use of either the cut-off technique or the weighting scheme causes a marked improvement in the values of the figures of merit. As can be seen from the tables of results, this is a fairly general trend. However, sometimes it can be a rather cosmetic effect as in the case of APAPA where, using the weighting scheme, the figures of merit improved but fewer atoms were found.

Care must be exercised in the selection of a cut-off value since too low a value will cause the convergence mapping to collapse because of a lack of contributors. A value of around 0.85 is normally fairly effective but it may be necessary to drop lower than this - values as low as 0.55 have been tried on some of the test structures without any great adverse affect on the convergence procedure. GOLDMAN2 was solved using a cut-off of 0.69 on triplets generated for the top 400 reflections.

The weighting scheme must also be used with a certain amount of caution. In cases where there are few phase relationships present, the weighting scheme may cause weak links in the convergence map. This problem can normally be rectified by simply increasing the number of reflections used to generate phase relationships.

As was noted in the previous section, the cut-off technique was also used to great success in solving the 138 atom MORNON structure. A 52 atom fragment was found using the cut-off scheme whereas no recognisable fragment could be found without it. At the time of writing this is the largest

structure to be solved at Glasgow using direct methods.

The result with APAPA, where more atoms were found without invoking either technique than with, illustrates that these methods are by no means infallible. In TVAL also, there was no noticeable improvement gained from using a cut-off value. The techniques do not work for every structure but in most cases they will help.

The fact that both the weighting scheme and the cut-off technique have worked proves that the original hypothesis behind the methods is correct i.e. that phases of reflections with high $(\sin^2\theta/\lambda^2)$ values will probably be unreliable.

Whilst there is no doubt that both methods have proved to be very helpful with the majority of structures they have been tested upon. The full range of their power is not yet known: this knowledge can only be gained by using them on many more structures and space groups.

In summary, both the weighting scheme and the cut-off method have proved to be very useful in solving difficult structures. If a structure should fail to solve using default parameters throughout the MITHRIL program, these methods should be among the first non-default options chosen to try to overcome the difficulty.

PART II THE EFFECT OF THREE ADDITIONAL CROSS TERMS ON QUARTET RELIABILITY

2.10 THE THREE ADDITIONAL CROSS TERMS

It was shown in section 1.9.2 that quartet reliability is dependent principally on the E-magnitudes of the four reflections involved followed by the E-magnitudes of the second neighbourhood i.e. $|E_{\underline{h}+\underline{k}}|$, $|E_{\underline{h}+\underline{l}}|$ and $|E_{\underline{k}+\underline{l}}|$. Using maximum determinant methods[22] it is possible to define the second neighbourhood of a quartet in a different way such that it contains the three cross terms above and also three new cross terms.

The maximum determinant method originates with Karle and Hauptman[23] who were able to prove, under the condition that the electron density is positive everywhere in the unit cell, that the determinant of any matrix of the form:

$$D_n = \begin{vmatrix} 1 & U(\underline{h}_1) & U(\underline{h}_2) & \cdots\cdots & U(\underline{h}_n) \\ U(-\underline{h}_1) & 1 & U(\underline{h}_2-\underline{h}_1) & \cdots\cdots & U(\underline{h}_n-\underline{h}_1) \\ U(-\underline{h}_2) & U(\underline{h}_1-\underline{h}_2) & 1 & \cdots\cdots & U(\underline{h}_n-\underline{h}_2) \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ U(-\underline{h}_n) & U(\underline{h}_1-\underline{h}_n) & U(\underline{h}_2-\underline{h}_n) & & 1 \end{vmatrix}$$

must be non-negative. A large variety of inequalities among unitary structure factors may be constructed from such determinants.

The maximum determinant method extends this priciple to allow probabilistic calculations to use the Karle and Hauptman determinants. Many probability distributions can be calculated using square matrices of varying size. Of particular interest is the 8 x 8 determinant shown in figure 2.2:

This determinant involves the quartet relationship $\phi_4 = \phi_{\underline{h}} + \phi_{\underline{k}} + \phi_{\underline{l}} + \phi_{\underline{m}}$ (with $\underline{h}+\underline{k}+\underline{l}+\underline{m} = 0$). The quartet probability distribution derived from this determinant is extremely complex, however, Tsoucaris[22] has shown that for the special case where the six cross terms $|E_{\underline{h}+\underline{k}}|$, $|E_{\underline{h}-\underline{k}}|$, $|E_{\underline{h}+\underline{l}}|$, $|E_{\underline{h}-\underline{l}}|$, $|E_{\underline{k}+\underline{l}}|$ and $|E_{\underline{k}-\underline{l}}|$ are all small then there is a higher probability that the quartet ϕ_4 is close to π.

This implies that in addition to being dependent upon the values of the cross terms $|E_{\underline{h}+\underline{k}}|$, $|E_{\underline{h}+\underline{l}}|$ and $|E_{\underline{k}+\underline{l}}|$, the value of a quartet is also dependent upon the three new cross terms $|E_{\underline{h}-\underline{k}}|$, $|E_{\underline{h}-\underline{l}}|$ and $|E_{\underline{k}-\underline{l}}|$. Specifically, maximum determinant methods suggest that a quartet is likely to have a value of when all six cross terms are small. It was thought therefore that negative quartets could be estimated with greater reliability in MITHRIL if the three additional cross terms were all small.

2.11 TESTING QUARTET RELIABILITY

Using MITHRIL it was already possible to list out quartets and their phase values by means of the LIST option in the quartet module. Code was added to the program to output

$$
D_8 = \begin{vmatrix}
1 & & & & & & & \\
U_{\underline{h}} & 1 & & & & & & \\
U_{\underline{k}} & U_{\underline{k}-\underline{h}} & 1 & & & & & \\
U_{\underline{1}} & U_{\underline{1}-\underline{h}} & U_{\underline{1}-\underline{k}} & 1 & & & & \\
U_{\underline{h}+\underline{k}} & U_{\underline{k}} & U_{\underline{h}} & U_{\underline{h}+\underline{k}-\underline{1}} & 1 & & & \\
U_{\underline{h}+\underline{1}} & U_{\underline{1}} & U_{\underline{h}+\underline{1}-\underline{k}} & U_{\underline{h}} & U_{\underline{1}-\underline{k}} & 1 & & \\
U_{\underline{k}+\underline{1}} & U_{\underline{k}+\underline{1}-\underline{h}} & U_{\underline{1}} & U_{\underline{k}} & U_{\underline{1}-\underline{h}} & U_{\underline{k}-\underline{h}} & 1 & \\
U_{-\underline{m}} & U_{\underline{k}+\underline{1}} & U_{\underline{h}+\underline{1}} & U_{\underline{h}+\underline{k}} & U_{\underline{1}} & U_{\underline{k}} & U_{\underline{h}} & 1
\end{vmatrix}
$$

Figure 2.2 The 8 x 8 determinant

the E-magnitudes of the additional cross terms (if present) together with the true phase value of a particular quartet and the phase value estimated by the program. Of course, in order to calculate the true value of a quartet it was necessary to use structures with known phases.

The following seven structures were used to test the dependence of negative quartet reliability on the presence of small (E's less than 1.0) additional cross terms:

Num	Name	Space Group	Formula	Z	Ref
1	NO55	Fdd2	$C_{20}H_{24}N_4$	16	17
2	GOLDMAN2	Cc	$C_{28}H_{16}$	8	15
3	MUNICH1	C2	$C_{20}H_{16}$	8	24
4	WINTER2	$P2_1$	$C_{52}H_{83}N_{11}O_{16}\cdot 3CH_2Cl_2$	2	25
5	PGE2	P1	$C_{20}H_{32}O_5$	1	14
6	SUOA	$P2_12_12_1$	$C_{28}H_{38}O_{19}$	4	26
7	NEWQB	$P\bar{1}$	$C_{24}H_{20}N_2O_5$	4	27

As with the structures used to test the techniques discussed in the first section of this chapter all the structures used here are taken from the database of difficult structures compiled by Sheldrick. They are examined in greater detail in Chapter 4.

Table 2.4 summarises the results obtained for six of the seven structures. NREF denotes the number of reflections used to generate negative quartets in each case. Columns 3 to 6 in the table contain the number of quartets found with 0, 1, 2 and 3 additional small cross terms present respectively. The final four columns contain the mean absolute differences, in degrees, between the true value of the quartet and the value calculated by MITHRIL. The number of cross terms quoted

	NREF	Number of additional cross terms found				Mean absolute phase difference			
Cross terms		0	1	2	3	0	1	2	3
NO55	62	15	54	110	77	79.9	78.3	89.6	90.8
GOLDMAN2	200	35	92	62	13	80.6	79.8	70.3	101.8
MUNICH1	100	19	52	44	19	63.4	83.0	70.6	88.3
WINTER2	190	15	48	44	12	85.3	87.1	77.7	81.7
PGE2	100	12	45	34	14	101.7	97.7	83.8	65.4
SUOA	140	38	43	21	2	81.4	70.1	85.2	133.0

Table 2.4 Quartet reliability and positive cross terms

in each case is the number with E-magnitudes less than 1.0 i.e. the number of small additional cross terms.

In centrosymmetric space groups invariants are restricted to the two values 0 and π. It is pointless, therefore, to use average values in these situations. The following table for NEWQB, the only centrosymmetric structure used, lists the number of correct and incorrect quartet value predicted by MITHRIL with respect to the number of cross terms. Negative quartets were generated using the top 120 reflections for NEWQB.

Additional Cross terms	No. right	No. wrong
0	10	5
1	30	10
2	33	13
3	12	5

2.12 CONCLUSIONS

Whereas the results obtained in the first part of this chapter were very encouraging, those obtained here are not as good as expected. Inspection of the results listed in table 2.4 shows that there appears to be no correlation between the number of small additional cross terms and negative quartet reliability. It can be seen that only in one case - that of PGE2 - did the quartet reliability increase with the number of cross terms present. In all the other structures the number of cross terms had little effect on the quartet reliability.

The average difference between true and calculated quartet values for quartets with all three additional cross terms present is 133° for SUOA which seems especially bad. However, it must be remebered that there are only two quartets contributing to the average and as such the sample size is far too small to draw any conclusions from this result.

The reasons for the poor results are rather unclear. One factor that should have been included but was unfortunately ignored in this analysis is the reliabilty attached to the quartets by MITHRIL. The program uses quartets according to their kappa values which it calculates in an analogous fashion to that shown for triplets in equation 2.4. It was observed that many of the negative quartets generated for the structures used had very low kappa values and so would not have been used by the program. It might be that these quartets are distorting the whole situation and could possibly (though not very probably) be hiding some sort of relationship between the number of small additional cross terms and negative quartet reliability. Repeating the analysis using only quartets whose kappa values were above a certain minimum value may prove to be interesting.

REFERENCES

1. H.Hauptman (1972). Crystal Structure Analysis: The Role of the Cosine Seminvariants, New York: Plenum Press

2. D.Viterbo & M.M.Woolfson (1973). Acta Cryst., A29, 205-208

3. C.Giacovazzo (1976). Acta Cryst., A32, 967-976

4. C.Giacovazzo (1977). Acta Cryst., A33, 527-531

5. J.C.Messager & G.Tsoucaris (1972). Acta Cryst., A28, 482-484

6. J.Karle (1979). Proc. Natl Acad. Sci. USA, 76, 2089-2093

7. J.Karle (1980). Proc. Natl Acad. Sci. USA, 77, 5-9

8. G.Cascarano, C.Giacovazzo, M.Camalli, R.Spagna, M.C.Burla, A.Nunzi & G.Polidori (1984). Acta Cryst., A40, 278-283

9. H.Hauptman (1985). Acta Cryst., A41, 454-456

10. C.J.Gilmore & H.Hauptman (1985). Acta Cryst., A41, 457-462

11. P.Main (1976) Crystallographic Computing Techniques, Copenhagen; Munksgaard

12. C.J.Gilmore (1984). J.Appl.Cryst., 17, 42-46

13. A.Colens, J.P.Declerq, G.Germain, J.P.Putzeys & M.Van Meerssche (1974). Cryst. Comm., 3, 119-122

14. G.T.DeTitta, D.A.Langs, J.W.Edmonds & W.L.Duax (1980). Acta Cryst., B36, 638-645

15. H.Irngartinger, W.R.K.Reibel & G.M.Sheldrick (1981). Acta Cryst., B37, 1768-1771

16. G.M.Sheldrick, B.E.Davison & J.Trotter (1978). Acta Cryst. B34, 1387-1389

17. G.M.Sheldrick & J.Trotter (1978). Acta Cryst. B34, 3122-3124

18. D.Suck, P.C.Manor & W.Saenger (1976). Acta Cryst., B32, 1727-1735

19. I.L.Karle (1975). J.Amer.Chem.Soc. 97, 4379-4386

20. G.D.Smith, W.L.Duax, D.A.Langs, G.T.DeTitta, J.W.Edmonds, D.C.Rohrer & C.M.Weeks (1975). J.Amer.Chem.Soc. 97, 7242-7247

21. C.K.Johnson (1976). ORTEP II, Report ORNL-5138, Oak Ridge National Laboratory, Oak Ridge, Tennessee

22. G.Tsoucaris (1980) in Theory and Practice of Direct Methods in Crystallography, Edited by M.F.C.Ladd & R.A.Palmer, Plenum Press, 287-360

23. J.Karle & H.Hauptman (1950). Acta Cryst., 3, 181

24. U.Szeimies-Seebach, J.Harnisch, G.Szeimies, M.Van Meerssche, G.Germain & J.P.Declerq (1978). Angew. Chem. Int. Edn., 17, 848-850

25. T.Butters, P.Hutter, G.Jung, N.Pauls, H.Schmitt, G.M.Sheldrick & W.Winter (1981). Angew. Chem., 93, 904-905

26. J.D.Oliver & L.C.Strickland (1984). Acta Cryst., C40, 820-824

27. R.Grigg, J.Kemp, G.M.Sheldrick & J,Trotter (1978). J.C.S. Chem. Comm., 109-111

CHAPTER 3

LSAM: THE USE OF SYMBOLIC ADDITION IN MITHRIL

3 LSAM: THE USE OF SYMBOLIC ADDITION IN MITHRIL

3.1 INTRODUCTION

LSAM is an acronym of Logical Symbolic Addition Method. It was distributed as a system of computer programs for the automatic solution of centrosymmetric crystal structures. LSAM was written in the 1960's by Germain, Main and Woolfson[1]. It was written for a 32K word computer which by the standards of today is tiny. The program as written in FORTRAN II consisted of three parts:

LSAM0 - Generated triple product sign relationships

LSAM1 - Fixed the origin, assigned symbols to a small number of reflections, developed signs for other reflections by symbolic addition and determined the relationships between the symbols.

LSAM2 - Solved the relationships between the sign symbols and, allowing for one such relationship to fail at a time, generated several sets of signs for the symbols. Signs for the reflections were then developed for each set of signs for the symbols and figures of merit were computed.

LSAM required the reflection indices and normalised structure factors to be read from files created by other computer programs, and parameters such as lattice type and number of symbols were input to the program system by means of cards. The output from LSAM2 was itself stored on file before

being fed into a separate Fourier program for E-map calculation and interpretation. Hence, LSAM was not a "complete" package for structure determination since it required these other programs.

With the advent of multisolution program packages, such as MULTAN[2], based on the tangent formula[3] for phase determination LSAM lost popularity and became obsolete. As pointed out earlier in this thesis, it should be noted that the tangent formula has some definite advantages over the symbolic addition method.

In spite of its limitations, symbolic addition is an extremely useful tool for determining structures and is still in use in modern Direct Methods program systems such as SIMPEL[4].

It was thought to be a worthwhile project to include LSAM as a module in the MITHRIL package for the following four reasons:

a) With the rapid and ongoing development of computer technology the program could be substantially upgraded to become more powerful.

b) It would provide an alternative to tangent refinement which could be of use in the case of difficult structures.

c) It would hopefully be much quicker than tangent refinement and so would cut down the amount of computer time needed to solve centrosymmetric structures.

d) It could be altered to use the previously unavailable quartet and quintet phase relationships.

3.2 LSAM - THE METHOD

Before the LSAM module in MITHRIL can be used, normalised structure factors, triplet - and, optionally, quartet and quintet - invariants must be calculated, or else read from a previously stored file.

The origin is fixed by the convergence module in MITHRIL and the program assigns symbols to a starting set of reflections also found in converge. The program contains options to let LSAM calculate its own origin (which it does in accordance with the rules given by Hauptman and Karle[5,6]) and allocate its own starting set.

There are two reasons for having a routine within LSAM to define an origin. The first is that this routine often generates a different origin and starting set from that found by the convergence module in MITHRIL and is therefore a useful alternative. Secondly, it allows the user to define the number of starting set symbols without being limited by the number of starting set reflections as is otherwise the case.

The number of symbols used can be chosen by the user, up to a maximum of 11 - the default is 4. However, unless the ORIGIN option is used the maximum number of symbols permitted is governed by the number of reflections in the starting set provided by the convergence module.

LSAM, as originally released, calculated triple product sign relationships for use in the symbolic addition

procedure. Within the MITHRIL package however, triplet and (sometimes) quartet and quintet phase relationships are calculated at an earlier stage so the LSAM module now uses these relationships rather than calculating them itself. The module accepts all triplet and quartet relationships (up to a maximum of 2000) which have values of $2P^{+}-1$ greater than a minimum value set by the user.

The probability of each sign relationship is computed using the Cochran & Woolfson formula (eqn. 1.25). Should the same reflection appear twice in a sign relationship the product $\left|E_{\underline{h}}E_{\underline{k}}E_{\underline{h}-\underline{k}}\right|$ is halved before calculating the probability.

Using the starting set symbols and origin reflections, signs are developed by symbolic addition[7] taking into account the probabilities involved. There are two cases to consider:

1) If two signs are known with probabilities P_1 and P_2 respectively and they enter into a sign relationship of probability P_3, the third sign in a relationship is indicated with a probability of:

$$P = P_1 + P_2 + P_3 - 2P_1P_2 - 2P_2P_3 - 2P_3P_1 + 4P_1P_2P_3$$

(assuming P_1, P_2 and P_3 are independent - which is not strictly true)

2) When the same sign is indicated with probabilities P_1 and P_2 by two different sign relationships, the combined probability is given by the expression

$$P = \frac{P_1P_2}{1 - P_1 - P_2 + 2P_1P_2}$$

The starting set signs are assigned probabilities of 1.0 (i.e. true) and the resulting sign indications are computed. The sign which is indicated with the highest probability is then accepted and this produces further sign indications which are added to the others. Again the sign with highest probability is accepted. The process continues in an iterative fashion until there are no signs with probabilities greater than a certain minimum value. This minimum probability is another of the user options in the LSAM module: the default value is 0.95.

Signs which have already been determined are constantly scrutinised throughout the process and if the probability of any sign drops below the minimum value it is rejected. It will sometimes happen that a sign has been accepted as one symbol and later it is indicated as a different symbol - in this case the sign which has the higher probability will be accepted.

When the sign determination process ends LSAM checks the consistency of the signs by rejecting all those with probabilities less than 0.9999 and re-determining the signs from those that remain. This continues until there is no change in signs from one re-determination to the next.

The program then sets up relationships between the symbols by locating sign relationships (triplets and quartets) in which the signs of all the reflections involved have been determined. The product of the signs is either +1 or a symbol relationship is created.

For example, suppose that:

$$S(\underline{h})S(\underline{k})S(\underline{h}-\underline{k}) = +$$

and that signs have been determined as follows:

$$S(\underline{h}) = A$$
$$S(\underline{k}) = ADF$$
$$S(\underline{h}-\underline{k}) = +$$

This gives the symbol relationship DF = + and its probability is calculated using the probabilities of the three signs and that of the sign relationship.

The program solves the strongest symbol relationships for the signs of symbols by an elimination process as illustrated by the above example. The possibility that one of these relationships may fail is taken into account which leads to several sets of signs being generated for the symbols. If there are n symbols, then the number of sign sets which will be generated is $(m+1)2^{n-m}$, where m is the number of symbols for which a solution has been found.

After signs have been obtained for the symbols, the sign determination for the reflections proceeds by using the results of the symbolic addition and rejecting all signs with a probability less than 0.9999. The remaining signs form a good starting point for further sign determination. Again using symbolic addition one set of signs is developed from this starting point for each set of signs for the symbols.

Figures of merit are computed along with each set of signs and these give an accurate indication of how good a particular solution is.

3.3 THE INCORPORATION OF LSAM INTO THE MITHRIL PACKAGE

Many changes have had to be made to the original LSAM system of programs in order to incorporate it as a module within the MITHRIL package. Some of these alterations have been carried out to include new theoretical developments whilst others were made to take advantage of the greater amounts of processor power available today. Another priority was to make LSAM as easy to use as possible with a high level of user interaction (should it be desired). Efforts were also made to incorporate features of FORTRAN 77, where possible, and to improve the commenting of the code.

There now follows a series of detailed accounts of the major changes made to LSAM together with the reasoning behind them. A table summarising the changes is also given.

3.3.1 FROM A THREE PROGRAM SYSTEM TO A SINGLE MODULE

As previously stated, LSAM in its original form consisted of three programs, LSAM0, LSAM1 and LSAM2, which had to be run consecutively. LSAM1 used data generated by LSAM0, while LSAM2 used data from LSAM1.

In order to facilitate the use of LSAM as a module in MITHRIL, LSAM0 was removed while LSAM1 and LSAM2 were combined and interfaces were written to enable the combination to be integrated into MITHRIL.

The reasoning behind the removal of LSAM0 was quite straightforward. It's role was to generate triple product sign relationships. However, since MITHRIL already contained

routines for the generation of triplet (as well as quartet and quintet) relationships, LSAM0 became redundant and was consequently removed.

3.3.2 DATA HANDLING

The original LSAM program system involved storing information on mass storage devices such as disks and magnetic tape. This situation was of course forced on the authors by the lack of memory available in the computers of the 1960's. With the greater amounts of memory available today, all the data required for the running of the LSAM module is either stored directly in memory, or else, in temporary files created by MITHRIL.

One example of this is that in the original system of programs, LSAM1 determined the relationships between the symbols and then wrote them on to a permanent file. LSAM2 then read the relationships off the file before going on to solve them. In the LSAM module the symbol relationships are stored in memory after being determined, ready for immediate use.

3.3.3 MAKING LSAM INTERACTIVE

In keeping with other computer programs of the time, the LSAM system of programs could only be run in batch mode. Parameters, such as lattice type and number of symbols, were entered in formatted form using cards. This had the effect of making LSAM a "black box" program: data and parameters for

running the program were input into LSAM and results duly came out but no information on the intervening processes was given.

As part of the effort to update LSAM, the LSAM module has been made interactive. Parameters for running the module are input via an on-screen menu which contains all the user options available (see later section). These options all have sensible default values so the program can be used without difficulty, even by inexperienced operators. There is output to the screen, to keep the user informed of what is happening, as well as to an output file which can be printed. As with the rest of MITHRIL, the LSAM module can also be run in batch mode.

3.3.4 THE USE OF HIGHER INVARIANTS IN LSAM

LSAM, in its original form used only the triple product sign relationship (eqn 1.23) to determine sign relationships. One major development which took place after LSAM was written was the derivation of formulae for estimating the magnitudes of four- and five- phase structure invariants[8,9,10,11]. These relationships contained new phase information and were subsequently incorporated successfully into direct methods computer programs[13].

In the case of centrosymmetric structures, quartet and quintet relationships become sign relationships (eqns 1.30,1.34). It was therefore a logical extension of the LSAM procedure to incorporate them into the module and take advantage of any new sign information they may generate.

A possible weakness in the LSAM module, as it currently stands, is that it can deal with a maximum of only

2000 invariants. If there are more than 2000 invariants available, LSAM chooses the top 2000, on the basis of probability, and discards the remainder. It may well prove advantageous to increase this number.

3.3.5 AN INCREASED NUMBER OF REFLECTIONS FOR LSAM

The LSAM program was altered so that the module can handle up to 800 reflections (i.e. the 800 reflections with the highest normalised structure factors which are to be phased). This compares to a previous maximum of 500.

The increase was made to make the LSAM module more uniform with the other modules in MITHRIL; each of which can deal with up to 800 reflections. Typically far fewer than 800 reflections will be used in a standard MITHRIL run. When dealing with difficult structures, however, it can often prove useful to extract phase information from as many reflections as possible.

3.3.6 MORE STARTING SET SYMBOLS

In certain cases where there is a paucity of relationships between the symbols it can be beneficial to increase the number of starting set symbols. The LSAM module now provides scope to do this: the maximum number of symbols to be assigned by LSAM has been increased from the original 6 to 11. The default number of starting symbols is 4. Increasing the starting set causes more sign sets to be generated.

3.3.7 MORE MEANINGFUL FIGURES OF MERIT FOR LSAM

The original LSAM system of programs produced four figures of merit for each sign set as follows:

$$F_1 = \sum_{\underline{h},\underline{k}} \sum s(\underline{h})s(\underline{k})s(\underline{h}-\underline{k})P_{\underline{hk}}$$

where $P_{\underline{hk}}$ is the probability that the product of the three signs is positive.

$$F_2 = \sum_{\underline{h},\underline{k}} \sum s(\underline{h})s(\underline{k})s(\underline{h}-\underline{k}) \left| E_{\underline{h}} E_{\underline{k}} E_{\underline{h}-\underline{k}} \right|$$

F_2 measured the same property as F_1 but tended to be more sensitive.

F_3 = The total number of sign relationships which hold

F_4 = The number of signs determined

For a correct set of signs all four figures of merit should be high.

In fact, these figures of merit were unique to the LSAM package and not particularly reliable. For these reasons it was decided to replace the LSAM figures of merit with those already used in MITHRIL: ABSFOM, PSIZERO, RESID, NQEST, NQINT and CFOM (refer to Introduction for a full description of these quantities). This had the added advantage of making the

LSAM module easier to use as well as giving it a more "MITHRIL-like" appearance.

Some figures of merit behave differently in the LSAM module than in the tangent refinement routine. ABSFOM and RESID both tend to be a bit lower in the LSAM module because there is no phase refinement. NQEST appears to be more sensitive in the LSAM module, often giving a large range of values.

The original LSAM system of programs had difficulty in dealing with space groups which contained no translational symmetry other than a centred lattice such as $P\bar{1}$. For such structures a set of signs is generated with all signs positive. This had the effect of making all four LSAM figures of merit the maximum possible but the solution was invariably wrong.

The LSAM module deals with the problem presented by these space groups in two ways. The first is by using negative quartet sign relationships. If there are no quartet relationships present then the module simply ignores the trivial solution with all signs positive and moves on to the next.

3.3.8 SUMMARY OF ALTERATIONS MADE TO LSAM

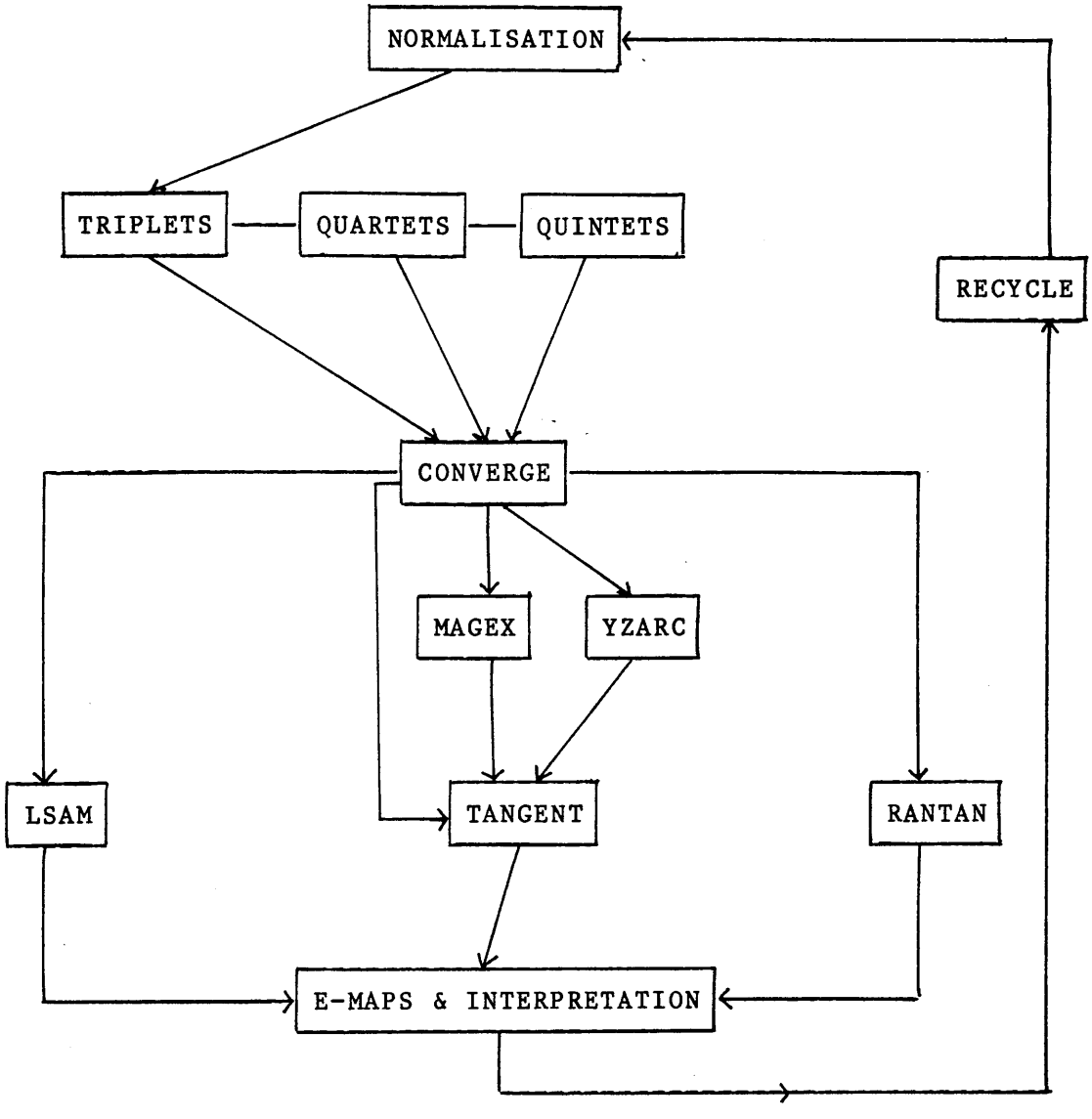
The following table summarises the changes made to LSAM to incorporate it, in a substantially upgraded form, into the MITHRIL package.

Originally	Now
3 separate programs	LSAM0 removed. LSAM1 & LSAM2 combined and integrated into MITHRIL as a single module
Information fed in by magnetic tape	Information stored in memory
Batch jobs only	Interactive (with option of running in batch mode)
Parameters input via cards	Parameters selected from on-screen menu
Triplet relationships	Triplet, quartet and quintet relationships
Maximum of 500 reflections	Maximum of 800 reflections
Up to 6 symbols	Up to 11 symbols
LSAM figures of merit	More meaningful figures of merit

A program listing of the LSAM module is included in the appendix. The code varies markedly from the original version of LSAM in order to accommodate the features listed above. It must be stressed, however, that in spite of all these changes the method used in LSAM remains essentially unaltered.

3.4 HOW LSAM FITS INTO THE MITHRIL PACKAGE

The following flowchart shows all the modules available in MITHRIL and illustrates how LSAM fits into the package.



3.5 RUNNING THE LSAM MODULE IN MITHRIL

All of the modules within the MITHRIL package are displayed on the master menu as follows:

```
The following modules are currently available:
 NORMAL    (IK,NB,ISC,MAXDUP )     TRIPLET (NO OF REFLECTIONS)
 QUARTET   (NO OF REFLECTIONS)     QUINTET (NO OF REFLECTIONS)
 RECYCLE   ( FOURIER / KARLE )     CONVERG (NUMB,IQ4,IQ5     )
 TANGENT   (                 )     LSAM    (                 )
 RANTAN    (NO OF PHASE SETS )     MAGEX   (NO OF PHASE SETS )
 YZARC     (NO OF PHASE SETS )     MAPS    (PHASE SETS NO(S) )
 PATTERSON(                  )     REVIEW  (                 )

In addition the commands: TITLE,END,MENU,LEVEL,NOPRINT,PRINT,
                          DEFAULT,HARD,VERY HARD,MODEL,SHOW,X
may be issued at any time.
Only the first 4 characters of any command are required.
<CR> or a call to another module terminates current input.
```

In order to run the LSAM module, the user simply types "LSAM" in response to the above menu. Provided that all of the other necessary modules (NORMAL, TRIPLET, CONVERGE and sometimes QUARTET) have already been run, the LSAM menu will appear. Alternatively, if LSAM is selected immediately after the NORMAL module has been run then the TRIPLET and CONVERGE (and possibly QUARTET*) modules will be run using default parameters rather than under user control.

The LSAM menu is of the form:

```
The following commands are currently available:
 LSAM   (                                                        )
 SYMBOL(NO OF SYMBOLS TO BE ASSIGNED                      (   4))
 SIGN   (MIN PROB ACCEPTABLE FOR A SIGN DETERMINATION (0.95))
 NUMBER (NO OF RELATIONSHIPS TO BE CONSIDERED             (  10))
 LIMIT  (SIGNS PREDICTED WITH PROB > LIMIT TO BE USED (0.80))
 ORIGIN (USE LSAM ORIGIN AND STARTING SET                        )
 LIST   (PRINT TABLES OF SIGNS AND SYMBOL RELATIONSHIPS          )

In addition the commands: TITLE,END,MENU,LEVEL,NOPRINT,PRINT,
                          DEFAULT,HARD,VERY HARD,MODEL,SHOW,X
Only the first 4 characters of any command are required.
<CR> or a call to another module terminates current input.
```

To run LSAM under default values, a carriage return is entered in response to this menu.

The LSAM module can only be entered if the structure is centrosymmetric. The symbolic addition method can be modified to deal with the non-centrosymmetric case[4,7], however it tends to be a great deal less succesful in dealing with such structures. It was, therefore, not thought to be a worthwhile project to alter the LSAM code to deal with the non-centrosymmetric case. It should also be noted that it is difficult to automate the symbolic addition procedure in non-centrosymmetric space groups.

If the "LSAM" command is selected for a non-centrosymmetric structure the following error message appears:

STRUCTURE IS NON-CENTROSYMMETRIC - CANNOT USE LSAM

The program then displays the master menu and awaits further instruction.

Notice that although these instructions refer to running LSAM interactively it is a simple matter to run the module in batch mode. In that instance commands are entered via a command file rather than through the keyboard.

(* - Footnote: As well as running when user requested, the QUARTET module is also run automatically where the space group is symmorphic, or where the HARD or VERY HARD command has been selected from the master menu.)

3.6 USER OPTIONS AVAILABLE IN THE LSAM MODULE

As can be seen from the LSAM menu given previously, there are 6 options available in the module. A description of each option is given below.

3.6.1 SYMBOL no. of symbols

This command allows the user to choose how many symbols, up to a maximum of 11, to be used in the symbolic addition procedure. The default value is 4, however using more symbols generates more sign sets which may be useful in the case of difficult structures.

Note that except in the case where the ORIGIN option is also used, the maximum number of symbols permitted is limited by the number of starting set reflections from convergence.

3.6.2 SIGN prob

This sets the minimum probability acceptable for a sign determination by symbolic addition. This should be set at a fairly high value - the default is 0.95.

3.6.3 NUMBER no. of relationships

This option determines the number of symbol relationships to be considered in the solution for the symbols. It may be set at a number higher than the actual number of relationships in order to ensure that all the

relationships present are considered. The default value is 10.

3.6.4 LIMIT prob

All signs predicted with a probability greater than LIMIT will be included in the final output. The signs determined by solving the symbol relationships are subject to the SIGN parameter above. All signs used will therefore have a high probability and so the parameter LIMIT can safely be set at a relatively low value. The default is 0.80.

3.6.5 ORIGIN

If this option is selected then the LSAM module determines for itself origin defining reflections and also the reflection in the starting set of signs. These may or may not be the same as those determined by the convergence procedure which are used by default. This command is often of use in tandem with SYMBOL.

3.6.6 LIST

This command causes tables of signs developed by symbolic addition and relationships between the symbols to be displayed in the output file.

3.7 THE USE OF LSAM IN STRUCTURE SOLVING: DIAMANTANE-4-OL

(Diamantane-4-ol is one of the difficult structures

which is considered in the next chapter)

MITHRIL was run on default values up to the LSAM module for the Diamantane-4-ol data set. That is to say: the structure factors were normalised using a Wilson plot, 5942 triplet relationships were generated using the top 250 reflections, 630 quartet relationships were generated using the top 100 reflections and the convergence map had 6341 contributors.

The LSAM module was also run on default parameters with the exception that the LIST command was used.

The origin defining reflections and the starting set of signs used by LSAM are those determined by the convergence module. The techniques used by LSAM to determine signs can be described in three parts:

(i) Symbolic Addition

Symbolic addition is used to determine new signs using as a starting point the origin reflections and the starting set reflections. These are shown in figurel.

The new indications of sign are accepted one at a time, the one accepted always being that with the highest probability. This continues until the highest probability for a new indication is lower than the user-fixed limit (in this case 0.95). For Diamantane-4-ol the sign determination process concludes after 175 signs have been determined.

Figure2 shows the sign (if any) determined for each reflection together with its associated probability. Note

LSAM (LOGICAL SYMBOLIC ADDITION METHOD)

DIAMANTANE-4-OL (FILE1)

ORIGIN DEFINING REFLECTIONS

CODE	H	K	L	SYMBOL	CODE	H	K	L	SYMBOL	CODE	H	K	L	SYMBOL
3	2	-1	6	+	4	9	7	3	+					

SYMBOLS HAVE BEEN ALLOCATED AS FOLLOWS

CODE	H	K	L	SYMBOL	CODE	H	K	L	SYMBOL	CODE	H	K	L	SYMBOL	CODE	H	K	L	SYMBOL
5	9	9	2	A	9	10	5	3	B	18	1	0	6	C	37	8	5	4	D

PROBABILITY LIMIT FOR ACCEPTANCE OF A SIGN SYMBOL IS 0.9500

DURING SIGN DETERMINATION LIMIT HAS BEEN CHANGED TO 0.9498

175 SIGNS HAVE BEEN DETERMINED OUT OF A POSSIBLE 250

Figure1: The Starting Set

TABLE OF SIGNS DEVELOPED BY SYMBOLIC ADDITION

1	C	0.0000	2	B	0.0000	3	+	1.0000	4	+	1.0000
5	A	1.0000	6	C	0.0000	7	+	0.0000	8	AC	0.0000
9	B	1.0000	10	C	1.0000	11	A	1.0000	12	AC	1.0000
13	B	1.0000	14	A	0.0000	15	C	1.0000	16	C	1.0000
17	C	0.0000	18	C	1.0000	19	C	0.0000	20	AC	0.0000
21	C	1.0000	22	B	1.0000	23	B	0.0000	24	B	1.0000
25	B	0.0000	26	B	1.0000	27	B	0.0000	28	B	0.0000
29	AC	1.0000	30	AC	1.0000	31	AC	0.9996	32	AC	0.0022
33	AC	0.0000	34	C	0.0000	35	C	1.0000	36	C	0.0000
37	D	1.0000	38	AC	1.0000	39	C	1.0000	40	B	0.0007
41	+	0.0000	42	B	0.0000	43	A	1.0000	44	AC	1.0000
45	AC	0.0552	46	B	1.0000	47	C	1.0000	48	C	0.0000
49	+	0.0000	50	A	0.0064	51	AC	0.0000	52	AC	1.0000
53	AC	0.0000	54	A	0.9986	55	AC	0.9998	56	C	0.0000
57	C	0.9963	58	+	1.0000	59	C	0.9999	60	+	1.0000
61	B	0.0000	62	C	0.0002	63	C	1.0000	64	B	0.0001
65	B	0.9744	66	B	0.9993	67	C	0.9241	68	B	0.9658
69	AC	0.0050	70	B	0.0001	71	B	0.0002	72	C	0.0000
73	AC	1.0000	74	+	0.9798	75	AC	0.9990	76	A	0.9416
77	B	0.0001	78	+	0.9998	79	AC	1.0000	80	C	0.0000
81	+	1.0000	82	AC	1.0000	83	C	1.0000	84	AC	0.0000
85	C	1.0000	86	B	0.0005	87	B	0.9785	88	AC	0.9975
89	B	0.9960	90	AC	0.1915	91	C	0.9794	92	+	0.0000
93	B	0.9968	94	B	0.9994	95	C	1.0000	96	A	0.9790
97	AC	0.0148	98	B	0.0016	99	C	0.0000	100	B	1.0000
101	C	0.0000	102	C	1.0000	103	C	0.0000	104	A	0.0000
105	AC	1.0000	106	C	1.0000	107	B	0.0000	108	A	0.0000
109	B	0.0007	110	B	0.0129	111	B	0.0063	112	B	0.0005
113	+	0.0005	114	A	0.0114	115	A	0.0067	116	B	0.0474
117	C	0.2891	118	B	0.8787	119	A	1.0000	120	+	0.0027
121	B	0.9790	122	+	0.0008	123	AC	0.0001	124	+	1.0000
125	AC	0.9677	126	B	0.0071	127	+	0.0009	128	+	0.0271
129	A	0.9999	130	+	0.9949	131	C	0.9552	132	C	0.0074
133	C	0.0001	134	AC	0.9759	135		0.5000	136	B	0.9990
137	B	0.9989	138	AC	0.0043	139	A	0.9965	140	AC	0.2108
141	C	0.8808	142	B	0.0879	143	AC	0.9876	144	C	0.9962
145	AC	0.9999	146	A	0.9729	147	A	0.9763	148	+	0.0119
149	C	0.0137	150	B	0.1016	151		0.5000	152	AC	0.1947
153	A	0.0015	154	B	0.0271	155	A	0.2850	156	B	0.9886
157	A	0.1443	158	A	0.0246	159	C	0.9995	160		0.5000
161	AC	0.9153	162		0.5000	163	C	0.0895	164	AC	1.0000

Figure2: Table of signs developed by Symbolic Addition

165	C	0.9979	166	AC	0.1708	167	AC	0.0003	168	AC	0.9020
169		0.5000	170	AC	0.7824	171	+	0.9874	172	AC	0.0895
173	B	0.0001	174	+	0.9910	175	+	0.0001	176	+	0.0399
177	A	0.9976	178		0.5000	179	B	0.9137	180	C	0.0004
181	B	0.0563	182	AC	0.1545	183	A	0.2244	184	C	1.0000
185	A	0.0691	186	A	0.2535	187	C	0.9754	188	B	0.8909
189	A	0.9585	190		0.5000	191	C	0.9866	192	B	0.9227
193	A	0.0817	194	C	0.0336	195	AC	0.0134	196	B	0.1111
197	B	0.9744	198		0.5000	199	AC	0.0502	200	AC	0.1171
201	C	0.0000	202		0.5000	203	B	0.7990	204	C	0.0666
205	C	0.0377	206	AC	0.9913	207		0.5000	208	A	0.9282
209	B	0.0573	210		0.5000	211	AC	0.9458	212		0.5000
213	B	0.3100	214		0.5000	215	AC	0.7068	216		0.5000
217	B	0.0206	218	AC	0.2108	219	AC	0.0911	220		0.5000
221	C	0.9689	222		0.5000	223	B	0.2497	224	C	0.7540
225	+	0.9213	226	AC	0.7311	227		0.5000	228		0.5000
229	+	0.9802	230	+	0.9881	231	AC	0.8085	232	+	0.9718
233	C	0.0787	234	C	0.2075	235		0.5000	236		0.5000
237	+	0.9997	238		0.5000	239	C	0.3015	240	B	0.0493
241	C	0.0176	242		0.5000	243		0.5000	244	AC	0.9526
245	C	0.0323	246	+	0.0630	247		0.5000	248	C	0.1301
249	C	0.9920	250		0.5000						

Figure2: (Continued)

that probabilities of less than 0.500 indicate the sign has been determined as negative.

For example:

S_{65}= B with probability 0.9744

S_{32}= AC with probability 0.0022

which is the same as saying

S_{32}= -AC with probability 0.9978

The table also shows other sign indications with probabilities lower than 0.95. These are not used to determine new signs.

(ii) Finding Relationships

LSAM finds relationships between the sign symbols by examining triplet and quartet relationships which involve reflexions whose signs have been determined above to be greater than 0.95. In the case of the triplet relationship, the sign symbols assigned to each of the three reflexions are combined into a symbol relationship.

If the same relationship is indicated several times then the overall probability of this relationship is calculated by combining the individual probabilities.

Figure3 shows the symbol relationships generated for Diamantane-4-ol together with their probabilities.

(iii) Solving the Symbol Relationships

LSAM finds the strongest relationship which can be solved for all or some of the symbols and finds that set of

DIAMANTANE-4-OL (FILE1)

RELATIONSHIPS BETWEEN SYMBOLS

RELATIONSHIP		PROB	RELATIONSHIP		PROB	RELATIONSHIP		PROB
C	= S(1)	1.0000	AC	= S(2)	0.0000	B	= S(3)	0.0000
AD	= S(4)	0.0000	BD	= S(5)	0.0000	C	= S(6)	0.9974
B	= S(7)	0.0047	AC	= S(8)	0.0075	BD	= S(9)	0.9794
A	= S(10)	0.9346	D	= S(11)	0.8629	BD	= S(12)	0.2932

THE TOP 10 SYMBOL RELATIONSHIPS WERE USED

SIGNS WITH PROBABILITY GREATER THAN 0.800 ARE ACCEPTED

Figure3: Symbol Relationships generated for Diamantane-4-ol

FIGURES OF MERIT

SET	ABS	PSIZERO	RESID	MQEST	MQINT	SIGNS OF SYMBOLS
1	0.55	1.977	35.65	0.02	0.00	+ + + + + + + + + + +
2	0.57	2.170	33.57	-0.05	0.00	- + + - + + + + + + +
3	0.59	1.856	31.60	0.04	0.00	+ + + - + + + + + + +
4	1.10	0.786	14.15	-0.55	0.00	+ - + - + + + + + + +
5	0.50	2.440	38.97	-0.06	0.00	- - - + + + + + + + +

Figure4: Figures of Merit for the Sign Sets

signs for the symbols which satisfies all the relationships or will only make one of them fail. Normally there are several such sign sets which fulfill these requirements. By use of the signs of the starting set symbols, signs are calculated for most of the other reflexions using Figure2.

LSAM terminates by computing figures of merit for each sign set (Figure4). The MAPS module was then selected and an electron density map was calculated for set 4 which yielded all 15 non-hydrogen atoms on interpretation.

3.8 RESULTS

Centrosymmetric structures tend to be easier to solve than non-centrosymmetric structures. This is illustrated by the fact that only 4 out of the 25 data sets in the Sheldrick database of structures that are resistant to the techniques of direct methods are centrosymmetric. It was therefore necessary to test LSAM on structures from outwith the database.

The following structures were used:

No.	Structure	Spacegroup	Formula	Z	Ref.
1	DIAMANTANE-4-OL	$P4_2/n$	$C_{14}H_{20}O$	8	14
2	NEWQB	$P\bar{1}$	$C_{24}H_{20}N_2O_5$	4	15
3	QUINOL	$R\bar{3}$	$C_6H_6O_2$	54	16
4	BSEMI	C2/c	$C_{10}H_{17}N_3O$	8	19
5	ECON	$P2_1/c$	$C_{18}H_{15}Cl_3N_2O$	4	17
6	ANDY2	$P2_1/c$	$C_{16}H_{14}F_3O_3N$	4	20
7	CSGLU	C2/m	$Cs_{10}C_{40}H_{64}O_{48}$	8	21
8	HOV1	C2/m	$Pr_{14}Ni_6Si_{11}$	4	18

Tangent refinement was also used on the structures to give an indication of how LSAM compares to it. The results are tabulated below. The figures of merit quoted are for the "best" phase set for each structure i.e. the phase set whose E-map yielded the most correct atoms. The rank refers to the CFOM value of the phase set considered with respect to those of the other phase sets produced.

Using LSAM:

Structure	ABSFOM	PSIZERO	RESID	NQEST	Rank	Atoms found*
1	1.10	0.79	14.15	-0.55	1	16/16
2	1.54	1.04	20.77	0.47	1	17/62
3	1.07	0.92	12.41	-0.69	1	24/24
4	1.34	0.85	26.08	-0.69	1	14/14
5	1.18	1.13	15.36	-0.72	1	24/24
6	1.05	0.87	19.59	-	1	23/23
7	1.09	1.46	7.15	-0.76	1	$1/1^+$
8	1.01	0.95	3.21	-0.48	1	$6/7^+$

Using multisolution methods:

Structure	ABSFOM	PSIZERO	RESID	NQEST	Rank	Atoms found*
1	1.18	0.68	17.89	-0.60	1	16/16
2	1.21	0.75	21.92		1	60/62
3	1.07	0.89	11.78	-0.69	1	24/24
4	1.37	0.85	27.65	-0.69	1	14/14
5	1.19	1.12	8.72	-0.72	1	24/24
6	1.08	0.84	15.41		1	23/23
7						0/1+
8						0/7+

* - "Atoms" refers to the number of non-hydrogen atoms in the asymmetric unit of the structure.

+ - In heavy atom cases, "atoms" refers to the number of heavy atoms in the asymmetric unit of the structure.

Note that in the multisolution table, no figures of merit are listed for CSGLU or HOV1 as no appropriate solutions could be found for these structures.

3.9 DISCUSSION

Examination of the results shows that, as expected, tangent refinement is a more powerful method - in the sense that it is more likely to find a recognisable molecular fragment - than symbolic addition. This is highlighted by the

results obtained with NEWQB where LSAM had particular difficulty, eventually finding only a 17 atom fragment as compared to the 60 atom fragment found by tangent refinement. Never the less the results obtained using LSAM are very encouraging. It was possible to solve most of the structures using LSAM. In fact, given a routine centrosymmetric structure, it can safely be said that application of LSAM will very probably lead to the correct structure.

It should be remebered, however, that the LSAM module in MITHRIL has only been tested on a limited number of structures and space groups. Whilst the results so far, even with difficult structures, have indeed been very encouraging it is recognised that the module requires even more rigorous testing.

The QUINOL structure proved to be interesting. Both LSAM and tangent refinement managed to find all the non-hydrogen atoms. However, in the case of LSAM, this was only achieved by reducing the number of starting set symbols from 4, the default value, to 3. This suggests that the fourth symbol was involved in bad triplet and/or quartet relationships; leading to wrong sign indications. So although increasing the number of starting set symbols can often be useful in the case of difficult structures, care must be taken in doing so. As with tangent refinement, the symbolic addition method depends crucially on the choice of starting set reflections.

The heavy atom structures, CSGLU (caesium glutarate) and HOV1, represent notable successes for LSAM. Exhaustive trials with tangent refinement methods, including the use of the Hull and Irwin statistically weighted tangent formula, were carried out on CSGLU with no success. This was despite investigating E-maps for many phase sets rather than just the top ranked one - it was thought that the figures of merit may

not be very reliable since the structure contains heavy atoms. The structure was solved by using symbolic addition, in the form of LSAM, instead of tangent refinement and indeed, this is a case of symbolic addition succeeding where tangent refinement failed. Using LSAM the structure solved easily: default parameters were used throughout the rest of the MITHRIL program as well as in the LSAM module itself. The top phase set was used to calculate an E-map from which a caesium atom was located - refinement techniques were used to find the rest of the structure. HOV1 which is one of the structures discussed in greater detail in Chapter4, was found to behave in a similar way to CSGLU: the Pr atoms were only located after tangent refinement methods had been given up in favour of symbolic addition.

The fact that LSAM dealt successfully with the heavy atom structures is not a great surprise. Tangent refinement can find difficulty with such structures because it refines the phase values to such an extent that the phase relationships become over-consistent. As mentioned in the introduction, the Hull and Irwin weighted tangent formula was devised to prevent over-consistency occurring. LSAM, of course, does not refine the phases and so avoids the problem.

It has been shown that LSAM can solve certain structures where all other methods fail. As such it fits in well with the philosophy of MITHRIL: to have as many alternatives available as possible to try when difficulties are encountered.

So succesful has the implementation of the LSAM module been that it is proposed, provided the program deals with more structures and space groups successfully, to make LSAM and not TANGENT (the tangent refinement module) the default module in MITHRIL for centrosymmetric structures. This may seem rather surprising considering the opening statement of this

discussion. However, it is thought that the valuable amounts of processor time saved by running LSAM, rather than TANGENT, will quickly justify this decision.

REFERENCES

1. G.Germain & M.M.Woolfson (1968). Acta Cryst.,B24,91

2. P.Main (1985). In Crystallographic Computing 3. edited by G.M.Sheldrick, C.Kruger & R.Goddard, Oxford University Press

3. H.Hauptman & J.Karle (1956). Acta Cryst.,9,635

4. H.Schenk & C.T.Kiers (1985). in Crystallographic Computing 3, edited by G.M.Sheldrick, C.Kruger & R.Goddard, Oxford University Press

5. H.Hauptman & J.Karle (1953). Solution of the Phase Problem.I. The Centrosymmetric Crystal. A.C.A. Monograph No.3. Pittsburgh: Polycrystal Book Service

6. H.Hauptman & J.Karle (1959). Acta Cryst.,12,93

7. J.Karle & I.L.Karle (1966). Acta Cryst.,21,849

8. H.Hauptman (1977). Acta Cryst., A33, 556-564

9. H.Hauptman (1977). Acta Cryst., A33, 565-568

10. C.Giacovazzo (1976). Acta Cryst., A32, 91-99

11. N.van der Putten & H.Schenk (1977). Acta Cryst., A33, 856-858

12. A.A.Freer & C.J.Gilmore (1980). Acta Cryst., A36, 470-475

13. G.Germain, P.Main & M.M.Woolfson (1974). In Direct Methods in Crystallography (Lecture Notes), N.A.T.O. Advanced Study Institute

14. J.Rogers & O.Kennard, unpublished

15. R.Grigg, J.Kemp, G.M.Sheldrick & J.Trotter (1978). J.C.S.

Chem. Comm. 109-111

16. S.C.Wallwork & H.M.Powell (1980). J.Chem.Soc. Perkin 2, 641-646

17. A.A.Freer, A.Pearson & E.G.Salole (1986). Acta Cryst. , C42, 1350-1352

18. E.Hovestreydt, R.Klepp & E.Parthé (1983). Acta Cryst., C39, 422-425

19. G.A.Sim, unpublished

20. A.A.Freer, unpublished

21. A.A.Freer & A.MacDonald, to be published

CHAPTER4

MITHRIL AND THE SHELDRICK DATABASE OF DIFFICULT STRUCTURES

4 MITHRIL AND THE SHELDRICK DATABASE OF DIFFICULT STRUCTURES

4.1 INTRODUCTION

There exists a database, compiled by Sheldrick, containing X-ray data for 25 structures which are known to be difficult to solve using the techniques of direct methods. Sheldrick himself made use of these structures when constructing the crystallographic package SHELXTL[1].

The reasons why these structures are resistant to direct methods computer programs are varied: some because of their space group symmetry; some because they contain heavy atoms; some because of poor quality X-ray data; and others simply because they have a large number of atoms in the asymmetric unit. For many of the structures, however, there is no clear cut reason why they should be difficult to solve and it is these structures which are of greatest interest.

It was felt that a great deal could be learned by using the direct methods program, MITHRIL, to try to solve these structures. It would help become apparent the strengths and weaknesses of a relatively new program package as well as, hopefully revealing some of the reasons why the structures were difficult to solve. The information gained thus could then be used to improve and strengthen the MITHRIL package.

The table below summarises the relevant crystallographic data for the 25 structures on the database.

Structure	Space gp	Formula	Z	Ref
DIAMANTANE-4-OL	$P4_2/n$	$C_{14}H_{20}O$	8	2
QUINOL	$R\bar{3}$	$C_6H_6O_2$	54	3
BOBBY	$P2_13$	$NaCaN(CH_2CO_2)_3$	4	4
HOV1	C2/m	$Pr_{14}Ni_6Si_{11}$	4	5
SELENID	$P2_1$	$C_{22}H_{28}O_2Se$	2	6
AZET	$Pca2_1$	$C_{21}H_{16}ClNO$	8	7
CUIMID	$P3_221$	C_6H_8ClCu	6	8
TUR10	$P6_322$	$C_{15}H_{24}O_2$	12	9
BED	I4	$C_{26}H_{26}N_4O_4$	8	10
LOGANIN	$P2_12_12_1$	$C_{17}H_{26}O_{10}$	4	11
DIOL	$I\bar{4}2d$	$C_{10}H_{18}O_2$	16	
APAPA	$P4_12_12$	$C_{30}H_{37}N_{15}O_{16}P_2\cdot 6H_2O$	8	12
TPALA	$P2_1$	$C_{28}H_{42}O_7N_4$	2	13
NO55	Fdd2	$C_{20}H_{24}N_4$	16	14
MGHEX	$P3_1$	$C_{48}H_{68}N_{12}O_{12}Mg\cdot 2ClO_4\cdot 4CH_3CN$	3	15
TOTC	$P6_1$	$C_{33}H_{36}O_6\cdot 0.2(C_{16}H_{33}OH)$	6	16
TVAL	P1	$C_{54}H_{90}N_6O_{18}$	2	17,18
TPH	$C222_1$	$C_{24}N_2H_{20}$	12	19
NEWQB	$P\bar{1}$	$C_{24}H_{20}N_2O_5$	4	20
GOLDMAN2	Cc	$C_{28}H_{16}$	8	21
MUNICH1	C2	$C_{20}H_{16}$	8	22
MBH2	P1	$C_{15}H_{24}O_3$	3	23
PGE2	P1	$C_{20}H_{32}O_5$	1	24
SUOA	$P2_12_12_1$	$C_{28}H_{38}O_{19}$	4	25
WINTER2	$P2_1$	$C_{52}H_{83}N_{11}O_{16}\cdot 3CH_2Cl_2$	2	26

4.2 MITHRIL UNDER DEFAULT PARAMETERS

When using MITHRIL on an X-ray data set for a particular structure it is sensible to begin by using default

parameters throughout the program. The following modules are entered in a standard run through the MITHRIL package:

NORMAL TRIPLET CONVERGE TANGENT MAPS

For structures without translational symmetry i.e. those in symmorphic space groups, the QUARTET module is run between TRIPLET and CONVERGE.

The default values, NT and NQ, for the number of reflections to be used in generating triplet and quartet phase relationships respectively are calculated as follows:

$$NT = 4 * NAT + 100 + 10 * ISYMP$$

$$NQ = 100 + 15 * ISYMP$$

where:

NAT = No. of atoms in the asymmetric unit

ISYMP = 0/1 for non-symmorphic/symmorphic space group

4.3 RESULTS

This section contains a series of detailed reports of how MITHRIL was used to solve each of the structures on the database. Any difficulties that were encountered are discussed together with an account of the strategies used to overcome them. Tables 4.1 and 4.2 (which are located at the end of this section) summarise some of the more relevant results obtained for the structures. Note that in the NORMAL column of Table 4.1 the numbers 0 and 1 indicate normalisation by Wilson plot and K-curve methods respectively whereas an

entry of -2 denotes the E-magnitudes were read directly from the data file.

4.3.1 STRUCTURE1: DIAMANTANE-4-OL

DIAMANTANE-4-OL was included in the database as an example of a centrosymmetric tetragonal space group ($P4_2/n$). This simple structure presented no problems to MITHRIL and was solved using default parameters throughout the program. Of the 16 phase sets generated by tangent refinement, 2 sets had much better figures of merit than the other 14. The E-map calculated for the set with the biggest combined figure of merit gave all 15 non-hydrogen atoms in the structure; in fact they were the top 15 peaks in the map.

This structure has also been used extensively in testing the new symbolic addition module, LSAM, in MITHRIL where again it solved under default parameters.

4.3.2 STRUCTURE2: QUINOL

This structure was originally solved by Patterson superposition, however, again MITHRIL had no difficulty in locating all 24 non-hydrogen atoms. The program solved the structure using default parameters throughout but notice that because the space group ($R\bar{3}$) is symmorphic, quartet invariants were automatically generated.

The E-magnitudes were read directly from the data file; triplets and negative quartets were generated using the

top 250 and 115 reflections respectively; and phase expansion and refinement were both performed by the tangent formula. Only 6 phase sets needed to be generated before MITHRIL found the correct solution which led to all 3 molecules in the asymmetric unit being discovered.

4.3.3 STRUCTURE3: BOBBY

BOBBY can be regarded as a heavy atom structure since it contains a calcium atom. The structure crystallises in space group $P2_13$ and so, while it is not centrosymmetric, it does have 12 symmetry operations. MULTAN was originally used to solve the structure which it did with relative ease. Similarly, few problems were encountered using MITHRIL, although the structure did not solve using default parameters throughout the program.

Despite the fact that the correct solution was not found, a few interesting features arose during the default run which were to point the way to solving the structure. The E-magnitudes read from the data file were poorly distributed with several very high values: the top 6 E's were all greater than 4.0. Large E-magnitudes such as these cause problems in the tangent refinement process as nearly all of the reflections will be assigned weights of unity and so the tangent formula will effectively be unweighted. Because of the large number of symmetry operations, a vast number - almost 21000 - triplets were generated for the top 180 reflections. Although only the top 9450 of these phase relationships were saved, the convergence map still had an enormous number of contributors.

The structure was found by doing two simple things:

firstly, letting the program calculate its own E-magnitudes and then reducing the number of triplets. Structure factors were normalised by means of a K-curve and this time no large E values were observed; the highest value being 2.24. Note that the E-magnitudes on the database were normalised using a Wilson plot method. This structure provides a rather extreme example of the different E-magnitude distributions that can come about from normalisation using the K-curve and Wilson plot methods. Triplets were generated using only the top 117 reflections. A total of 6198 phase relationships were produced leading to a convergence map which still had many contributors, but was not quite so enormous as before. The figures of merit generated by tangent refinement were poor but this is often the case when dealing with heavy atom structures. An E-map was calculated for the top ranked phase set from which the calcium, sodium and nitrogen atoms were located. All of these atoms lie on special positions which is a source of problems to programs like MITHRIL as all direct methods statistics are computed under the assumption that atoms are randomly distributed within the unit cell.

4.3.4 STRUCTURE4: HOV1

This is a good illustration of the use of direct methods to solve a typical inorganic structure. The space group, C2/m, is symmorphic and all atoms lie on special positions.

The major problem encountered with this structure was that the tangent formula caused the phases to become over-consistent. As a result all the phase sets look artificially good and the difficulty was in picking the true solution. Use of the Hull and Irwin statistically weighted tangent formula

improved the situation somewhat but still no solution could be found. The structure was eventually elucidated using the symbolic addition module, LSAM. This module is discussed in greater detail in Chapter 4.3. Unlike tangent formula methods, LSAM does not refine the phases and so the problem of over-consistency was avoided.

Note that rather than putting in a structure factor for praesodymium, atoms of this type were approximated to barium: the atom of nearest atomic number recognised by MITHRIL.

MITHRIL was run on default values up to the LSAM module i.e. structure factors were normalised using a least squares straight line through a Debye curve, triplet relationships were generated for the top 250 reflections and negative quartets for the top 115.

In the LSAM module, 7 starting symbols and 2 origin defining reflections were used to generate 180 sign indications with a probability limit for acceptance of 0.99. 41 symbol relationships with probability greater than 0.90 were produced from these sign indications. The E-map calculated for the best sign set yielded 6 Pr atoms. The remaining 8 Pr atoms were located using weighted Fourier syntheses.

4.3.5 STRUCTURE5: SELENID

SELENID is another of the heavy atom structures on the database: selenium has atomic number 34 and therefore has far greater scattering power than either carbon or oxygen. It was originally solved using SHELXTL but only after problems with

pseudosymmetry were overcome. MITHRIL managed to solve the structure with no major problems.

The convergence map generated using default parameters in MITHRIL had many contributors, however, many duplicate solutions were generated by tangent refinement: of the 24 solutions produced only 5 were unique. Even so, as with other heavy atom structures, there was still great variation in the figure of merit values within these 5 phase sets. A feature of the figures of merit were the high ABSFOM and low RESID values.

It was noticed that the scheme to weight triplets according to their squared $(\sin\theta/\lambda)$ values improved the figures of merit and it was therefore used. When quartets were introduced to the structure determination process it was found that very few negative quartets were produced. The situation was improved slightly by using a K-curve to normalise the structure factors rather than reading them directly from the data file.

SELENID was solved using the top 300 reflections to generate triplets and the top 225 to produce negative quartets; the triplet weighting scheme was also used. Phase refinement and expansion were carried out using the standard tangent formula. The figures of merit produced for the phase sets did not look encouraging. The best phase set was the one with the top ABSFOM value. An E-map was calculated for this phase set from which a 10 atom fragment, including a 6-membered ring, was found. The fragment did not contain the Se atom, however, it and most of the other atoms were located using Fourier recycling.

4.3.6 STRUCTURE6: AZET

AZET is one of the more famous structures on the database: it was one of a number of MULTAN failures reported by Lessinger[27] where it was shown to be unstable under tangent refinement. Using MITHRIL, AZET provided a few interesting but not too serious problems before being solved. This was another of the structures where the weighting scheme for triplets based on squared $(\sin\theta/\lambda)$ values proved to be useful - this scheme is described in detail in Chapter2.

Some useful information was gained by running MITHRIL using default parameters throughout. The convergence map looked very healthy with many contributors, however, the program had some difficulty in defining a good origin. AZET belongs to space group $Pca2_1$ which requires three origin defining reflections including one which defines the enantiomorph. MITHRIL chose a poor reflection to define the enantiomorph: it participated in very few phase relationships. The figures of merit produced by the standard tangent formula were poor and, although using the statistically weighted tangent formula caused an improvement, the correct solution was not found. Random phases were also tried but again without success.

It was decided to use quartet phase relationships as the structure was proving to be awkward and also because of the origin defining problems. Quartets were, in fact, to provide the key to the structure. Since relatively few negative quartets were generated for AZET, positive quartets were also used. This structure provides a good illustration of how higher invariants can be used to stabilise the tangent formula.

The structure was finally solved in a rather

unexpected fashion. E-magnitudes were calculated using a K-curve; triplets and quartets (positive as well as negative) were generated using the top 350 and 200 reflections respectively. As mentioned above the triplets were weighted. A total of 80 phase sets were produced by tangent refinement. The figures of merit were poorer than before, however, an E-map was calculated for the best set. Somewhat surprisingly, this map revealed recognisable fragments of both molecules in the asymmetric unit. 31 out of the 48 non-hydrogen atoms were found from this map and the rest were located using Fourier methods.

4.3.7 STRUCTURE7: CUIMID

It is often the case that a structure which solves easily using one computer program will present difficulties for another. CUIMID is reckoned to be one of the easier structures on the database but it still managed to pose a few problems for MITHRIL. It is thought that these difficulties are related to the fact that the structure crystallises in a trigonal space group ($P3_221$). Another awkward feature of this structure is that it does not contain a whole molecule in the asymmetric unit.

CUIMID is a heavy atom structure so the problem of solving the structure was immediately reduced to locating the three heavy atoms, one Cu and two Cl's, in the asymmetric unit. It was not possible to solve the structure using default parameters throughout MITHRIL. The E-magnitudes read from the data file were not very well distributed with a paucity of large values. The convergence map, however, looked extremely healthy with an abundance of contributors. As is often the case with heavy atom structures, the figures of

A relatively unusual feature of this structure is that it contains a 3 membered ring - the minimum bond angle was lowered to take this into account.

4.3.9 STRUCTURE9: BED

BED is one of very few compounds belonging to the space group I4; in fact it was only the second report of a structure in that space group. Despite its unusual symmetry, the structure behaved well using MITHRIL and was solved comparatively easily.

As the space group is symmorphic, quartet phase relationships were generated automatically and they participated greatly in the convergence procedure. A recognisable fragment of the molecule was obtained using default parameters throughout MITHRIL, however, a better solution was discovered using the WEIGHT command in the triplet module of the program. As discussed in Chapter2, this command causes unreliable triplet relationships to be downweighted and become of lesser importance in the phasing procedure.

The second best phase set - which, incidentally, had the best PSIZERO and RESID values - was used to calculate an E-map from which a 21 atom fragment was obtained. This fragment contained three 6 membered rings and involved the top 16 peaks of the E-map. As BED contains phenyl groups which show appreciable thermal motion, it was not surprising that some atoms were misssing from the map. The remaining 17 non-hydrogen atoms were located using weighted Fourier synthesis.

merit produced for the phase sets were rather poor.

A better distribution of E-magnitudes was obtained by normalisation using a K-curve and so it was these E's which were subsequently used. As was pointed out in Chapter2, reflections that are measured at both high and low angles are likely to be unreliable and so, in addition, it was decided to try eliminating these reflections. These two steps led to the structure being found. Only reflections which lay between the $\sin(\theta/\lambda)$ limits of 0.10 and 0.55 were used in normalisation. Triplets were generated for the top 250 reflections and phase expansion and refinement were carried out using the statistically weighted tangent formula. Once again, the figures of merit for the 36 phase sets were poor. An E-map was calculated using the best set, however, from which the three heavy atoms were located. As was to be expected, the three peaks corresponding to these atoms were far in excess of any of the other E-map peaks.

4.3.8 STRUCTURE8: TUR10

This structure is included in the database as an example of an unusual space group ($P6_322$). MITHRIL, however, experienced little difficulty in elucidating the structure: default parameters were used throughout the program. Because of the large number of symmetry operations a great many triplets were generated for this structure (13375 for the top 250 reflections). The program correctly found the phases of two reflections from $\sum_1$ relationships. These phases were used in the tangent refinement process. Of the 18 phase sets generated one looked significantly better than the rest. An E-map was calculated from which all 17 non-hydrogen atoms were found.

4.3.10 STRUCTURE10: LOGANIN

LOGANIN was first solved using SHELXTL, however, some difficulties were encountered before the structure was eventually found via a 2-dimensional projection. Using MITHRIL no such problems arose and the structure was solved easily using default parameters throughout the program.

The normalised structure factors were read off the data file and 3188 triplets were generated for the top 250 reflections. Although the phases of some 17 reflections were indicated by $\sum_1$ relationships, none was of a sufficiently high probability to be used in the phasing procedure. Among the phase sets produced by tangent refinement, was one which contained the best values for all three figures of merit used. The E-map generated using this set yielded all 27 non-hydrogen atoms.

4.3.11 STRUCTURE11: DIOL

DIOL is another structure which is included in the database as an example of an interesting space group: in this case $I\bar{4}2d$. Although the compound has only 12 atoms in the asymmetric unit it still managed to pose some interesting problems for MITHRIL.

The first of these problems was in the distribution of the E-magnitudes where there was a shortage of big values. Only 2 of the E-magnitudes read from the data file were greater than 2.1, however, it was found that normalisation of the structure factors using a K-curve improved the situation slightly.

As was to be expected of a structure containing 8 symmetry operations, very many triplet relationships were generated using a relatively small number of reflections. Most of these relationships, however, had low kappa values (i.e. were unreliable) and consequently were not used in either convergence or phasing procedures.

A strange feature of this structure was that although the convergence map looked good with many contributors, many undetermined phases showed up in the tangent refinement procedure. This led to quartets being used as it was felt that more phase relationships would help. As with triplets, many quartets were generated but, unfortunately, only a tiny fraction of these were negative: out of 7039 quartet relationships generated for the top 100 reflections only 62 were negative.

At that stage there were still undetermined phases occurring in the tangent refinement procedure. The problem was finally removed by issuing the command 'MISS 1' in the quartet module of the program. This option allows a quartet to be used even if one reflection is missing from its second neighbourhood and so causes more quartets, including negative quartets, to be generated. Note that the missing reflection in each case is assigned an E-value of 1.0. This is in accordance with the normalisation procedure which computes E-magnitudes in such a way that their average value is 1.0.

Since the structure was proving to be obstinate an extra reflection was added to the starting set so that more phase sets would be generated. The figures of merit for the phase sets were, in general, very poor: this was probably an effect caused by the rather exotic space group. Even so, of the 48 phase sets produced, one set stood out from the rest containing the best values for PSIZERO and RESID as well as

a good ABSFOM value. The complete structure was obtained from an E-map calculated from this phase set.

4.3.12 STRUCTURE12: APAPA

APAPA proved to be one of the more difficult structures on the database to solve. It was first solved only with extreme difficulty using MULTAN and other methods which gave a 25 atom fragment. Patterson search methods were not successful even although the structure contains two phosphorus atoms.

The structure was not solved by using default parameters throughout the MITHRIL program, however, by doing so several interesting points arose which were useful later. Amongst these features were that many triplet relationships were generated but, because of their associated reliabilities, very few were actually used in the convergence procedure. There were, consequently, not too many contributors to a rather sparse convergence map. The figures of merit generated for the phase sets looked encouraging but no solution could be found.

The first non-default option tried to overcome the difficulty was the introduction of quartets into the procedure. It was noticed, however, that very few negative quartets were produced even when a large number of reflections was used. Slightly more negative quartets were generated by using E-magnitudes calculated using a K-curve rather than those from the data file. The number of quartets being used was still very small and so the POSI command was issued in the quartet module of the program so that positive quartets were used as well as negative ones.

The structure was found by generating quartets - positive as well as negative - for the top 200 reflections. In addition, two reflections were added to the starting set so that the convergence procedure was stabilised. The other obvious advantage gained from using a large starting set with a difficult structure was that it caused more phase sets to be generated. A problem with this structure was that the E-maps were rather cluttered and difficult to interpret. However, the best phase set of the 128 generated was used to calculate an E-map from which a 26 atom fragment was identified. The remaining 37 non-hydrogen atoms of the structure were located using Fourier recycling methods.

4.3.13 STRUCTURE13: TPALA

This structure was originally solved by cosine invariant analysis together with the passive use of quartet phase relationships. Although TPALA did not solve under default parameters using MITHRIL, it presented few difficulties to the program and was solved soon afterwards.

Using default parameters throughout the program, relatively few triplets were generated: a consequence of there being only two symmetry operations in the space group. This led to a poor looking convergence map with very few contributors. A rather odd feature of this structure was that although the correct solution was not found, the figures of merit for the phase sets looked very good.

The structure was found by using the simple tactics of increasing the number of reflections for which to generate triplets and by introducing quartets. Triplets and negative

quartets were generated for the top 450 and 200 reflections respectively. Using these commands the convergence map had many more contributors with quartets playing an active part. The figures of merit, however, looked a good deal worse than those produced using default parameters - in fact, this is a common occurrence when the number of reflections used for generating phase relationships is increased. An E-map was calculated for the best of the 48 phase sets. The PSIZERO value of this phase set was relatively high (2.62). This is often an indication that a correct fragment has been found but it is wrongly orientated in the unit cell. However, a 20 atom fragment, including a 5-membered ring, was found which, when recycled using Fourier methods, gave the complete structure.

4.3.14 STRUCTURE14: NO55

Although the data for NO55 was known to be mediocre, it was included in the database as an example of an interesting space group (Fdd2). The structure was not expected to be easy to solve because of the poor data and also because it was known that it was only with great difficulty that it had originally been solved by direct methods.

The only features of note which came from using default parameters throughout MITHRIL were that the program managed to correctly assign a value to one phase from Σ_1 relationships and that the figures of merit were very poor. As usual, the first non-default option tried was the introduction of quartet phase relationships into the procedure. This caused the number of contributors to the convergence map to increase enormously with the quartets perhaps over-participating.

At this point it was thought - wrongly, as it later turned out - because everything up to and including the convergence procedure looked reasonable, that efforts should be concentrated on the phase expansion and refinement procedures. The statistically weighted tangent formula was tried instead of the standard tangent formula but little difference was detected by doing so. Similarly, running the MAGEX and YZARC modules prior to tangent refinement did not improve the figures of merit and no solution could be found. Another option used was the random phase module, RANTAN, but again without success.

NO55 was eventually solved by MITHRIL only after some radical changes in strategy. E-magnitudes were calculated using a K-curve rather than read from the data file and the use of quartets was abandoned. In addition, the WEIGHT command was issued in the triplet module so that the triplets were weighted according to the squared $(\sin\theta/\lambda)$ values of their contributing reflections. Using $\sum_1$ relationships, the program had identified the phase of one reflection but this was not used. Phase refinement and expansion were carried out using the statistically weighted tangent formula. The figures of merit for the 16 phase sets generated did not look encouraging, however, the second best set yielded a rather unlikely looking 10 atom fragment which when used in Fourier recycling gave the complete 24 atom structure.

A much better and more spectacular solution was later found by going back to using quartets. On this run the E-magnitudes were read from the data file and the weighting scheme was again used for the triplets. Rather than let them swamp the convergence map as they had done before, this time negative quartets were generated for only the top 50 reflections. Tangent refinement produced 18 phase sets, one of which was far superior to any previously generated. The E-map calculated from this phase set revealed a 21 atom fragment

which included the 16-membered ring as well as one of the two 6-membered rings.

4.3.15 STRUCTURE15: MGHEX

MGHEX was found to be one of the two most difficult structures on the database. At the time of writing, despite many attempts using MITHRIL, the structure could not be solved. It should be pointed out, however, that so vast is the range of options available in MITHRIL, that many combinations of options have still to be tried and it is possible that MITHRIL will solve the structure some time in the future.

From the outset there were reasons to believe that this structure would cause problems - it was originally solved only with difficulty, there are a large number of atoms in the asymmetric unit and too few psi-zero relationships were generated to produce a PSIZERO figure of merit. MGHEX consists of two macrocyclic rings coordinated around a central magnesium atom. It was thought that locating the Mg would provide the key to solving the structure but it was not possible to find it. The cause of the problem remains unclear - the E-magnitudes were well distributed and the convergence map reasonable. The figures of merit obtained on various runs were generally poor with low ABSFOM values, high RESID's and a low range of NQEST values. It was found that the figures of merit could be improved by raising the temperature factor but the E-maps produced were no better than those for other runs.

4.3.16 STRUCTURE16: TOTC

TOTC is the other structure on the database which, at the time of writing, could not be solved using MITHRIL. This time, however, there was no reason to expect any difficulty as the structure was solved easily using SHELX. As with MGHEX, almost all of the MITHRIL options were tried without success.

The problems encountered with TOTC were similar to those with MGHEX. The figures of merit were poor despite good E-magnitude distributions and convergence maps. Around 500 phase sets were generated for a typical MITHRIL run on TOTC and the top 10 or so E-maps were examined. Most of the program options tried proved ineffective although it was observed that the figures of merit could be improved somewhat by altering the temperature factor and also by use of the triplet weighting scheme. Even with this improvement, however, no recognisable fragment of the structure could be found from any of the E-maps generated.

4.3.17 STRUCTURE17: TVAL

TVAL was originally solved independently by hand symbolic addition and by translation, search and packing considerations. It should be noted that the latter methods were based on the monoclinic form of the structure which has only one molecule in the asymmetric unit whereas the data used in this instance was for the triclinic form which has two.

This structure caused difficulty to MITHRIL only

because the data set was incomplete: there were no weak reflections present. No real problems were encountered, however, after a technique was devised to add the missing reflections to the data set.

The following recipe was used to generate the missing reflections and assign structure factors to them:

(a) Find the indices of the missing reflections.

(b) Calculate $\sin(\theta/\lambda)$ for each reflection.

(c) Decide which region of $\sin(\theta/\lambda)$ each reflection lies in and assign an F value in accordance with the guidelines laid down by Vickovic and Viterbo[28] (see below)

(d) Add reflections to the data set

(e) Normalise

The structure factors for the unobserved reflections were assigned as follows:

Centrosymmetric structures $F = F_{min}/\sqrt{3}$

Non-centrosymmetric $F = F_{min}/\sqrt{2}$

where F_{min} is the minimum structure factor value in the particular $\sin(\theta/\lambda)$ region.

Using this method a total of 1557 reflections were added to the original 7484 on the data set. Normalisation was carried out using a K-curve. There were still no weak reflections present even in the enlarged data set: the above

recipe will provide weak reflections only if there are some on the original data set. Triplets were generated for the top 750 reflections and quartets for the top 115. A total of 12 reflections were used in the starting set as the structure was expected to be difficult to solve. The standard tangent formula was used to produce 173 phase sets. Three solutions stood out as being superior to the remainder and from the first of these, 80 out of the 156 non-hydrogen atoms in the structure were located including both the macrocyclic rings.

4.3.18 STRUCTURE18: TPH

TPH was originally solved by Patterson search methods using a C_6N search fragment. It presented problems to MITHRIL mainly because it was found that when the data was expanded over a full hemisphere, the store capacity of the program was exceeded. To get round this difficulty, a specially extended version of MITHRIL, with bigger array sizes, was employed. However, even using the bigger version there were still too many reflections and so it was necessary to reduce the data set. This was accomplished by eliminating the high angle reflections. All reflections with $(\sin\theta/\lambda)$ values greater than 0.55 were removed: this reduced the data set from an original size of 4758 reflections to just 2097 reflections.

Using this data it was noticed that 3 of the E-magnitudes had values very much greater than the rest. The temperature factor, B, was initially calculated to be around 1.5. It was found that increasing B to 6.0 produced a good distribution of E-magnitudes, however, it also led to weak links in the convergence map which caused the phasing procedure to break down. Eventually a compromise was reached which gave both a fairly good distribution of E-magnitudes and

a reasonable convergence map. It was achieved by setting the temperature factor to a value of 2.4 and limiting the maximum value for an E-magnitude to 3.5. The convergence map still contained a single weak link but this was not a problem since the reflection concerned was involved in very few phase relationships.

As the structure was proving to be quite troublesome and because there were not too many contributors to the convergence map, negative quartets were used in addition to triplets. The program correctly identified the phase of one reflection using $\sum_1$ relationships. This was input into the tangent refinement process which, in this case, used Hull and Irwin's statistically weighted tangent formula.

Of the 56 phase sets generated, one looked very much better than the rest. It contained, by a considerable margin, the best PSIZERO and RESID values, as well as a very good ABSFOM. The E-map calculated from this phase set was found to contain all 39 of the non-hydrogen atoms which made up the 1.5 molecules of TPH in the asymmetric unit.

4.3.19 STRUCTURE19: NEWQB

NEWQB was originally solved using SHELXTL, however, it should be pointed out that it was only with great difficulty that the structure was found. It was eventually solved by incorporating trio relations of two negative and one positive quartets into the starting set. MITHRIL also experienced great difficulty with this structure - many combinations of options were tried without success - before finally, and somewhat surprisingly, an almost perfect solution was found which revealed all but two of the non-hydrogen atoms in the

asymmetric unit.

Using MITHRIL, it was apparent right from the start that this structure was going to cause problems. Some of the difficulties were revealed by using default parameters throughout the program. The E-magnitudes read from the data file were poorly distributed with two values far in excess of the remainder. There was a paucity of phase relationships, caused by the low symmetry of the structure: only 875 triplets and 87 quartets were generated using the top 250 and 115 reflections respectively. Associated with the shortage of phase relationships was that there were very few contributors to a convergence map that had two weak links. On the bright side, however, the figures of merit produced by tangent refinement looked fairly good as did the peak distribution of some of the E-maps that were investigated.

It was thought that the structure would probably not be solved until there were no weak links in the convergence map. This was achieved by increasing the number of reflections for which triplets and negative quartets were generated so that the number of phase relationships was also increased. Although the convergence map produced thus had no weak links, it was far from ideal with few contributors and little quartet participation. As is often the case when the number of reflections used is increased, the figures of merit became a good deal worse. Experiments were carried out, without any particular success, in which the origin and starting set were defined by hand rather than letting the program do so automatically.

Satisfied that a better convergence map could not be easily obtained it was decided to concentrate on options after that point in an attempt to find the solution. The statistically weighted tangent formula was used instead of the standard formula but little difference was detected by doing

so. Similarly, use of the MAGEX module made little impact while employing random phase techniques only made the figures of merit worse. It should be remembered, however, that the RANTAN module often produces many very bad solutions before going on to find a perfect solution and it nay have been that the number of solutions produced in this case (256) was not large enough.

As has already been seen, the use of quartet phase relationships has proved to be very effective in the solution of many of the structures on the database. Since the space group of this structure is symmorphic, negative quartets were generated automatically by the program. However, they participated only very weakly in the convergence procedure and generally seemed to be of little effect. Several unsuccessful attempts were made to do something about this. Generating positive as well as negative quartets greatly increased the number of quartets produced but even when the phase relationships were comprised of 25% quartets, the convergence map still showed little quartet participation. This was mirrored by a lack of change in the figures of merit. Little change was again observed when, first of all, the third neighbourhood calculation was invoked and, subsequently, when reflections were allowed to be missing from both second and third neighbourhoods. Another, more radical, tactic tried was preventing quartets being generated by means of the "QUAR -1" command but this only caused the figures of merit to worsen.

In addition to the quartets work, a lot of time was devoted to the other phase relationships. For triplets, both the weighting scheme and cut-off technique based on squared $(\sin\theta/\lambda)$ values caused a slight improvement in the figures of merit but no solution could be found. Quintets were also generated for NEWQB which this time made the figures of merit worse.

Since no solution had been found it was decided at that point to go a stage further back in the structure determination process and investigate the E-magnitudes and the factors that affect them. As will be seen this proved to be the vital step in obtaining the solution to NEWQB.

It had already been noticed that calculating the E-magnitudes using a K-curve rather than reading them directly from the data file led to a slightly better distribution of E's. Indeed, these E-magnitudes had been used in many of the unsuccessful attempts to solve this structure described above. As with the other distribution, there were 2 E-magnitudes far bigger than the rest. Limiting the E-magnitudes to a maximum value of 3.0 made the figures of merit poorer and led to a reversal of tactics that was to prove successful.

The temperature factor, B, had been calculated by the program to have a value of around 1.5. It was increased to 5.0 and triplets and quartets were generated for the top 500 and 225 reflections respectively. The distribution of the E-magnitudes was worse than ever before and the convergence map was even more sparse than usual as the number of phase relationships had been decreased. Surprisingly though, tangent refinement produced one phase set with figures of merit far better than those of the other 127 sets produced and indeed, much better than any set produced previously. The E-map calculated from this phase set found almost the complete structure, revealing 60 out of the 62 atoms present in the asymmetric unit.

4.3.20 STRUCTURE20: GOLDMAN2

GOLDMAN2 proved to be one of the more difficult

structures on the database. Using MITHRIL, many different combinations of options were tried unsuccessfully before a quite spectacular solution was finally found which revealed the complete structure.

The structure was originally solved using SHELXTL but, here again, many difficulties were encountered before the solution was found: the problem was finally solved by incorporating three trio relations into the starting set. Trios comprise an internally consistent combination of two negative and one positive quartets. Approximately 1/2 of the atoms were found thus; the rest being located by Fourier methods.

Quartets were used on GOLDMAN2 from the outset as it was expected the structure would be difficult to solve. The E-magnitudes read from the data file were reasonably well distributed, however, very few negative quartets were generated. It was found that normalisation of the structure factors using a K-curve led to more negative quartets being produced and so it was these E-magnitudes which were used. Under default parameters, relatively few phase relationships were generated and it was decided at an early stage to increase the number of reflections used to generate triplets and quartets. This helped to strengthen the convergence procedure.

Among the many strategies employed in an attempt to solve this structure were the following:

(a) The number of quartets were increased by first allowing one reflection to be missing from the second neighbourhood and then invoking the third neighbourhood calculation, again allowing for one missing reflection. In both cases the figures of merit improved slightly overall.

(b) The MDKS formula was used to downweight unreliable triplets however no improvement was detected.

(c) The use of MAGEX and YZARC between the convergence and phase refinement procedures. Using MAGEX, RESID values improved slightly while PSIZERO values worsened but still no solution was found.

(d) As the structure consists largely of fused 6-membered rings, the coordinates of one such group were input to the program in the NORMAL module via the GROUP command. This tactic did improve the figures of merit quite noticeably but once again no solution was found.

The structure was eventually solved by using the cut-off technique for triplets described in Chapter2. E-magnitudes were calculated using a K-curve and triplets were generated for the top 400 reflections using a cut-off value of 0.69. This meant that all triplets with $\sin(\theta/\lambda)$ values greater than 0.69 times the maximum value were immediately rejected: in fact 926 triplets out of an original 5724 triplets were rejected. Negative quartets were produced for the top 150 reflections and phase expansion and refinement was carried out using the standard tangent formula. One phase set had the best, by some considerable margin, ABSFOM, PSIZERO and RESID values. The E-map calculated from this set yielded all 56 carbon atoms of the two molecules in the asymmetric unit; the atoms all being correctly connected with all rings present.

4.3.21 STRUCTURE21: MUNICH1

MUNICH1 was originally solved by Patterson search methods. A few minor difficulties were encountered before it was solved using MITHRIL. Although the structure did not actually solve using default parameters throughout the program enough clues were provided by doing so to enable it to be solved soon afterwards.

The E-magnitudes read from the data file were nicely distributed and so were not tampered with. It was noticed that few triplet and quartet phase relationships were generated for this structure and that, consequently, the convergence map was rather sparse with few contributors. In addition, the figures of merit produced for the phase sets did not look encouraging: in particular, both ABSFOM and RESID values were abnormally high.

The following three steps were taken to remedy the above problems:

(a) Triplets were generated for the top 450 reflections and negative quartets for the top 200 (c.f. the default values of 270 and 115 reflections respectively). There were, therefore, many more phase relationships which led to the convergence map having many more contributors.

(b) The convergence procedure was further stabilised by the addition of 2 more reflections to the starting set. A direct consequence of this was that the number of solutions generated was increased as was the probability of finding the correct solution.

(c) Excessively high values of ABSFOM and RESID are a symptom of the tangent formula causing the phases to become over-

consistent. The Hull and Irwin statistically weighted tangent formula was employed instead as it is known to perform better in such situations.

Using the above commands 160 phase sets were produced. One outstanding solution had the best values for both PSIZERO and RESID. An E-map was calculated using this phase set from which it was possible to locate all the non-hydrogen atoms of the 2 molecules of MUNICH1 in the asymmetric unit. Rather unusually, the structure contains two fused 3-membered rings and although all the atoms of these rings were found by the program they were not properly connected since the program has a default value for the minimum bond angle of 85°.

4.3.22 STRUCTURE22: MBH2

MBH2 proved to be fairly typical of structures which crystallise in space group P1 with very few phase relationships being generated. To avoid the problem of weak links in the convergence map, both triplets and negative quartets were generated using more than the default number of reflections: 500 and 225 instead of 326 and 115 respectively. For the reason just mentioned and also because MBH2 did not solve using default parameters, a relatively large starting set of reflections were used causing a large number (174) of phase sets to be generated.

The use of quartet phase relationships was crucial in the elucidation of this structure. There were three reasons for using quartets:

(a) The space group was symmorphic.

(b) It would provide more phase relationships and so strengthen the convergence procedure.

(c) NQEST would be computed so in the absence of PSIZERO which is not computed for this space group, there would still be three figures of merit on which to rank phase sets.

This structure was originally solved by SHELXTL using random start techniques. With MITHRIL, it was solved by invoking the third neighbourhood calculation for quartets in addition to the methods descibed above. The solution was found from an E-map calculated using the second best phase set: fragments from all three molecules present in the asymmetric unit were found. Using this E-map it was possible to locate 42 out of the 54 non-hydrogen atoms in the structure. The remaining atoms were found using Fourier methods.

4.3.23 STRUCTURE23: PGE2

PGE2 is a very rare example of a hairpin shaped molecule: the parallel side chains are only 5Å apart. It was originally solved with great difficulty using a combination of hand phasing techniques and tangent formula refinement. Using MITHRIL, the structure solved easily, however, problems were encountered in identifying the correct solution. As can be seen in Chapter 5 the use of interactive graphics techniques was of great benefit in the interpretation of E-maps generated for PGE2.

Unexpectedly, the structure was found using default parameters throughout MITHRIL, although it was some time

before this was actually realised. The E-magnitude distribution as read from the data file was not particularly good with a few rather large values. Triplets were generated for the top 250 reflections and negative quartets for the top 115. Taking the lack of symmetry in the space group into account, the number of triplets produced (2282) was quite reasonable whereas only a small number (165) of negative quartets were generated. The convergence map was moderately good but with no quartet participation. E-maps were generated for the top two phase sets, out of a total of 24, but no solution was found.

PGE2 belongs to space group P1 and as such no PSIZERO figure of merit was produced (because of a paucity of psi-zero relationships) while quartet phase relationships, and hence NQEST, were generated by default. It was noticed that some phase sets with good values for ABSFOM and RESID had low values for the combined figure of merit (CFOM) and so no maps were calculated for them. This was a direct result of the program placing more emphasis on NQEST than on either ABSFOM or RESID. Normally MITHRIL assigns weights of unity to all figures of merit for use in calculating CFOM, however, when the space group is symmorphic, greater weighting is given to PSIZERO and NQEST while less is given to ABSFOM and RESID. As PSIZERO was absent the weights assigned to ABSFOM, RESID and NQEST were 0.72, 0.72 and 1.56 respectively.

Given the fact that in this case so few quartets had been generated and those that had were largely inactive, this bias towards NQEST seemed unjustified. Maps were then investigated on the basis of ABSFOM and RESID values. The E-map calculated from the phase set with the second best RESID revealed a 12 atom fragment which included the 5-membered ring. Using Fourier recycling 4 more atoms were located. It was impossible to find the remaining atoms using MITHRIL due to the appreciable thermal vibrations along the side chains.

Refinement methods were needed to complete the structure.

A slightly better solution for this structure was subsequently found by using the weighting scheme based on squared ($\sin\theta/\lambda$) values on the triplet phase relationships. A 15 atom fragment was found in that instance.

4.3.24 STRUCTURE24: SUOA

The data for sucrose octa-acetate was collected at $-100^{o}C$ to avoid excessive thermal vibrations along the many side chains of the molecule. The structure was first solved, with some difficulty, using SHELXTL. Similarly, MITHRIL also experienced problems and many options were tried before the structure was solved by means of a very simple technique which, with the benefit of hindsight, should have been used much sooner.

Although the structure did not solve using default parameters throughout MITHRIL, the output did look encouraging. The E-magnitudes read from the data file were well distributed and the figures of merit produced for the phase sets were good, though not spectacularly so with no outstanding solution. The only problem that could be seen was the sparse convergence map caused in part by the fact that there was not a great number of triplet relationships.

The introduction of quartet phase relationships had proved useful in other structures but it made little impact on SUOA. Very few negative quartets were generated using the default number of reflections but even when more were produced, by using a greater number of reflections, they

contributed little to the convergence map and the figures of merit were unaffected. Quintet phase relationships were also tried on this structure but again without success.

E-magnitudes produced by either a K-curve or a Debye curve both had similar distributions to those read from the data file and it was found that their use made negligible difference to the structure determination process.

Many options in the MITHRIL program were used on this structure. Some such as the triplet weighting scheme, MAGEX and the statistically weighted tangent formula caused a slight improvement in the figures of merit but without the correct solution being found. On the other hand it was found that lowering the probability limit for acceptance of a $\sum_1$ relationship and also the use of random phase techniques only made things worse.

As no significant progress had been made since default parameters had been used it was decided to go back to that point and start again. The structure was elucidated shortly afterwards using a series of options that varied in only two respects from the default run: a larger starting set was used and the statistically weighted tangent formula was preferred to the standard formula. Of the 160 phase sets produced, one looked far better than any that had previously been observed. An E-map was calculated using this phase set from which 41 out of the 47 non-hydrogen atoms in the asymmetric unit were located. The remaining atoms were found using Fourier recycling.

4.3.25 STRUCTURE25: WINTER2

WINTER2 is generally thought to be one of the most difficult structures on the database and using MITHRIL it was discovered that this reputation is justified. After failing many times, MITHRIL did eventually find a very good solution that revealed a substantial fragment of the structure: this is almost certainly the best solution that has ever been found for WINTER2. This structure is another of those on the database that was originally solved using SHELXTL, but, as with MITHRIL, only after many problems had been overcome - in the successful run tangent refinement was used with random starting phases for all reflections. As with PGE2 the interpretation of E-maps for WINTER2 was greatly assisted by the use of interactive graphics techniques. This is described in some detail in the next chapter.

The sheer size of the structure is one of the main reasons why it is so difficult to solve: there are 88 non-hydrogen atoms in the asymmetric unit. The E-maps are, therfore, crowded and difficult to interpret. WINTER2 is a polypeptide containing a side chain with a terminal 6-membered ring. Unfortunately the ring is not easily identifiable from E-maps as the side chain exhibits appreciable thermal vibration since the data was collected at room temperature. The problem of interpreting E-maps is, therefore, made greater by the fact that there is no "foothold" to the structure i.e. there is no easily recognisable fragment that is unique to a certain part of the structure. The structure does, however, have the charcteristic 3 atom repeating unit of the peptide link along its spiral backbone.

Using MITHRIL, many combinations of options were tried before the solution was eventually found. It is probably true

to say, however, that correct fragments of the molecule were actually found several times but because of the problem of E-map interpretation, they were not recognised as being correct and, consequently, were overlooked.

Difficulties with this structure were apparent almost straight away when default parameters were used throughout MITHRIL. To begin with, the E-magnitudes were poorly distributed with several very large values. In addition, the convergence map was extremely sparse and the figures of merit were poor. Since the structure was expected to be very difficult, the tactics of generating a lot of phase sets, by using a bigger starting set, and investigating several E-maps, rather than just the best one, were used from that point on. The bigger starting set also had the added advantage of strengthening the convergence procedure.

Of the many options tried, it was the use of the MAGEX module, the statistically weighted tangent formula and the cut off scheme for triplets, which brought about the most noticeable improvements in the figures of merit. It must be emphasised, however, that these improvements were not all that great. Most of the other options tried made little impression but it was found that a few had a derogatory effect on the structure determination process. Amongst these were: the use of quartet and quintet phase relationships, which were difficult to generate in any case; lowering the probability limit for acceptance of Σ_1 relationships; and finally, using YZARC.

In retrospect, however, none of the MITHRIL options mentioned above made as much impact, or were as significant, as the use of program options which directly affected the E-magnitudes. Normalisation by use of either a Wilson plot or a K-curve led to even worse E-magnitude distributions but the latter method was used since it led to more triplets being

generated and also a better convergence map. It was found that varying the temperature factor, B, had quite profound effects. Increasing B made the E distribution poorer, led to fewer phase relationships being generated which caused the convergence map to become sparser but the figures of merit became significantly better. Despite some good phase sets being generated by increasing B, the correct solution couyld not be found. Lowering the temperature factor only led to poorer figures of merit and was quickly abandoned.

It was only when attention was focussed on removing potentially unreliable reflections from the data set that the structure was finally elucidated. It had been found previously that removing all high angle reflections with ($\sin\theta/\lambda$) values greater than 0.50 led to a better distribution of E-magnitudes and better convergence map but poorer figures of merit. The structure was solved by editing some of the low angle reflections from the data set. All reflections with ($\sin\theta/\lambda$) values lower than 0.125 were removed from the data set: 65 reflections out of a total of 6980 were removed. In addition, structure factors were normalised using a K-curve, and phase refinement and expansion were carried out using the statistically weighted tangent formula. The starting set consisted of 13 reflections which led to 568 phase sets being generated. The best phase set contained the top values for both PSIZERO and RESID. An E-map was calculated for this set which revealed a 50 atom fragment. All the other non-hydrogen atoms, including the 6-membered ring, were located by Fourier recycling.

No.	Space gp	NORMAL*	TRIPLETS			QUARTETS						LSAM	TANGENT	
			No refs	No triplet	Weight	No refs	No quartets	Miss	Neig	Posi	Weight	No sets	No sets	SWTR
1	$P4_2/n$	-2	250	5856									16	
2	$R\bar{3}$	-2	250	3795		115	816						16	
3	$P2_13$	1	117	6198									15	
4	$C2/m$	1	280	4511		130	564					8		
5	$P2_1$	1	300	7059	Y	225	1372						24	
6	$Pca2_1$	1	350	11751	Y	200	25617			Y	Y		80	
7	$P3_221$	0	250	14826	Y								36	Y
8	$P6_322$	-2	250	13775									18	
9	$I\bar{4}$	-2	250	3100	Y	115	677				Y		24	
10	$P2_12_12_1$	-2	250	3188									24	
11	$I\bar{4}2d$	1	250	16068		100	243	Y					48	
12	$P4_12_12$	1	350	11857	Y	200	6020			Y	Y		128	
13	$P2_1$	-2	450	11527		200	2690						48	
14	$Fd\bar{d}2$	-2	250	5763	Y	50	101						18	Y
15	$P3_1$													
16	$P6_1$													
17	$P\bar{1}$	1	750	6581	Y	115	7						120	Y
18	$C222_1$	1	256	4331		125	596						56	Y
19	$P\bar{1}$	1	525	4841		250	63						128	
20	Cc	1	400	4644	Y	150	1355						40	
21	$C2$	-2	450	11315		200	1881						160	Y
22	$P\bar{1}$	1	500	6216		225	974		Y				173	
23	$P\bar{1}$	-2	250	2106	Y	115	165						24	
24	$P2_12_12_1$	-2	288	2865									160	Y
25	$P2_1$	1	452	4185									596	Y

(* -2 = E's from test file; 0 = Wilson plot; 1 = K-curve)

Table 4.1: Summary of methods used to solve the database structures

No.	ABSFOM	PSIZERO	RESID	NQEST	RANK	ATOMS FOUND
1	1.21	0.79	20.37	-	1	18/18
2	1.09	0.91	12.75	-0.67	1	24/24
3	1.62	3.81	35.22	-	1	3/3*
4	1.01	0.95	3.21	-0.48	1	1/1*
5	1.30	2.23	11.20	-0.07	1	10/25
6	1.58	2.71	37.87	-0.06	1	17/24
7	0.33	2.41	58.63	-	1	3/3*
8	1.12	1.64	24.72	-	1	17/17
9	1.12	1.06	21.94	-0.03	2	18/34
10	1.10	0.93	16.18	-	1	27/27
11	0.76	1.90	27.67	0.09	1	12/12
12	1.09	1.11	25.28	-0.05	1	20/69
13	1.08	2.68	24.40	-0.04	1	19/39
14	0.66	1.70	28.29	-0.03	1	21/24
15						
16						
17	1.25	-	22.83	-	1	65/156
18	1.01	0.84	13.75	0.01	1	39/39
19	1.21	0.75	21.92	-	1	60/62
20	1.14	0.86	18.84	0.01	1	56/56
21	0.92	1.00	18.75	-0.03	1	40/40
22	1.04	-	16.83	0.00	2	42/54
23	1.15	-	16.93	0.07	9	15/25
24	0.82	1.32	20.33	-	1	41/47
25	0.87	1.63	21.53	-	1	50/79

Table 4.2: Figures of merit & ranks of the solutions

(* - number of atoms refers to the number of heavy atoms)

4.4 CONCLUSIONS

It is apparent from the results obtained in the previous section that MITHRIL has performed well, managing to solve 23 out of the 25 structures on the database. Great care must be taken when drawing conclusions from these results because of the variation in the quality of the X-ray data for the structures and also because of the wide range of space groups that were present. Frequently, it was observed that a certain option would greatly assist the solution of one structure while being detrimental to that of another. Another problem which makes it difficult to generalise is that there is usually more than one way to solve a structure. When dealing with the structures on the database, every attempt was made to approach their determination in a logical and sensible manner, but, even so the methods of solution used will, in most cases, not be unique. It is important to realise that there is no magical panacea which will solve every structure. Each structure must be considered strictly on its own merit before a strategy is devised that will, hopefully, lead to its elucidation - this strategy can be altered to take into account any information gained from unsuccessful attempts.

Although very few of the database structures were solved using default parameters throughout MITHRIL, it is necessary to show great caution before suggesting that any of these parameters should be altered. It must be remembered that the structures on the database are there because they are resistant to the techniques of direct methods. To think about changing the default parameters on the basis of the results obtained here would, therefore, not be appropriate in many cases as the database structures are far from being typical of the sort of structure MITHRIL is routinely used on.

It may be more than a coincidence that the two

structures which could not be solved show similar symmetry: MGHEX belongs to $P3_1$ whereas TOTC belongs to $P6_1$. MITHRIL could be having some problems in handling the symmetry in such space groups and further investigation and tesing with other structures will have to be carried out in this area to see whether or not this is the case.

There now follows a series of critical appraisals of some of the techniques of direct methods, as presented in the MITHRIL package, based on the information amassed from using the program on the database of difficult structures.

4.4.1 NORMALISATION

The NORMAL module is perhaps the most important module of the MITHRIL package since decisions made at this stage have a fundamental effect on all the ensuing direct methods procedures. For example, the simple act of altering the temperature factor by even a small amount will cause noticeable differences in the distribution of the E-magnitudes, the number of phase relationships generated and in the phase sets produced by tangent refinement. If a stalemate should be reached when trying to solve a structure, it will often prove profitable to go back to the NORMAL module and experiment with some of the options there.

There are three ways of obtaining normalised structure factors in MITHRIL: by use of a K-curve, by means of a Wilson plot and, lastly, by reading in pre-calculated E's. As can be seen from the results in section 4.3, the majority of the difficult structures were solved using E's calculated by means of a K-curve. There is no doubt, however, that the other two methods provide useful alternatives.

The two options of the NORMAL module in MITHRIL that were used with most significant effect were the commands LIMIT and BSCL. LIMIT allows the user to remove high and/or low angle reflections from the data set on the basis of their sin(theta)/lambda values, as well as limiting the maximum E-magnitude value. It is well known that reflections measured at high or low angles are likely to be unreliable. This is illustrated in the successful elucidations of WINTER2, TPH and CUIMID: low angle reflections were eliminated for WINTER2, high angle reflections for TPH and both types for CUIMID. In a similar fashion, it was found that changing the temperature factor, B, by means of the BSCL command contributed greatly to the solutions of TPH and NEWQB - B was increased in both instances.

The NORMAL module has recently been enhanced by the addition of the MISS command which causes the program to find reflections missing from the original data set, assign structure factors and add them to the data set before normalisation takes place. The recipe for doing this was described in section 4.3.17 along with an account of how the option was successfully used in the elucidation of TVAL. It is thought that MISS will continue to prove useful since nowadays it is an option for some computer-controlled diffractometers to skip the measurement of very weak reflections in order to save time in data collection. Additionally, it should be remembered that speed in data collection is often required to reduce the extent of radiation damage on sensitive crystals. In recent years direct methods have been used to solve more and more crystal structures. A major factor in this success is that the quality of the X-ray data has improved markedly.

4.4.2 TRIPLETS, QUARTETS AND QUINTETS

The simple triple phase relationship has been the cornerstone of direct methods for over thirty years and continues to be so. The use of triplets is essential to the determination of any structure. It has been discovered that the introduction of a weighting scheme and a cut-off technique - both of which are based on squared $(\sin\theta/\lambda)$values - has proved to be very effective in weeding out unreliable triplets. One or other of the two methods was used in the successful solutions of SELENID, AZET, BED, NO55, GOLDMAN2 and PGE2.

There are two other options, MDKS and L.E., in the TRIPLET module of MITHRIL, both of which are meant to give an indication of triplet reliability. They were used only sparingly here because of the considerable amount of processor time required to do so. However, on the several occasions on which they were used, neither method appeared to contribute in a positive way to any of the structures being solved.

One problem common to many of the structures with low symmetry was that too few triplet relationships were generated using default parameters but this was remedied simply by increasing the number of reflections on the TRIPLET command.

The use of quartet phase relationships proved to be invaluable in the elucidation of many of the database structures. Indeed so successful were they, that it is recommended that quartets should be one of the first options invoked when difficulties arise.

In most of the difficult structures where the QUARTET module was invoked, the use of negative quartets alone was

sufficient to generate a reasonable number of quartets. In other cases, however, where there was a paucity of these relationships - even after the number of reflections on the QUARTETS command was increased - it was necessary to use alternative methods. Positive quartets were generated with great success for both AZET and APAPA, a similar result was obtained for MBH2 only after the third neighbourhood calculation was invoked and, finally, in the case of DIOL, the structure was found only after one reflection was permitted to be absent from the second neighbourhood. In addition to these three methods, more quartets can be produced by allowing reflections to be missing from the third neighbourhood also. Note that positive quartets are closely correlated with triplets.

Although, in general, quartets were found to be helpful, there were a few cases, most noticeably for NO55 and SUOA, where their use was found to be detrimental to the business of finding the structures. Like all invariants quartets are probabilistic and should, therefore, not be used blindly and a close check must be kept on their effects.

Quintet phase relationships were used only a few times mainly because of the considerable computational time necessary for their generation. Even when they were used, however, there was not a single instance of their being beneficial to the structure determination process. It is doubtful whether quintets will ever be able to provide constructive help since it must be remembered that they have a $1/N^{3/2}$ probability dependence.

4.4.3 CONVERGENCE MAPPING

Convergence mapping plays a vital role in the multisolution approach to direct methods employed by MITHRIL. It was observed that only in very few cases was a structure solved before even a moderately good convergence map, with a reasonable number of contributors, had been obtained. The most noticeable exceptions to this were for TPH, where the structure was solved despite a weak link in the convergence map, and for NEWQB, where raising the temperature factor led to a sparse convergence map but yielded the structure.

For many of the database structures, the convergence map produced by using default parameters was very thin with many phases detemined by only one or two relationships. In such circumstances the phasing often failed, however, the situation could be remedied by increasing the size of the starting set or by using more reflections to generate phase relationships. The first of these options had the added advantage of producing more phase sets - this is often useful when dealing with difficult structures. It was also important to ensure that the starting set used was reasonable with all the starting set reflections involved early in the phase determination.

Caution had to be exercised in the use of the sigma-1 determined phases as a wrongly determined phase could cause havoc in the phasing process. The SIGMA NONE command in MITHRIL enables the user to exclude these phase indications. This option was used to good effect in the solution of NO55. The results obtained here suggest that it would be better to change the default setting in this case so that Σ_1 determined phases could only be used if specifically requested. This is one instance where it is appropriate to change a default setting on the evidence of results obtained here.

MITHRIL contains three modules all of which can be used as alternatives to convergence mapping: MAGEX, RANTAN and YZARC. Not one of the successful solutions to the structures on the database involved the use of any of these options. In their defence, however, it must be said that these modules were only invoked in situations where a reasonable convergence map could not be obtained or else only after other options had been tried and failed. RANTAN was used several times without success but usually only the default number of phase sets were generated. It is a feature of RANTAN that after generating many very bad phase sets it will suddenly produce a good solution. The chances of doing so are obviously increased for a large number of phase sets but due to restrictions imposed on processor time it was not possible to do this. MAGEX mainly seemed to make little impression on the structure determination process, however, there were occasions, as in the case of WINTER2, where its use appeared to have a positive effect. The same cannot be said of YZARC whose use in almost all cases made things worse.

4.4.4 PHASE EXPANSION AND REFINEMENT

There is only one method available with which to examine the phase expansion and refinement procedures and that is by inspection of the figures of merit generated for the phase sets. It is usually sufficient, for routine structures, to look at phase sets on the basis of their combined figure of merit (CFOM) values. For the structures on the database, however, it was found to be necessary as well as fruitful to inspect the individual figures of merit. This is nicely illustrated by the rather extreme case of PGE2 where a phase

set with a poor CFOM but good RESID led to the structure being elucidated. PSIZERO was found to be the most reliable of the figures of merit used in MITHRIL while ABSFOM was the least reliable. High ABSFOM and low RESID values were often an indication of over-consistency among the phases and could usually be remedied using Hull-Irwin weights.

It was observed that the heavy atom structures on the database, such as BOBBY and CUIMID, tended to give rather extreme values for the figures of merit. Even the correct solutions had rather odd figures of merit. Although the absolute values were meaningless in themselves, when compared to one another in a relative manner they were still of use.

The traditional weighted tangent formula was used to find the correct solution for the majority of the database structures but the Hull-Irwin statistically weighted tangent formula also performed with some distinction. It was used in the determination of seven of the structures: CUIMID, NO55, TVAL, TPH, MUNICH1, SUOA and WINTER2. It may be more than a coincidence that TVAL and WINTER2 are the two largest structures on the database. The Hull-Irwin scheme was generally found to be useful when dealing with heavy atom and symmorphic structures. The statistically weighted tangent formula is another of the options recommended to be tried early if difficulties are encountered in solving a structure.

The recent addition of the LSAM module to the MITHRIL package means that symbolic addition can now be used instead of tangent refinement for centrosymmetric structures. As was explained in the introduction, symbolic addition methods are fundamentally different to multisolution methods and, as such, provide a real alternative to them. LSAM was used in the determination of HOV1 where tangent refinement failed to find the correct solution. As was reported in the previous chapter, it has also been shown to work for all the other

centrosymmetric structures on the database.

4.4.5 E-MAPS

The interpretation of E-maps is the final hurdle to be overcome in the structure determination process. It is often far from being a trivial process especially when dealing with large or complex structures. A good illustration of this was given when inspecting the E-maps generated for WINTER2 where great difficulties were encountered in trying to ascertain which particular part of the molecule was being looked at. It is hoped that the use of interactive colour graphics routines, as described in the following chapter, will reduce these problems.

Several interesting observations were made when using MITHRIL to solve the database of difficult structures. If the E-map had one or two peaks much bigger than the rest, and there were no heavy atoms present, the phases were usually incorrect. If heavy atoms were present then the maps normally revealed only those atoms. For most of the structures some atoms were missing from the correct solution but it was found that Fourier recycling usually located the missing atoms.

4.5 DEALING WITH DIFFICULT STRUCTURES

MITHRIL will normally solve routine structures using default parameters throughout the program. In some cases, however, difficulties will arise and the E-map will not yield the solution. This section is designed to help users of the program decide what to do when confronted with such situations. There follows a list of ten non-default options that can be tried, either individually or in combination, to solve troublesome structures. The options are listed in an order of priority which is based on the experiences of using MITHRIL on the database of difficult structures - in fact this section is a corollary of the conclusions reached in the previous section.

1. Try to get as good a set of E-magnitudes as possible with the given data set. In MITHRIL there is a variety of options available to enable the user to achieve this. Among the most important of these are the commands LIMIT and BSCL. For example, in cases where some of the E-magnitudes are large, it can often be productive to use a LIMIT command to reduce or exclude them. The BSCL command can be used to input a temperature factor rather than let the program calculate its own - this can have a dramatic effect on the distribution of E-magnitudes. Additionally, the results obtained in this chapter suggest that the K-curve method of normalisation should be preferred to other methods, at least in the early stages of a structure determination.

2. Ensure that the convergence map is adequate. A common problem is for the convergence map to have a shortage of contributors and perhaps even weak links. Simple steps

which can be taken to remedy this are to introduce negative quartets (if they are not generated automatically for the structure), to increase the number of reflections for which phase relationships are generated and, lastly, to increase the number of starting set reflections.

3. If the structure is non-symmorphic introduce negative quartets into the procedure and ensure they participate actively in the convergence map. This may mean increasing the number of reflections for which quartets are generated. Note that MITHRIL generates quartets automatically for symmorphic space groups.

4. Use of the statistically weighted tangent formula can be helpful in many cases. The formula is accessed by means of the SWTR command in both the TANGENT and RANTAN modules of MITHRIL.

5. As was seen in Chapter2 employing either the weighting scheme or the cut-off technique - both of which are based on $(\sin^2\theta/\lambda^2)$ values - can often prove to be very useful.

6. If the structure is centrosymmetric symbolic addition, in the form of LSAM, can be used as an alternative to tangent refinement.

7. Generate more phase sets by using the SETS command in the CONVERGE module of the program. This increases the number of starting set reflections.

8. Other types of quartets such as positive quartets and those with reflections missing from their second neighbourhood can be introduced by means of various options in the QUARTET module.

9. Use random phases by calling the RANTAN module.

10. Run MAGEX between the convergence and tangent refinement procedures.

4.6 REVIEW - A NEW MODULE FOR MITHRIL

A new module, REVIEW, has been added to the MITHRIL package as a direct consequence of the experience gained by using the program on the database of difficult structures. For many of the structures it was found necessary to generate a lot of phase sets, in some cases more than 500 phase sets were produced. It is a laborious task to sift through hundreds of phase sets looking for good ones and mistakes can easily be made. The REVIEW module has been written and added to MITHRIL in an effort to eradicate this problem. It prints out the best phase sets for each of the individual figures of merit as well as CFOM together with an indication of whether a set has already been investigated.

The master menu in MITHRIL is now of the form:

```
The following modules are currently available:
NORMAL    (IK,NB,ISC,MAXDUP )    TRIPLET (NO OF REFLECTIONS)
QUARTET   (NO OF REFLECTIONS)    QUINTET (NO OF REFLECTIONS)
RECYCLE   ( FOURIER / KARLE )    CONVERG (NUMB,IQ4,IQ5     )
TANGENT   (                 )    LSAM    (                 )
RANTAN    (NO OF PHASE SETS )    MAGEX   (NO OF PHASE SETS )
YZARC     (NO OF PHASE SETS )    MAPS    (PHASE SETS NO(S) )
PATTERSON(                  )    REVIEW  (                 )

In addition the commands: TITLE,END,MENU,LEVEL,NOPRINT,PRINT,
                          DEFAULT,HARD,VERY HARD,MODEL,SHOW,X
may be issued at any time.
Only the first 4 characters of any command are required.
(CR) or a call to another module terminates current input.
```

REVIEW can only be run after TANGENT or LSAM have been run, otherwise an error message will appear saying that there are no phase sets to investigate. The module is entered simply by typing "REVIEW" which causes a message of the form:

```
Please wait for phase sets to be sorted.
```

to appear on the screen (if running interactively). When the phase sets have been sorted the REVIEW menu is displayed. It is of the form:

```
Choose from the following options:

 a - absfom
 b - psizero
 c - resid
 d - nqest
 e - nqint
 f - cfom
 x - exit module

a, b, c, d, e, f or x ? -
```

If "a" is selected from this menu then the figures of merit for the top 18 phase sets with respect to ABSFOM are displayed on the screen. Should a phase set have already been used to calculate an E-map then the word "map" will appear rather than a CFOM value to indicate that is the case. The options b-f behave in an exactly analogous manner to a. A warning message is output when a figure of merit is selected when it was not used. For example, if "d" is selected when no quartets were used then the message:

```
Sorry, nqest not used. Please try again.
```

is displayed on the screen. Selecting "x" causes MITHRIL to

exit from the REVIEW module and revert to displaying the master menu where it awaits further instruction.

REFERENCES

1. G.M.Sheldrick (1985). in Crystallographic Computing 3, eds.: G.M.Sheldrick, C.Kruger & R.Goddard, Oxford: Clarendon Press, 184-189

2. J.Rogers & O.Kennard, Unpublished results

3. S.C.Wallwork & H.M.Powell (1980). J.Chem.Soc.Perkin 2, 641-646

4. B.L.Barnett & V.A.Uchtman (1979). Inorg. Chem., 18, 2674-2678

5. E.Hovestreydt, K.Klepp & E.Parthé (1983). Acta Cryst., C39, 422-425

6. W.Clegg, K.Harms, G.M.Sheldrick, G. von Kiedrowski & L.F.Tietze (1980). Acta Cryst., B36, 3159-3162

7. A.Colens, J.P.Declerq, G.Germain, J.P.Putzeys, M. van Meersche (1974). Cryst. Struct. Comm., 3, 119-122

8. W.Clegg, S.R.Acott & C.D.Garner (1984). J.Chem.Soc. (Dalton), 2581

9. J.C.Braekman, D.Daloze, A.Dupont, B.Tursch, J.P.Declerq, G.Germain & M. van Meerssche (1981). Tetrahedron, 37, 179

10. G.M.Sheldrick, B.E.Davison & J.Trotter (1978). Acta Cryst., B34, 1387-1389

11. P.G.Jones, G.M.Sheldrick, K-H.Glusenkamp & L.F.Tietze (1980). Acta Cryst., B36, 481-483

12. D.Suck, P.C.Manor & W.Saenger (1976). Acta Cryst., B32, 1727-1737

13. G.D.Smith (Medical Foundation of Buffalo), Unpublished.

14. G.M.Sheldrick & J.Trotter (1974). Acta Cryst., B34, 3122-3124

15. I.L.Karle & J.Karle (1981). Proc. Natl. Acad. Sci. USA, 78, 681-685

16. D.J.Williams & D.Lawton (1975). Tertrahedron Letters, 111-114

17. I.L.Karle (1975). J. Amer. Chem. Soc., 97, 4379-4386

18. G.D.Smith, W.L.Duax, D.A.Langs, G.T.DeTitta, J.W.Edmonds, D.C.Rohrer, & C.M.Weeks (1975). J. Amer. Chem. Soc., 97, 7242-7247

19. A.Hoekstra, A.Vos, P.B.Braun & J.Hornstra (1975). Acta Cryst., B31, 1708-1715

20. R.Grigg, J.Kemp, G.M.Sheldrick & J.Trotter (1978). J.C.S. Chem. Comm., 109-111

21. H.Irngartinger, W.R.K.Reibel & G.M.Sheldrick (1981). Acta Cryat., B37, 1768-1771

22. U.Szeimies-Seebach, J.Harnisch, G.Szeimies, M. van Meerssche, G.Germain & J.P.Declerq (1978). Angew. Chem. Int. Edn., 17, 848-850

23. M.B.Hursthouse, Unpublished results

24. G.T.DeTitta, D.A.Langs, J.W.Edmonds & W.L.Duax (1980). Acta Cryst., B36, 638-645

25. J.D.Oliver & L.C.Strickland (1984). Acta Cryst., C40, 820-824

26. T.Butters, P.Hutter, G.Jung, N.Pauls, H.Schmitt, G.M.Sheldrick & W.Winter (1981). Angew. Chem., 93, 904-905

27. L.Lessinger (1976). Acta Cryst., A32, 538

28. I.Vickovic & D.Viterbo (1979). Acta Cryst., A35, 500-501

CHAPTER5

THE USE OF INTERACTIVE COLOUR GRAPHICS TO AID THE INTERPRETATION OF E-MAPS IN MITHRIL

5 THE USE OF INTERACTIVE COLOUR GRAPHICS TO AID THE INTERPRETATION OF E-MAPS IN MITHRIL

5.1 INTRODUCTION

During a crystal structure determination using direct methods, an important step is the interpretation of E-maps. In most direct methods computer programs, the interpretation consists of determining a set of peaks according to their height in the E-map and then making use of sensible chemical criteria - such as maximum and minimum peaks distances and angles - to eliminate spurious peaks. A line printer plot on to a least squares plane along with a table of the peak-peak distances and angles completes the interpretation.

In the first release of MITHRIL the interpretation was assisted by the use of some relatively primitive computer graphics. A stick diagram of any molecular fragment found in the E-map interpretation could be displayed on a graphics terminal provided it supported the SIMPLEPLOT[1] library of graphics routines. It was found that even simple graphics facilities, such as these, could quite transform the time-consuming and sometimes arduous task of searching E-maps for recognisable molecular fragments.

As direct methods become more successful they are being applied to increasingly larger and more complex structures. The corresponding E-maps of these structures are, of course, much more difficult to interpret. The approaches discussed above are perfectly adequate for dealing with simple structures but are far too simplistic for difficult structures. It is felt that the interpretation of E-maps can

be greatly simplified by the utilisation of colour computer graphics facilities. This view is enhanced by the fact that such facilities are becoming commonplace in crystallographic laboratories.

Two quite separate graphics modules have been written to aid the interpretation of E-maps in MITHRIL: MITHGRAPH and CONTOUR.

MITHGRAPH enables the user to display, manipulate and edit molecular fragments found by MITHRIL. The display is in colour and the program is menu driven. There are of course a number of molecular modelling programs already in existence that can be used for displaying molecules. The functions of these programs vary considerably. Some such as ORTEP[2], PLUTO[3] and SPACEFILL[4] are used for the production of publication illustrations; while others such as MOLBUILD[5] and CHARMM[6] perform model building for energy minimisation. Somewhat more sophisticated than these is FRODO[7] which is used in macromolecular structure determination. In addition to these a program - IMAGE[8] - has recently been reported which uses the peaks from E-maps calculated by direct methods programs like MULTAN for the visualisation of molecular structures. MITHGRAPH has most in common with the last of these programs.

In a sense MITHGRAPH deals with the product of the E-map interpretation rather than with the interpretation itself. It was felt that direct observation of the 3-dimensional contoured E-map could well reveal information that would otherwise be obscured. For this reason a new module called CONTOUR has been written and added to the MAPS module of MITHRIL. This is a routine which will contour planes of either an electron density map or a Fourier map on to a colour graphics screen.

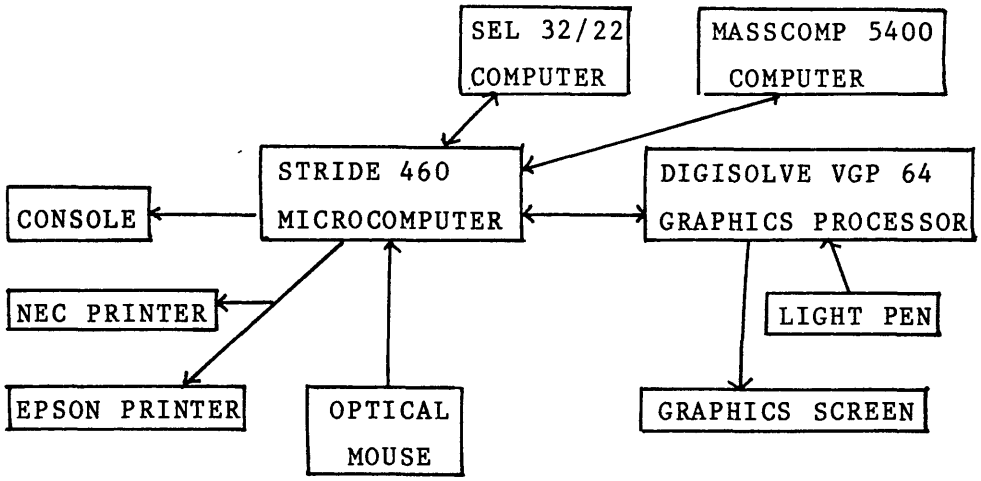
Both MITHGRAPH and CONTOUR are discussed in greater

detail in forthcoming sections of this chapter. As was stated previously, they were written for use with the MITHRIL program. It was discovered, however, that it was not practical to run a full version of MITHRIL on the colour graphics workstation at Glasgow due to lack of both memory and processing power. In spite of this problem it remained a priority to use the colour graphics routines to their best advantage. This has been accomplished by putting together a program called SUBMIT which, as the name suggests, is basically a subset of the MITHRIL program. SUBMIT is made up from the RECYCLE, MAPS and peak search modules from MITHRIL together with the new graphics routines CONTOUR and MITHGRAPH.

SUBMIT performs three separate functions: weighted Fourier recycling[9,10] of molecular fragments, E-map calculation and E-map inspection. Again it is discussed in greater detail in a later section.

5.2 THE GLASGOW COLOUR GRAPHICS WORKSTATION

The colour graphics workstation at Glasgow used in this thesis is shown in diagrammatic form overleaf. As can be seen the system is built around a STRIDE 460 microcomputer. This is a machine based on the Motorola M68000 processor chip with 2 Mbytes of memory and 16 Mbytes of hard disk capacity. The STRIDE drives the DIGISOLVE VGP 64 which is a dedicated graphics processor. The 19", colour graphics screen is 512 by 512 pixels and is, therefore, of medium resolution. There is a choice of printers for screen dumps.



5.3 MITHGRAPH

A new module, titled MITHGRAPH, has been added at the end of the MITHRIL package. It is designed to run automatically after the MAPS module. MITHGRAPH enables the user to display and manipulate molecular fragments found by the SEARCH part of MITHRIL. The display is in colour and atoms are identified by their colour code. The program is menu driven - all input is by means of a light pen with the exception of the input of numbers which is done more efficiently using the keyboard.

In addition to the functions mentioned above MITHGRAPH also provides a direct interface to the CHEMMOD[11] system of molecular modelling programs - it is possible to create a CHEMMOD data file using MITHGRAPH.

There are two main menus in the program:

(a) Master menu - performs manipulations, such as rotation and translation, of molecular fragments.

(b) Edit menu - performs edit functions, such as removing an atom from a molecular fragment, as well as geometry calculations.

The next section contains details of all the options available in the MITHGRAPH module. There are descriptions of what each of the options does together with, where appropriate, instructions for their use. The flowchart of the MITHGRAPH options (Figure 5.1) illustrates all of the options in the module as well as giving an indication of what happens after a particular command is selected. When MITHGRAPH is entered a title page is displayed on the graphics screen (Figure 5.2). The module can then be initialised by placing the light pen over the '*' on this display.

5.4 Options available in Master menu of MITHGRAPH

Figure 5.3 shows the full range of options available under the master menu: this illustration is typical of the display on the colour graphics screen immediately after the module is initialised. An option is selected by pointing the light pen at the first two letters of the command. There are 13 options available in the master menu of MITHGRAPH as follows:

5.4.1 QUIT

This command causes the program to terminate immediately. An "END OF PROGRAM" message is displayed on the

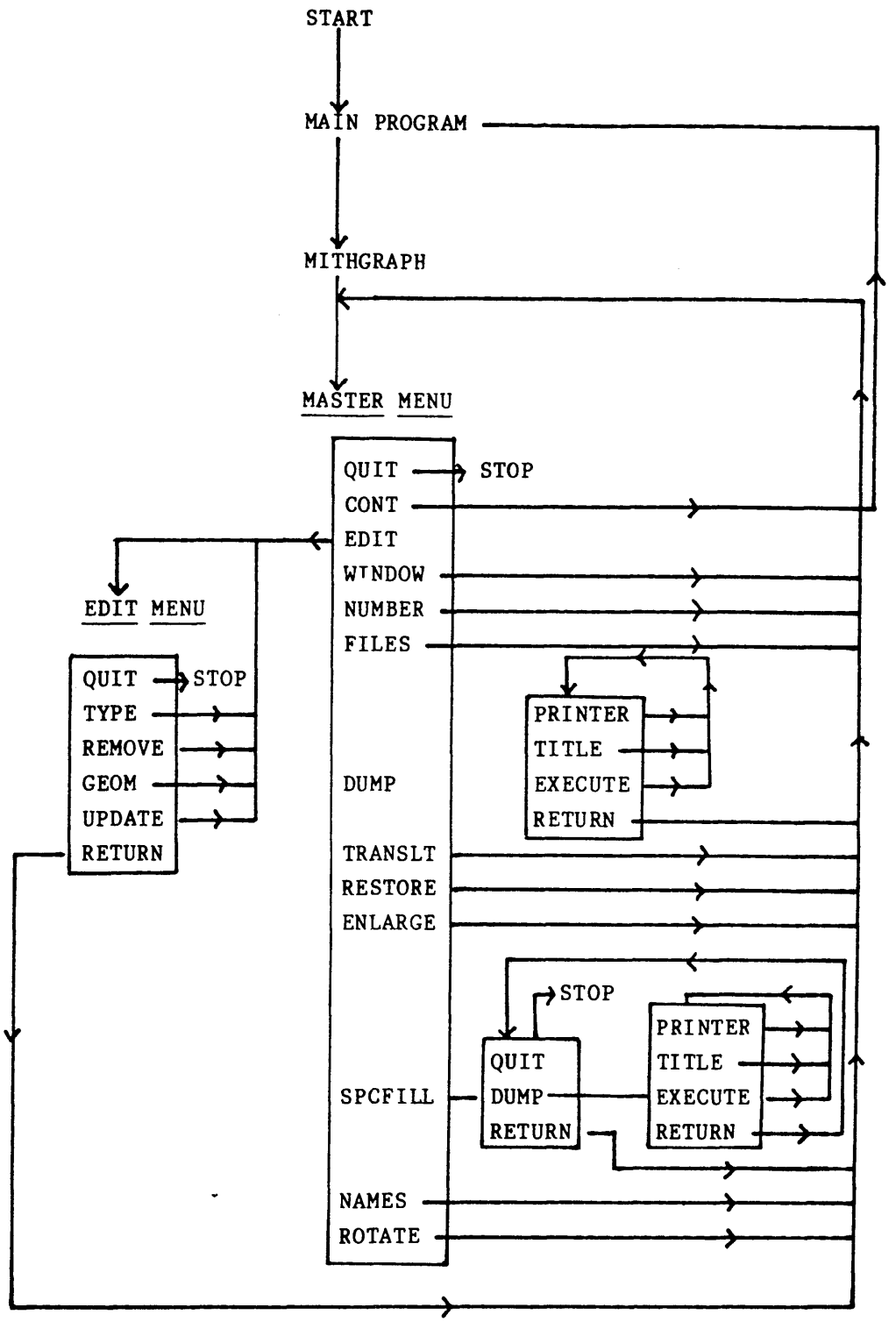


Figure 5.1 Flowchart of the MITHGRAPH options

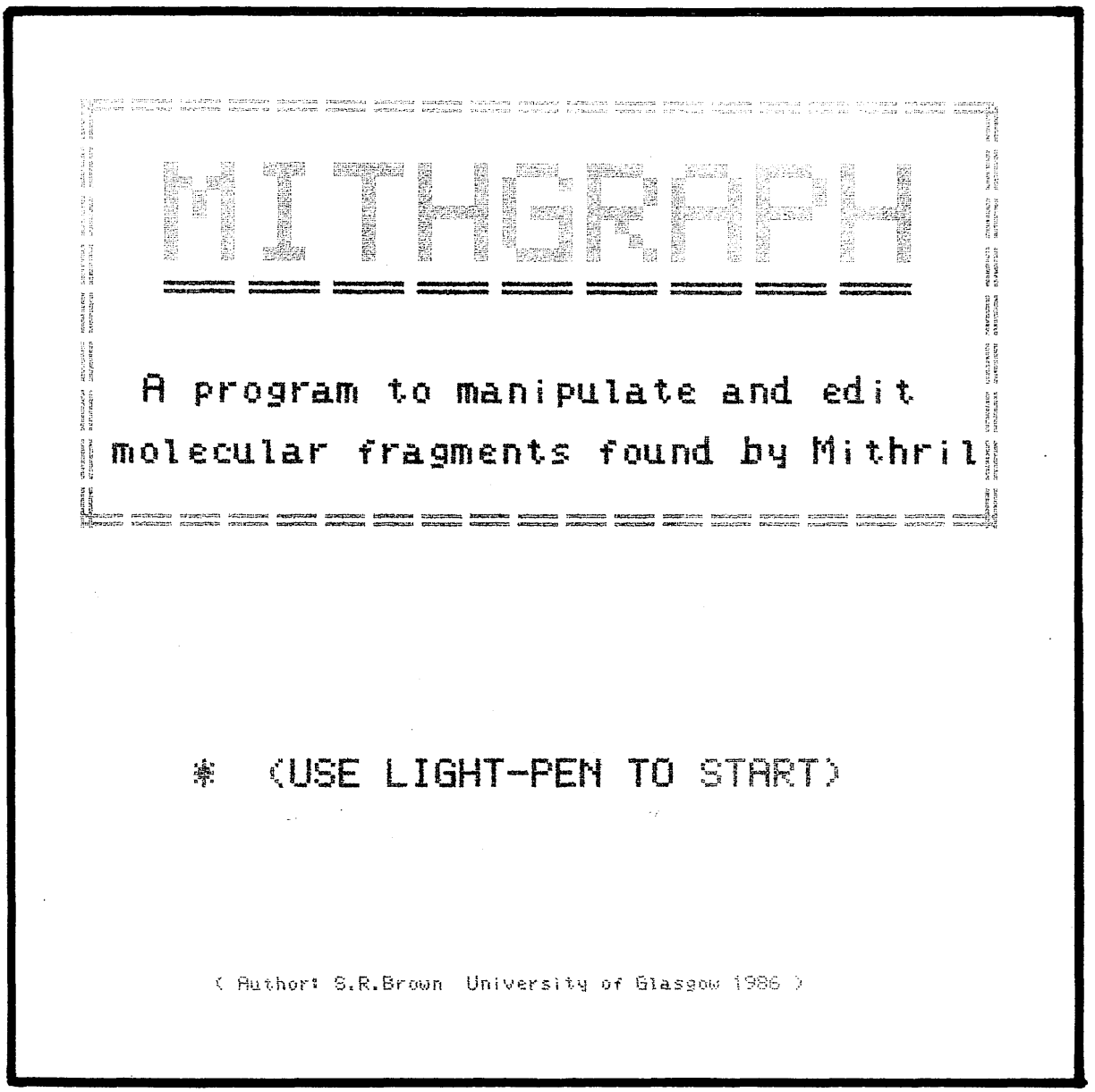


Figure 5.2 MITHGRAPH title page

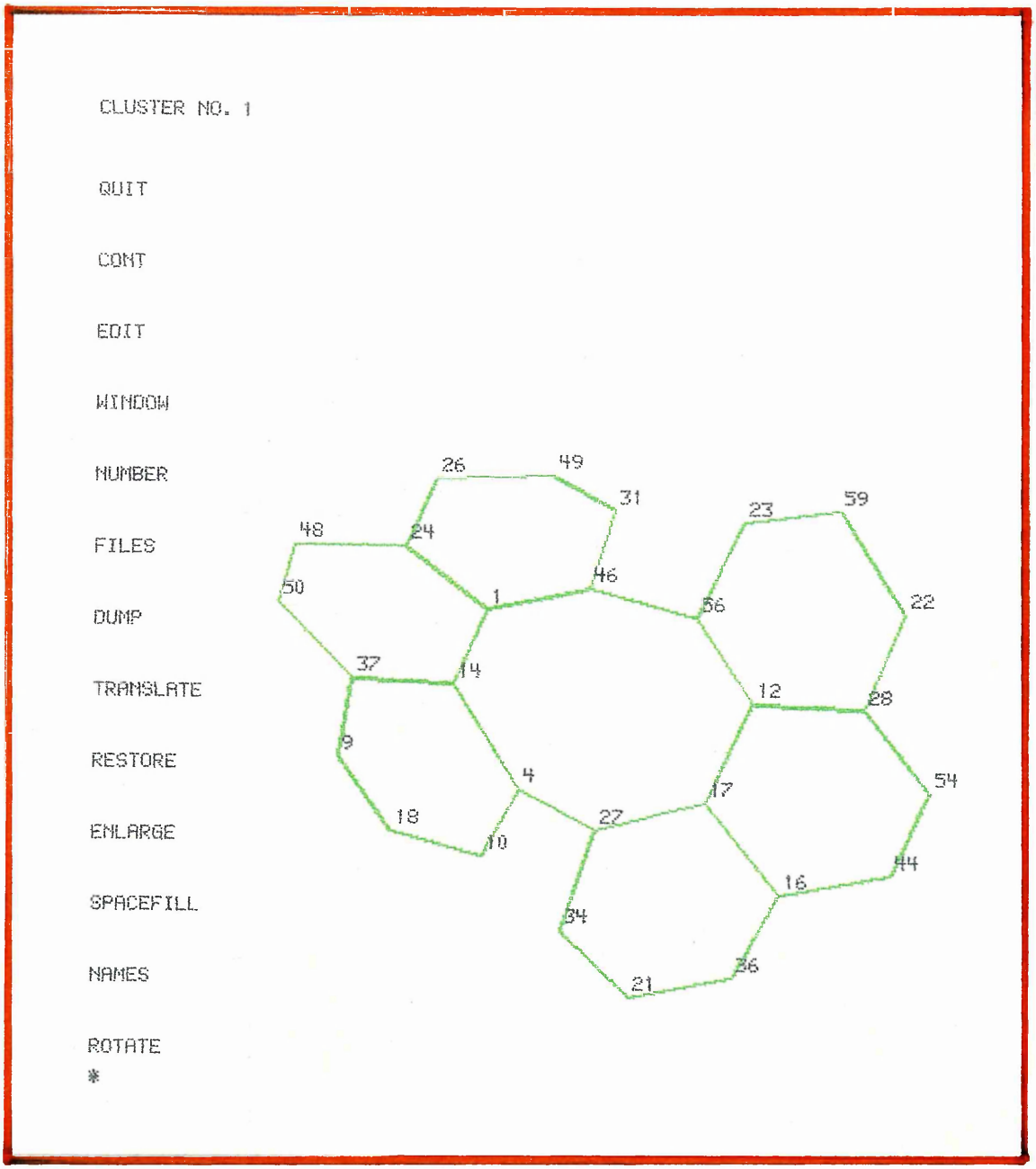


Figure 5.3 Display of GOLDMAN2 with master menu of MITHGRAPH

graphics screen.

5.4.2 CONT

CONT(INUE) causes the program to finish with the molecular fragment currently on display and then moves on to display the next peak interpretation or cluster. If all the peak clusters have been investigated MITHGRAPH is exited and an "END OF PROGRAM" banner is displayed on the colour graphics screen.

5.4.3 EDIT

Selection of this option causes the program to go into edit mode which has a separate menu. This is discussed in greater detail in section 5.5. All of the edit functions in the module are accessed through this command.

5.4.4 WINDOW

This option alternately inserts and removes a window round the fragment on display - a cosmetic effect to enhance the appearance of the screen display.

5.4.5 NUMBER

The atoms in the fragment(s) are numbered according to their peak heights in the E-map. If the atom types are on display when this command is selected then they are removed to make way for the numbers. NUMBER can be useful in determining whether a peak interpretation is likely to be correct - for example, if a peak interpretation contains

several very small peaks in key areas of the structure then it is probably (but not definitely) wrong.

5.4.6 FILES

Using this option the user has the choice of creating either a data file which can be used in the CHEMMOD system of molecular modelling routines, or else a file which can be used in MITHGRAPH2 - a stand alone version of MITHGRAPH. MITHGRAPH2 is considered in greater detail in section 5.6.

When the FILES option is selected, the master menu is removed and the prompt:

Enter type of file required

is displayed at the top of the graphics screen. At the same time the following menu is displayed at the foot of the screen:

*** MITHGRAPH2 * CHEMMOD * NEITHER**

If FILES was selected by mistake, or if the user no longer wishes a data file to be created, the 'NEITHER' option will cause the program to return to the main menu without any files having to be written.

If the user selects the 'CHEMMOD' option, the following message will appear on the system console :

Enter the name of the CHEMMOD data file -

The user enters a file name and it is created automatically. However, if the file name entered by the user is that of a file which already exists the program will await

permission from the user before over-writing it.

An exactly analogous procedure operates for the creation of MITHGRAPH2 files.

5.4.7 DUMP

This command is used to obtain hard copies of the graphics screen display (minus any menu which may be on view).

When this option is selected the hardcopy menu replaces the master menu. The display at this stage is shown in Figure 5.4. There are four options available under the hardcopy menu:

5.4.7(a) PRINTER

This option allows the user to select the printer to be used for the screen dump. In the Glasgow system there is a choice of two black and white printers: an Epson FX-80 and a NEC pinwriter P3. When this option is selected a message of the form:

Printer is presently set to NEC.
Do you wish to change to Epson ? (y/n) :

appears on the console screen. The same printer will be used for all hard copies unless the other one is selected by means of this command.

5.4.7(b)TITLE

A title, up to a maximum of fifty characters, can be

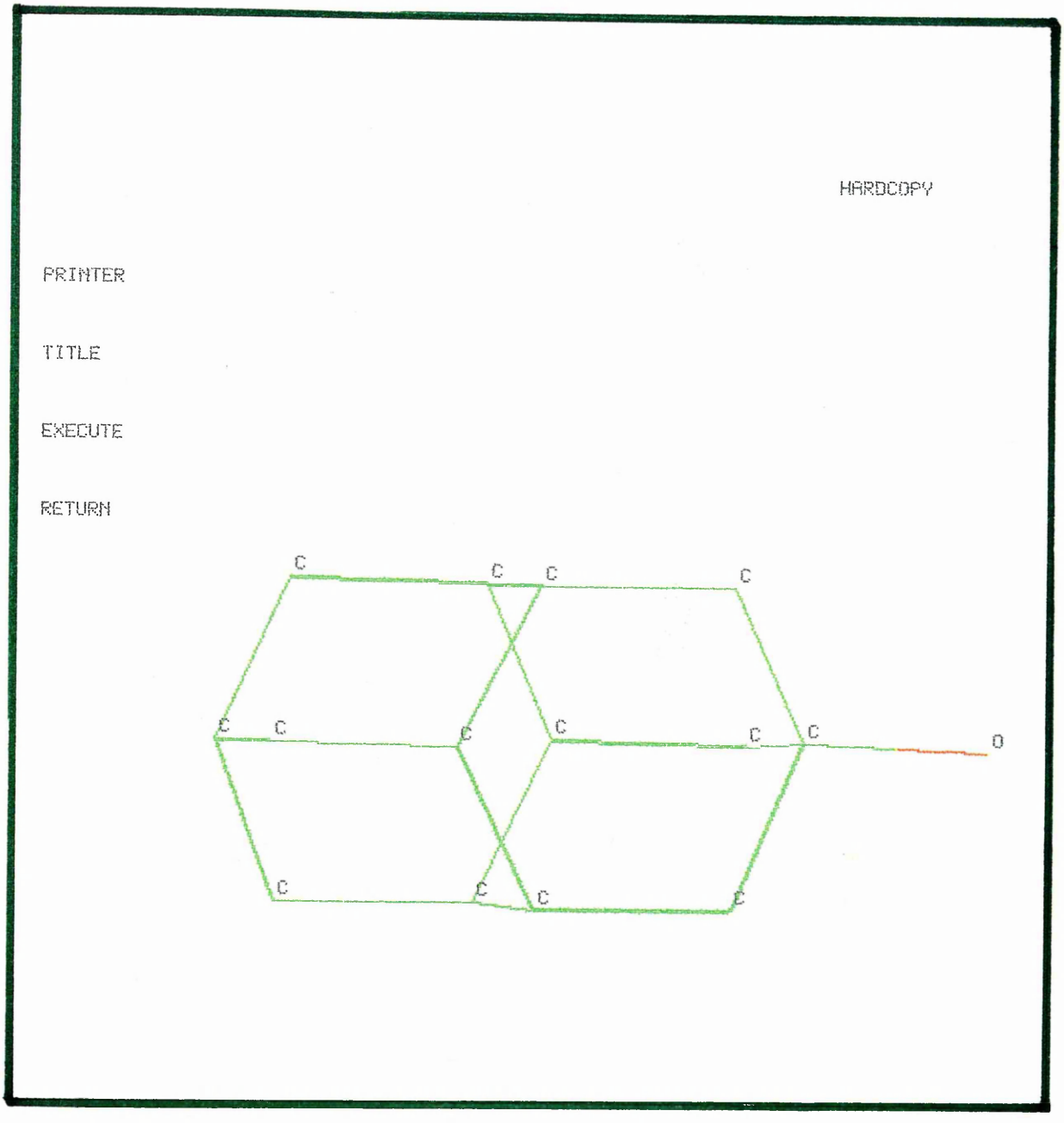


Figure 5.4 Hardcopy menu of MITHGRAPH with display of DIAMANTANE-4-OL

added to the screen display using this option. A three button optical mouse is used to position the text.

When TITLE is selected the following instructions appear on the console screen:

Use the cursor to locate the point where text will start. Press the middle key of the mouse at that point.

Place the mouse in the centre of the pad and press any of its keys.

Depression of a mouse key causes a small yellow cursor to appear in the middle of the graphics screen. The cursor is moved by moving the mouse. When the cursor is at the desired position press the middle key of the mouse. This causes the following prompt to appear on the console screen:

Enter title (maximum of 50 characters)

The title is entered via the keyboard. When the (CR) key is pressed the title appears on the graphics screen and the program displays the hardcopy menu.

5.4.7(c)EXECUTE

Selection of this command causes the screen dump to be carried out immediately - the master menu is displayed afterwards. Care should be taken to ensure that the desired printer for the hard copy has in fact been selected. This can be verified using the PRINTER command described above.

5.4.7(d) RETURN

RETURN causes the program to return to displaying the master menu without carrying out a screen dump. This is useful where the DUMP option was selected by mistake.

5.4.8 TRANSLATE

TRANSLATE allows the molecular fragment on display to be moved both horizontally and vertically (positively as well as negatively). When this option is selected the graphics screen display is as shown in Figure 5.5. The program displays the menu:

```
  * X     * Y     * STOP
* Q   * S   * -   * +
```

and the message:

Choose + or - first

is displayed in the upper left part of the screen. The user selects the positive or negative direction for the translation by pointing the light pen at the '*' to the left of either the '+' or the '-'. The message:

Now choose direction of translation

is then displayed. The translation is started by choosing 'X' (for horizontal) or 'Y' (for vertical). As soon as the translation begins the messages in the upper left part of the screen are removed.

The direction of translation can be changed at any time by pointing the light pen at the desired axis. Similarly the sense of the translation can be altered using the '+' and '-' signs. The translation can also be speeded up or slowed

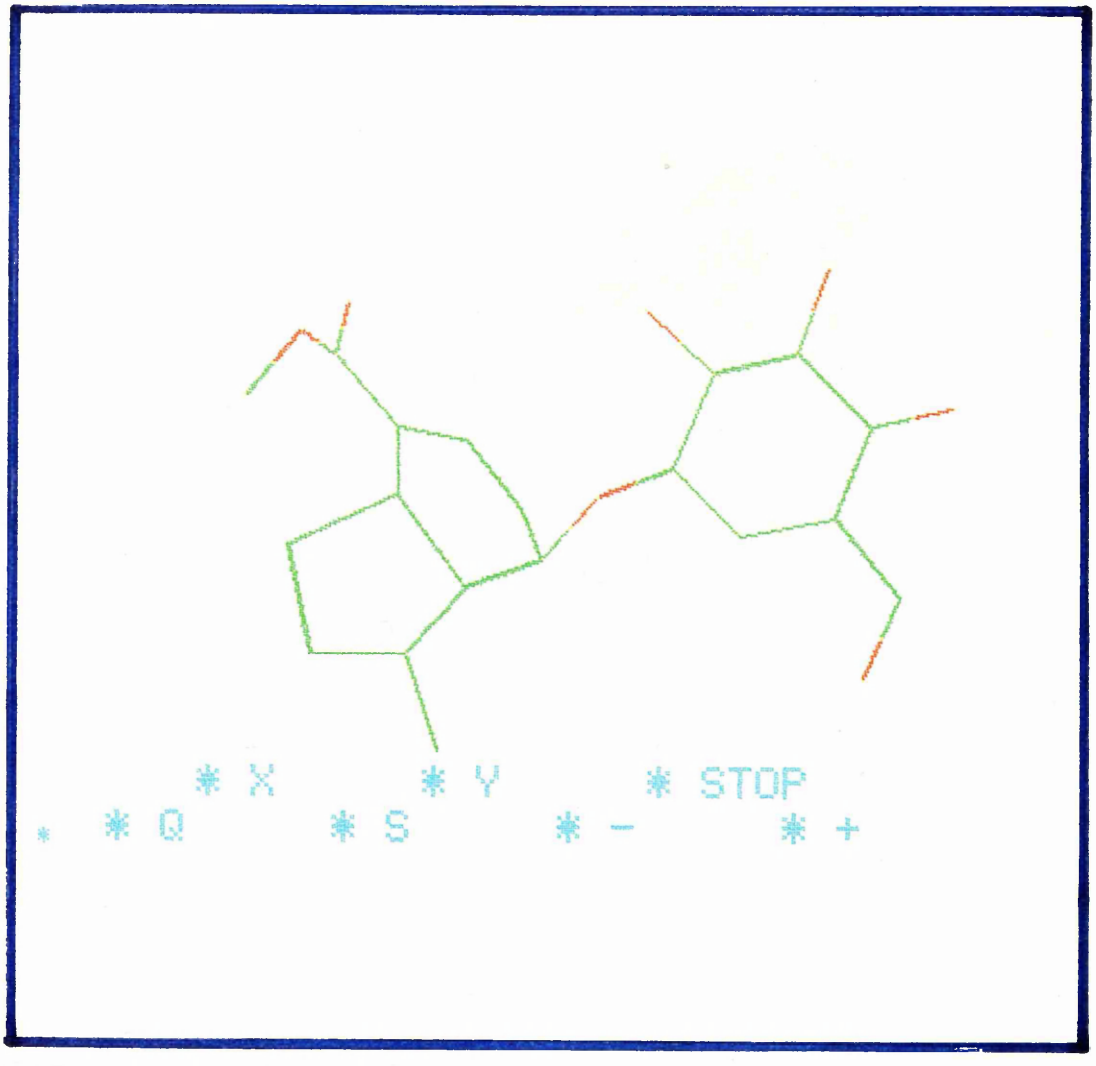


Figure 5.5 Display of LOGANIN with the translate menu of MITHGRAPH

down by pointing the light pen at 'Q' (quick) or 'S' (slow) respectively. Pointing the light pen at 'STOP' ends the translation and the program returns to displaying the master menu.

Note that translation along the Z axis is exactly equivalent to the molecule (or fragment) on display being enlarged or reduced. There is a separate option, called ENLARGE, included in the program to perform these functions.

5.4.9 RESTORE

The RESTORE option resets the orientation and size of the fragment on display to what they originally were when MITHGRAPH was first entered. It should be noted that any editing which may have been carried out on the fragment does not get undone. This means that any atoms that have been removed are not restored. Similarly, any atoms that have changed types do not revert to what they originally were.

5.4.10 ENLARGE

This option is particularly useful when looking at large or complicated fragments as it allows the user to enlarge or reduce the size of the fragment displayed. As previously mentioned, enlarging/reducing is equivalent to translation along the Z-axis.

After the ENLARGE option is selected, the fragment is displayed together with the menu (Figure 5.6):

*** ENLARGE * REDUCE * STOP**

Selecting 'ENLARGE' with the light pen enables the

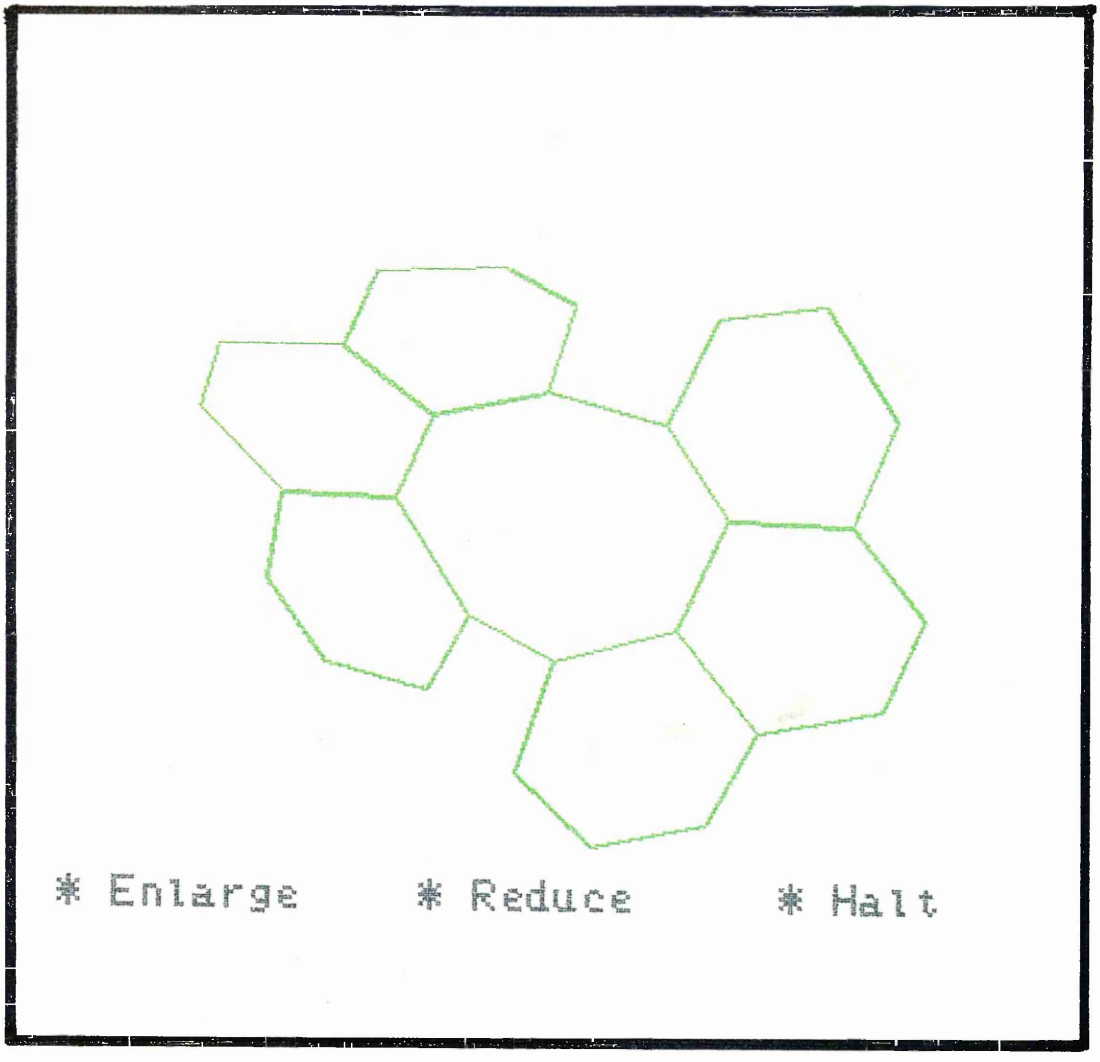


Figure 5.6 Display of GOLDMAN2 with the enlarge menu of MITHGRAPH

user to effectively zoom in on the structure and it is scaled up in size. 'REDUCE' has the opposite effect: it causes the fragment to decrease in size. To stop the zoom and exit from the option select 'STOP' with the light pen. Enlarging / reducing will continue indefinitely unless either the opposite command or 'STOP' is selected.

Notice that the shape of the fragment is always retained using this option. This is achieved by scaling from the "centre of gravity" of the fragment.

5.4.11 SPCFILL

This produces a spacefilled display of the current molecular fragment. The size of sphere used for an atom is directly proportional to its atomic radius. The atoms are drawn in order of increasing Z-coordinate to ensure the correct perspective is displayed.

After the spacefiled fragment is drawn, a menu is displayed in the upper left part of the screen (Figure 5.7). There are three options in this menu:

5.4.11(a) QUIT

Exit from program immediately (i.e. program abort). "END OF PROGRAM" message is displayed on the graphics screen.

5.4.11(b) DUMP

This option allows the user to obtain a screen dump of the spacefilled display. After DUMP has been selected, the spacefilled fragment is displayed together with the hardcopy

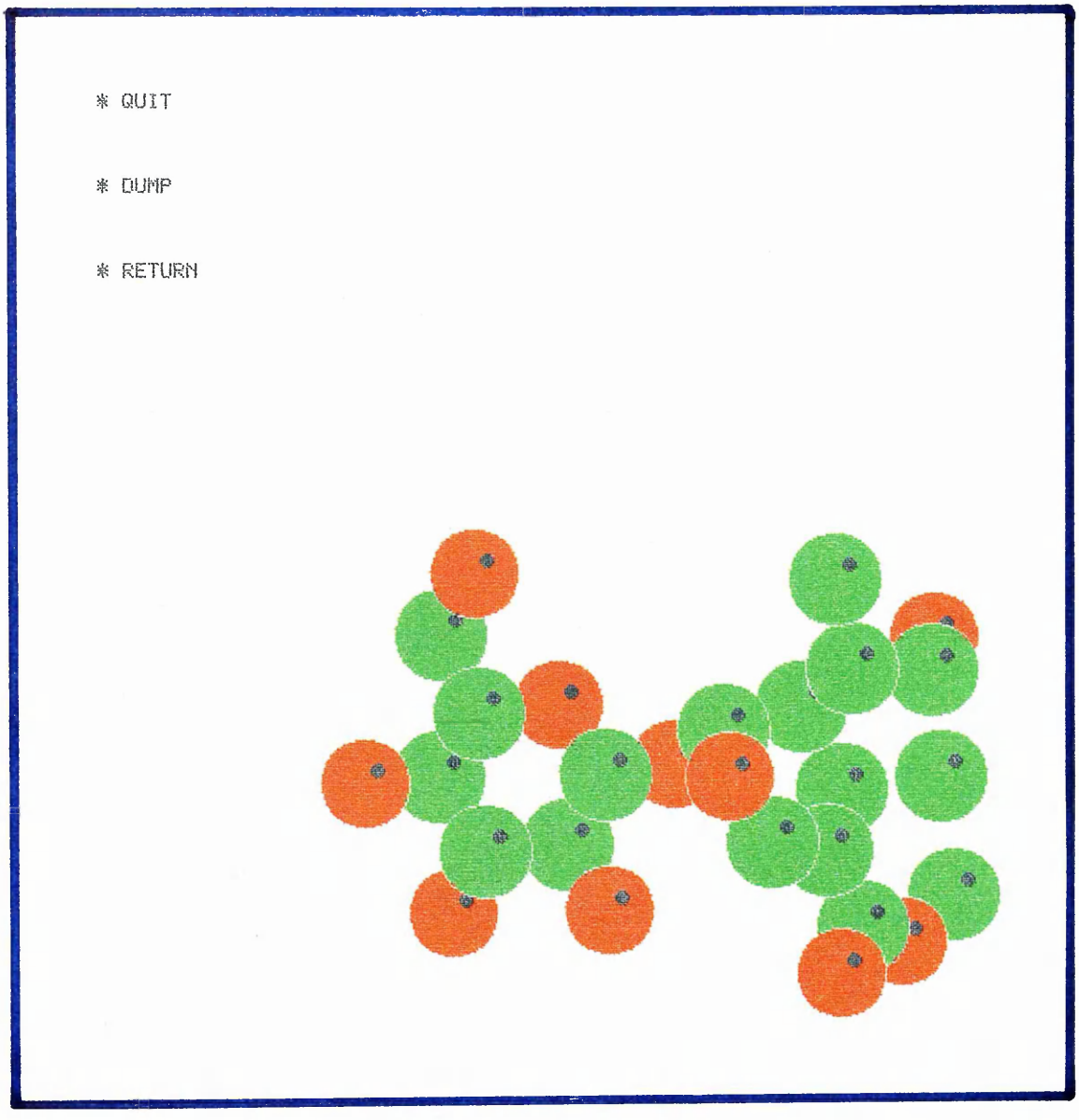


Figure 5.7 Spacefilled display of LOGANIN with spacefill menu

menu. The options available are identical to those already described for the DUMP command in Section 5.4.7 with the exception that in this case RETURN causes the spacefill menu to be displayed rather than the master menu.

5.4.11(c) RETURN

Exit from spacefill mode. The program reverts to displaying a stick diagram of the fragment together with the master menu.

5.4.12 NAMES

The atoms in the fragment(s) are labelled with their chemical symbols. If the atom numbers are on display when this option is chosen then they are removed to make way for the names.

5.4.13 ROTATE

This option allows the user to rotate the fragment about the X, Y or Z axes. It is especially useful when dealing with large or crowded molecular fragments as it enables the user to view the structure from a variety of orientations. The option works in a similar way to the TRANSLATE option described above. When ROTATE is selected a menu of the form:

*** X * Y * Z * STOP**
*** Q * S * + * -**

is displayed followed by the commands:

Choose + or - first

and:

Now axis for rotation

The menu is used in exactly the same way as for the translate option. The rotation is achieved by matrix transformation of atom coordinates[12].

5.5 OPTIONS AVAILABLE UNDER EDIT MENU

After the EDIT option is selected the master menu disappears immediately and three seconds later the edit menu is displayed. This delay is designed to allow the user time to take the light pen away from the screen before the edit menu appears - so helping to avoid problems of options being selected inadvertently. Once again, the menu is situated in the upper left part of the graphics screen (Figure 5.8).

The atoms in the fragment on display are automatically numbered as many of the options require the input of these numbers. There are six options available in this menu as follows:

5.5.1 QUIT

QUIT causes the program to terminate immediately. An "END OF PROGRAM" message is displayed on the graphics screen.

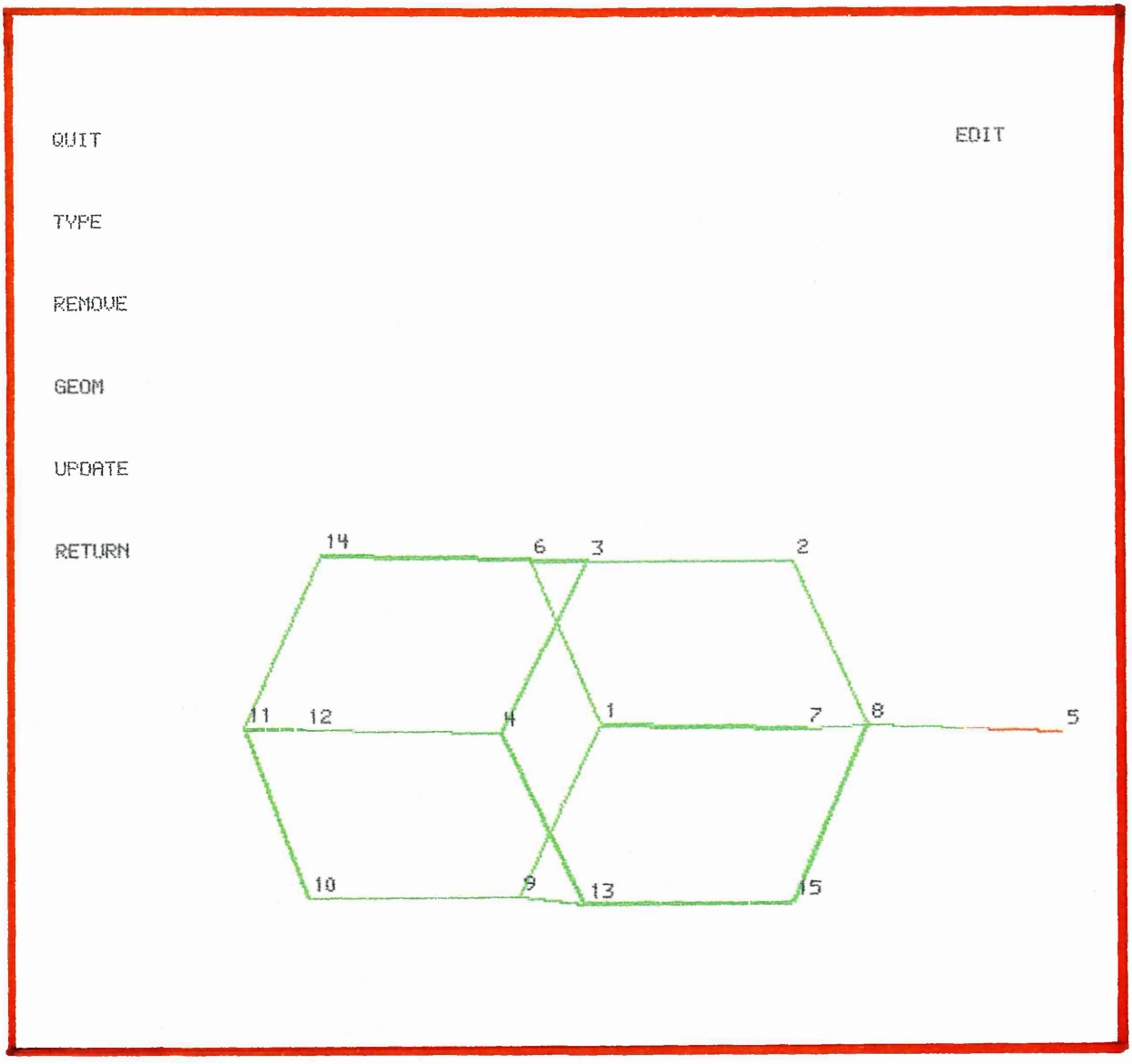


Figure 5.8 Display of DIAMANTANE-4-OL with edit menu of MITHGRAPH

5.5.2 TYPE

This option allows the user to change the atom type of an individual atom or that of a group of atoms all of the same type. The program checks the connectivities of each atom before changing its type in case the change is disallowed by valency. All atoms in a fragment are initially assumed to be carbon sp3.

After the option is selected two commands are displayed in the upper left part of the screen:

*** Atom type**
*** Single atom(s)**

5.5.2(a) Atom type

A menu of atom types of the form:

*** C sp3**	*** O Double bond**
*** C sp2**	*** S Digonal**
*** C Aromatic**	*** S Tetrahedral**
*** C sp**	*** Phosphorus**
*** N (+)R(1)R(2)**	*** Hydrogen**
*** N Tetrahedral**	*** Fluorine**
*** N Aromatic**	*** Chlorine**
*** N (-)Aromatic**	*** Bromine**
*** O Single bond**	*** Other**

is displayed along with, at the top of the screen, the message:

CHOOSE ATOM TYPE TO BE CHANGED
FROM THE LIST BELOW

The user places the lightpen to the left of the atom type to be changed. This causes a new message to appear above the menu of atom types:

PICK NEW ATOM TYPE FROM THE LIST BELOW

Again the user selects the desired atom type usingthe light pen. The program then displays the fragment containing the new atom types together with the EDIT menu.

5.5.2(b) Single atom(s)

After this option is selected the message:

Type in number of atom to be changed

appears on the console screen. Input of a valid atom number causes the menu of atom types shown above to be displayed on the graphics screen together with a message of the form:

ATOM IS PRESENTLY: C sp3
PICK NEW ATOM TYPE FROM THE LIST BELOW

The new atom type is selected using the light pen. The fragment is then displayed along with the menu:

Change more atoms ?
*** Yes * No**

Choosing 'Yes' causes the whole process to repeat whereas 'No' exits from the option.

5.5.3 REMOVE

This option enables the user to remove a specified atom from the fragment on view. The edit menu disappears when this option is selected and the prompt:

Remove which atom ? ⟶

is printed at the bottom of the graphics screen. The arrow points towards the console screen where the question:

Atom number ?

is displayed. Input of a valid atom number causes that atom to be removed from the fragment. The edited fragment is again displayed with the menu:

Remove more atoms ?
*** Yes * No**

appearing at the foot of the screen. Choosing 'Yes' causes the whole removal procedure to be repeated. 'No' exits from the option and the program returns to displaying the edit menu.

5.5.4 GEOM

GEOM prints out tables of interpeak distances and bond angles on the console screen. The tables are printed out a page at a time. When all distances and angles have been shown the message:

---- Geometry calculations complete ----

appears on the console screen.

5.5.5 UPDATE

This option is for use whenever a group of atoms have been fed into the program by means of a model file. The file

is altered so that the atoms of the fragment on display replace the original group of atoms.

As will be discussed in section 5.8 the MITHRIL subset SUBMIT contains a NORMAL module which has been substantially modified in order to make it re-enterable. This enables the user to do succesive cycles of weighted Fourier syntheses without having to exit from the program each time. It also means that the user no longer has to type in the coordinates of the new group of atoms to be recycled.

After UPDATE is selected the message:

This option will store the atoms in this fragment as a group in a model file for use in Fourier recycling. Proceed ? (y/n) -

is written on the console screen. Typing 'n' cancels the option whereas 'y' causes the fragment atoms to be automatically written on to the model file.

Note that in the SUBMIT program this option only works in the Fourier recycling part of the program. If the option is selected when SUBMIT is being used to perform some other function, the message:

Option not available

appears on the console screen while the edit menu and fragment are re-displayed on the graphics screen.

5.5.6 RETURN

This option replaces the edit menu with the master menu.

5.6 MITHGRAPH2

MITHGRAPH2 is a quite separate, stand alone version of MITHGRAPH. As the name implies, it performs almost all of the same routines as MITHGRAPH. The CONT and GEOM routines are unavailable in MITHGRAPH2. All of the options available in the program, with the exception of UPDATE, work exactly as described previously. The program allows the user to manipulate and/or edit a molecular fragment without having to re-enter MITHRIL. No time is wasted, therefore, in carrying out duplicate calculations.

MITHGRAPH2 uses a data file written under the FILES option in MITHGRAPH. In fact, FILES is one of the most important options in MITHGRAPH as it allows the user to save any version of the molecular fragment that has been formed using the other options. So, in a sense, FILES can also be regarded as a "save structure" routine.

5.6.1 USING MITHGRAPH2

To enter the program the user types "Mith2". This causes the display:

```
*****************************************
*        WELCOME TO MITHGRAPH2          *
*        ====================           *
*                                       *
*  A program to manipulate and edit     *
* molecular fragments found by MITHRIL  *
*****************************************
```

Enter the name of the MITHGRAPH2 file -

to appear on the console screen (a similar title is written on the graphics screen). Entry of a valid file name causes the

program to display the structure along with the master menu. However, should an incorrect file name be input, the message:

File does not exist. Try again ? (y/n):

will be shown on the console. A 'y' causes the file name prompt to be re-displayed whereas an 'n' terminates the program.

5.6.2 THE UPDATE OPTION IN MITHGRAPH2

The UPDATE option works in a slightly different way in MITHGRAPH2 than in MITHGRAPH: here it requires the model file to be specified. When UPDATE is selected the message:

Enter the name of the model file -

is written on the console. When a valid file name is entered in response to this prompt, the program updates the file and then displays the message:

---- New model file written ----

on the console to indicate the update has taken place. If an incorrect file name is entered, the program gives the user the choice of trying again or leaving the option.

5.7 THE CONTOUR MODULE

It has been shown, using the MITHGRAPH module as an

example, how the use of interactive colour graphics can assist the interpretation of E-maps by enabling the user to manipulate displays of the various peak interpretations. It must be remembered, however, that even when used with routines such as MITHGRAPH the actual interpretation of the E-map is still performed automatically by computer and remains a "black box" process. It seems obvious that more information on the E-map interpretation would become available if it were possible to visualise the E-map itself - the CONTOUR module has been written to do just that. As the name implies it is a contouring routine which can display contoured sections of an electron density map or a Fourier map on a colour graphics screen. The CONTOUR routine has been added to the direct methods program MITHRIL as an option in the MAPS module.

The contouring routine uses Diamond's[13] algorithm for contouring a square grid of points. As with MITHGRAPH, the CONTOUR module is menu driven with light pen input.

In order to run the CONTOUR program, the user must select the CONT option when the menu for the MAPS module is displayed. After the E-map (or Fourier map) has been calculated and interpreted the title page for the CONTOUR module is displayed on the graphics screen (Figure 5.9) while the caption:

```
*********************************************
*                                           *
*    WELCOME TO THE CONTOURING PROGRAM      *
*   ----------------------------------      *
*********************************************
```

appears on the console screen.

The program is started by placing the light pen over

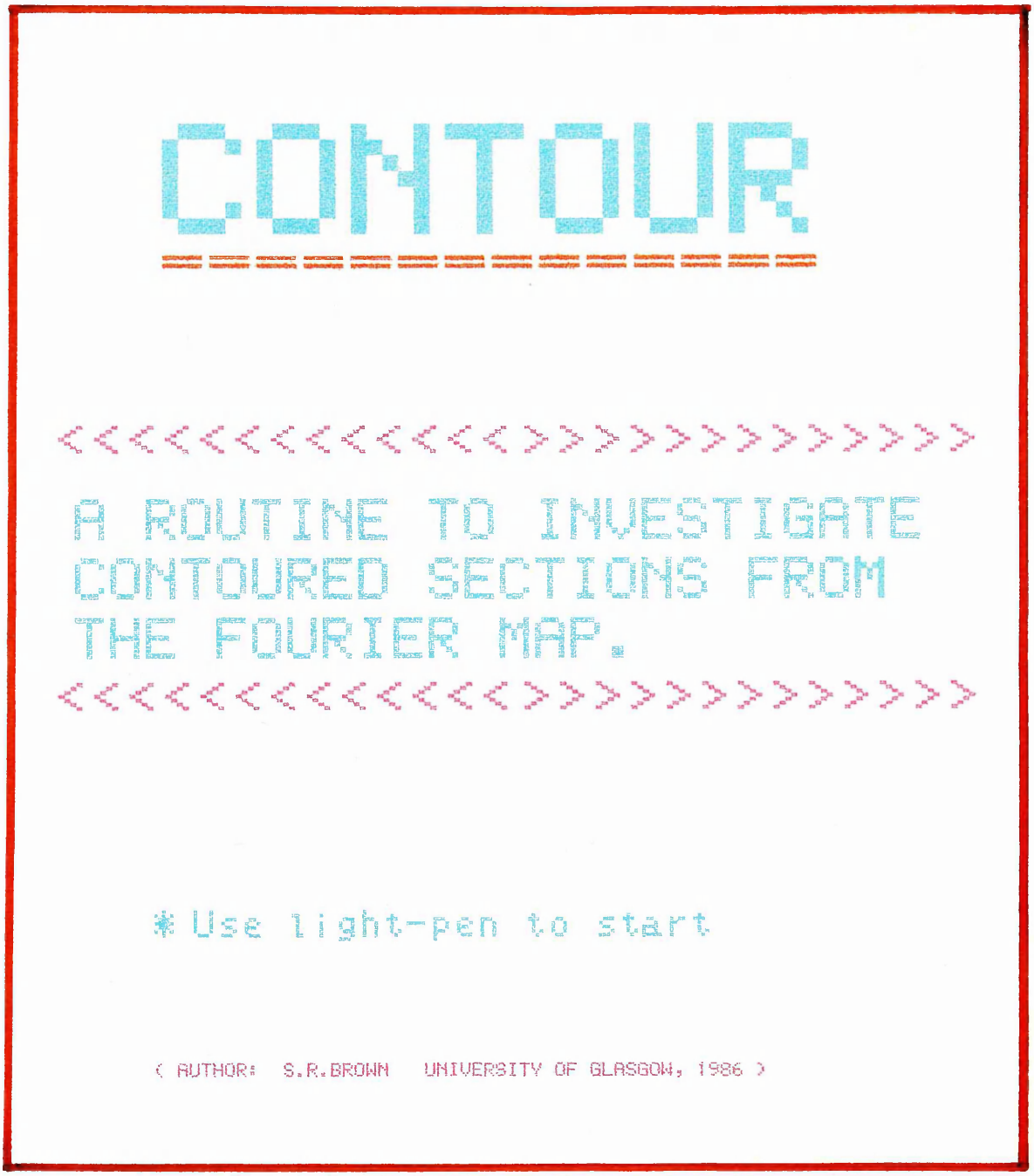


Figure 5.9 CONTOUR title page

the '*' on the title page. This causes the first contoured section of the E-map to be displayed (Figure 5.10). The program will initially display a section of the shortest axis. That means, for example, if the y-axis is the shortest then the first section to be displayed will be Y=0 (i.e. all map points with y=0). The section is orientated so that the longer of the two axes on display is always the horizontal.

The map is contoured in green and peaks found in the SEARCH part of MITHRIL are denoted by a '+'. The peaks in the input asymmetric unit are coloured white whereas the symmetry equivalents, if any, are purple. The peaks are numbered according to their height.

As can be seen from Figure 5.10 the user has a choice of how to proceed from this point. There is a main menu which contains the full list of options available however the program is structured so that a user can inspect succesive contoured layers of the E-map without having to refer to this menu each time.

The commands:

*** MENU * NEXT SECTION**

appear in the bottom right hand corner of the graphics screen whenever a contoured map section is displayed.

To obtain the menu of options available, the user places the light pen over the '*' to the left of 'MENU'. A full explanation of what each of the options does and instructions for their use is given in the next section.

Selecting 'NEXT SECTION' causes the next layer of the E-map to be displayed on the graphics screen. If the 'NEXT SECTION' option is chosen when the last section is displayed,

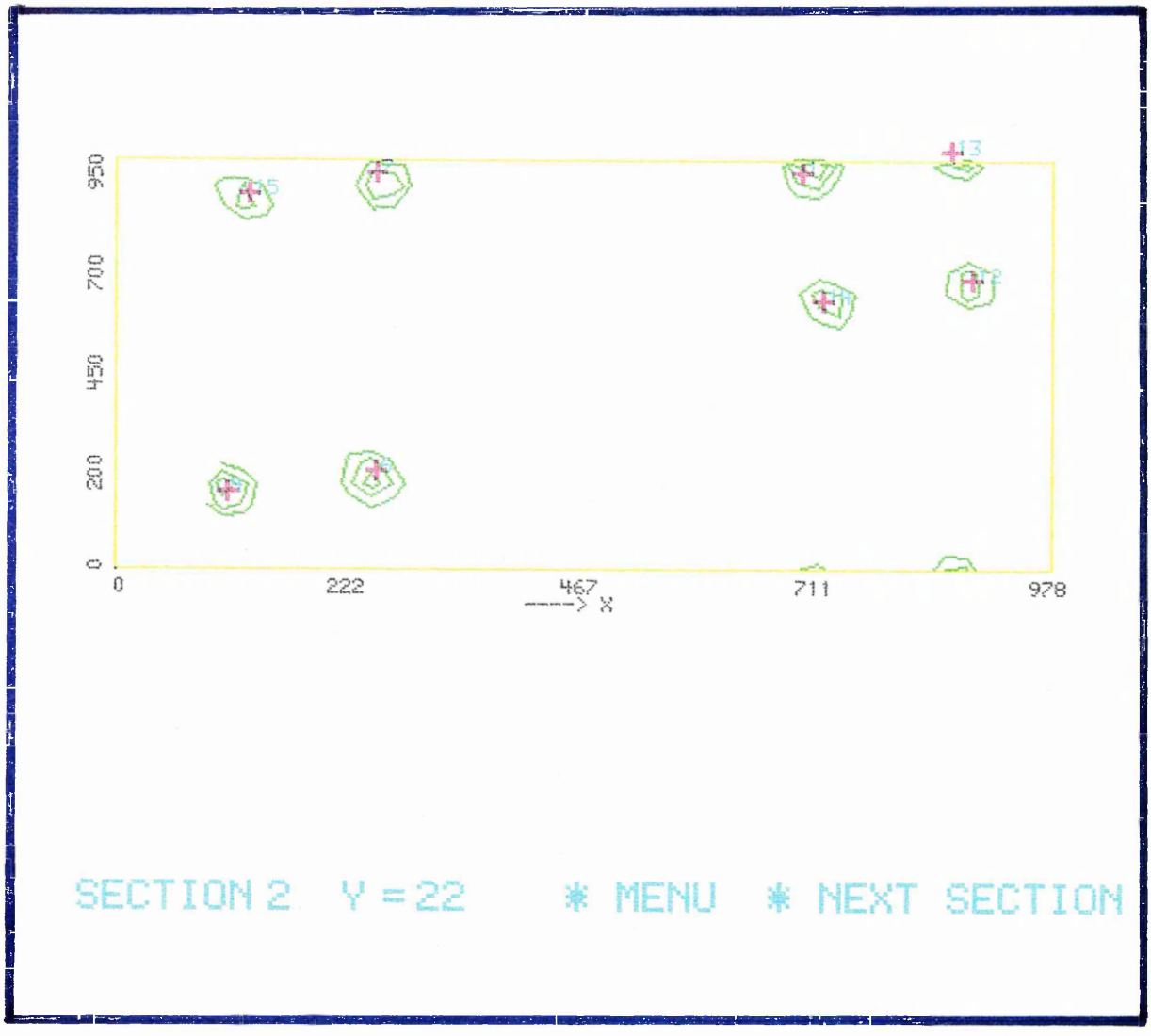


Figure 5.10 Contoured E-map section of DIAMANTANE-4-OL

the program will proceed to display the first section - this gives a realistic illustration of the repeating nature of the unit cell. Using this command the program will go on indefinitely displaying successive sections along the same axis.

5.8 OPTIONS AVAILABLE IN THE CONTOUR MODULE

The main menu in the CONTOUR module contains twelve options and is of the form:

THE FOLLOWING OPTIONS ARE NOW AVAILABLE:

*** EXIT FROM CONTOURING PART OF THE PROGRAM**
*** SUPERIMPOSE PREVIOUS PLANE**
*** SUPERIMPOSE NEXT PLANE**
*** CHOOSE SECTION TO BE DRAWN**
*** ZOOM IN ON A SECTION USING THE MOUSE**
*** CHANGE VALUE OF THE CONTOURS**
*** CHANGE LOWER LIMIT FOR CONTOURING**
*** DUMP CONTOUR MAP TO PRINTER**
*** CHOOSE DIRECTION TO BE VIEWED**
*** VIEW GENERAL PLANE**
*** CONTINUE**
*** QUIT - PROGRAM ABORT**

5.8.1 EXIT FROM CONTOURING PART OF THE PROGRAM

This option causes the program to leave CONTOUR and continue on to the next part of the program: the MITHGRAPH routines.

5.8.2 SUPERIMPOSE PREVIOUS PLANE

The section of electron density which comes immediately before the section on display is superimposed in

blue when this option is chosen. Any peaks contained in the previous section are also displayed. Figure 5.11 shows a contoured section with the previous plane superimposed in blue.

5.8.3 SUPERIMPOSE NEXT PLANE

This option is identical to the one just described with the exceptions that it is the next section rather than the previous one and this time the new section is contoured in red. Note that it is possible to display a section with both the next and previous planes superimposed.

Often, a part of a map which is of interest overlaps from one section on to the next. Using the two superimpose options just described, it is possible to gain a better understanding of parts of a map like this.

5.8.4 CHOOSE SECTION TO BE DRAWN

When this option is selected, the graphics screen goes blank and a message similar to:

There are 20 sections along the Z direction.
Which one do you wish to look at ?

appears on the console screen. Input of a valid section number causes the relevant contoured section to be displayed on the graphics screen. However, should an illegal section number be entered, the message:

Invalid section number. Try again

is displayed on the console.

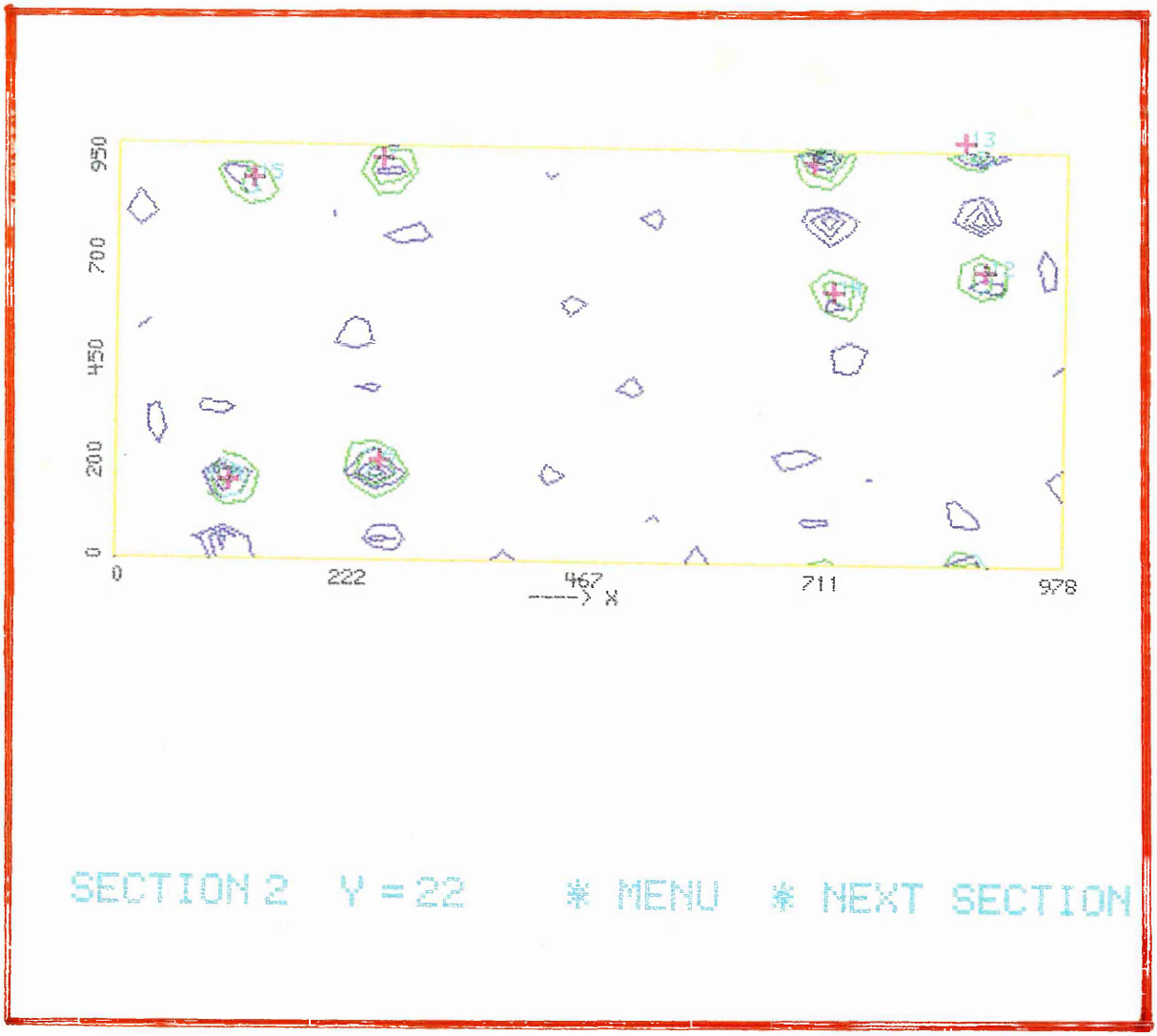


Figure 5.11 Contoured E-map section of DIAMANTANE-4-OL with previous plane superimposed

This command allows the user to go immediately from viewing one section to any other section along the same axis without any time being wasted in contouring the intermediate sections.

5.8.5 ZOOM IN ON A SECTION USING THE MOUSE

This option allows the user to define an area of the section on display to be enlarged. Use is again made of a 3-button optical mouse and mouse pad.

Selection of this option causes the instructions:

The switches on the mouse operate as follows:

Left - keep this button depressed to outline the area to be enlarged
Middle - press this button to change cursor colour (yellow/cyan)
Right - return to main program to see contouring of enlarged section

Place the mouse on the centre of the pad and press any of its keys

to appear on the console screen. Touching any one of the three mouse buttons causes a cursor (in the form of a '*') to be displayed in the middle of the graphics screen. The cursor is moved by moving the mouse on the pad.

The user defines the area to be enlarged by moving the cursor around the relevant portion on screen whilst simultaneously depressing the left hand button of the mouse. Pressing the right hand button causes the defined portion to

be displayed, in an enlarged form, on the graphics screen.

It may happen that the portion defined by the user contains too few grid points for it to be contoured in any sensible way. When this situation occurs, the message:

Too few grid points to zoom

will be displayed on the console screen and the program reverts to displaying the original contoured section on the graphics screen.

An important point to note is that zooming does not improve the resolution of a contoured section. The resolution is selected by the user via the GRID command in the MAPS menu. No new information is going to magically appear simply by magnifying. However, by using this option the map will become 'clearer' (in the sense of being less cluttered) and may become easier to interpret.

5.8.6 CHANGE VALUE OF THE CONTOURS

This option allows the user to alter the interval at which contours are drawn - this can often improve the appearance of a contoured section. If the map section is too crowded, raising the contouring level will de-congest it whereas lowering the level can add more detail to a sparse map. All of the map points are scaled to lie in the range -127 to +127 (so they can be stored efficiently by the computer) and the contouring level is initially set to 25.0. A message of the form:

Contouring level is at present 25.0. Please enter new value:

is displayed on the console screen when this option is

selected. The map section on display prior to this option being selected will be re-displayed with the new contouring interval after a suitable value is input. There are no restrictions on the contouring level but, in general, values in the range 20-30 give a sensible contour diagram.

5.8.7 CHANGE LOWER LIMIT FOR CONTOURING

Map sections are contoured at values which begin at the lower limit and then increase in regular intervals. The intervals can be changed by means of the previous command. Altering the lower limit for contouring can also help to clarify the map. A message of the type:

Lower limit is at present 25.0. Please enter new value:

is displayed on the console screen when this option is selected. The default value for the lower limit is 25.0. Raising the limit will certainly make a map less crowded but care must be taken as going too high will result in a loss of detail.

This option and the previous one are essential in order that the program can deal with both E-maps and Fourier maps in a sensible manner. These maps have intrinsic differences which mean that fixed values for the contouring level and lower limit for contouring would not give good contoured map sections in both cases. Fourier maps are different in appearance to E-maps: they generally have a smaller range of values and they also tend to have better defined peaks.

5.8.8 DUMP CONTOUR MAP TO PRINTER

This option enables the user to obtain a hard copy of any map section displayed. There is a choice of printers and a title can be given to the section. The command works in exactly the same manner as described for the DUMP command in MITHGRAPH (section 5.4.7). The only differences are that the menu is of a different format and that it appears on its own (Figure 5.12).

5.8.9 CHOOSE DIRECTION TO BE VIEWED

Initially, the program will display map sections along the shortest axis of the unit cell. Using this option it is possible to view sections along the other two axes also. A message of the type:

Sections are currently being viewed along the Y direction.
Type in new direction (X, Y or Z):

is shown on the console screen. The only permissible responses are 'X', 'Y' and 'Z'. Any other input causes the warning:

Invalid choice of input. Please try again.

to appear on the console screen. Input of a direction will cause the first section in that direction to be displayed on the graphics screen.

5.8.10 VIEW GENERAL PLANE

Using the previous option it has been demonstrated how sections can be viewed along any of the three axes of the unit

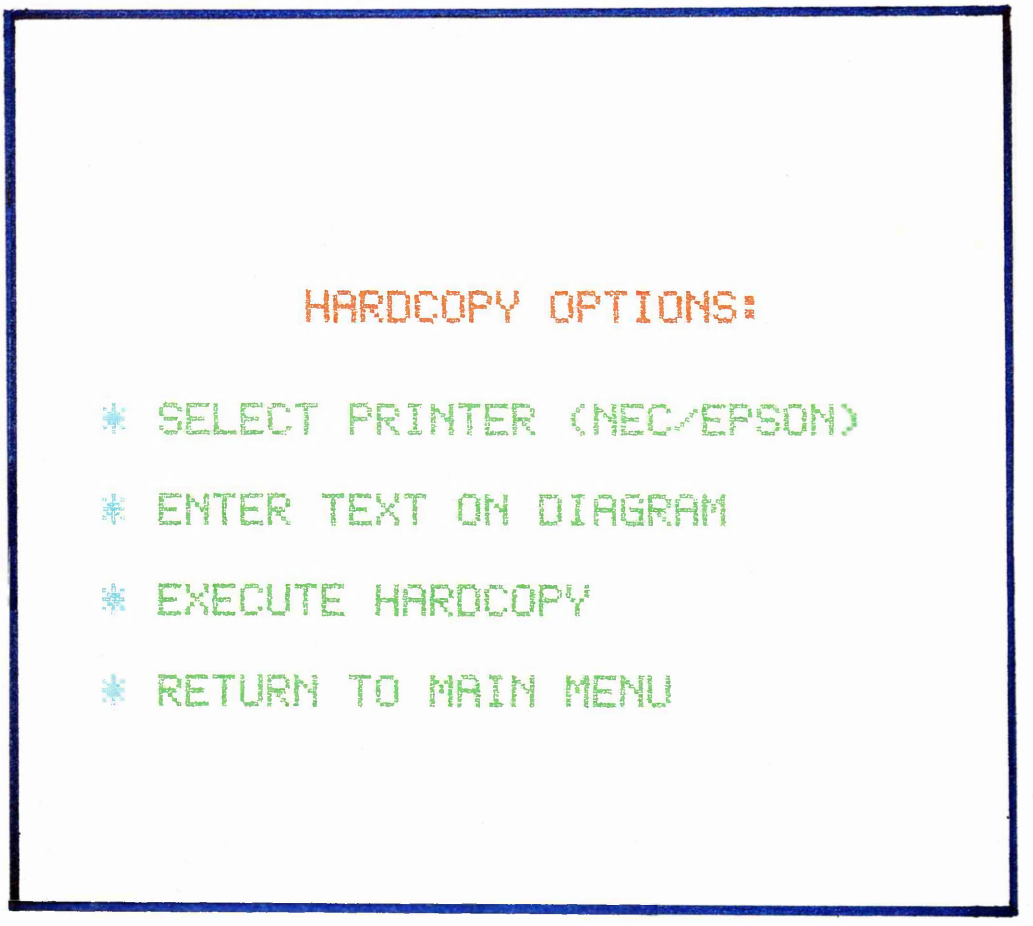


Figure 5.12 The hardcopy menu of CONTOUR

cell. However, it is only very rarely that a part of a map which is of interest will lie parallel to a face of the unit cell. More often it will cut across sections at an angle.

This option gets round the problem, to a certain extent, by allowing the user to define a plane in terms of its Miller Indices and then contouring it. In general, very few map points will actually lie exactly on planes of this type and so the program is forced to estimate points to make up the plane by interpolation of the nearest map points in each case.

When this option is chosen the menu is removed from the graphics screen and the message:

Input the Miller Indices of the plane to be viewed:

appears on the console. The three indices, separated by commas, are input via the keyboard before the plane is displayed on the graphics screen. The display is rather different to that of a normal section. As can be seen from Figure 5.13, the title 'GENERAL PLANE' is written in the top left part of the screen and the indices, rather than the section number, are located in the bottom left hand corner of the screen.

Like the zoomed section, it is not possible to superimpose another contoured section on the general plane display. If either of the superimpose options is selected while the program is in the general plane mode, the message:

Superimpose is not available for the current display.
Please select another option.

appears on the console screen.

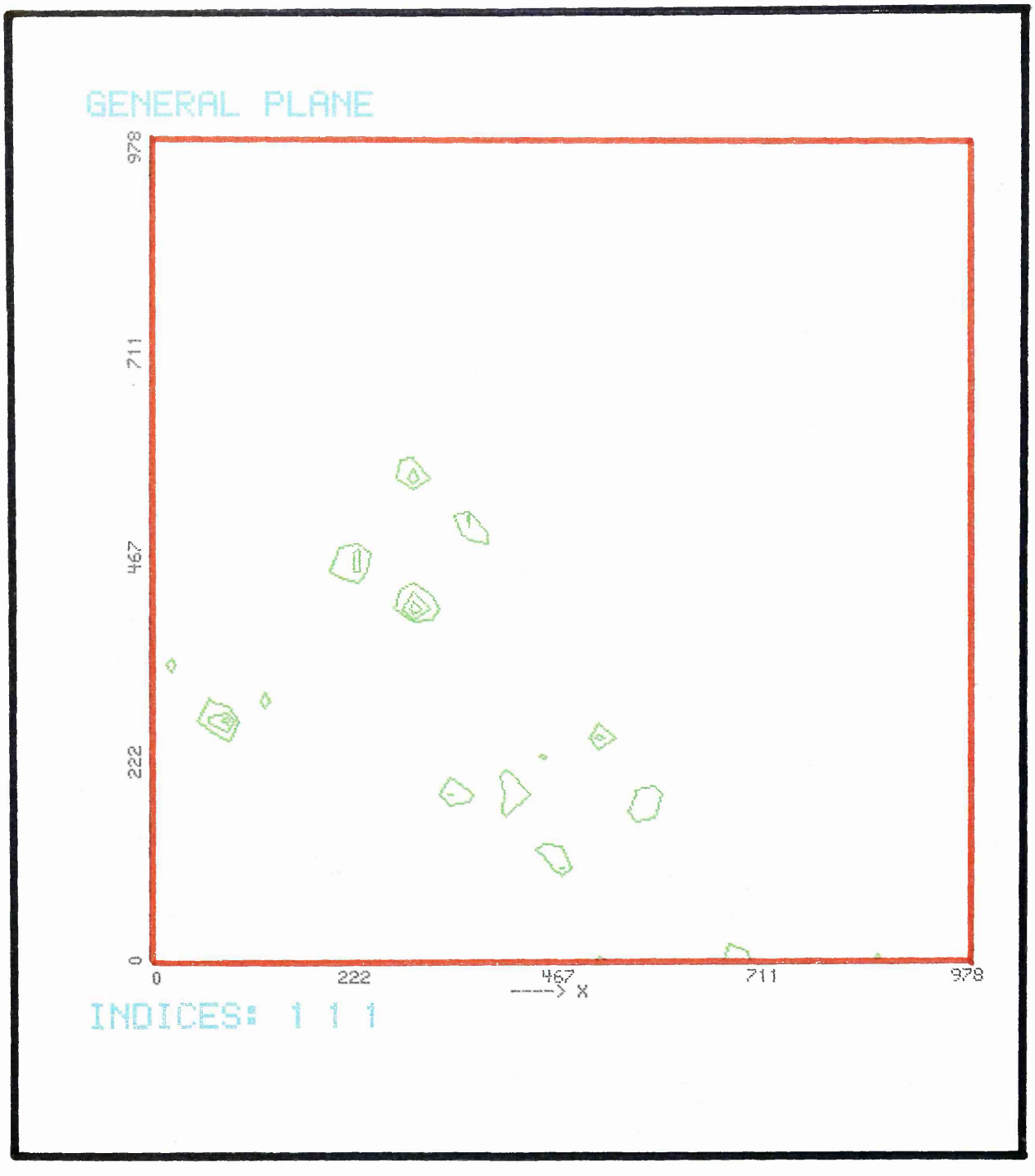


Figure 5.13 Contoured (1 1 1) plane of DIAMANTANE-4-OL

5.8.11 CONTINUE

This option will cause the program to cease displaying the main menu on the graphics screen; instead it will revert to showing the map section on display before 'MENU' was called. A useful option if the menu was entered by mistake or if the user does not wish to use any of the options.

5.8.12 QUIT - PROGRAM ABORT

Selection of this command causes the program to terminate immediately. The message 'END OF PROGRAM' is displayed on the graphics screen to indicate the program has finished.

5.9 TWO EXAMPLES OF USING CONTOUR: PGE2 AND WINTER2

As was noted in the previous chapter both PGE2 and WINTER2 proved to be among the most difficult of the database structures to solve. The use of colour interactive computer graphics, particularly the contouring routine, was an important factor in the determination of both structures.

5.9.1 PGE2

Several E-maps were investigated for PGE2 using the CONTOUR routine. It was noticed that the E-maps generated for unsuccessful solutions were rather difffuse and noisy with a fairly random scattering of peaks. In marked contrast, the E-

map generated for what turned out to be the succesful solution was very clean with sharply defined peaks and an ordered, rather than a random, peak distribution. Many of the peaks were found to lie in the vicinity of the section Y=0. Figures 5.14, 5.15 and 5.16 show the successive Y-sections 32, 0 and 1 respectively.

The peak height distribution for this solution was quite poor with one peak very much bigger than the rest. In fact such a peak distribution is often an indication of an incorrect solution. In this case, however, direct observation of the three dimensional E-map provided enough evidence to make it worthwhile pursuing the solution. A 15 atom fragment was subsequently found which was used in Fourier recycling to locate most of the remaining atoms in the structure (some atoms could not be found due to large thermal vibrations).

It had been hoped to observe the hairpin structure that is known to be a characteristic of PGE2. However, despite viewing E-map sections along each of the three axes of the unit cell it was not possible to do so. It must be concluded, therefore, that the hairpin cannot lie parallel to any of the faces of the unit cell.

5.9.2 WINTER2

As was discussed in section 4.3.25 the interpretation of E-maps was found to be all important in the determination of WINTER2. Several promising phase sets had to be discarded because it was not possible to match the molecular fragments produced by automatic E-map interpretation with the known structure. As with PGE2 above, direct observation of the three dimensional E-map using the CONTOUR module proved to be

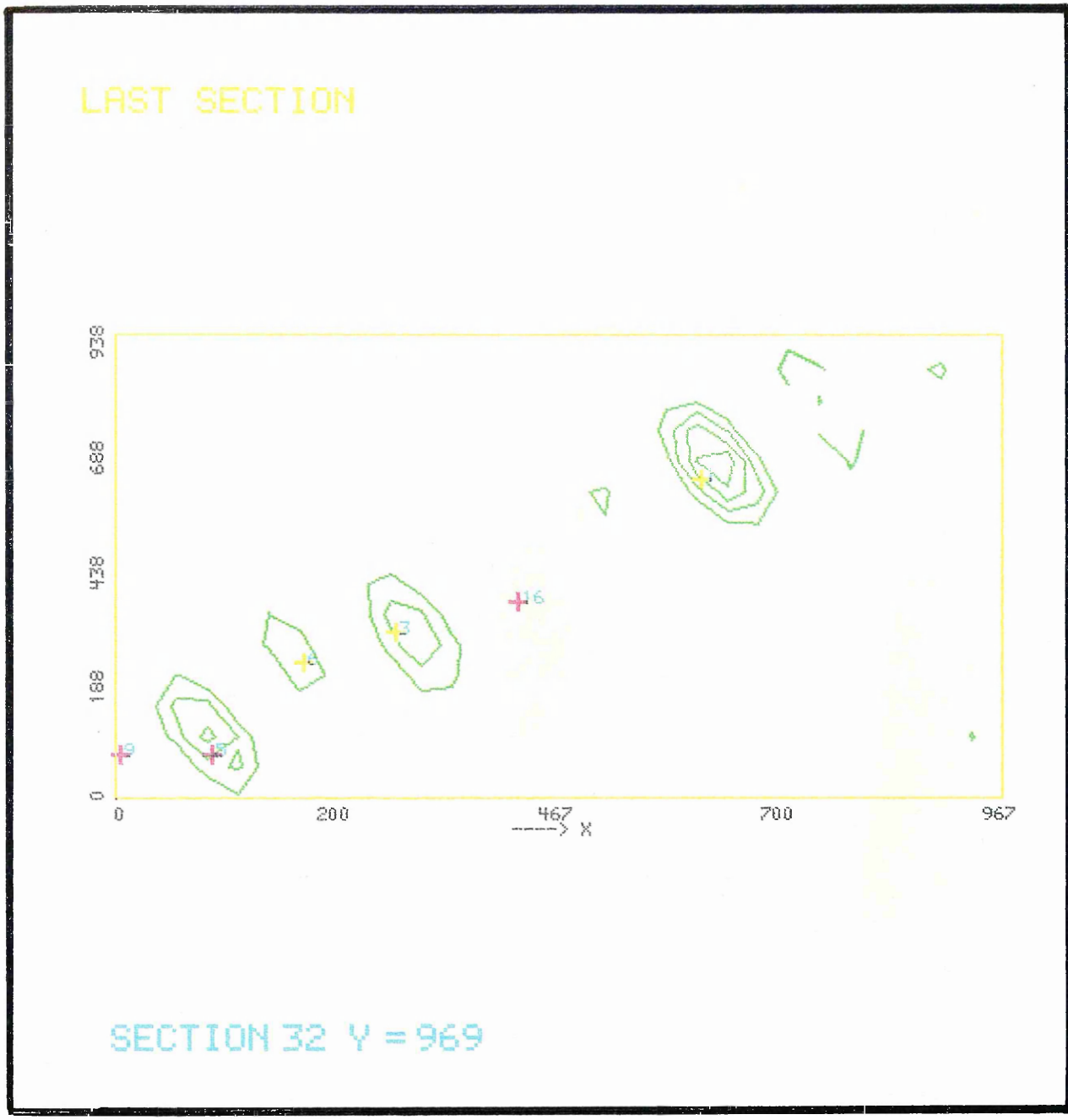


Figure 5.14 Y-section number 32 of PGE2

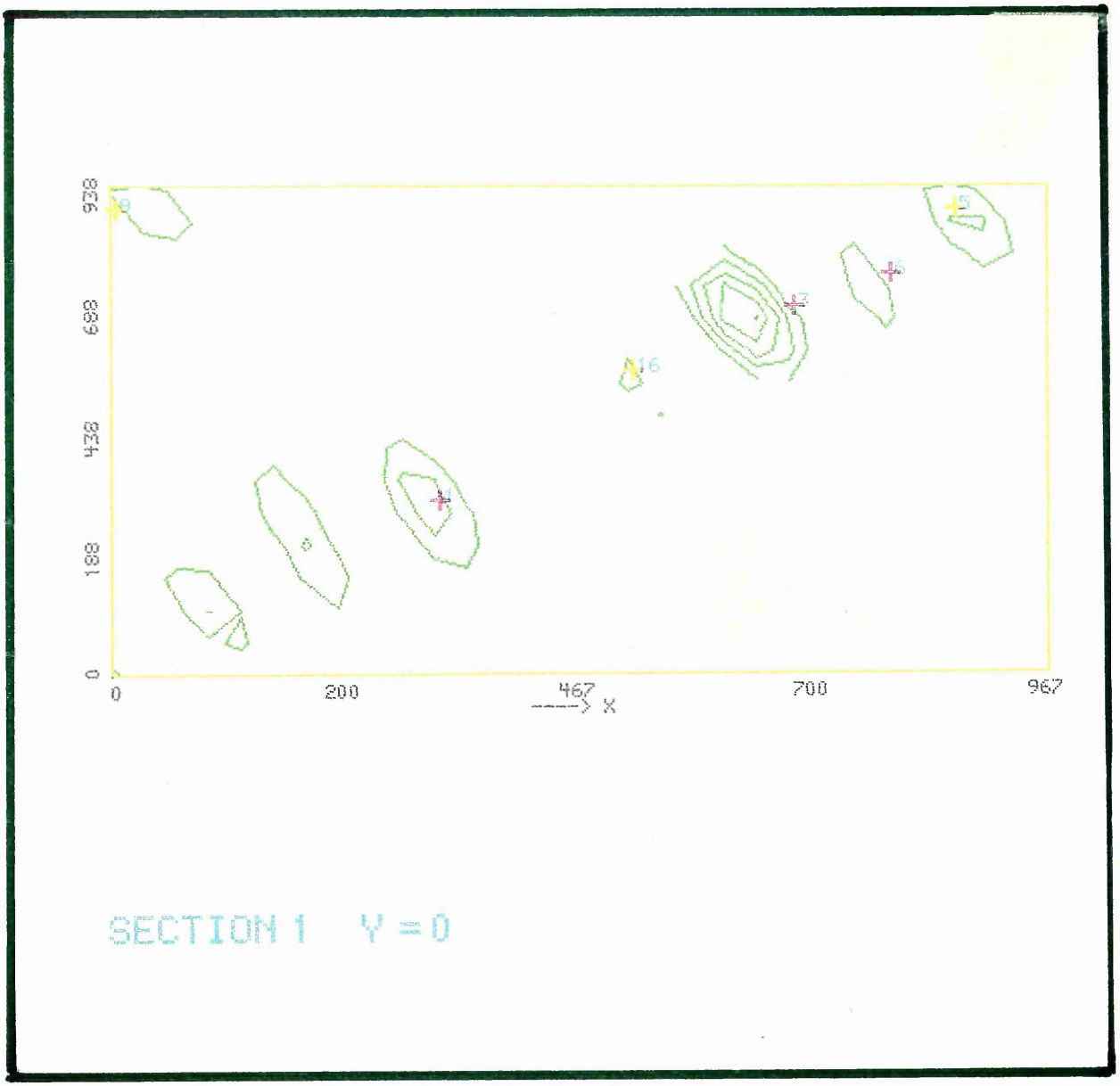


Figure 5.15 Y-section number 1 of PGE2

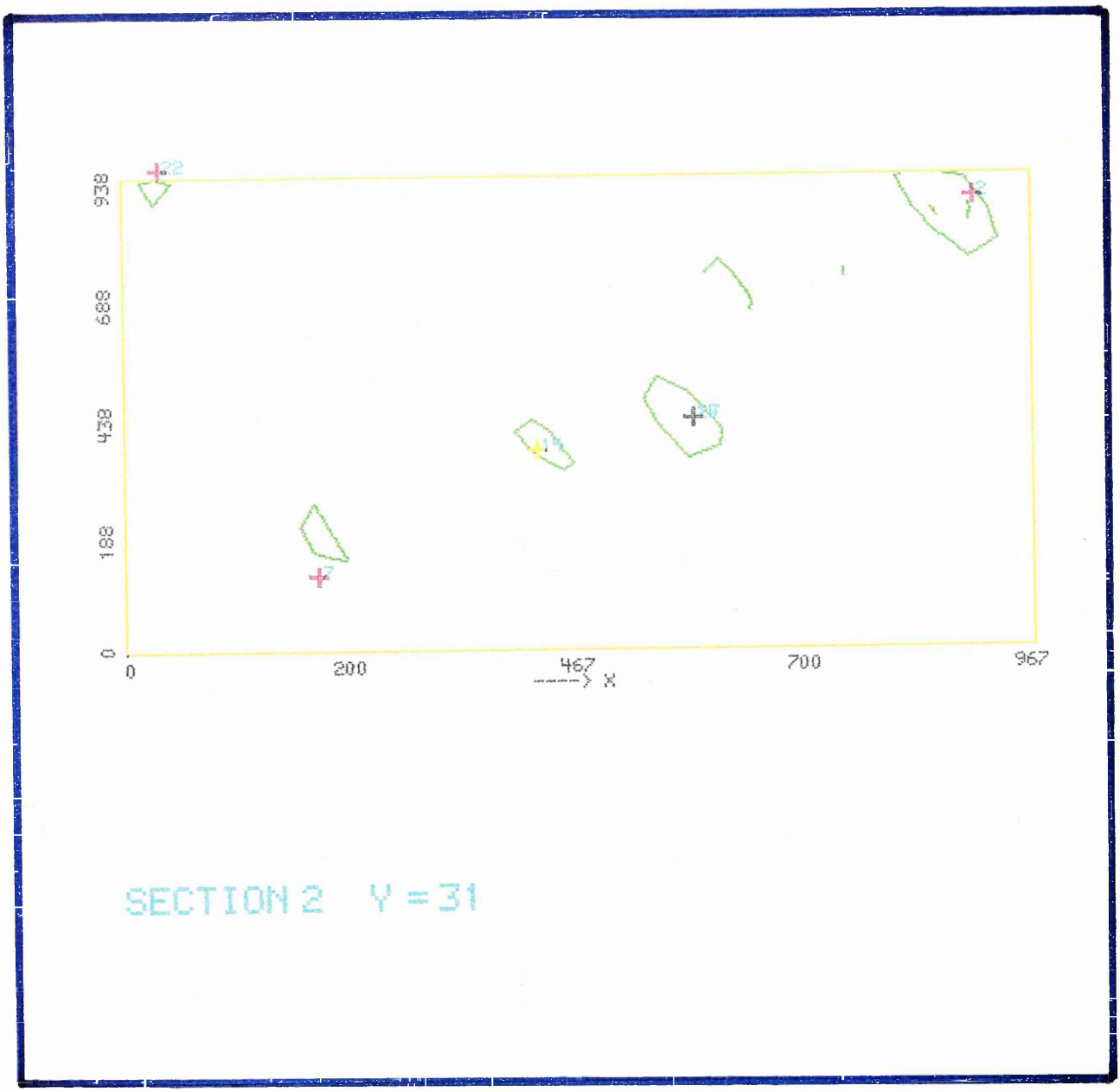


Figure 5.16 Y-section number 2 of PGE2

an invaluable tool in identifying the correct solution. E-maps generated for WINTER2 were found to be more complex than those of PGE2 mainly because it had a far greater number of atoms in the unit cell.

Once again the successful E-map was relatively clean with very few spurious peaks whereas unsuccessful maps were less well defined and had an abundance of smaller peaks. The Z-sections 5 and 6 shown in Figures 5.17 and 5.18 respectively illustrate the sharpness of the successful E-map. In marked contrast Figure 5.19 illustrates a typical section from an unsuccessful E-map. In the successful E-map it was observed that none of the X, Y, or Z-sections contained more than a handful of peaks. This is because WINTER2, typical of polypeptide structures, has a long spiral backbone which does not run parallel to any of the unit cell axes.

5.10 SUBMIT - A MITHRIL SUBSET WITH IMPROVED GRAPHICS ROUTINES

SUBMIT is a program which has been written in order to utilise the MITHGRAPH and CONTOUR modules described previously in this chapter. It is based on a subset of the subroutines in the MITHRIL program - it contains the RECYLE, MAPS and peak search modules - together with the graphics modules. The program can be used for three distinct purposes:

(a) To perform Fourier recycling of molecular fragments found by MITHRIL (or by any other means). This involves running the RECYCLE module of MITHRIL, computing an R-factor and calculating and intepreting a weighted Fourier map. Unlike MITHRIL, the code here is re-enterable and it is possible to do continuous Fourier

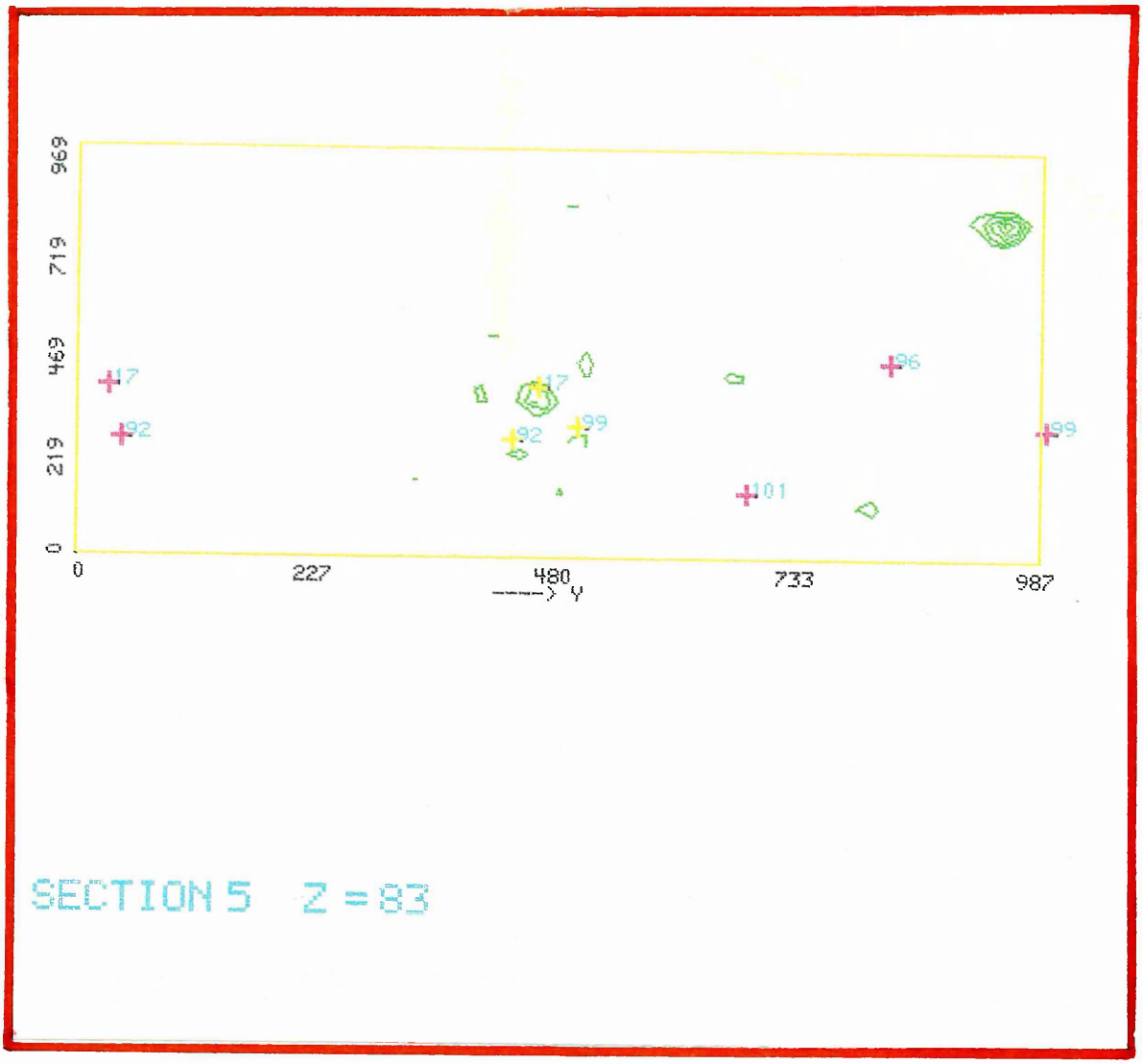


Figure 5.17 Z-section number 5 of WINTER2

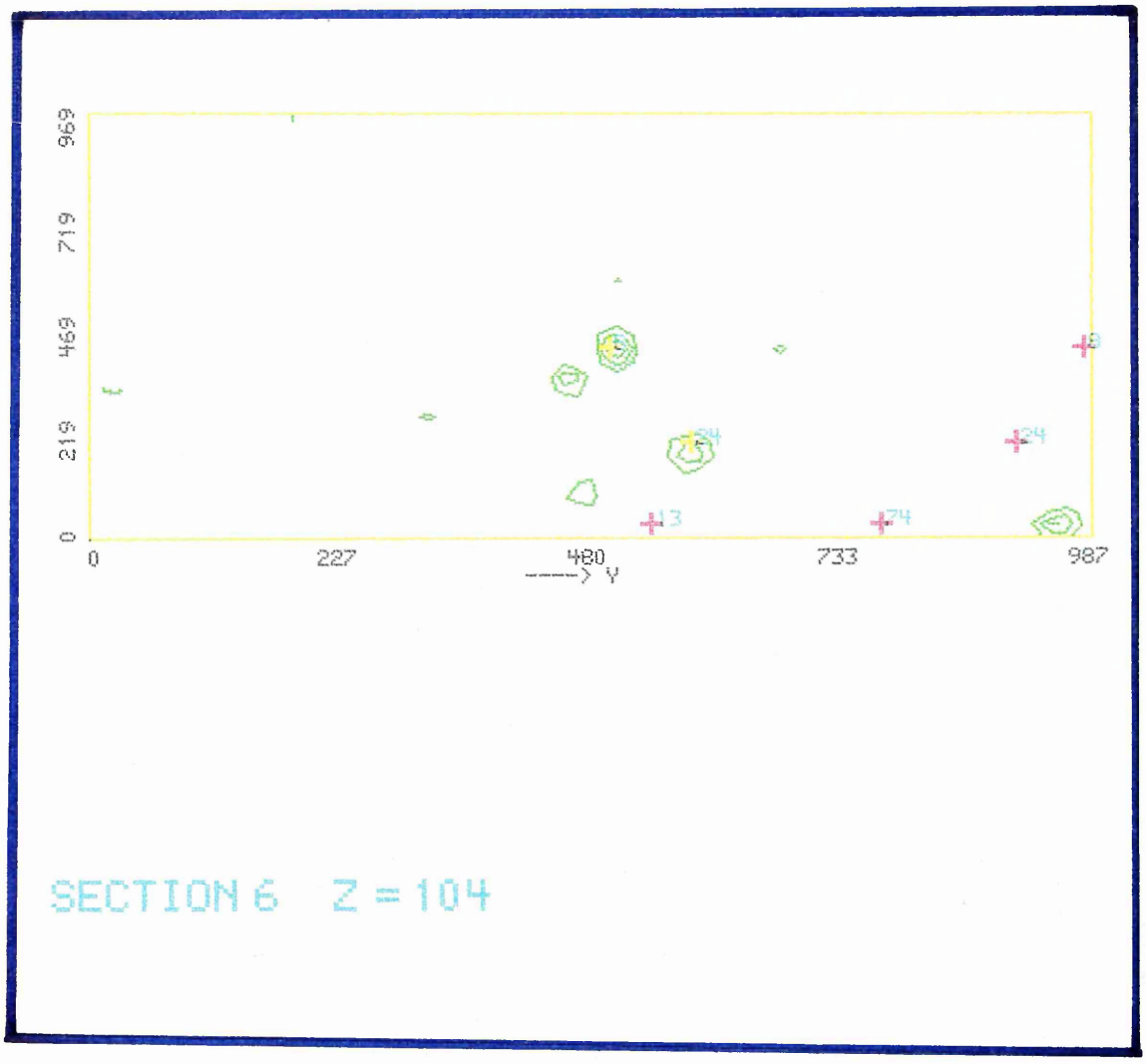


Figure 5.18 Z-section number 6 of WINTER2

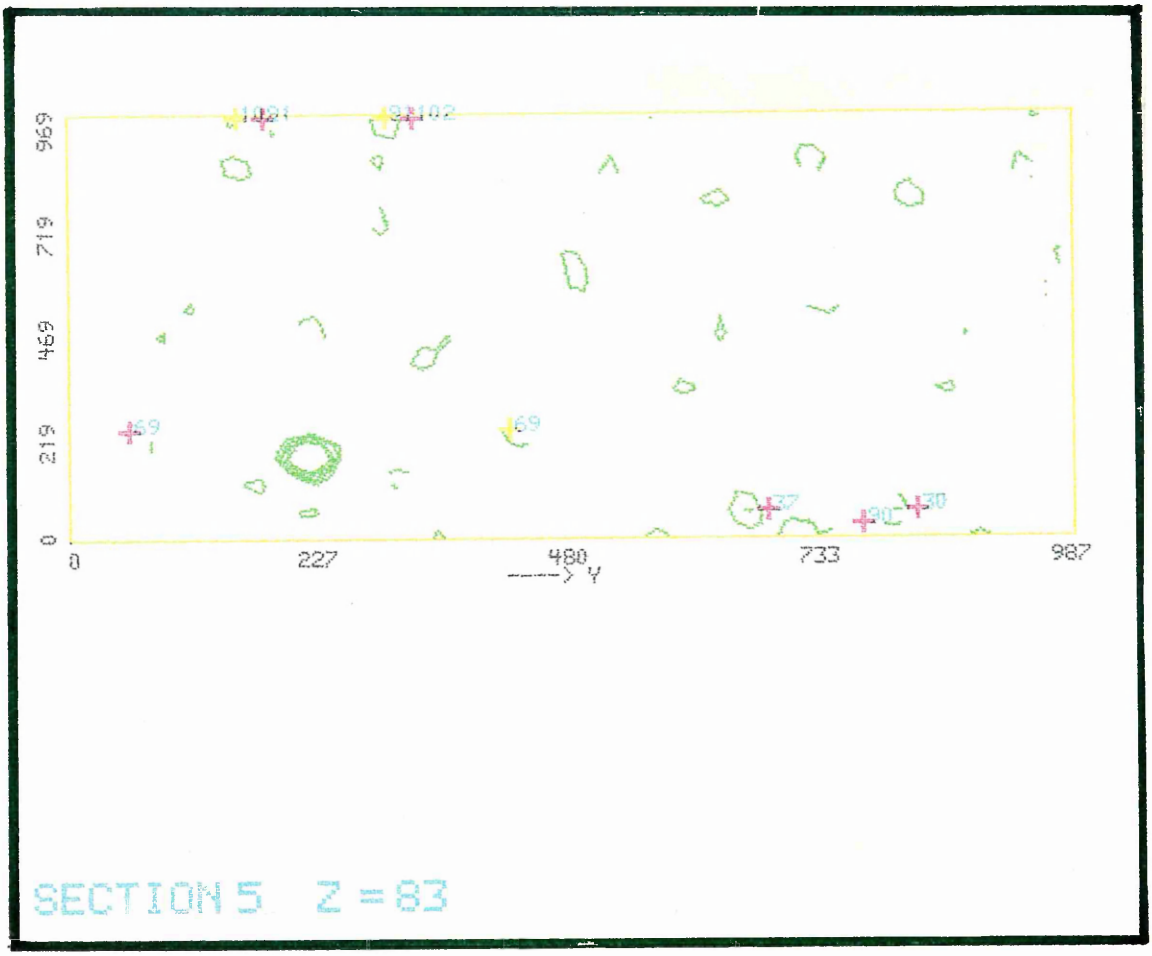


Figure 5.19 Z-section from an unsuccessful E-map calculated for WINTER2

recycling without having to leave the program.

(b) To calculate E-maps using phase sets computed previously by MITHRIL.

(c) To inspect E-maps produced by MITHRIL.

The idea is to use other computers to do all the heavy computational work involved in MITHRIL before storing the results on file and transferring them to the graphics workstation for use in SUBMIT. At Glasgow MITHRIL is run on either the MASSCOMP 5400 or SEL 32/22 computers before the appropriate file is transferred to the colour graphics workstation for use with SUBMIT. A flowchart of the program is given in Figure 5.20. A point to note is that SUBMIT has no output file which can be printed, however, it is possible to obtain hard copies of peak interpretations and contoured sections of electron density or Fourier maps by using the screen dump facilities in MITHGRAPH and CONTOUR respectively.

5.11 USING SUBMIT

SUBMIT is initialised by simply typing "SUB", without any parameters. A title page is immediately displayed on the graphics screen (Figure 5.21) and the heading:

```
**********************************************************
*                    WELCOME TO THE                      *
*         STRIDE CRYSTALLOGRAPHIC FOURIER PACKAGE        *
*              (Now with Fourier recycling)              *
**********************************************************
```

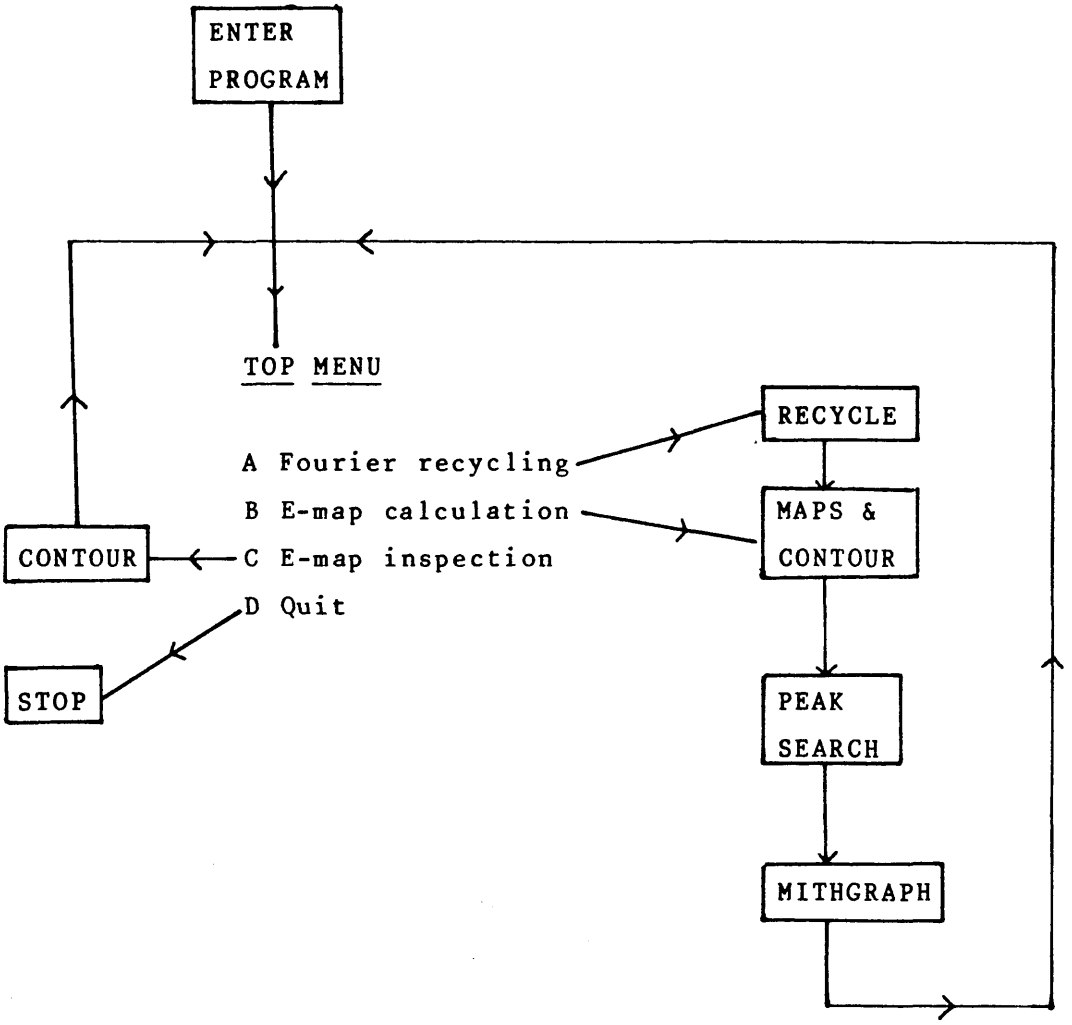


Figure 5.20: Flowchart of the SUBMIT program

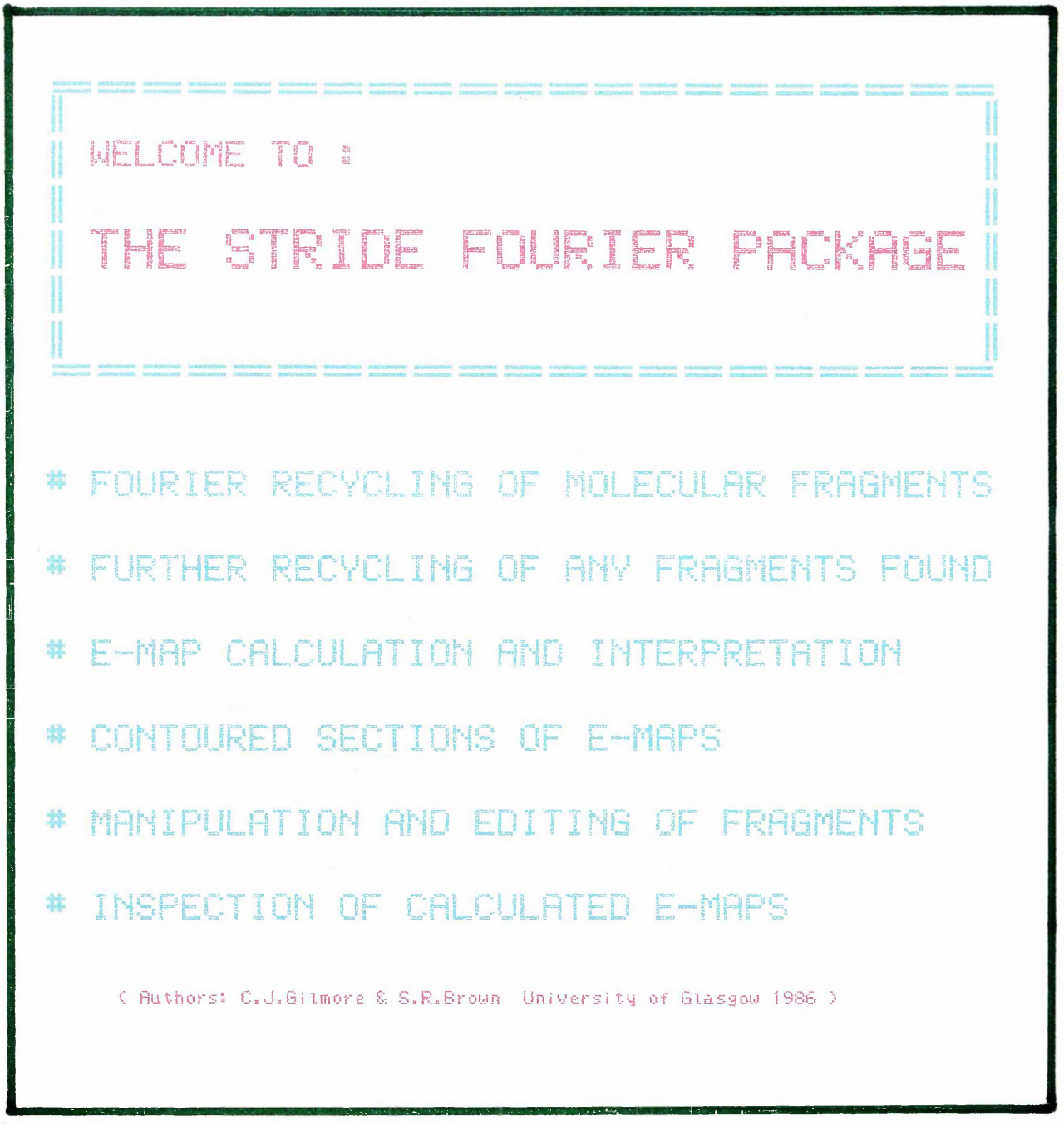


Figure 5.21 Title page of the SUBMIT program

appears on the console screen together with the top menu of the program which is of the form:

Choose from one of the following options:

A: Fourier recycling
B: Calculate E-maps
C: Inspect E-maps
D: Quit

Which option ?

If the user should enter anything other than an 'A', 'B', 'C', 'D', or one of their lower case equivalents then the warning:

Unrecognised option - try again

is shown on the console screen before the top menu is displayed again. Entry of an 'A', 'B' or 'C' will cause the program to go into Fourier recycling, calculation of E-map or inspection of E-map modes respectively - instructions for their use are given in the following sections. Selection of 'D', however, causes the program to terminate immediately with an 'END OF PROGRAM' message displayed on the graphics screen.

5.11.1 FOURIER RECYCLING

This option is for use when a fragment of a structure has been found and the whole structure is required.

When an 'A' is entered in response to the top menu above, the message:

Fourier recycling chosen

is displayed on the console screen followed by the message:

Enter the name of the peaks file -

If a file name is entered at this point then it will be used to store the coordinates of the peaks from the weighted Fourier map. If, on the other hand, a carriage return is entered then a scratch file will be used for the peaks: this file will be erased automatically when the program ends. The program requires that a data file containing the intensities of the reflections collected for the structure is used. Hence the request:

Enter the name of the reflection file -

is the next message to appear on the console screen. Should a non-existent file name be entered in response, then the following prompt is displayed:

File does not exist. Do you wish to try again ? (y/n) -

If an 'n' is entered in reply to this question then the program terminates immediately with an 'END OF PROGRAM' banner displayed on the graphics screen. Entry of a 'y' causes the question to be re-displayed. Once a valid file name has been entered the program displays:

Enter the name of the model file -

Again the program will wait until it has received a valid file name. The format of the model file for SUBMIT is the same as that used in MITHRIL.

After all the files have been input satifactorily, the full RECYCLE menu from MITHRIL is displayed on the console screen. The option all work in exactly the same way as for MITHRIL. At the end of the recycle module, statistics are shown on the console screen as regards the numbers of maximum duplicates, systematic absences and excluded reflections together with an R-factor. The R-factor has a value of approximately 55 percent for a random distribution of atoms within the unit cell. Values lower than this are often an indication that the fragment is correct and that things are proceeding well. The message:

****** **** NORMAL COMPLETE **** ******

is displayed on the console screen to show the RECYCLE module has been completed. This is quickly followed by:

****** **** MAPS program called **** ******

Notice that, unlike MITHRIL, in SUBMIT the MAPS module is called automatically during the Fourier recycling procedure. The MAPS menu that is then displayed is the same as that in MITHRIL with a few notable exceptions. The VDU and LIST commands are redundant on the graphics workstation and have been removed from the menu while CONT has been added to it. Notice also that the default map resolution has been changed from 0.333 Å to 0.400 Å - this means that the map will be of lower resolution but that it will be calculated more quickly.

After the Fourier calculation has been completed, a message of the type:

Fourier points range from -414.83 to 1362.45
Do you wish to proceed with peak search ?

The only valid responses to this question are 'y' and 'n'. Input of an 'n' will cause the program to abandon the current map and go back to displaying the top menu where it will await further instruction. Entry of a 'y' causes the peak interpretation to proceed exactly as in MITHRIL. Whereas MITHRIL finishes after outputting the number of clusters it has found, SUBMIT continues on from that point to enter the CONTOUR (if it was selected) and MITHGRAPH routines. Instructions for the use of these modules were given in the previous two chapters. Note that there is a very useful option in MITHGRAPH for use in conjunction with Fourier recycling. It is called UPDATE and is located amongst the EDIT functions of MITHGRAPH. This option enables the user to store the fragment on display at that time as a group for use in Fourier recycling - the coordinates of the previous group of atoms are overwritten. After MITHGRAPH is exited the program reverts to displaying the top menu. This means it is perfectly possible to recycle a group of atoms, find some more atoms, add them to the group and undertake further Fourier recycling without having to leave the program.

5.11.2 CALCULATE E-MAPS

This option enables the user to calculate E-maps using phase sets computed by MITHRIL on another computer beforehand. MITHRIL contains an option to create a permanent file to store the phase sets. Unfortunately this is a binary file and as such cannot be easily transferred from one computer to another. To get round this problem, a program, called CONVERT, has been written which allows the user to store desirable phase sets on a file in ASCII format. It is then a simple task to transfer this file to the graphics workstation for use in SUBMIT. Normally the file will contain only the

best 2 or 3 phase sets, but in difficult cases it can often prove fruitful to investigate many more than this. (A listing of the source code of CONVERT is given in APPENDIX D).

To calculate E-maps using SUBMIT, select option 'B' from the top menu. A message of the form:

E-map option chosen

will appear on the console screen to confirm the selection. It is accompanied by the prompt:

Enter the name of the peaks file -

As with the Fourier recycling option, if a file name is given then that file is used to store the peaks from the E-map, otherwise a scratch file is used. The program then looks for the file containing the phase sets. It does so by printing:

Enter the name of the phase file -

on the console. If an invalid file name is entered in response then the message:

File does not exist. Do you wish to try again ? (y/n) -

appears. Typing 'n' will cause the program to terminate immediately whereas a 'y' will cause the file prompt to be repeated. After a valid file name has been received, the caption:

****** **** MAPS program called **** ******

appears on the console to indicate that the MAPS module has been entered. Rather than immediately displaying the MAPS

menu, as in the Fourier recycling option, in this instance a smaller menu is produced first. It is of the form:

Select E-map option:

0 for best map
n for set #n
R for review phase sets
Q for quit

COMMAND:

Phase sets are ranked in SUBMIT according to their combined figure of merit values. Selecting '0' from the above menu will cause the best phase set to be used in the E-map calculation. After a phase set has been used for an E-map calculation it is assigned a flag that prevents it from being chosen as the best set again. It may happen that all the sets on the file have already been used. If this is the case when '0' is selected then the message:

All phase sets have now been investigated -
there is now no best set

appears on the console followed by the menu above. Phase set number n can be used for the E-map calculation by simply entering the number n in response to the menu. The 'R' option allows the user to review the phase sets on the file. It prints the phase set number, its figures of merit as well as an indication of whether it has already been used, for each of the sets on the file. The menu is again displayed after this option has been run. The quit option terminates the program immediately.

After a valid phase set has been entered for the E-map calculation, the MAPS menu is displayed and the program

proceeds from this point in exactly the same way as was described for Fourier recycling in section 5.11.1. The only difference is that rather than displaying the top menu when MITHGRAPH is exited, this time the program re-displays the E-map option menu.

5.11.3 INSPECT E-MAP

This option enables the user to inspect contoured sections of interesting E-maps that have already been calculated using MITHRIL. There is an option in MITHRIL that makes it possible to create a permanent binary file to store all the points in the E-map. As with the previous option, this file is transferred into ASCII format using convert before being passed to the graphics workstation for use in SUBMIT.

To inspect an E-map the user selects option 'C' from the top menu. The message:

Inspect E-map option chosen

is displayed on the console to confirm the selection. It is accompanied by the request:

Enter the name of the maps file -

As with the other options, if an incorrect file name is given, the program prints a warning to that effect as well as enquiring whether the user wishes to continue. After a valid file name has been received there is a delay as the map points are read from the data file - this file can be very large - before the title page of the CONTOUR module is

displayed on the graphics screen. Instructions for use of the CONTOUR module were given in a previous section. When CONTOUR is exited, the top menu is once again displayed on the console awaiting further input.

5.12 DISCUSSION

The graphics routines MITHGRAPH and CONTOUR, as presented in the SUBMIT program, have proved to be very useful in the interpretation od E-maps. Using MITHGRAPH, no time is wasted in drawing out peak interpretations by hand and, in most cases, it can be seen at a glance whether the structure, or a fragment of it, has been found. For more complex structures the facilities to manipulate, especially the rotate option, can be invaluable in their determination. The problem of not being able to recognise a correct solution should occur much less frequently using the graphics routines.

CONTOUR offers some significant advantages over the previously automatic E-map intepretation. One instance is if the map is disordered with a random scattering of peaks, it will probably not lead to the structure being found. It is sometimes possible to locate missing atoms using this routine - for example, in a structure known to have a 6-membered ring only 5 of the atoms may appear in the peak list while the other peak is clearly visible but too small to make the peak list. Use of the CONTOUR module has highlighted the differences between E-maps and weighted Fourier maps: the latter have a smaller range of values and they also tend to be sharper with better defined peaks.

Although the SUBMIT program and the two graphics

modules work and are undoubtedly of great value in their present form, they are still at a fairly early stage in development. As the program is used on more structures, more information will become available on modifications and new routines needed to improve it. An example of one such modification is that it is hoped to soon provide a function which enables the user to employ an optical mouse to pick out a group of peaks from a set of contoured sections and that these peaks will comprise the complete structure.

It would obviously be an improvement, from the point of view of convenience as well as flexibility, if the graphics routines could be integrated into a full direct methods program rather than a cut-down version. It was not possible to do this here because of the lack of processing power on the graphics workstation used but with graphics facilities becoming common on fairly powerful computers this situation should soon be rectified.

5.13 PROGRAMMING CONSIDERATIONS

Due to restrictions imposed on file size by the editor on the Glasgow system, it was found necessary to divide the source code into three: it is contained in the files NORMAL.FOR, MAPS.FOR and SEARCH.FOR. The full source for SUBMIT is listed in APPENDIX B.

The Fortran code was written for the colour graphics workstation at Glasgow and is, as a result, very machine specific. It uses the GRAPHDRVR library of graphics routines. A full list of these routines and their functions is given in APPENDIX C.

References

1. J.Butland. SIMPLEPLOT User's Handbook, Report No. 253, University of Bradford.

2. C.K.Johnson (1976). ORTEP II, Report ORNL-5138, Oak Ridge National Laboratory, Oak Ridge, Tennessee

3. S.Motherwell & W.Clegg (1978). PLUTO, Program for plotting molecular and crystal structures, Cambridge University

4. D.R.Henry (1983). Comput. Chem., 7(3), 119

5. T.Liljefors (1983). J. Mol. Graph., 1(4), 111-117

6. B.M.Brooks, R.E.Bruccoleri, B.D.Olafson, D.J.States, S.Swaminathan & M.Karplus (1983). J. Comput. Chem., 4(2), 187-217

7. T.A.Jones (1985). Methods Enzymol., 115, 157-171

8. C.Nayl, L.Lazzerini & M.Pierrot (1987). J. Appl. Cryst., 20, 269-270

9. G.A.Sim (1959). Acta Cryst. 12, 813-815

10. G.A.Sim (1960). Acta Cryst. 13, 511-512

11. M.Lindley, J.K.Tyler & D.N.J.White (1984). J.Mol.Graph, 2, 56

12. J.D.Foley & A. Van Dam (1982). in "Fundamentals of Interactive Computer Graphics", Addison Wesley, pp 245-266

13. R.Diamond (1982). in Computational Crystallography, Ed. D.Sayre, Oxford University Press, pp 266-272

NEW APPROACHES TO DIRECT METHODS IN X-RAY CRYSTALLOGRAPHY

APPENDIX

Stephen Robert Brown

CONTENTS

APPENDIX A

Program listing of the LSAM module

APPENDIX A - PROGRAM LISTING OF THE LSAM MODULE

```
      SUBROUTINE LSAM1
C****
C****   L        SSSSS    AAA     M     M
C****   L        S       A   A    MM   MM
C****   L        SSSSS   AAAAA    M MM M
C****   L            S   A   A    M     M
C****   LLLLL    SSSSS   A   A    M     M
C****
C**** ORIGINAL PROGRAM BY G.GERMAIN (UNIVERSITE DE LOUVAINE) ,P.MAIN
C**** & M.M.WOOLFSON (BOTH UNIVERSITY OF YORK)  1969
C****
C**** DEVELOPED AS A MODULE FOR 'MITHRIL' BY S.R.BROWN
C**** UNIVERSITY OF GLASGOW 1986
C****
      COMMON/IO/NSPEC,NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(156),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEM
      COMMON/BLK1/IPH1(30000),EEE(30000),IZ(800),LO(800,11),
     1 IPH2(30000),TS(3,24),IS(2,3,24),CX(6),PAD1(178)
      COMMON/BLK2/LIM(901),IZRO,PAD2(698)
      COMMON/BLK3/STABLE(450),NQ1(500),NQ2(500),AQ(500),NQTOT,
     1 NQ4,NQ5,PAD3(629)
      COMMON/BLK4/NUM,NUM2,NYMBOL,PROLO,NT,NSRT,PROB,KCUT,MITOR,NSYMB,
     # NORI,IKARLE,IPROLO,NUMIN,LATT,IEF3,IEF4,IEF5,IGIN(4),IND,IPRO(4),
     1 LINE(14),IHKL(800),NAT,ICENT,PAD4(1305)
      COMMON/BLK6/ILIST,IQT,E(800),IP(2000),RPSIZO(1000),NW(10),NO(10),
     1 CVR(10),VVR(10),MAXHKL(3),PAD6(1555)
      DIMENSION IA(200,4),IROBA(200),IEE(30000)
      EQUIVALENCE(IEE(1),EEE(1)),(LK5,LK(5)),(KUSE20,KUSE(20)),
     1 (LK6,LK(6)),(LK10,LK(10)),(LK11,LK(11)),(LK16,LK(16)),(LK17,
     2 LK(17)),(LK21,LK(21)),(LK7,LK(7)),(LK8,LK(8))
      CHARACTER*1 LTEXTE(14),LIST(14)
      DATA LTEXTE/'A','B','C','D','E','F','G','H','I','J','K','+','-',
     1 ' '/
      IF(LEVEL.GT.0) WRITE(NSPEC,9000)
 9000 FORMAT(/' **** ****  LSAM PROGRAMME CALLED **** ****')
C**** FLAG FOR QUARTETS PRESENT OR ABSENT (1/0)
      IQT=0
C**** SET DEFAULT VALUES
      NSYMB=4
      PROLO=0.95
      KCUT=10
      PROB=0.80
      MITOR=1
      ILIST=1
C**** READ DATA FROM THE FILE WRITTEN AT THE END OF CONVERGE
      REWIND NTAPEC
      READ(NTAPEC)IZRO,NT,NSRT,JUNK,IEF3,IEF4,IEF5,NQTOT,
     1 (IPH1(J),J=1,KUSE20),(EEE(J),J=1,KUSE20),LIM
      IF(IEF3.NE.0) READ(NTAPEC) (IPH2(J),J=1,KUSE20)
      READ(NTAPEC)ICENT,ITLE,LAT,CX,NAT,(JUNK,J=1,2),NASU,MAXHKL
```

```
      READ(NTAPEC)NSYM,TS,IS,ISYMP,IAPX,(JUNK,J=1,3),
     1 NK,CVR,VVR,NW,NO,IVDIF,IDIF,(JUNK,J=1,2),IKARLE
      READ(NTAPEC)(JUNK,J=1,107),IHKL,E,(JUNK,J=1,800),IZ
C**** IF(NQTOT.GT.0) READ(NTAPEC)NQ1,NQ2,AQ,NQ4,NQ5
C**** CHECK STRUCTURE IS CENTROSYMMETRIC BEFORE PROCEEDING
      IF(ICENT.EQ.0)THEN
        IF(LEVEL.GT.0) WRITE(NSPEC,9020)
 9020   FORMAT(/' STRUCTURE IS NOT CENTROSYMMETRIC - CANNOT USE LSAM ')
        RETURN
      ENDIF
C**** WRITE FIRST RECORD OF FOURIER FILE
    5 MARK=0
      REWIND NTAPED
      WRITE(NTAPED)MARK,ITLE,ICENT,LAT,NSYM,TS,IS,MAXHKL,(CX(J),J=1,6)
     1,NASU,NT,
     2(IHKL(I),E(I),I=1,NT),NW,NO,CVR,VVR,NK,IAPX,ISYMP,IVDIF,IDIF
      ENDFILE NTAPED
      REWIND NTAPED
      NUMIN=NT/3
C**** SET LATTICE TYPE
      LATT=1
      IF(LAT.EQ.2.OR.LAT.EQ.3) LATT=2
      IF(LAT.EQ.4.OR.LAT.EQ.5) LATT=2
      IF(LAT.EQ.6) LATT=3
C**** DISPLAY MENU IN INTERACTIVE MODE
      IF(LEVEL.GT.0) CALL MENU12
      IF(IDEF.NE.0) GOTO 150
C**** DECODE ANY INSTRUCTIONS RECEIVED
   10 CALL DEC
      IF(IEND.NE.0) GOTO 150
      IF(IDEF.NE.0) GOTO 150
      IF(NREAD.EQ.50) CALL MENU12
      IF(NREAD.EQ.50) GOTO 10
      IF(NREAD.EQ.93) GOTO 20
      IF(NREAD.EQ.152) GOTO 30
      IF(NREAD.EQ.153) GOTO 40
      IF(NREAD.EQ.154) GOTO 50
      IF(NREAD.EQ.58)  GOTO 60
      IF(NREAD.EQ.74) GOTO 65
C**** CALL TO ANOTHER MODULE
      INXT=1
      GOTO 150
C**** USE LSAM ORIGIN &STARTING SET
   20 MITOR=0
      GOTO 10
C**** NO OF SYMBOLS TO BE ASSIGNED
   30 NSYMB=IABS(INT(A(1)))
C**** CHECK FOR GARBAGE INPUT
      IF(NSYMB.GT.11)THEN
        NSYMB=4
        IF(LEVEL.GT.0)WRITE(NSPEC,9002)
 9002   FORMAT(/' INVALID NO. OF SYMBOLS'/)
      ENDIF
      GOTO 10
C**** MIN PROBABILITY ACCEPTABLE FOR A SIGN DETERMINATION
   40 PROLO=A(1)
      IF(PROLO.LT.0.0.OR.PROLO.GT.1.0)THEN
        IF(LEVEL.GT.0) WRITE(NSPEC,9003)
```

```
 9003   FORMAT(/' PROBABILITY MUST BE BETWEEN 0 & 1'/)
        PROLO=0.95
      ENDIF
      GOTO 10
C**** NO OF EQUATIONS TO BE CONSIDERED
   50 KCUT=IABS(INT(A(1)))
      GOTO 10
C**** PROBABILITY LIMIT FOR USING SIGNS
   60 PROB=A(1)
      IF(PROB.LT.0.0.OR.PROB.GT.1.0)THEN
        IF(LEVEL.GT.0)WRITE(NSPEC,9003)
        PROB=0.80
      ENDIF
      GOTO 10
   65 ILIST=0
      GOTO 10
C**** INPUT COMPLETE
C**** GET PSI-ZERO RELATIONSHIPS
  150 IF(LEVEL.GT.0)WRITE(NSPEC,9012)
 9012 FORMAT(/' CARRYING OUT SYMBOLIC ADDITION - PLEASE WAIT'/)
C**** SORT INVARIANTS
 1600 IC=0
C**** UNPACK PHASE RELATIONSHIPS
      DO 14 IHH=1,NT
C**** ROUTINE TO STORE MILLER INDICES IN ARRAY LO(I,J)
        LO(IHH,4)=IHKL(IHH)/LK16
        LO(IHH,5)=MOD(IHKL(IHH),LK16)/LK8-LK7
        LO(IHH,6)=MOD(IHKL(IHH),LK8)-LK7
C**** NOW UNPACK PHASE RELATIONSHIPS
        LIM1=LIM(IHH)+1
        LIM2=LIM(1+IHH)
        IF(LIM1.GT.LIM2) GOTO 14
        DO 13 J=LIM1,LIM2
          IC=IC+1
          IEE(IC)=IFIX(100*EEE(IC))
          IK=IABS(IPH1(J)/LK16-LK10)
          IL=IABS(MOD(IPH1(J),LK16)/LK5-LK10)
          IM=0
          IF(IEF4.EQ.1.OR.IEF5.EQ.1) IM=IABS(IPH2(J)/LK17-LK10)
          IN=0
          IF(IEF5.EQ.1) IN=IABS(MOD(IPH2(J),LK17)/LK6-LK10)
          IPP=MOD(IPH1(J),LK5)
C**** PACK RELATIONSHIPS IN A DIFFERENT FORM
          IPH1(IC)=IHH*LK21+IK*LK11+IL
          IPH2(IC)=IM*LK21+IN*LK11+IPP
   13   CONTINUE
   14 CONTINUE
C**** SORT RELATIONSHIPS WITH RESPECT TO THEIR KAPPA VALUES
      M=2
  820 INTT=IC/M
      IF(INTT.EQ.0) INTT=1
      IFIN=IC-INTT
  830 MARK=0
      DO 840 I=1,IFIN
        J=I+INTT
        IF(IEE(I).GE.IEE(J)) GOTO 840
        M1=IEE(I)
        IEE(I)=IEE(J)
```

```
          IEE(J)=M1
          M1=IPH1(I)
          IPH1(I)=IPH1(J)
          IPH1(J)=M1
          M1=IPH2(I)
          IPH2(I)=IPH2(J)
          IPH2(J)=M1
          MARK=1
  840 CONTINUE
      IF(MARK.EQ.1) GOTO 830
      IF(INTT.EQ.1) GOTO 850
      M=2*M
      GOTO 820
C**** END OF SORTING ROUTINE
  850 JJ=0
      K=0
      REWIND NTAPEE
      DO 801 J=1,IC
        IHH=IPH1(J)/LK21
        IK=MOD(IPH1(J),LK21)/LK11
        IL=MOD(IPH1(J),LK11)
        IM=IPH2(J)/LK21
        IN=MOD(IPH2(J),LK21)/LK11
        IF(IN.NE.0) GOTO 801
        JJ=JJ+1
C**** STORE PHASE SHIFTS IN ARRAY IP( )
        IP(K+1)=0
        JS=1
        IPP=MOD(IPH2(J),LK11)
        IF(IPP.EQ.12)THEN
          IP(K+1)=180
          JS=-1
        ENDIF
C**** STORE INVARIANTS IN ARRAY IA( )
        IA(JJ,1)=IHH
        IA(JJ,2)=IK
        IA(JJ,3)=IL
        IA(JJ,4)=IM
        IF(IM.NE.0) IQT=1
C**** TRANSFER KAPPA VALUES (* 50) INTO IROBA( )
        IROBA(JJ)=IEE(J)*JS/2
        K=K+1
C**** PREVENT ARRAY IA( ) OVERFLOWING
        IF(K.EQ.2000) GOTO 100
C**** WRITE INVARIANTS AND KAPPA VALUES ONTO NTAPEE
        IF(JJ.NE.200) GOTO 801
C**** BUFFER FULL
        JJ=0
        WRITE(NTAPEE) IA,IROBA
  801 CONTINUE
      JJ=JJ+1
  100 IROBA(JJ)=-999
      NSRT=K
      WRITE(NTAPEE)IA,IROBA
      ENDFILE NTAPEE
      REWIND NTAPEE
C**** PREVENT INVARIANT WITH IROBA( )=-999 BEING USED IN ORIGIN DEFN.
      IF(NSRT.EQ.2000) NSRT=1999
      CALL PRELIM(LTEXTE,LIST)
```

```
      CALL RIGOLE(NNUM)
      CALL PRINTB(LTEXTE)
      CALL LOLA(LTEXTE)
      CALL LSAM2(LTEXTE,NNUM)
      IF(LEVEL.GT.0) WRITE(NSPEC,9001)
 9001 FORMAT(/5X,' **** **** LSAM COMPLETE **** ****')
      RETURN
      END
C
      SUBROUTINE PRELIM(LTEXTE,LIST)
C**** PRELIM FIXES THE ORIGIN AND ASSIGNS SYMBOLS
      COMMON/IO/NSPEC,NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(156),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEM
      COMMON/BLK1/IPH1(30000),EEE(30000),IZ(800),LO(800,11),
     1 IRO(800,11),IA(2000,4),IROBA(2000),JUNK(1200),PAD1(10400)
      COMMON/BLK4/NUM,NUM2,NYMBOL,PROLO,NT,NSRT,PROB,KCUT,MITOR,NSYMB,
     # NORI,IKARLE,IPROLO,NUMIN,LATT,IEF3,IEF4,IEF5,IGIN(4),IND,IPRO(4),
     1 LINE(14),IHKL(800),NAT,ICENT,NCODE(4,11),MCODE(4,11),PAD4(1217)
      DIMENSION IB(200,4),IROBB(200)
      CHARACTER*1 LTEXTE(14),LIST(14),LIST1(14)
      EQUIVALENCE (LK16,LK(16)),(LK8,LK(8)),(LK7,LK(7))
C**** READ INVARIANTS OFF NTAPEE
      II=0
      REWIND NTAPEE
   50 READ(NTAPEE)IB,IROBB
      DO 60 I=1,200
        IF(IROBB(I).EQ.-999) GOTO 70
        II=II+1
        IA(II,1)=IB(I,1)
        IA(II,2)=IB(I,2)
        IA(II,3)=IB(I,3)
        IA(II,4)=IB(I,4)
        IROBA(II)=IROBB(I)
   60 CONTINUE
      GOTO 50
C**** INITIALISE STORE
   70 DO 120 I=1,NT
        LO(I,1)=0
        LO(I,2)=0
  120 IRO(I,1)=0
      DO 130 I=1,4
  130   IGIN(I)=0
      NYMBOL=0
      NUM=0
      IND=0
      NORI=MAX0(1,4-IKARLE)
      NORI=IABS(NORI-LATT+1)
C**** COMPILE REFLEXION STATISTICS
  138 DO 160 I=1,NSRT
        DO 140 J=1,4
          K=IA(I,J)
          IF(K.EQ.0) GOTO 160
  140     LO(K,2)=LO(K,2)+(IABS(IROBA(I))+5)/10
  160 CONTINUE
C**** FIX ORIGIN
```

```
  620 IF(MITOR.EQ.1) GOTO 100
      IF (NORI.LE.0) GOTO 722
      J=1
      DO 680 I=1,NT
        IF(LO(I,2).LE.LO(J,2)) GOTO 680
        J=I
680   CONTINUE
      IF(LO(J,2).GT.0) GOTO 695
      WRITE(NOUT,690)
690   FORMAT(35H0ORIGIN CAN NOT BE PROPERLY DEFINED)
      STOP
695   CALL LSAMOR(LO(J,4),LO(J,5),LO(J,6),JUMP)
      IF(JUMP.EQ.2) GOTO 720
      IRO(J,1)=100000
      IZ(J)=0
      IND=IND+1
      LINE(IND)=J
      LIST(IND)=LTEXTE(12)
720   LO(J,2)=-LO(J,2)
      GOTO 620
C**** USE MULTAN  ORIGIN
100   DO 200 I=1,NT
        IF((IZ(I)+1).LE.0) GOTO 200
        IWT=IZ(I)-1000*(IZ(I)/1000)
        IF((IWT-100).LE.0) GOTO 200
        IND=IND+1
        IZ(I)=0
        IRO(I,1)=100000
        LINE(IND)=I
        LIST(IND)=LTEXTE(12)
  200 CONTINUE
C**** ASSIGN SYMBOLS
  722 JJ=NSYMB
      L=1
      M=0
      IF(MITOR.EQ.1) GOTO 10
      DO 15 I=1,NT
        LO(I,2)=IABS(LO(I,2))
        IF(IRO(I,1).NE.0) LO(I,2)=-LO(I,2)
   15 CONTINUE
   10 DO 800 I=1,JJ
        IF(IAND(NYMBOL,L))790,740,790
  740   IF(NSYMB.LE.0) GOTO 800
        IF(MITOR.EQ.1) GOTO 20
C**** LSAM SYMBOLS
        K=1
        DO 780 J=2,NT
          IF(LO(J,2).GT.LO(K,2)) K=J
  780   CONTINUE
        GOTO 25
C**** MULTAN SYMBOLS
   20   DO 910 K=1,NT
         IF(IZ(K).LT.0) GOTO 25
  910   CONTINUE
        GOTO 800
   25   NSYMB=NSYMB-1
        NYMBOL=NYMBOL+L
        LO(K,1)=L
```

```
        IRO(K,1)=100000
        LO(K,2)=-LO(K,2)
        M=M+1
        IZ(K)=M
        LIST(IND)=LTEXTE(M)
        LINE(IND)=K
  790   NUM=NUM+1
        L=2*L
  800 CONTINUE
  810 NUM2=(2**NUM)-1
C**** OUTPUT FIXED SIGNS AND REFLEXION WEIGHTS
      DO 820 I=1,NT
  820   LO(I,2)=IABS(LO(I,2))
      WRITE(NOUT,8009)ITLE
 8009 FORMAT(1H1138X,'LSAM  (LOGICAL SYMBOLIC ADDITION METHOD)'/
     1138X,40(1H-)///68A1/////45X,' ORIGIN DEFINING REFLECTIONS'//)
      IST=0
      IORG=0
C===> LIST OUT THE STARTING SET OF REFLECTIONS
      DO 1200 I=1,NT
      IF(IRO(I,1).LE.0)GOTO 1200
C===> UNPACK THE REFLECTION INDICES
      IHH=IHKL(I)/LK16
      IK=MOD(IHKL(I),LK16)/LK8-LK7
      IL=MOD(IHKL(I),LK8)-LK7
      M=IZ(I)
      IF(M.GT.0)THEN
C===> STARTING SET REFLECTION
        IST=IST+1
        NCODE(1,IST)=I
        NCODE(2,IST)=IHH
        NCODE(3,IST)=IK
        NCODE(4,IST)=IL
        LIST1(IST)=LTEXTE(M)
      ELSE
C===> ORIGIN DEFINING REFLECTION
        IORG=IORG+1
        MCODE(1,IORG)=I
        MCODE(2,IORG)=IHH
        MCODE(3,IORG)=IK
        MCODE(4,IORG)=IL
        LIST(IORG)=LTEXTE(12)
      ENDIF
 1200 CONTINUE
      WRITE(NOUT,8010)
 8010 FORMAT(1H //9X,3(' CODE    H    K    L  SYMBOL',15X))
      WRITE(NOUT,8011)((MCODE(I,J),I=1,4),LIST(J),J=1,IORG)
 8011 FORMAT(1H /10X,3(4I4,A5,19X)//)
C===> STARTING SET REFLECTIONS
      WRITE(NOUT,8012)((NCODE(I,J),I=1,4),LIST1(J),J=1,IST)
 8012 FORMAT(////39X,' SYMBOLS HAVE BEEN ALLOCATED AS FOLLOWS'///
     114(1H 'CODE    H    K    L  SYMBOL',5X)//4(1H 4I4,A5,8X))
      RETURN
      END
C
      SUBROUTINE LSAMOR(IH,IK,IL,JUMP)
      COMMON/BLK4/NUM,NUM2,NYMBOL,PROLO,NT,NSRT,PROB,KCUT,MITOR,NSYMB,
     # NORI,IKARLE,IPROLO,NUMIN,LATT,IEF3,IEF4,IEF5,IGIN(4),IND,IPRO(4),
```

```
     1 LINE(14),IHKL(800),NAT,ICENT,PAD4(1305)
C**** FIND OUT WHETHER REFLEXION IH,IK,IL IS SUITABLE FOR FIXING ORIGIN
C****      JUMP=1 IF IT IS     ;    JUMP=2 IF IT IS NOT
      IF(IKARLE.EQ.1)THEN
        LOG=MOD (IABS(IH),2)+2* MOD(IABS(IK),2)+4*MOD (IABS(IL),2)
      ELSE IF (IKARLE.EQ.2) THEN
        LOG=MOD(IABS(IH+IK),2)+2*MOD(IABS(IL),2)
      ELSE IF (IKARLE.EQ.3) THEN
        LOG=MOD(IABS(IL),2)
      ELSE IF (IKARLE.EQ.4) THEN
        LOG=MOD(IABS(IH+IK+IL),2)
      ENDIF
      J=LOG
      DO 100 I=1,4
        J=IEOR(J,IGIN(I))
        IF (IEOR(LOG,IGIN(I)).EQ.0) GOTO 140
  100 CONTINUE
      IF(J.EQ.0) GOTO 140
      IGIN(NORI)=LOG
      NORI=NORI-1
      JUMP=1
      RETURN
  140 JUMP=2
      RETURN
      END
C
      SUBROUTINE RIGOLE(NNUM)
C**** RIGOLE DEVELOPS SIGNS BY SYMBOLIC ADDITION
      COMMON/BLK1/IPH1(30000),EEE(30000),IZ(800),LO(800,11),
     1 IRO(800,11),IA(2000,4),IROBA(2000),JUNK(1200),NNN(800),LOG(5),
     2 IAD(5),KOUNT(800),IROG(5),IS(5),PAD1(8780)
      COMMON/BLK4/NUM,NUM2,NYMBOL,PROLO,NT,NSRT,PROB,KCUT,MITOR,NSYMB,
     # NORI,IKARLE,IPROLO,NUMIN,LATT,IEF3,IEF4,IEF5,IGIN(4),IND,IPRO(4),
     1 LINE(14),IHKL(800),NAT,ICENT,PAD4(1305)
C**** INITALISE STORE
C**** KOUNT=-1 FOR UNKNOWN SIGN  =0 FOR FIXED SIGN  =1 FOR KNOWN SIGN
      DO 180 I=1,NT
        KOUNT(I)=-1
        IF (IRO(I,1)) 160,180,160
  160   KOUNT(I)=0
  180 CONTINUE
      IVIEUX=0
      IPRO(1)=0
      IPRO(2)=34
      IPRO(3)=54
      IPRO(4)=68
      NITER=50
185   IPROLO=50.0*ALOG(PROLO/(1.0-PROLO))
190   INEW=0
      ISWAP=1
      NNUM=0
C**** BEGIN ITERATION ISWAP.NE.0 TO SET UP COMPLETE TABLE OF INDICATIONS
C****                 ISWAP.EQ.0 TO ADD NEW INDICATIONS ONLY TO TABLE
200   DO 220 I=1,NT
        NNN(I)=(KOUNT(I)+3)/3
        IF (KOUNT(I).LE.0) GOTO 220
        KOUNT(I)=2
        NNUM=NNUM+1
```

```
  220 CONTINUE
  240 DO 1000 I=1,NSRT
        ISS=ISIGN(1,IROBA(I))
        IF(ISWAP.NE.0) GOTO 300
        DO 280 J=1,4
          IF(IA(I,J).EQ.0) GOTO 280
          IF (INEW-IA(I,J)) 1000,300,280
  280   CONTINUE
        GOTO 1000
  300   L=1
        N=3
C**** LOOK FOR SUITABLE RELATIONSHIP
        DO 500 J=1,4
          K=IA(I,J)
          IF(K.EQ.0) GOTO 500
          IF (KOUNT(K)) 480,460,320
  320     M=1
          IF(ISWAP.EQ.0) GOTO 420
          K1=NNN(K)
          IF(K1.LE.1) GOTO 420
          DO 400 K2=2,K1
            IF(IABS(IRO(K,M)).LT.IABS(IRO(K,K2))) M=K2
  400     CONTINUE
  420     IAD(L)=K
  440     IROG(L)=IABS (IRO(K,M))
          LOG(L)=LO(K,M)
          IS(L)=ISIGN (1,IRO(K,M))
          L=L+1
          GOTO 500
  460     IAD(L)=INEW
          M=1
          N=N-1
          GOTO 440
  480     IAD(3)=K
  500    CONTINUE
C**** ADD SIGN INDICATIONS TO TABLE
        IF(L-3) 1000,520,540
  520   K1=1
        GOTO 620
  540   IF(N.LE.0) GOTO 1000
        IAD(4)=IAD(1)
        IAD(5)=IAD(2)
        LOG(4)=LOG(1)
        LOG(5)=LOG(2)
        IROG(4)=IROG(1)
        IROG(5)=IROG(2)
        IS(4)=IS(1)
        IS(5)=IS(2)
        K1=3
  620   DO 900 J=1,K1
          JJ=J+1
C**** TEST IF SIGN IS FIXED
          IF(IAD(J+2).EQ.INEW) GOTO 900
          K=IAD(J+2)
C**** INDICATED SYMBOL AND PROBABILITY
          LOPUT =IEOR(LOG(J),LOG(JJ))
          IROPUT=MIN0 (IROG(J),IROG(JJ),IABS(IROBA(I)))-IPRO(N)
          JSIGN=IS(J)*IS(JJ)*ISS
```

```
          IF(NNN(K).GT.0) GOTO 680
          K2=0
          GOTO 720
  680     K2=NNN(K)
C**** LOOK FOR SYMBOL IN TABLE
          DO 700 M=1,K2
            IF (LO(K,M).EQ.LOPUT) GOTO 840
  700     CONTINUE
C**** NEW SYMBOL, MAX NUMBER OF ELEVEN DIFFERENT SYMBOLS POSSIBLE
          IF(K2.GE.11) GOTO 740
  720     M=K2+1
          GOTO 820
  740     M=1
          DO 780 L=2,11
            IF (IABS (IRO(K,M)).GT.IABS (IRO(K,L))) M=L
  780     CONTINUE
          IF (IABS (IRO(K,M)).GE.IROPUT) GOTO 900
          K2=5
  820     NNN(K)=K2+1
          LO(K,M)=LOPUT
          GOTO 860
  840     IF(KOUNT(K).LE.1) GOTO 880
          KOUNT(K)=1
  860     IRO(K,M)=0
C**** ACCUMULATE PROBABILITIES
  880     IRO(K,M)=IRO(K,M)+ISIGN (IROPUT,JSIGN)
  900   CONTINUE
 1000 CONTINUE
C**** LOOK FOR NEW SIGN AND REARRANGE TABLE
       MAXX=0
       ISWAP=0
       DO 1200 I=1,NT
         IF (NNN(I)-1)  1200,1120,1020
 1020    M=1
         K2=NNN(I)
         DO 1080 L=2,K2
           IF (IABS (IRO(I,M)).LT.IABS (IRO(I,L))) M=L
 1080    CONTINUE
         IF (M-1) 1200,1120,1100
 1100    NSWAP=LO(I,M)
         LO(I,M)=LO(I,1)
         LO(I,1)=NSWAP
         NSWAP =IRO(I,M)
         IRO(I,M)=IRO(I,1)
         IRO(I,1)=NSWAP
         IF (KOUNT(I).NE.1) GOTO 1120
         ISWAP=1
 1120    IF (KOUNT(I)) 1140,1200,1180
 1140    IF (MAXX.GE.IABS(IRO(I,1))) GOTO 1200
         MAXX=IABS (IRO(I,1))
         INEW=I
         GOTO 1200
 1180    IF(IABS(IRO(I,1)).GE.IPROLO) GOTO 1200
         KOUNT(I)=-1
         ISWAP=1
         NNUM=NNUM-1
 1200 CONTINUE
C**** IS NEW SIGN ACCEPTABLE
```

```
 1210 IF (MAXX.LT.IPROLO) GOTO 1240
      KOUNT(INEW)=1
      NNUM=NNUM+1
      IF (ISWAP) 190,240,1230
 1230 NITER=NITER-1
      IF (NITER) 1380,1380,190
 1240 IF(NNUM.GT.NUMIN) GOTO 1250
      IPROLO = MAXX
      NITER=50
      IF (MAXX.GT.0) GOTO 1210
 1250 IOLD=0
      DO 1320 I=1,NT
        IF (KOUNT(I).LE.0) GOTO 1320
        IF (IABS (IRO(I,1)).LT.800) GOTO 1300
        IOLD=IOLD+1
        GOTO 1320
 1300   KOUNT(I)=-1
 1320 CONTINUE
      IF(IVIEUX.GE.IOLD) GOTO 1380
      IVIEUX=IOLD
      GOTO 185
 1380 RETURN
      END
C
      SUBROUTINE PRINTB(LTEXTE)
      COMMON/IO/NSPEC,NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(156),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEM
      COMMON/BLK1/IPH1(30000),EEE(30000),IZ(800),LO(800,11),
     1 IRO(800,11),IA(2000,4),IROBA(2000),JUNK(1200),PAD1(10400)
      COMMON/BLK4/NUM,NUM2,NYMBOL,PROLO,NT,NSRT,PROB,KCUT,MITOR,NSYMB,
     # NORI,IKARLE,IPROLO,NUMIN,LATT,IEF3,IEF4,IEF5,IGIN(4),IND,IPRO(4),
     1 LINE(14),IHKL(800),NAT,ICENT,PAD4(1305)
      COMMON/BLK6/ILIST,IQT,E(800),IP(2000),RPSIZO(1000),LLO(800),
     1 KNT(11),ENIL(4),PAD6(783)
      CHARACTER*11 IWORD(4),JWORD(11),LTEXTE(14)*1
      DO 700 I=1,NT
        LLO(I)=LO(I,1)
  700 CONTINUE
      DO 1400 I=1,11
 1400   KNT(I)=0
      IOLD=0
      DO 1500 I=1,NT
        IF(IABS (IRO(I,1)).LE.IPROLO) GOTO 1500
        L=1
        IOLD=IOLD+1
        DO 1460 J=1,NUM
          IF(IAND(LO(I,1),L).GT.0) KNT(J)=KNT(J)+1
          L=2*L
 1460   CONTINUE
 1500 CONTINUE
      PRONEW=0.5+0.5*TANH(0.01*FLOAT(IPROLO))
      WRITE (NOUT,1520)PROLO,PRONEW,IOLD,NT
 1520 FORMAT(////28X,'PROBABILITY LIMIT FOR ACCEPTANCE OF A SIGN SYMBOL
     1IS',F8.4//28X,'DURING SIGN DETERMINATION LIMIT HAS BEEN CHANGED TO
     2',F9.4//28X,I5,' SIGNS HAVE BEEN DETERMINED OUT OF A POSSIBLE
```

```
     3',I5)
      IF(LEVEL.GT.0)WRITE(NSPEC,1525)IOLD,NT
 1525 FORMAT(/1H ,I5,' SIGNS HAVE BEEN DETERMINED OUT OF A POSSIBLE'
     1,I5)
      IF(ILIST.EQ.0)THEN
      WRITE(NOUT,1530)
 1530 FORMAT(1H1,40X,45HTABLE OF SIGNS DEVELOPED BY SYMBOLIC ADDITION/39
     1X,47(1H-)//)
      ENDIF
      M=0
      DO 500 I=1,NT
        M=M+1
        LINE(M)=I
        K=LO(I,1)
        L=IRO(I,1)
        IF(K.NE.0) GOTO 5
        IWORD(M)=LTEXTE(12)
        IF(L.EQ.0) IWORD(M)=LTEXTE(14)
        GOTO 60
    5   JJ=0.5*(NUM2+1)
        DO  50 J=NUM,1,-1
          IF(K.LT.JJ) GOTO 10
          K=K-JJ
          IF(J.LT.NUM)JWORD(J)=LTEXTE(J)//JWORD(J+1)
          IF(J.EQ.NUM)JWORD(NUM)=LTEXTE(NUM)
          GOTO  50
   10     IF(J.LT.NUM)  JWORD(J)=JWORD(J+1)
          IF(J.EQ.NUM)  JWORD(NUM)=LTEXTE(14)
          JJ=JJ/2
   50   CONTINUE
        IWORD(M)=JWORD(1)
   60   ENIL(M)=0.5+0.5*TANH(0.01*FLOAT(IRO(I,1)))
        IF(M.NE.4) GOTO 500
        IF(ILIST.EQ.0)WRITE(NOUT,100)
     1                 (LINE(J),IWORD(J),ENIL(J),J=1,4)
  100   FORMAT(4(I12,2X,A11,F7.4))
        M=0
  500 CONTINUE
      IF(M.GT.0.AND.ILIST.EQ.0) WRITE(NOUT,100)
     1                      (LINE(J),IWORD(J),ENIL(J),J=1,M)
      RETURN
      END
C
      SUBROUTINE LOLA(LTEXTE)
C**** LOLA LOOKS FOR RELATIONSHIPS BETWEEN THE SYMBOLS
      COMMON/IO/NSPEC,NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(156),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEM
      COMMON/BLK1/IPH1(30000),EEE(30000),IZ(800),LO(800,11),
     1 IRO(800,11),IA(2000,4),IROBA(2000),LIP(2047),LILO(2047),
     2 LIP2(2047)
      COMMON/BLK4/NUM,NUM2,NYMBOL,PROLO,NT,NSRT,PROB,KCUT,MITOR,NSYMB,
     # NORI,IKARLE,IPROLO,NUMIN,LATT,IEF3,IEF4,IEF5,IGIN(4),IND,IPRO(4),
     1 LINE(14),IHKL(800),NAT,ICENT,PAD4(1305)
      COMMON/BLK6/ILIST,IQT,E(800),IP(2000),RPSIZO(1000),LLO(800),II(3),
     1 PROBL(3),PAD6(792)
```

```
      CHARACTER*11 IWORD(4),JWORD(11),LTEXTE(14)*1
      IF(ILIST.EQ.0)WRITE (NOUT,100) ITLE
  100 FORMAT(1H1,10X,68A1///40X,29HRELATIONSHIPS BETWEEN SYMBOLS/40X,29
     1(1H-)////3(12X,12HRELATIONSHIP,10X,4HPROB)/)
      DO 120 I=1,NUM2
        LIP(I)=0
        LIP2(I)=0
  120 CONTINUE
      DO 220 I=1,NSRT
        N=4
        LOPUT=0
        JSIGN=ISIGN(1,IROBA(I))
        IROPUT=IABS(IROBA(I))
        DO 180 J=1,4
          K=IA(I,J)
          IF(K.EQ.0) GOTO 180
          IF (IRO(K,1))     140,220,160
  140     JSIGN=-JSIGN
  160     LOPUT=IEOR(LOPUT,LLO(K))
          IF (IABS (IRO(K,1)).LE.99999) GOTO 175
          N=N-1
          GOTO 180
  175     IROPUT=MIN0(IROPUT,IABS(IRO(K,1)))
  180   CONTINUE
        IF(LOPUT.GT.0)LIP2(LOPUT)=LIP2(LOPUT)
     1                                  +ISIGN(IROPUT-IPRO(N),JSIGN)
  220 CONTINUE
      L=0
      DO 320 I=1,NUM2
        M=1
        DO 260 J=2,NUM2
          IF(IABS(LIP2(M)).GE.IABS(LIP2(J))) GOTO 260
          M=J
  260   CONTINUE
        IF (LIP2(M).EQ.0) GOTO 340
        LIP(I)=LIP2(M)
        LILO(I)=M
        K=M
        L=L+1
    5   JJ=0.5*(NUM2+1)
        DO  50 J=NUM,1,-1
          IF(K.LT.JJ) GOTO 10
          K=K-JJ
          IF(J.LT.NUM) JWORD(J)=LTEXTE(J)//JWORD(J+1)
          IF(J.EQ.NUM) JWORD(NUM)=LTEXTE(NUM)
          GOTO  50
   10     IF(J.LT.NUM) JWORD(J)=JWORD(J+1)
          IF(J.EQ.NUM) JWORD(NUM)=LTEXTE(14)
          JJ=JJ/2
   50   CONTINUE
        IWORD(L)=JWORD(1)
        PROBL(L)=0.5+0.5*TANH(0.01*FLOAT (LIP2(M)))
        II(L)=I
        IF(L.NE.3)GOTO 310
        IF(ILIST.EQ.0)WRITE(NOUT,300)
     1                    (IWORD(J),II(J),PROBL(J),J=1,3)
  300   FORMAT(3(10X,A11,5H = S(,I3,1H),F8.4))
        L=0
```

```
  310    LIP2(M)=0
  320 CONTINUE
  340 IF(ILIST.EQ.0)WRITE(NOUT,300)
     1                    (IWORD(J),II(J),PROBL(J),J=1,L)
      RETURN
      END
C
      SUBROUTINE LSAM2(LTEXTE,NNUM)
C**** LSAM2 SOLVES EQUATIONS BETWEEN SYMBOLS AND GENERATES SETS OF SIGNS
      COMMON/IO/NSPEC,NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(156),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEM
      COMMON/BLK1/IPH1(30000),EEE(30000),IZ(800),LO(800,11),
     1 IRO(800,11),IA(2000,4),IROBA(2000),LIP(2047),LILO(2047),
     2 LIP2(2047),ISS(2047,3),LH(686),CFOM(2048)
      COMMON/BLK3/STABLE(450),NQ1(500),NQ2(500),AQ(500),NQTOT,
     1 NQ4,NQ5,PAD3(629)
      COMMON/BLK4/NUM,NUM2,NYMBOL,PROLO,NT,NSRT,PROB,KCUT,IRE(11,12),
     1 IRS(11,11),JA(11),IAD(5),IROG(5),IVS(11),KE(11),KS(11,3),FM1,
     2 FM2,IFM3,IFM4,IFM5,NUMSET,IB(11),IS(12),ISI(5),GARB(1777)
      COMMON/BLK6/ILIST,IQT,E(800),IP(2000),ALFA(800),JUNK(200),
     1 LLO(800),ALFRAN,PAD6(797)
      CHARACTER*1 LTEXTE(14)
C**** SET UP SINE/COSINE LOOK-UP TABLE
      DO 50 J=1,450
        ARG=(J-1)*DTOR
   50 STABLE(J)=SIN(ARG)
      NUMSET=0
C**** SKIP THE FIRST READ OF NTAPED
      READ (NTAPED)
      K=1
      DO 120 I=1,11
        DO 110 J=1,3
          KS(I,J)=0
 110    CONTINUE
        DO 115 J=1,11
          IRE(I,J)=0
          IRS(I,J)=0
 115    CONTINUE
        IB(I)=0
        IS(I)=0
        KE(I)=K
        IRE(I,12)=1
        IF (IAND(NYMBOL,K).NE.0) GOTO 118
        KE(I)=0
        IRE(I,12)=0
 118    IVS(I)=0
        K=2*K
 120  CONTINUE
      K=1
      DO 200 I=1,31
        DO 160 J=1,3
          L=31*(J-1)+I
          IF(L.GT.2047) GOTO 180
          ISS(L,J)=K
 160    CONTINUE
```

```
180     LW(I)=K
        K=2*K
200   CONTINUE
      DO 240 I=1,NUM2
        IF (LIP(I)) 220,260,240
220     LILO(I)=LILO(I)+NUM2+1
240   CONTINUE
260   IF (KCUT.LT.I) GOTO 300
      KCUT=I-1
300   IT=0
      IF (KCUT.LE.0) GOTO 525
      DO 520 I=1,KCUT
        IF (IAND(LILO(I),NUM2).LE.0) GOTO 520
        IT=IT+1
        IVS(IT)=I
        DO 340 J=1,NUM
          IF (IAND(LILO(I),LW(J)).GT.0) GOTO 360
340     CONTINUE
360     KE(J)=IEOR(LILO(I),LW(J))
        DO 380 K=1,3
          KS(J,K)=ISS(I,K)
380     CONTINUE
        DO 440 L=I,KCUT
          IF (IAND(LILO(L),LW(J)).LE.0) GOTO 440
          LILO(L)=IEOR(IEOR(LILO(L),LW(J)),KE(J))
          DO 420 K=1,3
           ISS(L,K)=IEOR(ISS(L,K),KS(J,K))
420       CONTINUE
440     CONTINUE
        DO 500 L=1,NUM
          IF (IAND(KE(L),LW(J)).LE.0) GOTO 500
          KE(L)=IEOR(IEOR(KE(L),LW(J)),KE(J))
          DO 480 K=1,3
            KS(L,K)=IEOR(KS(L,K),KS(J,K))
480       CONTINUE
500     CONTINUE
        IF(IT.GE.NNUM) GOTO 525
520   CONTINUE
525   DO 680 I=1,NUM
        IF(IRE(I,12).EQ.0) GOTO 550
        IF (IAND(KE(I),LW( 7)).GT.0) IRE(I,12)=-1
550     DO 600 J=1,NUM
          IF (IAND(KE(I),LW(J)).LE.0) GOTO 600
          IRE(I,J)=1
          IB(J)=1
600     CONTINUE
        IF(IT.LE.0) GOTO 680
        DO 660 J=1,IT
          L=IVS(J)/31+1
          K=IVS(J)-31*(L-1)
          IF (IAND(KS(I,L),LW(K)).LE.0) GOTO 660
          IRS(I,J)=1
          IS(J)=1
660     CONTINUE
680   CONTINUE
      IS(IT+1)=0
C**** GET NO OF PHASE SETS
      NSS=(IT+1)*(2**(NUM-IT))-ISYMP+IQT
      IF(LEVEL.GT.0) WRITE(NSPEC,710)NSS
```

```
  710 FORMAT (/' NO OF SIGN SETS TO BE GENERATED =',I4/)
C      WRITE (NOUT,700)ITLE,KCUT,(LTEXTE(I),I=1,11),IVS,(LTEXTE(I),
C     1 (IRE(I,J),J=1,11),(IRS(I,J),J=1,11),IRE(I,12),I=1,11)
C 700  FORMAT(1H1/20X,68A1///10H THE FIRST,I3,27H EQUATIONS TO BE CONSIDE
C     1RED// 16H THE SOLUTION IS/20X,11A2,5X,2HS(,10(I2,4H) S(),I2,
C     3  7H)  SIGN//11(13X,A2,5H  =  ,11I2,3X,12I6/))
      IF(KCUT.LE.2)THEN
        IF(LEVEL.GT.0) WRITE(NSPEC,8000)
 8000   FORMAT(//
     1' ****** ****** ******  WARNING !!  ****** ****** ******'//
     #' TOO FEW SYMBOL RELATIONSHIPS WERE GENERATED TO PROCEED'/
     3' WITH LSAM - EXITING FROM MODULE IMMEDIATELY'//
     4' ****** ****** ******  **********  ****** ****** ******'//)
        RETURN
      ENDIF
      WRITE(NOUT,720)KCUT,PROB
  720 FORMAT(/35X,' THE TOP',I4,' SYMBOL RELATIONSHIPS WERE USED'//
     128X,'SIGNS WITH PROBABILITY GREATER THAN',F7.3,' ARE ACCEPTED')
      CALL GENE(LTEXTE)
C**** CALCULATE COMBINED F.O.M. AND OUTPUT SUMMARY
      CALL RESULT
      REWIND NTAPED
      RETURN
      END
C
      SUBROUTINE GENE(LTEXTE)
      COMMON/IO/NSPEC,NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(156),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEM
      COMMON/BLK1/IPH1(30000),EEE(30000),IZ(800),LO(800,11),
     1 IRO(800,11),IA(2000,4),IROBA(2000),LIP(2047),LILO(2047),
     2 LIP2(2047),ISS(2047,3),LW(686),CFOM(2048)
      COMMON/BLK4/NUM,NUM2,NYMBOL,PROLO,NT,NSRT,PROB,KCUT,IRE(11,12),
     1 IRS(11,11),JA(11),IAD(5),IROG(5),IVS(11),KE(11),KS(11,3),FM1,
     2 FM2,IFM3,IFM4,IFM5,NUMSET,IB(11),IS(12),ISI(5),K(11),GARB(1766)
      COMMON/BLK6/ILIST,IQT,E(800),IP(2000),ALFA(800),JUNK(200),
     1 LLO(800),ALFRAN,PAD6(797)
      CHARACTER*1 LTEXTE(14)
      DO 60 J=1,NT
   60 ALFA(J)=0.0
      DO 70 J=1,NSRT
C**** USE ONLY TRIPLETS
        IF(IA(J,4).NE.0) GOTO 70
C**** CALCULATE KAPPA SQUARED
        RKAPPA=0.0004*(FLOAT(IROBA(J))**2)
        IHH=IA(J,3)
        ALFA(IHH)=ALFA(IHH)+RKAPPA
   70 CONTINUE
      WRITE(NOUT,10)
   10 FORMAT(/////9X,' FIGURES OF MERIT'/10X,16(1H-)//18X,
     1 '  SET    ABS    PSIZERO  RESID    NQEST    NQINT',20X,
     2 ' SIGNS OF SYMBOLS'//)
      CALL CLSCN
      IF(LEVEL.GT.0) WRITE(NSPEC,20)
   20 FORMAT(' FIGURES OF MERIT:'//
     1 '  SET    ABS    PSIZERO  RESID    NQEST    NQINT'/)
      NO=0
      IPROLO=50.0*ALOG(PROLO/(1.0-PROLO))
```

```
      IPRO=50.0*ALOG(PROB/(1.0-PROB))
  140 DO 200 I=1,11
        JA(I)=ISIGN (1,IRE(I,12))
  200 CONTINUE
      DO 400 I=1,11
        DO 320 J=1,11
          IF (IRE(I,J).LE.0.OR.IB(J).GT.0) GOTO 260
          JA(I)=-JA(I)
          K(I)=I
  260     IF (IRS(I,J).LE.0.OR.IS(J).GE.0) GOTO 320
          JA(I)=-JA(I)
          K(I)=I
  320   CONTINUE
  400 CONTINUE
      NO=NO+1
C**** IGNORE FIRST SET IF THERE IS NO TRANSLATIONAL SYMMETRY PRESENT
C**** TRIVIALLY THE BEST SOLUTION
      IF(NO.EQ.1.AND.ISYMP.EQ.1.AND.IQT.EQ.0) GOTO 441
C**** CALCULATE PHASES
      CALL LOLITA(LTEXTE,IPRO,NO,IPROLO)
  441 DO 440 I=1,11
        IF (IB(I)) 420,440,460
  420   IB(I)=1
  440 CONTINUE
      GOTO 500
  460 IB(I)=-1
      GOTO 140
  500 DO 520 I=1,12
        IF (IS(I).LE.0) GOTO 540
  520 CONTINUE
  540 IS(I)=1
      IF(I.LE.1) GOTO 580
      IS(I-1)=-1
      GOTO 140
  580 RETURN
      END
C
      SUBROUTINE LOLITA(LTEXTE,IPRO,NO,IPROLO)
      COMMON/IO/NSPEC,NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(156),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEM
      COMMON/BLK1/IPH1(30000),EEE(30000),IZ(800),LO(800,11),
     1 IRO(800,11),IA(2000,4),IROBA(2000),LIP(2047),LILO(2047),
     2 LIP2(2047),ISS(2047,3),LH(686),CFOM(2048)
      COMMON/BLK2/LIM(901),IZRO,PAD2(698)
      COMMON/BLK3/STABLE(450),NQ1(500),NQ2(500),AQ(500),NQTOT,
     1 NQ4,NQ5,I4,I5,IQTOT,CCFOM,RN4,RN5,PAD3(623)
      COMMON/BLK4/NUM,NUM2,NYMBOL,PROLO,NT,NSRT,PROB,KCUT,IRE(11,12),
     1 IRS(11,11),JA(11),IAD(5),IROG(5),IVS(11),KE(11),KS(11,3),FM1,
     2 FM2,IFM3,IFM4,IFM5,NUMSET,IB(11),IS(12),ISI(5),GARB(1777)
      COMMON/BLK6/ILIST,IQT,E(800),IP(2000),ALFA(800),JUNK(200),
     1 LLO(800),ALFRAN,PAD6(797)
      DIMENSION  CTABLE(360),JTEXTE(3)
      CHARACTER*1 LTEXTE(14),NWORD(11)
      EQUIVALENCE (LK10,LK(10)),(CTABLE(1),STABLE(91)),(LK18,LK(18)),
     1 (LK8,LK(8)),(LK9,LK(9)),(LK13,LK(13)),(LK7,LK(7)),(LK5,LK(5))
```

```
      DATA JTEXTE/'   + ','   - ','     '/
      ABSFOM=0.0
      RESID=0.0
      DO 300 I=1,NT
        IRO(I,2)=0
        IF (IABS (IRO(I,1)).LT.500) GOTO 300
        IRO(I,2)=IRO(I,1)
        DO 260  J=1,11
          IF (IAND(LLO(I),LW(J)).LE.0) GOTO 260
          IF (JA(J).LT.0) IRO(I,2)=-IRO(I,2)
  260   CONTINUE
  300 CONTINUE
      CALL RIG2(IPROLO)
      DO 320 I=1,NT
        IRO(I,4)=0
        IF (IABS(IRO(I,2)).LE.IPRO) GOTO 320
        IRO(I,4)=ISIGN(1,IRO(I,2))
  320 CONTINUE
      NUMSET=NUMSET+1
      ALFRAN=0.0
      DO 100 J=1,NT
        IIS=IRO(J,4)
        IPHAZ=0
        IF (IIS.EQ.-1) IPHAZ=180
C**** CALCULATE A RANDOM ALPHA
        IF(IIS.NE.0)ALFRAN=ALFRAN+SQRT(ALFA(J))
C**** FOR A KNOWN PHASE THE WEIGHT EQUALS 1
        IWT=100
C**** FOR AN UNKNOWN PHASE THE WEIGHT EQUALS 0
        IF (IIS.EQ.0) IWT=0
C**** IZ( ) NOW CONTAINS THE PHASES
        IZ(J)=IPHAZ*LK10+IWT
 100  CONTINUE
C**** CALCULATE NQEST & NQINT FIGURES OF MERIT
      I4=0
      I5=0
      CCFOM=0.0
      RN4=0.0
      RN5=0.0
C**** CANNOT CALCULATE NQEST/NQINT IF NO QUARTETS/QUINTETS PRESENT
      IF(NQTOT.EQ.0) GOTO  410
      IF(NQTOT.GT.500)NQTOT=500
      RNUM4=0.0
      RDEN4=0.0
      RNUM5=0.0
      RDEN5=0.0
      DO 20 J=1,NQTOT
        IHH=NQ1(J)/LK18
        IF(IRO(IHH,4).EQ.0) GOTO 20
        IK=MOD(NQ1(J),LK18)/LK9-LK8
        IKA=IABS(IK)
        IF(IRO(IKA,4).EQ.0) GOTO 20
        IL=MOD(NQ1(J),LK9)-LK8
        ILA=IABS(IL)
        IF( IRO(ILA,4).EQ.0)GOTO 20
        IM=NQ2(J)/LK13-LK8
        IMA=IABS(IM)
        IF(IRO(IMA,4).EQ.0) GOTO 20
        INA=0
```

```
          IF( NQ5. EQ. 0)  GOTO 30
          IN=MOD( NQ2( J) , LK13) /LK5-LK7
          IF( IN. EQ. 0)  GOTO 30
          INA=IABS( IN)
          IF( IRO( INA, 4) . EQ. 0)  GOTO 20
    30    IPP=MOD( NQ2( J) , LK5) *15
          IPHAZH=IZ( IHH) /LK10
          IPHAZK=IZ( IKA) /LK10
          IPHAZL=IZ( ILA) /LK10
          IPHAZM=IZ( IMA) /LK10
          ISUM=IPHAZH +ISIGN( IPHAZK, IK)  +ISIGN( IPHAZL, IL)
      1    +ISIGN( IPHAZM, IM)  +IPP+2340
          IF( INA. EQ. 0) GOTO 40
C**** QUINTET
          IPHAZN= IZ( INA) /LK10
          ISUM=MOD( ISUM+ISIGN( IPHAZN, IN) , 360)  +1
          RNUM5=RNUM5+AQ( J) *CTABLE( ISUM)
          RDEN5=RDEN5+AQ( J)
          I5=I5+1
          GOTO 20
C**** QUARTET
    40    ISUM=MOD( ISUM, 360) +1
          RNUM4=RNUM4+AQ( J) *CTABLE( ISUM)
          RDEN4=RDEN4+AQ( J)
          I4=I4+1
    20 CONTINUE
C**** CALCULATE FIGURES OF MERIT
        IF( ABS( RDEN4) . LT. 0. 00001) THEN
          RN4=1. 0
        ELSE
          RN4=RNUM4/RDEN4
        ENDIF
        IF( I5. GT. 20. AND. ABS( RDEN5) . GT. 0. 00001)   RN5=RNUM5/RDEN5
        IQTOT=I4+I5
        IF( IQTOT. LT. 20)  IQTOT=0
        IF( IQTOT. GT. 0)  CCFOM=( RN4*FLOAT( I4) +RN5*FLOAT( I5)) /
       1  FLOAT( IQTOT)
C**** GET ABSFOM & RESID FIGURES OF MERIT
   410 CALL ABFOM( ABSFOM, RESID)
C**** GET PSI-ZERO FIGURE OF MERIT
        CALL PSZO( VALUE)
        IF ( ABSFOM. LE. 0. 0) THEN
          VALUE=5. 0
        ELSE
          VALUE=VALUE/AMIN1( 1. 3, ABSFOM)
        ENDIF
        WRITE ( NTAPED)  NUMSET, ABSFOM, VALUE, RESID, RN4, RN5, ( IZ( J) , J=1, NT)
        DO 600 I=1, NT
          IF ( IRO( I, 4) . LT. 0) THEN
            IRO( I, 4) =JTEXTE( 1)
          ELSE IF( IRO( I, 4) . EQ. 0) THEN
            IRO ( I, 4) =JTEXTE( 3)
          ELSE IF( IRO( I, 4) . GT. 0) THEN
            IRO( I, 4) =JTEXTE( 2)
          ENDIF
   600 CONTINUE
        DO 620 I=1, 11
          K=12+( 1-JA( I)) /2
```

```
      IF(K.GT.14)K=14
      NWORD(I)=LTEXTE(K)
  620 CONTINUE
      WRITE(NOUT,650)NO,ABSFOM,VALUE,RESID,RN4,RN5,(NWORD(I),I=1,11)
  650 FORMAT(18X,I5,F7.2,F8.3,F8.2,F7.2,F8.2,12X,11A3)
      IF(LEVEL.GT.0)WRITE(NSPEC,670)NO,ABSFOM,VALUE,RESID,RN4,RN5
  670 FORMAT(I5,F7.2,F8.3,F8.2,F7.2,F8.2)
      RETURN
      END
C
      SUBROUTINE RIG2(IPROLO)
      COMMON/IO/NSPEC,NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(156),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEM
      COMMON/BLK1/IPH1(30000),EEE(30000),IZ(800),LO(800,11),
     1 IRO(800,11),IA(2000,4),IROBA(2000),LIP(2047),LILO(2047),
     2 LIP2(2047),ISS(2047,3),LW(686),CFOM(2048)
      COMMON/BLK4/NUM,NUM2,NYMBOL,PROLO,NT,NSRT,PROB,KCUT,IRE(11,12),
     1 IRS(11,11),JA(11),IAD(5),IROG(5),IVS(11),KE(11),KS(11,3),FM1,
     2 FM2,IFM3,IFM4,IFM5,NUMSET,IB(11),IS(12),ISI(5),GARB(1777)
      NNN=30
      DO 180 I=1,NT
        IRO(I,3)=-1
        IF (IRO(I,2).EQ.0) GOTO 180
        IF(IABS(IRO(I,2)).LT.10000) GOTO 160
        IRO (I,3)=0
        GOTO 180
  160   IRO(I,3) =2
  180 CONTINUE
  190 INEW=0
      ISWAP=1
      DO 220 I=1,NT
        IF (IRO(I,3).GT.0) IRO(I,3)=2
  220 CONTINUE
  240 DO 1000 I=1,NSRT
        IF (ISWAP.NE.0) GOTO 300
        DO 280 J=1,4
          IF(IA(I,J).EQ.0) GOTO 280
          IF (INEW-IA(I,J)) 1000,300,280
  280   CONTINUE
        GOTO 1000
  300   L=1
        DO 500 J=1,4
          K=IA(I,J)
          IF(K.EQ.0) GOTO 500
          IF (IRO(K,3)) 480,460,420
  420     IAD(L)=K
  440     IROG(L)=IABS(IRO(K,2))
          ISI(L)=ISIGN (1,IRO(K,2))
          L=L+1
          GOTO 500
  460     IAD(L)=INEW
          GOTO 440
  480     IAD(3)=K
  500   CONTINUE
        IF (L-3) 1000,520,540
```

```
  520    K1=1
         GOTO 620
  540    DO 600 J=1,2
           IAD(J+3)=IAD(J)
           IROG(J+3)=IROG(J)
           ISI(J+3)=ISI(J)
  600    CONTINUE
         K1=3
  620    DO 900 J=1,K1
           IF (IAD(J+2).EQ.INEW) GOTO 900
           K=IAD(J+2)
           IROPUT=MIN0 (IROG(J),IROG(J+1),IABS (IROBA(I)))-54
           JSIGN=ISI(J)*ISI(J+1)*ISIGN (1,IROBA(I))
           IF (IRO(K,3).LE.1) GOTO 880
           IRO(K,3)=1
           IRO(K,2)=0
  880      IRO(K,2)=IRO(K,2)+ISIGN (IROPUT,JSIGN)
  900    CONTINUE
 1000 CONTINUE
       MAXX=0
       ISWAP=0
       DO 1200 I=1,NT
         IF (IRO(I,3)) 1140,1200,1180
 1140    IF (MAXX.GE.IABS(IRO(I,2))) GOTO 1200
         MAXX=IABS (IRO(I,2))
         INEW=I
         GOTO 1200
 1180   IF (IABS(IRO(I,2)).GE.IPROLO) GOTO 1200
         IRO(I,3)=-1
         ISWAP=1
 1200 CONTINUE
       IF (MAXX.LT.IPROLO) GOTO 1240
       IRO(INEW,3)=1
       IF (ISWAP.EQ.0) GOTO 240
       NNN=NNN-1
       IF (NNN.GT.0) GOTO 190
1240   RETURN
       END
C
       SUBROUTINE ABFOM(ABSFOM,RESID)
C**** CALCULATES ABSFOM AND RESIDUAL FIGURES OF MERIT
       COMMON/IO/NSPEC,NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
      1 NTAPEF,NTAPEG,IH(156),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
      2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
      3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
      4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEM
       COMMON/BLK1/IPH1(30000),EEE(30000),IZ(800),LO(800,11),
      1 IRO(800,11),IA(2000,4),IROBA(2000),LIP(2047),LILO(2047),
      2 LIP2(2047),ISS(2047,3),LW(686),CFOM(2048),PALF(800),ALPHA(800),
      3 EALF(800),TALF(800)
       COMMON/BLK4/NUM,NUM2,NYMBOL,PROLO,NT,NSRT,PROB,KCUT,IRE(11,12),
      1 IRS(11,11),JA(11),IAD(5),IROG(5),IVS(11),KE(11),KS(11,3),FM1,
      2 FM2,IFM3,IFM4,IFM5,NUMSET,IB(11),IS(12),ISI(5),GARB(1777)
       COMMON/BLK6/ILIST,IQT,E(800),IP(2000),ALFA(800),JUNK(200),
      1 LLO(800),ALFRAN,PAD6(798)
       EQUIVALENCE(LK10,LK(10))
       VEC(U)=U*(U+0.4807)/((U+0.8636)*U+1.3943)
C**** INITIALISE ARRAYS
```

```
      DO 10 I=1,NT
        ALPHA(I)=0.0
        EALF(I)=0.0
        TALF(I)=0.0
   10 CONTINUE
C**** LOOP OVER ALL RELATIONSHIPS
      DO 100 J=1,NSRT
        IM=IA(J,4)
        IF(IM.NE.0) GOTO 100
        IHH=IA(J,3)
        IK=IA(J,2)
        IL=IA(J,1)
C**** IF RELATIONSHIP IS A QUARTET THEN IGNORE IT
C**** CHECK ALL PHASES ARE KNOWN
        IF(IRO(IHH,4).EQ.0.OR.IRO(IK,4).EQ.0.OR.IRO(IL,4).EQ.0)
     1     GOTO 100
C**** IROBA( )=KAPPA*50
        RKAPPA=ABS(FLOAT(IROBA(J))*0.02)
        VECX=VEC(RKAPPA)*RKAPPA
        IESTH=IZ(IK)/LK10+IZ(IL)/LK10+IP(J)
        IF(IESTH.LT.0) IESTH=IESTH+360
        IF(IESTH.GT.360) IESTH=IESTH-360
        IF(IESTH.EQ.180) THEN
          ALPHA(IHH)=ALPHA(IHH)-RKAPPA
        ELSE
          ALPHA(IHH)=ALPHA(IHH)+RKAPPA
        ENDIF
        EALF(IHH)=EALF(IHH)+VECX
        TALF(IHH)=TALF(IHH)+RKAPPA*RKAPPA-VECX*VECX
  100 CONTINUE
      DO 150 J=1,NT
        PALF(J)=EALF(J)*EALF(J)+TALF(J)
  150 CONTINUE
      SUMALF=0.0
      ALFEST=0.0
C**** CALCULATE ABSFOM
      DO 300 LL=1,NT
C**** IGNORE REFLECTION IF IT IS UNDETERMINED
        IF(IRO(LL,4).EQ.0) GOTO 300
        ALPHA(LL)= ABS(ALPHA(LL))
        PALF(LL)=SQRT(PALF(LL))
        ALFEST=ALFEST+PALF(LL)
        SUMALF=SUMALF+ALPHA(LL)
  300 CONTINUE
C**** NOW GET ABSFOM
      ABSFOM=(SUMALF-ALFRAN)/(ALFEST-ALFRAN)
C**** CALCULATE RESIDUAL
      RESID=0.0
      DO 400 L=1,NT
C**** IGNORE REFLECTION IF IT IS UNDETERMINED
        IF(IRO(L,4).EQ.0) GOTO 400
        RESID=RESID+ABS(PALF(L)-ALPHA(L))
  400 CONTINUE
      RESID=100.0*RESID/ALFEST
      RETURN
      END
C
      SUBROUTINE PSZO(VALUE)
C**** CALCULATES PSI-ZERO FIGURE OF MERIT
```

```
      COMMON/IO/NSPEC,NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(156),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEM
      COMMON/BLK1/IPH1(30000),EEE(30000),IZ(800),LO(800,11),
     1 IRO(800,11),IA(2000,4),IROBA(2000),LIP(2047),LILO(2047),
     2 LIP2(2047),ISS(2047,3),LW(686),CFOM(2048)
      COMMON/BLK2/LIM(901),IZRO,PAD2(698)
      COMMON/BLK3/STABLE(450),NQ1(500),NQ2(500),AQ(500),NQTOT,
     1 NQ4,NQ5,I4,I5,IQTOT,CCFOM,RN4,RN5,PAD3(623)
      COMMON/BLK4/NUM,NUM2,NYMBOL,PROLO,NT,NSRT,PROB,KCUT,IRE(11,12),
     1 IRS(11,11),JA(11),IAD(5),IROG(5),IVS(11),KE(11),KS(11,3),FM1,
     2 FM2,IFM3,IFM4,IFM5,NUMSET,IB(11),IS(12),ISI(5),GARB(1777)
      COMMON/BLK6/ILIST,IQT,E(800),IP(2000),ALFA(800),JUNK(200),
     1 LLO(800),ALFRAN,PAD6(798)
      EQUIVALENCE (LK5,LK(5)),(LK10,LK(10)),(LK16,LK(16))
      PSIZRO=0.0
      RANPSI=0.0
      VALUE=2.0
C**** CHECK TO SEE IF THERE ARE ANY PSI-ZERO RELATIONSHIPS PRESENT
      IF(IZRO.LE.0) THEN
        VALUE=0.0
        RETURN
      ENDIF
      DO 1500 I=1,IZRO
        LL=I+NT
C**** CHECK FOR ARRAY LIM GOING OUT OF BOUNDS
        IF(LL.GT.901.AND.LEVEL.GT.0) WRITE(NSPEC,9001)
 9001   FORMAT(/' PROGRAMMING PROBLEM - ARRAY LIM HAS GONE OUT OF
     1 BOUNDS')
        LI=LIM(LL)+1
        LS=LIM(LL+1)
        IF(LI.GT.LS) GOTO 1500
        RAN=0.0
        TOP=0.0
        DO 1100 JJ=LI,LS
C**** UNPACK RELATIONSHIPS
          IL=IPH1(JJ)/LK16-LK10
          ILA=IABS(IL)
C**** CHECK PHASE IS KNOWN
          IF(IRO(ILA,4).EQ.0) GOTO 1100
          IRK=MOD(IPH1(JJ),LK16)/LK5-LK10
          IRA=IABS(IRK)
          IF(IRO(IRA,4).EQ.0)GOTO 1100
C**** KAPPA VALUE
          EE=EEE(JJ)
          RAN=RAN+EE*EE
C**** PHASE SHIFT
          IPS=15*MOD(IPH1(JJ),LK5)
          IPHAZL=IZ(ILA)/LK10
          IPHAZR=IZ(IRA)/LK10
          ISUM=IPHAZL+IPHAZR+IPS
          IF(MOD(ISUM,360).NE.0) EE=-EE
          TOP=TOP+EE
 1100   CONTINUE
C**** GET FINAL FIGURE OF MERIT
 1400   PSIZRO=PSIZRO+ABS(TOP)
        RANPSI=RANPSI+SQRT(RAN)
```

```
 1500 CONTINUE
 1600 IF(RANPSI.GT.0.1) VALUE=PSIZRO/RANPSI
      RETURN
      END
C
      SUBROUTINE RESULT
      COMMON/IO/NSPEC,NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(156),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEM
      COMMON/BLK1/IPH1(30000),EEE(30000),IZ(800),LO(800,11),
     1 IRO(800,11),IA(2000,4),IROBA(2000),LIP(2047),LILO(2047),
     2 LIP2(2047),ISS(2047,3),LW(686),CFOM(2048)
      COMMON/BLK3/STABLE(450),NQ1(500),NQ2(500),AQ(500),NQTOT,
     1 NQ4,NQ5,WTFOM(5),AMX(5),AMN(5),PAD3(614)
      COMMON/BLK4/NUM,NUM2,NYMBOL,PROLO,NT,NSRT,PROB,KCUT,IRE(11,12),
     1 IRS(11,11),JA(11),IAD(5),IROG(5),IVS(11),KE(11),KS(11,3),FM1,
     2 FM2,IFM3,IFM4,IFM5,NUMSET,IB(11),IS(12),ISI(5),GARB(1777)
      COMMON/BLK6/ILIST,IQT,E(800),IP(2000),ALFA(800),JUNK(200),
     1 LLO(800),ALFRAN,PAD6(797)
C**** OUTPUT SUMMARY OF RESULTS FOR F.O.M. AND CALCULATE C.F.O.M.
      DIMENSION CTABLE(360),FM(2250,6)
C**** COMPUTE COMBINED FIGURES OF MERIT AND OUTPUT SUMMARY OF RESULTS
      IF(NUMSET.EQ.1) GOTO 1190
      ENDFILE NTAPED
      REWIND NTAPED
      READ (NTAPED)
C**** INITIALISE AMX,AMN AND NORMALISE CFOM TO 2.0,3.0,4.0 OR 5.0
      DO 10 J=1,3
   10 WTFOM(J)=1.0
      IF(NQTOT.NE.0)THEN
        WTFOM(4)=1.0
      ELSE
        WTFOM(4)=1.0
      ENDIF
      WTFOM(5)=0.0
      DO 80 I=1,5
      AMX(I)=-1.0E20
      AMN(I)=1.0E20
   80 CONTINUE
      DO 120 I=1,NUMSET
      READ (NTAPED)NSET,(FM(I,J),J=1,5),(IZ(K),K=1,NT)
      IF(NUMSET.EQ.1) GOTO 1190
      DO 100 J=1,5
      AMX(J) = AMAX1(AMX(J), FM(I,J))
  100 AMN(J) = AMIN1(AMN(J), FM(I,J))
  120 CONTINUE
      WRITE (NOUT,140) ITLE,AMX,AMN,WTFOM
  140 FORMAT(1H1,20X,68A1/// 33X,' SUMMARY OF FIGURES OF MERIT OUTPUT '
     1 ,'BY LSAM'///35X,'ABS FOM',11X,'PSI ZERO',11X,'RESID',13X,
     2 'NQEST',12X,'NQINT'//10X,'MAXIMUM VALUE',F19.4,F18.3,3F17.2//
     3 10X,'MINIMUM VALUE',F19.4,F18.3,3F17.2//9X,'RELATIVE WEIGHTS',
     4 F15.2,F19.2,F18.2,2F17.2/8X,'(FOR COMBINED FOM)'/
     5 1H0,11X,'NUMSET',7X,'ABS FOM',11X,'PSI ZERO',10X,'RESID',
     6 13X,'NQEST',9X,'NQINT',7X,'COMBINED FOM')
      CALL CLSCN
      IF(LEVEL.GT.0) WRITE(NSPEC,7003) AMX,AMN,WTFOM
```

```
 7003 FORMAT(////10X,'**** SUMMARY OF RESULTS ****'/
     1 '          ABS FOM    PSI ZERO    RESID    NQEST    NQINT'/
     2 '  MAX    ',F7.3,F11.3,3F8.3/'  MIN    ',F7.3,F11.3,3F8.3/
     3 '  WEIGHTS',F7.3,F11.3,3F8.3///
     4 ' NUMSET  ABS FOM    PSI ZERO    RESID    NQEST    NQINT    CFOM'/)
      IF (AMX(1) .EQ. AMN(1)) AMN(1) = 0.0
      IF (AMX(2) .EQ. AMN(2)) AMX(2) = 100.0
      IF (AMX(3) .EQ. AMN(3)) AMX(3) = 100.0
      IF (AMX(4) .EQ. AMN(4)) AMN(4) = 100.0
      IF (AMX(5) .EQ. AMN(5)) AMN(5) = 100.0
      DO 180 I=1,NUMSET
  180 FM(I,6) =WTFOM(1) * (FM(I,1) - AMN(1)) / (AMX(1) - AMN(1)) +
     1          WTFOM(2) * (AMX(2) - FM(I,2)) / (AMX(2) - AMN(2)) +
     2          WTFOM(3) * (AMX(3) - FM(I,3)) / (AMX(3) - AMN(3))+
     3          WTFOM(4) * (AMX(4) - FM(I,4)) / (AMX(4) - AMN(4))+
     4          WTFOM(5) * (AMX(5) - FM(I,5)) / (AMX(5) - AMN(5))
      DO 1000 I=1,NUMSET
      WRITE(NOUT,400)I,(FM(I,J),J=1,6)
  400 FORMAT(7X,I8,F16.4,F18.3,F17.2,  F17.2,F14.2,F16.4)
      IF(LEVEL.GT.0) WRITE(NSPEC,7004)I,(FM(I,J),J=1,6)
 7004 FORMAT(1H ,I4,2F11.3,3F8.3,F7.2)
 1000 CONTINUE
 1190 NIX=-1
      WRITE(NTAPED) NIX
      WRITE(NTAPED) NUMSET,(FM(I,6),I=1,NUMSET)
      ENDFILE NTAPED
      RETURN
      END
      SUBROUTINE MENU12
      COMMON/IO/NSPEC,NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(156),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEM
      IF(LEVEL.EQ.0) RETURN
      CALL CLSCN
      WRITE(NSPEC,9001)
9001  FORMAT('OTHE FOLLOWING COMMANDS ARE CURRENTLY AVAILABLE:'/
     1'  LSAM   <                                                  >'/
     2'  SYMBOL < NO OF SYMBOLS TO BE ASSIGNED              (   4) >'/
     3'  SIGN   < MIN PROB ACCEPTABLE FOR SIGN DETERMINATION (0.95) >'/
     4'  NUMBER < NO OF EQUATIONS TO BE CONSIDERED          (  10) >'/
     5'  LIMIT  < SIGNS PREDICTED WITH PROB>LIMIT TO BE USED (0.80) >'/
     6'  ORIGIN < USE LSAM ORIGIN & STARTING SET                    >'/
     7'  LIST   < OUTPUT TABLES OF SIGNS AND SYMBOL RELATIONSHIPS   >'/
     8)
      CALL MENU
      RETURN
      END
```

APPENDIX B

PROGRAM LISTING OF SUBMIT

APPENDIX B - PROGRAM LISTING OF SUBMIT

```
C
C      SSSSS   U   U   BBBB     M   M  III  TTTTT
C      S       U   U   B   B    MM MM   I     T
C      SSSSS   U   U   BBBB     M M M   I     T
C          S   U   U   B   B    M   M   I     T
C      SSSSS    UUU    BBBB     M   M  III    T
C
C
C===> SUBMIT - a subset of MITHRIL for use on a STRIDE 460 micro.
C===>
C===> Authors:   S.R.Brown & C.J.Gilmore
C===>            University of Glasgow, 1986
C
C===> It contains facilities for:
C===> (a) Fourier recycling of molecular fragments found by MITHRIL
C===> (b) Further recycling of any fragments found
C===> (c) E-map calculation and interpretation
C===> (d) Contouring sections of E-maps/weighted Fourier maps
C===> (e) Manipulation and editing of fragments
C===> (f) Inspection of E-maps calculated by MITHRIL
C
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFL,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INIT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEH
      COMMON/BLK1/SIZE1(40000)
      COMMON/BLK2/SIZE2(1600)
      COMMON/BLK3/STABLE(450),SIZE3(2132)
      COMMON/BLK4/SIZE4(1592)
      COMMON/BLK5/NUMB,NUMSET,NRAL,NANT(4),
     1 ALFRAN,NSX,IEF3,IEF4,IEF5,NDET,NAT,IFOM(3),IEFOM,IZRO,IXRAN,
     2 IYRAN,IRAND,HMIN,IHMIN,NSREQ,CUTT(3),EXTRA(7)
      COMMON/BLK6/ SIZE6(5400)
      COMMON/KEEP/SIZE7(625)
      CHARACTER*1 CH
C
C===> FILES:
C===> FILE NAME      LOGICAL NO.    USE
C===>
C===> NOUT(PRINT)          6        NULL
C===> NOUT                 5        PRINTER OUTPUT
C===> NTAPEB               9        MAPS FILE
C===> NTAPEE              12        SEARCH DATA
C===> NTAPEF              13        SCRATCH
C===> NTAPEG              14        INTENSITY DATA
C===> NTAPEH              16        MODEL
C===> NTAPEI              17        PEAKS
C===>
C===> BLOCK SIZES ARE AS FOLLOWS: /BLK1/  40000 WORDS
C===>                             /BLK2/   1600 WORDS
C===>                             /BLK3/   2582 WORDS
C===>                             /BLK4/   2148 WORDS
C===>                             /BLK5/     35 WORDS
C===>                             /BLK6/   5400 WORDS
```

```
C===>                                  /KEEP/     625 WORDS
C===>                                  / IO /     669 WORDS
C===>                                 /SINCOS/ 18000 WORDS
      NTAPEH=16
      NTAPEI=17
      NSYMOP=0
      PI=ATAN(1.0)*4.0
      DTOR=PI/180.0
      RTOD=1.0/DTOR
C===> SET UP SINE-COSINE LOOKUP TABLE
      DO 50 J=1,450
        ARG=(J-1)*DTOR
   50 STABLE(J)=SIN(ARG)
      IFRST=0
      NDIFF=1
      NREAD=0
      KUSE(25)=KUSE(7)+1
C===> PACKING CONSTANTS (POWERS OF 2)
      LK(1)=2
      DO 40 J=2,30
  40    LK(J)=2*LK(J-1)
      NULL=LK(27)+LK(16)
C===> CLEAR SCREEN
      CALL CLSCN
      WRITE(*,9001)
 9001 FORMAT('  ************************************************************'/
     #       '  *                    WELCOME TO THE                        *'/
     1       '  *       STRIDE CRYSTALLOGRAPHIC FOURIER PACKAGE            *'/
     #       '  *            (now with Fourier recycling)                  *'/
     2       '  ************************************************************'//)
C===> ALSO WRITE TITLE ON GRAPHICS SCREEN
      CALL TITLE2
C===> PROMPT USER FOR THE REQUIRED OPTION
  5   WRITE(*,9010)
 9010 FORMAT(/'Choose from one of the following options:'//
     1 'A : Fourier recycling '/'B : Calculate E-maps'/
     2 'C : Inspect E-maps '/'D : Quit'//'Which option ? '\)
      READ(*,9003) CH
 9003 FORMAT(A)
C===> GET THE OPTION CHOSEN
      IF(CH.EQ.'A'.OR.CH.EQ.'a') THEN
        WRITE(*,9011)
 9011   FORMAT(/'Fourier recycling chosen'/)
C===> SET MARKER FOR MAPS ROUTINE
        NELSE=2
      ELSEIF(CH.EQ.'B'.OR.CH.EQ.'b') THEN
        NELSE=1
        WRITE(*,9012)
 9012   FORMAT(/'E-map option chosen'/)
C===> INSPECT E-MAPS
      ELSEIF(CH.EQ.'C'.OR.CH.EQ.'c') THEN
        NELSE=3
        WRITE(*,9014)
 9014   FORMAT(/'Inspect E-map option chosen'/)
C===> QUIT
      ELSEIF(CH.EQ.'D'.OR.CH.EQ.'d') THEN
        CALL EXIT
      ELSE
C===> ILLEGAL INPUT OPTION
```

```
        WRITE(*,9013)
 9013   FORMAT(/' Unrecognised option - try again'/)
        GOTO 5
       ENDIF
C===> SET UP THE APPROPRIATE FILES
       CALL FILE1(NELSE)
       IF(NELSE.EQ.3)THEN
C===> READ THE MAP OFF THE FILE AND CONTOUR IT
         CALL MAPS(NELSE,NSYMOP)
         GOTO 5
       ENDIF
   98  IF(NELSE.EQ.2)CALL NORMAL(NSYMOP)
       WRITE(*,8008)
 8008  FORMAT(/'**** **** Maps program called **** ****'/)
       CALL MAPS(NELSE,NSYMOP)
C===> CHECK FOR FURTHER FOURIER RECYCLING
       IF(NELSE.EQ.2)THEN
  102    WRITE(*,8011)
 8011  FORMAT(//' Choose an option:'//'   A: Recycle fragment found in last run'/
     1 '   B: End'//'(N.B. Option A is only worthwhile if model file has been
     2 updated using UPDATE'/'    option in MITHGRAPH)'//' A or B ? - '\)
         READ(*,8012)CH
 8012    FORMAT(A)
C===> FURTHER FOURIER RECYCLING
         IF(CH.EQ.'A'.OR.CH.EQ.'a')THEN
           GOTO 98
C===> QUIT
         ELSE IF(CH.EQ.'B'.OR.CH.EQ.'b')THEN
           GOTO 99
         ELSE
C===> ILLEGAL INPUT OPTION
           WRITE(*,8013)
 8013      FORMAT(/10X,'---- Invalid option - try again ----')
         ENDIF
         GOTO 102
       ENDIF
C===>
C===> ALL DONE
C===>
   99  WRITE(*,8007)
 8007  FORMAT(/'**** **** Run Completed **** ****'/)
       CALL EXIT
       STOP
       END
C
       BLOCK DATA  INITAL
       COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
      1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
      2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFL,ZERO,LK(32),
      3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INIT,IRDY,RTOD,NULL,
      4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEH
       COMMON/BLK1/IESN(24000),IX(1000),EX(1000),FX(1000),X(200),Y(200),
      1 Z(200),NZ(200),NCG1(200),NCG2(200),LLH(60),LLK(60),LLL(60),FO(60),
      2 ID(60),ED(60),EH(60),RHO(60),SIGMA(60),JUNK(11260)
       COMMON/BLK2/IS(2,3,24),TS(3,24),P(6),CX(9),NSYM,ICENT,NORI,SIGMAQ,
      1 NEX,LATT,LAT,PTS,KSYS,IAPX,NGP,NTOT,SIGD(20),IAD1(129),IAD2(129)
      2 ,IHE(24),IKE(24),ILE(24),NSH(24),KHL(6,128),XX(3),HKL(4),NASU,NRD,
      3 RUBISH(206)
       COMMON/BLK3/STABLE(450),E(800),IFN(800),SCAL(8),FLGH(30),FLGD(30),
```

```
     1 AVR(30),DCV(50),VST(10,5),NST(5),ZT(25,5),EE(10),JUNK1(194)
       COMMON/BLK4/IOBUF(200),LEVEL1(129),LEVEL2(129),EDITF(100),
     1 IEDIT(100),TRANS(12),GIS(142),GIW(142),NINF(10),NAG(10),
     2 GPOP(10),GCELL(6,10),NCSF1(10),NCSF2(10),NASF(10),TAS(10),
     3 TAL(10),TBS(10),TBL(10),TCL(10),AL(10),AS(10),BL(10),BS(10),
     4 CL(10),NW(10),NO(10),F(10),KH(10),KK(10),KL(10),KD(10),SIG(10),
     5 NC1(10),NC2(10),NA(10),IFORM(68),CVR(10),VVR(10),TCVR(10),TVVR(10),
     6 NEWE(100),ENEW(100)
       COMMON/BLK5/NPC,NAT,NK,
     1 MAXDUP,EMAX,RHOMAX,NSF,LST,IK,NB,ISC,SLOPE,FLGK,BT,SC,EPS,
     2 DEL,KS,MULT,IND,NZR,TMUL,SINLMN,SINLMX,JUMP,NNN,NN,INEW
     3 ,NEDIT,ITRANS,MM,MZ,EN,ER,GARB
       COMMON/KEEP/N1(50),N2(50),ALT(50),AST(50),BLT(50),BST(50),CLT(50),KX(14)
     1 ,LTC(7),AVA(10),AVC(10),AVH(10),CPH(25),PCH(30),TABCVR(50),TABBVR(50),
     2 JFORM(68),JLATT
       DATA IH/'0','1','2','3','4','5','6','7','8','9','.','-','+','X',
     1 'Y','Z',',','=','A','B','C','D','E','F','G','H','I','J',
     2 'K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y',
     3 'Z',' ','TITL','NOPR','PRIN','END ','MENU','LEVE','DEFA','NORM',
     4 'CELL','SYMM','LATT','CONT','LIMI','DATA','BSCL','SFAC',
     5 'GROU','ATOM','EDIT','TRAN','RECY','NEWE','VERY','HARD',
     6 'NOCH','TRIP','MDKS','L.E.','LIST','QUAR','POSI','NEIG','LIST',
     7 'MISS','REST','QUIN','LIST','MISS','REST','CONV','SIGM',
     8 'EXCL','REMO','RELA','LIST','KMIN','MAGI','ORIG','ENAN',
     9 'PERM','KNOW','SETS','SPEC','GENE','ANY ','MAX ','CHEC','DIVE',
     # 'TANG','RANT','SWTR','SKIP','WTFO','EFOM','SERI','MARK','WEIG',
     $ 'PHAS','SETS','NOST','MAGI','STAR','WTMI','MAGE','LIST','PRIM',
     $ 'FUNC','MAGI','ALIM','KALI','WSPE','SELE','EXPA','YZAR','NREF',
     $ 'L.S.','STAR','NQES','TOP ','RAND','MAPS','GRID','PEAK',
     $ 'NOJO','PROJ','DOUT','DMIN','DMAX','AMIN','AMAX','MOLE',
     $ 'VDU ','PATT','SHOW','MODE','LSAM','SYMB','SIGN','NUMB','LIMI'/
       DATA NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,NTAPEF,
     1  NTAPEG/5,5,8,9,10,11,12,13,14/
       DATA KUSE/24000,4000,800,200,200,200,128,10,100,10,100,20,
     168,200,150,24,60,500,300, 9450,1000,900,80,2250,0,3615,100,20100,
     1  3000,10000,200,25100,8*0/
       DATA ISPCH/'!','/','**','(',',56X','   /','   /','   /','   /',
     1 '4X, ','I4) ','1X, ','I3) ',',',',( 1X',',I6,','I13,',' 7X,','3F10',
     2 '.4, ','(','ZERO','PI  ','*',6*' '/
       DATA LTC/'P','A','B','C','I','F','R'/
       DATA KX/'1','2','3','4','5','6','X','Y','Z',' ',',','+','-','/'/
C===> 50 ATOM TYPES. ATOMIC SYMBOLS ARE STORED IN N1 + N2
       DATA N1/'H','L','B','B','C','N','O','F','N','M','A','S','P',
     1 'S','C','K','C','T','V','C','M','F','C','N','C','Z','A','S',
     2 'B','R','S','Z','M','R','R','P','A','C','S','S','I','C','B',
     3 'N','O','P','A','H','P','B'/
       DATA N2/' ','I','E',' ',' ',' ',' ',' ',' ','A','G','L','I',' ',
     1 ' ','L',' ','A','I',' ','R','N','E','O','I','U','N','S','E',
     2 'R','B','R','R','O','U','H','D','G','D','N','B',' ','S','A',
     3 ' ','S','T','U','G','B','I'/
C===> S ATOMIC SCATTERING FACTORS IN THE FORM :
C===>      F = AL * EXP(-AS * RHO) + BL * EXP(-BS * RHO) + CL
C===>  SEE ACTA CRYST (1963) 16 1169
       DATA ALT/0.388,1.560,1.261,1.207,2.112,3.188,4.197,5.155,7.488,
     1 7.426,7.276,6.988,6.509,5.967,5.557,9.544,10.25,11.69,12.50,
     2 13.67,14.17,15.01,15.84,16.69,17.88,18.32,18.62,18.48,18.18,
     3 17.27,17.95,18.21,18.38,19.56,20.42,20.02,22.37,23.74,25.20,
     4 25.41,25.51,25.57,26.03,36.53,36.50,36.08,36.22,37.13,38.75,
     5 39.16/
```

```
      DATA AST/7.151,3.264,2.620,5.745,7.827,7.341,6.327,5.392,4.821,
     1 3.770,3.143,2.739,2.602,2.753,3.176,7.683,7.176,6.300,5.929,
     2 5.789,5.252,4.958,4.687,4.440,4.357,4.019,3.091,2.815,2.638,
     3 3.382,4.361,4.515,4.490,4.686,4.797,4.398,4.881,4.953,4.631,
     4 4.290,3.653,3.556,4.022,3.505,3.460,3.333,3.360,3.585,3.820,
     5 3.714/
      DATA BLT/0.601,1.059,2.008,2.530,2.462,2.305,2.218,2.172,1.280,
     1 2.267,3.192,4.169,5.107,5.925,6.573,2.843,3.086,3.419,3.482,
     2 3.131,3.554,3.573,3.578,3.558,3.088,3.501,5.729,6.744,7.710,
     3 7.397,5.919,6.578,7.602,7.598,7.360,8.892,6.844,6.153,6.411,
     4 7.297,9.410,10.22,9.230,8.741,9.627,11.22,11.46,10.38,9.179,
     5 9.659/
      DATA BST/30.18,108.3,54.77,38.23,31.65,26.84,22.83,19.61,96.71,
     1 69.72,55.67,43.28,34.46,28.44,24.26,53.75,79.64,62.92,57.74,
     2 46.70,49.62,46.50,43.88,41.79,35.63,38.46,35.40,30.87,27.32,
     3 28.90,44.38,42.30,30.85,28.20,27.65,20.92,26.38,32.06,40.81,
     4 39.07,33.09,31.70,38.86,33.47,29.80,23.06,22.12,25.33,33.99,
     5 35.23/
      DATA CLT/0.008,0.3747,0.7163,1.243,1.412,1.498,1.578,1.668,2.216,
     1 2.291,2.491,2.801,3.343,4.073,4.839,6.494,6.616,6.846,6.966,
     2 7.147,7.226,7.375,7.541,7.712,7.986,8.136,8.601,8.727,9.070,
     3 12.09,13.92,15.06,15.88,16.72,17.11,17.04,17.69,18.02,18.30,
     4 18.22,18.02,18.93,20.42,28.57,29.71,30.56,31.19,32.35,33.92,
     5 34.06/
C===> TABLE OF (VERY APPROXIMATE) COVALENT RADII
      DATA TABCVR/0.37,0.68,0.90,0.80,0.77,0.74,0.74,0.72,1.00,0.68,
     1 0.50,1.17,1.10,1.04,0.99,1.33,0.99,0.63,0.59,0.58,0.80,0.75,
     2 0.71,0.67,0.96,0.70,1.21,1.17,1.14,2.11,1.92,0.80,6*1.4,1.37,
     3 1.41,1.33,1.40,1.34,1.45,1.40,1.35,1.37,1.03,1.75,1.70/
C===> VAN DER WAALS RADII    (MOSTLY VERY APPROXIMATE)
      DATA TABBVR/1.20,3*1.70,1.70,1.55,1.40,1.47,3*1.70,2.10
     1 ,1.80,1.80,1.75,11*1.8,1.85,1.90,1.85,9*1.90,2.27,
     2 2.09,1.98,9*2.0/
C===> TABLES OF THEORETICAL DISTRIBUTIONS
      DATA AVA/0.886,1.0,1.329,2.0,3.323,6.0,0.736,1.0,2.0,2.415/
      DATA AVC/0.798,1.0,1.596,3.0,6.383,15.0,0.968,2.0,8.0,8.691/
      DATA AVH/0.718,1.0,1.916,4.5,12.26,37.5,1.145,3.5,26.0,26.903/
      DATA PCH/84.2,73.4,64.5,56.9,50.3,44.4,39.2,34.5,30.4,26.7,
     1 23.4,20.4,17.8,15.5,13.4,11.6,10.0,8.6,7.3,6.2,5.3,4.5,3.8,3.2,
     2 2.7,2.2,1.8,1.5,1.3,1.0/
      DATA CPH/0.368,0.463,0.526,0.574,0.612,0.643,0.670,0.694,0.715,
     1 0.733,0.765,0.791,0.813,0.832,0.848,0.863,0.875,0.886,0.896,
     2 0.905,0.913,0.920,0.926,0.932,0.938/
      DATA IEND,NREAD,IDEF,ICHK,MAXH,NDIFF,IDIF,IVDIF,IFLOW/33*0/
      DATA ZERO,ISYMP,SCAL,LEVEL/0.0001,1,8*0.0,2/
      DATA JFORM,ITLE,JLATT,KARLE,IHVY,IGPFL,INXT,IRDY/136*' ','P',5*0/
      END
C
      SUBROUTINE FILE1(NELSE)
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFL,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEH
      CHARACTER*12 PEAKS,DATAFL,MODEL,OPTION,NAME*9
      LOGICAL*4 FILE
      DATA NAME/'PRINT.OUT'/
      NTAPEH=16
      NTAPEI=17
```

```
C===>  FIRST SORT OUT FILES NEEDED TO INSPECT E-MAPS
       IF(NELSE.EQ.3)THEN
C===>  GET THE MAPS FILE
         WRITE(*,9030)
 9030    FORMAT(/'Enter the name of the maps file - '\)
         READ(*,9031)OPTION
 9031    FORMAT(A)
         CALL SETUP(OPTION,NTAPEB,1)
         RETURN
       ENDIF
C===>  PEAKS FILE
  100  WRITE(*,8005)
 8005  FORMAT('Enter the name of the peaks file - '\)
       READ(*,8003) PEAKS
 8003  FORMAT(A)
C===>  IF <CR> IS PRESSED SET UP A SCRATCH FILE FOR THE PEAKS
       IF(PEAKS.EQ.' ') THEN
         OPEN(UNIT=NTAPEI,STATUS='SCRATCH')
         WRITE(*,8015)
 8015    FORMAT('Scratch file used for peaks')
       ELSE
         CALL SETUP(PEAKS,NTAPEI,0)
       ENDIF
C
C===>  PRINTER OUTPUT FILE
       CALL SETUP(NAME,NOUT,0)
C
C===>  SEARCH DATA FILE
       INQUIRE(FILE='SRCH.FIL',EXIST=FILE)
       IF(FILE)THEN
         OPEN(FILE='SRCH.FIL',UNIT=NTAPEE,FORM='UNFORMATTED')
       ELSE
         OPEN(FILE='SRCH.FIL',UNIT=NTAPEE,STATUS='NEW',FORM='UNFORMATTED')
       ENDIF
C
C===>  NULL FILE FOR THE NOPRINT OPTION
       OPEN(UNIT=6,STATUS='SCRATCH')
C
       IF(NELSE.EQ.2)THEN
C
C===>  NOW THE REFLECTION FILE
         WRITE(*,9005)
 9005    FORMAT(/'Enter the name of the reflection file - '\)
         READ(*,9003) DATAFL
 9003    FORMAT(A)
         CALL SETUP(DATAFL,NTAPEG,1)
C
C===>  MODEL FILE
         WRITE(*,8007)
 8007    FORMAT(/'Enter the name of the model file - '\)
         READ(*,9003) MODEL
         CALL SETUP(MODEL,NTAPEH,1)
C
       ELSE IF (NELSE.EQ.1)THEN
C
C===>  E-MAP REQUESTED - ASSIGN THE PHASE FILE
         WRITE(*,9021)
 9021    FORMAT(/'Enter the name of the phase file - '\)
         READ(*,9022) OPTION
```

```
 9022    FORMAT( A)
         CALL SETUP( OPTION, NTAPED, 1)
C
       ENDIF
C===>  SCRATCH FILE
       OPEN( UNIT=NTAPEF, STATUS=' SCRATCH' , FORM=' UNFORMATTED' )
       RETURN
       END
C
       SUBROUTINE SETUP( NAME, UNITNO, MUST)
C===>  ROUTINE TO CREATE AND/OR OPEN FILES
C===>  MUST=1 IF A FILE MUST EXIST FOR PROGRAM TO PROCEED
       CHARACTER*12 NAME, CH*1
       INTEGER UNITNO
       LOGICAL*4 FILE
C===>  SEE WHETHER FILE EXISTS OR NOT
 100   INQUIRE( FILE=NAME, EXIST=FILE)
       IF( FILE) THEN
         OPEN( FILE=NAME, UNIT=UNITNO)
       ELSE
C===>  FILE DOES NOT EXIST.  CHECK WHETHER IT IS AN ESSENTIAL FILE
         IF( MUST. EQ. 1)  THEN
C===>  CANNOT CONTINUE WITHOUT THIS FILE.  GIVE USER ANOTHER CHANCE TO
C===>  GET THE NAME CORRECT
           WRITE( *, 1000)
 1000      FORMAT( /' File does not exist.  Do you wish to try again ? (y/n) - '
     1\)
           CALL YN( NN)
           IF( NN. EQ. 1)  CALL EXIT
           WRITE( *, 1001)
 1001      FORMAT( /' Please enter file name - '\)
           READ( *, 1002) NAME
 1002      FORMAT( A)
           GOTO 100
         ELSE
C===>  SET UP NEW FILE
           OPEN( FILE=NAME, UNIT=UNITNO, STATUS=' NEW' )
         ENDIF
       ENDIF
       RETURN
       END
C
       SUBROUTINE FINDCH
C===>  SCANS A LINE FOR CHARACTERS (A-Z ONLY) AND STORES THEM AS
C===>  TWO CHARACTER ELEMENTS IN ICH1 AND ICH2
C===>  USED TO SCAN CARDS WHERE CHARACTER INFORMATION IS IMPORTANT
C===>  USED IN CONJUNCTION WITH DEC WHICH IS THE MAIN DECODE ROUTINE
       COMMON/IO/NIN, NOUT, NTAPEA, NTAPEB, NTAPEC, NTAPED, NTAPEE,
      1 NTAPEF, NTAPEG, IH( 155), A( 200), ICH1( 10), ICH2( 10), ITLE( 68), IR( 68),
      2 ICALL, NRC, NCH, NREAD, IEND, NREF, PI, DTOR, MAXH, ISPFL, ZERO, LK( 32),
      3 IDEF, ICHK, ISYMP, LEVEL, IFLOW( 25), KUSE( 40), INIT, IRDY, RTOD, NULL,
      4 IDIF, IVDIF, KARLE, IHVY, IGPFL, NDIFF, ISPCH( 30), NTAPEH
       CHARACTER*4 ICH1, ICH2, IR, IH
       EQUIVALENCE( KUSE13, KUSE( 13))
       NCH=0
       J=1
  100  DO 130 K=19, 44
       IF( IR( J). NE. IH( K))  GOTO 130
       IF( J. EQ. KUSE13)  GOTO 120
```

```
      DO 140 L=19,44
       IF(IR(J+1).EQ.IH(L)) GOTO 150
  140 CONTINUE
C===> SECOND CHARACTER IS NOT A-Z
       GOTO 120
  130 CONTINUE
       J=J+1
       IF(J.GT.KUSE13) RETURN
       GOTO 100
C===> JUST ONE CHARACTER IN STRING
  120 NCH=NCH+1
       ICH1(NCH)=IR(J)
       ICH2(NCH)=IH(45)
       IF(NCH.EQ.KUSE(12)) RETURN
       J=J+1
       IF(J.GT.KUSE13) RETURN
       GOTO 100
C===> TWO CHARACTERS
  150 NCH=NCH+1
       ICH1(NCH)=IR(J)
       ICH2(NCH)=IR(J+1)
       J=J+2
       IF(J.GT.KUSE13) RETURN
       IF(NCH.EQ.KUSE(12)) RETURN
       GOTO 100
       END
       SUBROUTINE DEC(MODE)
C===> FREE FORMAT LINE DECODING
C===> IF MODE=0 THEN THE FIRST 4 CHARACTERS ARE A COMMAND
C===> IF MODE=1 THEN NO COMMAND IS PRESENT IN FIRST 4 CHARACTERS
       COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
      1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
      2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH    ,ISPFL,ZERO,LK(32),
      3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
      4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEH
       COMMON/BLK2/IS(2,3,24),TS(3,24),P(6),CX(9),NSYM,ICENT,NORI,SIGMAQ,
      1 NEX,LATT,LAT,PTS,KSYS,IAPX,NGP,NTOT,NASU,NRD,PAD2(1355)
       COMMON/BLK4/IOBUF(200),LEVEL1(129),LEVEL2(129),EDITF(100),
      1 IEDIT(100),TRANS(12),GIS(142),GIH(142),NINF(10),NAG(10),
      2 GPOP(10),GCELL(6,10),NCSF1(10),NCSF2(10),NASF(10),TAS(10),
      3 TAL(10),TBS(10),TBL(10),TCL(10),AL(10),AS(10),BL(10),BS(10),
      4 CL(10),NH(10),NO(10),F(10),KH(10),KK(10),KL(10),KD(10),SIG(10),
      5 NC1(10),NC2(10),NA(10),IFORM(68),CVR(10),VVR(10),TCVR(10),TVVR(10),
      6 NEWE(100),ENEW(100)
       COMMON/KEEP/N1(50),N2(50),ALT(50),AST(50),BLT(50),BST(50),CLT(50),KX(14)
      1 ,LTC(7),AVA(10),AVC(10),AVH(10),CPH(25),PCH(30),TABCVR(50),TABBVR(50),
      2 JFORM(68),JLATT
       CHARACTER*4 IR,ICALL,ITLE,IH,ISPCH,MMM,KX
       EQUIVALENCE (KUSE13 ,KUSE(13)),(KUSE14,KUSE(14)),
      1 (KUSE15,KUSE(15)),(KUSE16,KUSE(16))
       IF(IEND.EQ.-1) IEND=0
       IF(INXT.EQ.0) GOTO 3
       INXT=0
       RETURN
   3   L=73
       NRC=0
       DO 25 J=1,KUSE14
   25  A(J)=0.0
C===> PROMPT USER TO ENTER COMMAND
```

```
      IF(ISPFL.EQ.0)WRITE(*,8090)
 8090 FORMAT('COMMAND:   '\)
C===> MODEL FILE - READ DATA OFF NTAPEH
      IF(NIN.EQ.16)THEN
      READ(NIN,9001,END=120,ERR=129 ) ICALL,IR
C===> ALL OTHER INSTRUCTIONS FROM KEYBOARD
      ELSE
      IF(MODE.EQ.0)  READ(*,9001,END=120,ERR=129 ) ICALL,IR
      IF(MODE.EQ.1)  READ(*,8001,END=120,ERR=129) IR
      ENDIF
 9001 FORMAT(A4,68A1)
 8001 FORMAT(68A1)
      IF(MODE.EQ.1) GOTO 160
C===> FIRST CHECK FOR ABORT CODE (X)
      IF(ICALL.EQ.IH(14)) GOTO 99
C===> NOW CHECK FOR OTHER OPTIONS DEALT WITH IN THIS ROUTINE
      IF(ICALL.EQ.IH(45)) GOTO 101
      IF(ICALL.EQ.IH(46)) GOTO 100
      IF(ICALL.EQ.IH(47)) GOTO 110
      IF(ICALL.EQ.IH(48)) GOTO 112
      IF(ICALL.EQ.IH(49)) GOTO 125
      IF(ICALL.EQ.IH(50)) GOTO 121
      IF(ICALL.EQ.IH(52)) GOTO 180
      IF(ICALL.EQ.IH(149)) GOTO 900
      IF(ICALL.EQ.IH(55)) GOTO 800
      IF(ICALL.EQ.IH(150)) GOTO 811
      IF(ICALL.EQ.IH(146)) GOTO 813
      GOTO 135
C===> INPUT ERROR
  129 IF(MODE.EQ.0)WRITE(*,9070) ICALL,IR
 9070 FORMAT(/'Read error on input command:'//A4,68A1//
     1'Input command has been ignored'/)
      IF(MODE.EQ.1) WRITE(*,9071) IR
 9071 FORMAT(/'Read error. Input line reads:'//68A1//
     1'Input command has been ignored'/)
      GOTO 3
C===> A BLANK LINE-HOW WE TREAT IT DEPENDS ON INTERACTION LEVEL
  101 IF(LEVEL.EQ.0.OR.ISPFL.EQ.1) GOTO 3
      IEND=-1
      INXT=0
      NREAD=45
      RETURN
C===> TITLE
  100 DO 140 J=2,KUSE13
  140 ITLE(J)=IR(J)
      GOTO 3
C===> NOPRINT
 110  NOUT=6
      GOTO 3
C===> PRINT
  112 NOUT=5
      GOTO 3
C===> END OF FILE ENCOUNTERED
  120 IF(ISPFL.NE.1) GOTO 125
C===> END OF FILE ON MODEL FILE
      NIN=5
      ISPFL=0
      GOTO 3
C===> DEFAULTS CARD
```

```
  180 IDEF=1
      NREAD=52
      RETURN
C===> END COMMAND ENCOUNTERED
  125 IEND=1
      NREAD=52
      RETURN
C===> MENU INSTRUCTION
  121 NREAD=50
      INXT=0
      RETURN
C===> SCAN THE MODEL FILE
  811 ISPFL=1
      NIN=NTAPEH
      REWIND NTAPEH
      GOTO 3
C===> SHOW PARAMETERS
  900 CALL CLSCN
      WRITE(*,9050) IDIF,IVDIF,LEVEL
 9050 FORMAT('The hard structure flag is ',I1/'The very hard flag is',
     1 I2/'The level parameter is',I2/'(1=ON 0=OFF for flags)')
      IF(NOUT.EQ.1) WRITE(*,9051)
 9051 FORMAT('The printing is switched OFF'/)
      IF(NOUT.NE.1) WRITE(*,9052)
 9052 FORMAT('The printing is switched ON'/)
      GOTO 3
C===> MOLECULE COMMAND DECODED IN SEARCH
  813 NREAD=146
      RETURN
C===> DETERMINE PROGRAMME CALLED
  135 DO 150 J=51,KUSE15
      JX=J
      IF(ICALL.EQ.IH(J)) GOTO 160
  150 CONTINUE
C===> INSTRUCTION NOT AVAILABLE-THIS IS O.K. IF READING MODEL FILE
      IF(ISPFL.EQ.1) GOTO 3
      WRITE(*,9011)
 9011 FORMAT('**** Unrecognised instruction ****'/)
      GOTO 3
  160 NREAD=JX
C===> DECODE NUMERICAL PORTIONS OF THE CARD
      N=0
      ASSIGN 16 TO JJ
      GOTO 24
    6 W=1.0
    7 V=0.0
      NB=0
      Y=1.
      U=10.0
      Z=1.
      GOTO 10
    8 Z=Y*Z
      V=U*ABS(V)+Z*X
      NB=1
      IF(V) 9,10,9
    9 V=SIGN(V,W)
      W=V
   10 N=N+1
      K=10
```

```
      IF(KUSE13-N) 15,11,11
   11 X=0.
      IF(IR(N).EQ.ISPCH(1)) GOTO 34
      DO 12 M=1,10
C===> STOP SCANNING AFTER A ! SIGN
      IF(IR(N).EQ.IH(M)) GOTO 8
   12 X=X+1.0
      GOTO 14
   13 IF(IR(N).EQ.IH(K+9)) GOTO 15
   14 K=K-1
      IF (K-2) 15,13,13
   15 GOTO JJ,(16,26)
   16 GOTO (10,17,31,31,18,18,18,19,21,19),K
   17 U=1.
      Y=0.1
      GOTO 10
   18 K=K+NJ
      GOTO 6
   19 L=L+1
      NJ=NJ+3
      IF(L-NJ-9) 20,6,6
   20 IN1=IN1+12
      IF(L-12-IN1) 22,3,22
   21 IF(ISPFL.EQ.0)WRITE(*,8091)
 8091 FORMAT('CONTINUE: '\)
      IF(MODE.EQ.0) READ(*,9001,END=120,ERR=129) MMM,IR
      IF(MODE.EQ.1) READ(*,8001,END=120,ERR=129) IR
      N=0
      IF(IH(45).EQ.MMM) GOTO 6
   22 IN1=1
   24 ASSIGN 26 TO JJ
      GOTO 6
   26 IF(K-2) 27,17,27
   27 NRC=NRC+NB
      IF(KUSE14-NRC) 22,28,28
   28 IF(-NRC) 29,30,30
   29 A(NRC)=V+A(NRC)
   30 IF(K-9) 32,21,34
   31 CONTINUE
   32 IF(K-3) 6,33,6
   33 N=-1.
      GOTO 7
   34 IF(MODE.EQ.1) RETURN
      IF(NREAD.EQ.51) GOTO 122
      IF(NREAD.EQ.68) GOTO 123
      IF(NREAD.EQ.69) GOTO 124
      RETURN
C===> VERY HARD STRUCTURE
  123 IF(NRC.EQ.0) IVDIF=IVDIF+1
      IF(NRC.GT.0) IVDIF=IABS(INT(A(1)))
      IF(IVDIF.GT.1) IVDIF=0
      GOTO 3
C===> HARD STRUCTURE
  124 IF(NRC.EQ.0) IDIF=IDIF+1
      IF(NRC.GT.0) IDIF=IABS(INT(A(1)))
      IF(IDIF.GT.1) IDIF=0
      GOTO 3
C===> A LEVEL CARD - CHANGE THE LEVEL KEY
  122 LEVEL=IABS(INT(A(1)))
C===> SELECT OUTPUT DEVICE
```

```
      GOTO 3
C===> SYMMETRY CARD
  800 NSYM=NSYM+1
      IF(NSYM.GT.KUSE16) THEN
      WRITE(*,8003)
 8003 FORMAT(/'Too many symmetry commands - run aborted')
      CALL EXIT
      ENDIF
C===> IF THE SYMM CARD IS BLANK THEN RESET THE SYMMETRY OPERATIONS
      DO 812 J=1,KUSE13
      IF(IR(J).NE.IH(45)) GOTO 814
  812 CONTINUE
C===> CARD IS BLANK
      NSYM=1
      WRITE(*,9005)
 9005 FORMAT('Blank symmetry card-all previous symmetry cards ignored')
      GOTO 3
  814 IC=1
      ISN=1
      TRV=-1.0
      IN=1
C===> FIND FIRST SPACE
      DO 810 J=1,KUSE13
      JX=J
      IF(IR(J).EQ.IH(45)) GOTO 805
      IF(IR(J).EQ.IH(17))GOTO 805
  810 CONTINUE
C===> COULD BE A PROBLEM ON THIS CARD
      JX=5
      WRITE(*,8006)
 8006 FORMAT(/'Undecipherable input command')
  805 JJ=KUSE13+1
      DO 890 I=JX,JJ
      IF(I.EQ.JJ) GOTO 830
      DO 815 J=1,14
      IF(IR(I).NE.KX(J)) GOTO 815
      JGO=J
      GOTO 820
  815 CONTINUE
      WRITE (*,8071)
 8071 FORMAT(/'Undecodable SYMM command')
      CALL EXIT
  820 GOTO(870,870,870,870,870,870,850,850,850,890,830,890,840,890),JGO
C===> COMMA
  830 IF(IN.EQ.2) IS(2,IC,NSYM)=0
      IF(TRV.LT.0.0) TS(IC,NSYM)=0.0
      IN=1
      TRV=-1.0
      IC=IC+1
      GOTO 890
C===> MINUS SIGN
  840 ISN=-1
      GOTO 890
C===> X OR Y OR Z
  850 IS(IN,IC,NSYM)=ISN*(J-6)
      ISN=1
      IN=IN+1
      GOTO 890
C===> NUMBER 1-6
  870 IF(TRV.LT.0.0) TRV=FLOAT(J)
```

```
      IF(TRV.GT.0.0)  TS(IC,NSYM)=TRV/FLOAT(J)
  890 CONTINUE
C===> CHECK THAT THE IDENTITY OPERATION HAS NOT BEEN INPUT
      DO 895 J=1,3
      IF(IS(1,J,NSYM).NE.J)  GOTO 3
      IF(IS(2,J,NSYM).NE.0)  GOTO 3
  895 CONTINUE
      NSYM=NSYM-1
      GOTO 3
C===> ABORT
   99 WRITE(*,9004)
 9004 FORMAT (/20X,'****** User abort ******'/)
      CALL EXIT
      END
C
      SUBROUTINE NORMAL(NSYMOP)
C===>
C===>  N    N  00000  RRR     M        M       A       L
C===>  NN   N  O   O  R  R    MM      MM      A A      L
C===>  N N N  O   O  RRRR    M M    M M     A   A     L
C===>  N  NN  O   O  R R     M  M M  M    AAAAAAA    L
C===>  N   N  00000  R  R    M   M   M    A     A    LLLLL
C===>*
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFL,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEH
      COMMON/BLK1/IESN(24000),IX(1000),EX(1000),FX(1000),X(200),Y(200),
     1 Z(200),NZ(200),NCG1(200),NCG2(200),LLH(60),LLK(60),LLL(60),
     2 FO(60),ID(60),ED(60),EW(60),RHO(60),SIGMA(60),JUNK(11260)
      COMMON/BLK2/IS(2,3,24),TS(3,24),P(6),CX(9),NSYM,ICENT,NORI,SIGMAQ,
     1 NEX,LATT,LAT,PTS,KSYS,IAPX,NGP,NTOT,SIGD(20),IAD1(129),IAD2(129)
     2 ,IHE(24),IKE(24),ILE(24),NSH(24),KHL(6,128),XX(3),HKL(4),NASU,NRD,
     3 RUBISH(206)
      COMMON/BLK3/STABLE(450),E(800),IFN(800),SCAL(8),FLGW(30),FLGD(30),
     1 AVR(30),DCV(50),VST(10,5),NST(5),ZT(25,5),EE(10),JUNK1(194)
      COMMON/BLK4/IOBUF(200),LEVEL1(129),LEVEL2(129),EDITF(100),
     1 IEDIT(100),TRANS(12),GIS(142),GIW(142),NINF(10),NAG(10),
     2 GPOP(10),GCELL(6,10),NCSF1(10),NCSF2(10),NASF(10),TAS(10),
     3 TAL(10),TBS(10),TBL(10),TCL(10),AL(10),AS(10),BL(10),BS(10),
     4 CL(10),NH(10),NO(10),F(10),KH(10),KK(10),KL(10),KD(10),SIG(10),
     5 NC1(10),NC2(10),NA(10),IFORM(68),CVR(10),VVR(10),TCVR(10),TVVR(10),
     6 NEWE(100),ENEW(100)
      COMMON/BLK5/NPC,NAT,NK,
     1 MAXDUP,EMAX,RHOMAX,NSF,LST,IK,NB,ISC,SLOPE,FLGK,BT,SC,EPS,
     2 DEL,KS,MULT,IND,NZR,TMUL,SINLMN,SINLMX,JUMP,NNN,NN,INEW
     3 ,NEDIT,ITRANS,MM,MZ,EN,ER,GARB
      COMMON /BLK6/ IRR(68),FILLER(5332)
      COMMON/KEEP/N1(50),N2(50),ALT(50),AST(50),BLT(50),BST(50),CLT(50),KX(14)
     1 ,LTC(7),AVA(10),AVC(10),AVH(10),CPH(25),PCH(30),TABCVR(50),TABBVR(50),
     2 JFORM(68),JLATT
      DIMENSION MAXHKL(3)
      EQUIVALENCE(LK7,LK(7)),(LK8,LK(8)),(LK16,LK(16)),(KUSE4,KUSE(4))
     1 ,(KUSE(8),KUSE8),(KUSE9,KUSE(9)),(KUSE13,KUSE(13)),
     2 (KUSE(10),KUSE10)
C===> ENTRY POINT INTO THE PACKAGE. IT NORMALISES STRUCTURE FACTORS
C===> AND PREPARES FILES FOR KARLE OR WEIGHTED FOURIER RECYCLING
C===> NDIFF=0 FOR USUAL NORMAL
C===>       =1 FOR FOURIER RECYCLING
```

```
C===>       =2 FOR KARLE RECYCLING
C===>
C
C===> ==== THIS VERSION ONLY DEALS WITH FOURIER RECYCLING ==== ====
C
      IFLAG=0
C===> ==== INITIALISE VARIABLES  (N.B. CAN'T USE BLOCK DATA STATEMENT AS
C===> ====   CODE IS RE-ENTRANT)
      DO 10 J=1,24
      DO 5 I=1,3
      TS(I,J)=0.0
      DO 3 K=1,2
      IS(K,I,J)=0
    3 CONTINUE
    5 CONTINUE
   10 CONTINUE
      DO 2 J=1,3
    2 IS(1,J,1)=J
      DO 15 J=1,129
      LEVEL1(J)=0
      LEVEL2(J)=0
   15 CONTINUE
      NREF=0
      RHOMAX=0.0
      NSYM=1
      LST=0
      EMAX=8.1
      NTOT=0
      MM=800
      MZ=100
      EN=1.0
      ER=0.3
      ITRANS=0
      NEDIT=0
      NK=1
      NAT=1
      JUMP=-1
      BT=0.0
      SINLMX=100.0
      SINLMN=0.0
      NNN=0
      NGP=0
      NSF=0
      INEH=0
      IAPX=1
      ICENT=1
      CX(1)=-10.0
      LATT=JLATT
      DO 18 J=1,68
      IFORM(J)=JFORM(J)
   18 CONTINUE
      CALL MENU1
  104 IK=0
      MAXDUP=50
      JUMP=1
      EN=0.0
      ER=0.0
  109 BT=4.5
      NTAPEH=16
```

```
      SCAL(1)=1.0
      SC=1.0
      ISC=1
      NB=1
C===> SCRATCH FILE
      OPEN(UNIT=NTAPEF,STATUS='SCRATCH',FORM='UNFORMATTED')
C===> OTHER INITIALISATIONS
 108  NC1(1)=IH(21)
      NC2(1)=IH(45)
      NC1(2)=IH(26)
      NC2(2)=IH(45)
      IF(IDEF.EQ.1) GOTO 100
  105 CALL DEC(0)
  107 IF(IDEF.EQ.1) GOTO 100
      IF(IEND.NE.0) GOTO 100
      IF(NREAD.EQ.50) CALL MENU1
      IF(NREAD.EQ.50) GOTO 105
      IF(NREAD.EQ.74) GOTO 176
      I =NREAD-52
      IF(I .LE.18) GOTO 112
C===> SET A FLAG TO SIGNIFY THAT THE NEXT CARD IS ALREADY DECODED
      INXT=1
      GOTO 100
  112 GOTO(104,110,105,120,130,140,150,160,170,180,179,190,200,100,
     1 145,105,105,175),I
C===> CELL CARD (CELL)
  110 DO 111 J=1,6
  111 CX(J)=A(J)
      IF(NRC.LT.3) CALL STOPP(26,1,1)
      IF((A(4)+A(5)+A(6)).GT.5.0) GOTO 21
C===> COSINES INPUT
      DO 22 J=4,6
   22 CX(J)=RTOD* ACOS(CX(J))
C===> VOLUME AND RECIPROCAL CELL FUNCTIONS
  21  DO 20 J=4,6
      IF(CX(J).LT.0.01) CX(J)=90.0
  20  CONTINUE
      ARG=1.0
      DO 25 I=4,6
      CX(I+3)=SIN(DTOR*CX(I))
      CX(I)=COS(DTOR *CX(I))
   25 ARG=ARG-CX(I)*CX(I)
      V=SQRT(ARG+2.0*CX(4)*CX(5)*CX(6))
      VV=CX(1)*CX(2)*CX(3)*V
      V=1.0/(V*CX(1)*CX(2)*CX(3))
      P(1)=CX(2)*CX(3)*CX(7)*V
      P(2)=CX(1)*CX(3)*CX(8)*V
      P(3)=CX(1)*CX(2)*CX(9)*V
      P(4)=0.5*P(1)*P(2)*(CX(4)*CX(5)-CX(6))/(CX(7)*CX(8))
      P(5)=0.5*P(1)*P(3)*(CX(4)*CX(6)-CX(5))/(CX(7)*CX(9))
      P(6)=0.5*P(2)*P(3)*(CX(5)*CX(6)-CX(4))/(CX(8)*CX(9))
      DO 40 I=1,3
      P(I)=0.25*P(I)*P(I)
  40  CX(I+3)=180.0*ATAN2(CX(I+6),CX(I+3))/PI
C===> CHECK CELL IS REASONABLE
      I=0
      DO 60 J=1,3
      IF(CX(J).LT.5.0.OR.CX(J).GT.60.0) I=1
      IF(CX(J+3).LT.60.0.OR.CX(J+3).GT.130.0) I=1
```

```
   60  CONTINUE
       IF(I.EQ.1) CALL STOPP(18,0,1)
       GOTO 105
C===>  LATTICE CARD (LATTICE,LATT)
  120  CALL FINDCH
       IF(NCH.EQ.0) GOTO 105
       IF(ICH1(1).EQ.IH(19)) ICENT=0
       IF(NCH.EQ.1) GOTO 105
       LATT=ICH1(2)
       GOTO 105
C===>  CELL CONTENTS (CONTENTS,CONT)
  130  CALL FINDCH
       IF(NCH.NE.NRC) CALL STOPP(3,0,1)
       IF(NRC.EQ.0.OR.NCH.EQ.0) GOTO 105
       IF(NRC.GT.KUSE8) CALL STOPP(21,0,1)
       NRC=MIN0(NRC,KUSE8)
       IAPX=0
       DO 1610 J=1,NRC
       NC1(J)=ICH1(J)
       NC2(J)=ICH2(J)
 1610  NA(J)=INT(A(J))
       NK=NRC
       GOTO 105
C===>  SFAC CARD (SFAC)
  170  IF(NRC.GT.7) CALL STOPP(5,0,1)
       NSF=NSF+1
       NCSF1(NSF)=ICH1(1)
       NCSF2(NSF)=ICH2(1)
       TAS(NSF)=A(1)
       TAL(NSF)=A(2)
       TBS(NSF)=A(3)
       TBL(NSF)=A(4)
       TCL(NSF)=A(5)
C===>  COVALENT AND VAN DER WAALS RADII
       TCVR(NSF)=A(6)
       TVVR(NSF)=A(7)
       IF(TCVR(NSF).LT.ZERO) TCVR(NSF)=0.8
       IF(TVVR(NSF).LT.ZERO) TVVR(NSF)=2.0
       GOTO 105
C===>  FULL E-LIST NEEDED
  176  LST=1
       GOTO 105
C===>  LIMITS CARD (LIMIT,LIMITS,LIMI)
  140  IFLAG=0
       IF(A(3).LE.ZERO) A(3)=9.9
       SINLMN=A(1)
       SINLMX=A(2)
       EMAX=AMIN1(A(3),9.9)
       IF(SINLMN.GT.ZERO.OR.SINLMX.GT.ZERO) IFLAG=1
       IF(SINLMX.LT.ZERO) SINLMX=100.0
       GOTO 105
C===>  TEMPERATURE AND SCALE FACTOR(S) (BSCL)
C===>  SCALE CARD IGNORED IF NO NORMALISATION REQUESTED (IK=-1)
  160  IF(IK.EQ.-1) GOTO 105
       BT=A(1)
       DO 165 J=1,8
  165  SCAL(J)=A(J+1)
C===>  SET JUMP
C===>  JUMP=0 B SUPPLIED-CALCULATE SCALE
C===>  JUMP=1 B AND SCALE SUPPLIED
C===>  JUMP=-1 CALCULATE BOTH B AND SCALE
```

```
      IF(SCAL(1).GT.ZERO) GOTO 166
C===> NO SCALE FACTOR INPUT
      IF(BT.GT.(-ZERO).AND.BT.LT.ZERO) GOTO 105
      JUMP=0
      IK=0
      GOTO 105
C===> ONE OR MORE SCALE FACTORS INPUT
  166 JUMP=1
      IF(SCAL(2).LE.ZERO) ISC=1
      IF(SCAL(2).GT.ZERO) ISC=0
      GOTO 105
C===> OVERIDE E-MAGNITUDE CALCULATED BY ECAL (NEWE)
  145 INEW=INEW+1
      IF(INEW.GT.KUSE9) GOTO 146
      IF(A(1).LT.0.0) GOTO 147
      ENEW(INEW)=A(4)
      NEWE(INEW)=INT(A(1))*LK16+(INT(A(2))+LK7)*LK8
     1 +INT(A(3))+LK7
      GOTO 105
C===> TOO MANY NEWE CARDS
  146 INEW=KUSE9
      CALL STOPP(30,0,1)
      GOTO 105
C===> NEGATIVE H NOT PERMITTED HERE
  147 CALL STOPP(31,0,1)
      INEW=INEW-1
      GOTO 105
C===> EDIT CARD (EDIT)
  190 NEDIT=NEDIT+1
      IF(NEDIT.LE.KUSE9) GOTO 191
C===> TOO MANY EDIT CARDS
      CALL STOPP(14,0,1)
      GOTO 105
  191 EDITF(NEDIT)=A(4)
      IEDIT(NEDIT)=(INT(A(1))+LK7)*LK16+(INT(A(2))+LK7)*LK8
     1 +INT(A(3))+LK7
      GOTO 105
C===> TRANS CARD (TRANS,TRAN)
  200 IF(NRC.LT.12)CALL STOPP(10,0,0)
      ITRANS=1
      DO 205 J=1,12
  205 TRANS(J)=A(J)
      GOTO 105
C===> INTENSITY DATA FORMAT CARD (DATA)
  150 IF(NRC.EQ.0) GOTO 105
      NNN=INT(A(1))
      IF(NNN.LE.0) GOTO 105
C===> FIND FIRST OPEN BRACKET
      DO 156 J=1,KUSE13
      JX=J
      IF(IR(J).EQ.ISPCH(21)) GOTO 157
  156 CONTINUE
      ISTOP=0
      CALL STOPP(23,ISTOP,1)
      GOTO 105
  157 DO 159 K=JX,KUSE13
  159 IFORM((K-JX)+1)=IR(K)
C===> DATA CARD MUST BE LAST CARD IF DATA FOLLOW ON CARDS
      GOTO 100
C===> ATOM CARD BUT IN THE WRONG PLACE
C===> ASSUME THAT A WEIGHTED FOURIER/KARLE RECYCLING/TYPE=2 REQUIRED
```

```
  179 NGP=NGP+1
      IF(NGP.LE.KUSE10) GOTO 188
C===> TOO MANY GROUPS
      CALL STOPP(12,0,1)
      GOTO 181
C===> FIX THE GROUP TYPE
  188 IF(NDIFF.EQ.1) NINF(NGP)=6
      IF(NDIFF.EQ.2) NINF(NGP)=5
      IF(NDIFF.EQ.0) NINF(NGP)=2
      GPOP(1)=0.0
      IRD=0
      DO 151 J=1,6
  151 GCELL(J,1)=0.0
      NGP=1
      GOTO 152
C===> GROUP CARD (PARAMETERS: TYPE, POPULATION, CELL)  (GROUP, GROU)
  180 NGP=NGP+1
      IF(NGP.LE.KUSE10) GOTO 182
C===> TOO MANY GROUPS
      CALL STOPP(12,0,1)
      GOTO 181
  182 NINF(NGP)=INT(A(1))
      IF(NDIFF.EQ.1) NINF(NGP)=6
      IF(NDIFF.EQ.2.AND.NINF(NGP).EQ.0) NINF(NGP)=5
      IF(IABS(NINF(NGP)).EQ.3.OR.IABS(NINF(NGP)).EQ.4) IGPFL=1
      GPOP(NGP)=A(2)
      NAG(NGP)=0
      DO 185 J=1,6
      GCELL(J,NGP)=A(J+2)
      IF(J.LE.3) GOTO 185
      IF(GCELL(J,NGP).LE.ZERO) GCELL(J,NGP)=90.0
  185 CONTINUE
C===> GET THE ATOMS FOR THIS GROUP (ATOM)
  181 CALL DEC(0)
C===> CHECK FOR A BRACKET
  152 ISS=0
      DO 187 I=1,KUSE13
      IF(IR(I).EQ.ISPCH(21)) ISS=1
  187 CONTINUE
      IF(NREAD.NE.63) GOTO 107
      IF(NGP.GT.KUSE10) GOTO 181
C===> ATOM CARD (ATOM)
      CALL FINDCH
      NTOT=NTOT+1
      IF(NTOT.LE.KUSE4) GOTO 186
      CALL STOPP(13,0,1)
      NTOT=KUSE4
      GOTO 181
  186 IF(NCH.EQ.0) GOTO 189
      NCG1(NTOT)=ICH1(1)
      NCG2(NTOT)=ICH2(1)
      GOTO 183
C===> NO ATOM NAME ASSUME C
  189 CALL STOPP(6,0,1)
      NCG1(NTOT)=IH(21)
      NCG2(NTOT)=IH(45)
C===> NOW LOAD THE ATOM COORDINATES
  183 IF(NRC.GE.3) GOTO 184
      CALL STOPP(7,0,1)
      NRC=3
  184 X(NTOT)=A(ISS+1)
```

```
      Y(NTOT)=A(ISS+2)
      Z(NTOT)=A(ISS+3)
      NAG(NGP)=NAG(NGP)+1
      GOTO 181
C===> TURN OFF INPUT CHECKING (NOCHECK)-NOT ALLOWED IN DIFFICULT CASES
  175 IF(IDIF.EQ.1.OR.IVDIF.EQ.1) GOTO 105
      ICHK=1
      GOTO 105
C===> ALL PARAMETERS NOW INPUT
  100 IF(CX(1).LT.ZERO) CALL STOPP(16,1,1)
      IF(IAPX.EQ.0) GOTO 413
C===> NO VALID CONTENTS CARD
      J=0.0645*VV
      NA(1)=(J/NSYM+1)*NSYM
      NA(2)=2*NA(1)
      NK=2
C===> DETERMINE LATTICE MULTLIPLICITY
  413 DO 412 J=1,7
      JX=J
      IF(LATT.EQ.LTC(J)) GOTO 414
  412 CONTINUE
      CALL STOPP(24,0,1)
      JX=1
  414 LAT=JX
      PTS=1.0
      IF(LATT.EQ.LTC(7)) GOTO 300
C===> A,B,C OR I
      IF(LATT.NE.LTC(1)) PTS=2.0
C===> F
      IF(LATT.EQ.LTC(6)) PTS=4.0
      GOTO 400
  300 DO 310 I=1,NSYM
C===> PRIMITIVE RHOMBOHEDRAL
      IF(IABS(IS(1,1,J)).EQ.3) GOTO 400
  310 CONTINUE
C===> CENTRED HEXAGONAL
      PTS=3.0
C===> DETERMINE CRYSTAL SYSTEM
  400 DO 450 I=1,2
      DO 440 J=1,NSYM
      DO 430 IN=1,2
      IF(IABS(IS(IN,1,J)).EQ.4-I) GOTO 460
  430 CONTINUE
  440 CONTINUE
  450 CONTINUE
C===> 1,2,3 FOR TRICLINIC,MONOCLINIC,ORTHORHOMBIC
      KSYS=MIN0(NSYM,3)
      GOTO 500
  460 IF(I.EQ.1) GOTO 480
C===> TETRAGONAL
      KSYS=4
      IF(NSYM.EQ.4.OR.NSYM.EQ.8) GOTO 500
C===> TRIGONAL OR RHOMBOHEDRAL ON HEXAGONAL AXES
      KSYS=5
C===> TEST FOR 6-FOLD AXIS (-X,-Y,Z+T)
      DO 470 J=2,NSYM
      IF(IABS(IS(1,1,J)).GT.1.OR.IABS(IS(2,1,J)).GT.1) GOTO 470
      IF(IABS(IS(1,2,J)).EQ.1.OR.IABS(IS(2,2,J)).EQ.1) GOTO 470
C===> HEXAGONAL
      KSYS=6
```

```
  470 CONTINUE
      GOTO 500
C===> CUBIC
  480 KSYS=8
C===> PRIMITIVE RHOMBOHEDRAL
      IF(LATT.EQ.LTC(7)) KSYS=7
  500 NN=IFIX(PTS)
      WRITE(NOUT,9120) ITLE
 9120 FORMAT('1'//25X,68A1)
      IF(IK.NE.1) WRITE(NOUT,9150)
 9150 FORMAT(/28X,'Normalisation by least squares straight line thr
     1ough Debye curve')
      IF(LST.EQ.1) WRITE(NOUT,9160)
 9160 FORMAT(/45X,'List complete set of E-values')
      IF(LST.NE.1) WRITE(NOUT,9170)
 9170 FORMAT(/41X,'No listing of complete set of E-values')
      IF(ISC.EQ.1) WRITE(NOUT,9180)
      IF(IFLAG.EQ.1) WRITE(NOUT,9001) SINLMN,SINLMX
 9001 FORMAT(/12X,'Only data between the sin(theta)/lambda limits of'
     1 ,F8.4,' and ',F8.4,' will be used')
      SINLMX=SINLMX*SINLMX
      SINLMN=SINLMN*SINLMN
 9180 FORMAT(/48X,'No index group rescaling')
      IF(ISC.NE.1) WRITE(NOUT,9190)
 9190 FORMAT(/32X,'Apply separate scale factors to appropriate inde
     1x groups')
C===> OUTPUT SYMMETRY OPERATIONS AND LATTICE TYPE
      IF(ICENT.EQ.0) WRITE(NOUT,9185) LATT
 9185 FORMAT(/34X,'The structure is non-centrosymmetric with lattice
     1type ',A1)
      IF(ICENT.EQ.1) WRITE(NOUT,9186) LATT
 9186 FORMAT(/34X,'The structure is centrosymmetric with lattice type
     1 ',A1)
      NSYMOP=NSYM
      WRITE(NOUT,9175) NSYM
 9175 FORMAT(/37X,'The',I3,' symmetry operations are as follows:'/)
      DO 896 J=1,NSYM
C===> PREPARE FORMATTED OUTPUT OF SYMMETRY OPERATIONS
      DO 895 I=1,21
  895 IRR(I)=KX(10)
      DO 970 IC=1,3
C===> TRANSLATIONAL PART
      INC=7*IC-5
      IF(TS(IC,J).LT.0.1) GOTO 920
C===> SET  SYMMORPHIC SPACE GROUP FLAG-THIS SPACE GROUP NOT SYMMORPHIC
      ISYMP=0
      NUM=1
      IF(TS(IC,J).GT.0.6) NUM=TS(IC,J)/(1.0-TS(IC,J))+0.1
      NOM=FLOAT(NUM)/TS(IC,J)+0.1
      IRR(INC-1)=KX(NUM)
      IRR(INC)=KX(14)
      IRR(INC+1)=KX(NOM)
C===> SYMBOLIC PART
  920 DO 960 IN=1,2
      INC=INC+2
      IF(IS(IN,IC,J)) 930,960,940
C===> NEGATIVE SYMBOL
  930 IRR(INC)=KX(13)
      GOTO 950
C===> POSITIVE SYMBOL
  940 IF(IN.EQ.2.OR.TS(IC,J).GT.0.1)IRR(INC)=KX(12)
```

```
C===> X OR Y OR Z
  950 IARG=IABS(IS(IN,IC,J))+6
      IRR(INC+1)=KX(IARG)
  960 CONTINUE
  970 CONTINUE
C===> OUTPUT IT
      WRITE(NOUT,9187) (IRR(I),I=1,21)
 9187 FORMAT(33X,3(7X,3A1,4(1X,A1)))
  896 CONTINUE
C===> CHECK THAT THERE IS ONLY ONE GROUP INPUT FOR WEIGHTED FOURIER
C===> (IF NOT THEN ONLY THE FIRST INPUT GROUP IS USED)
      IF(NGP.EQ.1) GOTO 775
C===> TOO MANY GROUPS
      NGP=1
      NINF(1)=6
      CALL STOPP(19,0,0)
  775 IF(ISYMP.EQ.1) WRITE(NOUT,9135)
 9135 FORMAT(/37X,'Note that this space group is symmorphic')
      IF(IVDIF.EQ.1) WRITE(NOUT,9136)
 9136 FORMAT(/34X,'This structure is expected to be very difficult')
      IF(IDIF.EQ.1) WRITE(NOUT,9137)
 9137 FORMAT(/37X,'This structure is expected to be difficult')
      IF(JUMP.EQ.-1.AND.IK.NE.-1) WRITE(NOUT,9130)
 9130 FORMAT(/37X,'Scale and temperature factor to be calculated')
      IF(JUMP.GE.0) WRITE(NOUT,9320) BT
 9320 FORMAT(/40X,'Supplied temperature factor (B) =',F6.2)
      IF(JUMP.EQ.0.AND.IK.NE.-1) WRITE(NOUT,9330)
 9330 FORMAT(/49X,'Scale to be calculated')
      IF(JUMP.EQ.1.AND.ISC.EQ.1) WRITE(NOUT,9340) SCAL(1)
 9340 FORMAT(/45X,'Supplied scale factor =',F6.2)
      IF(JUMP.EQ.1.AND.ISC.NE.1) WRITE(NOUT,9350) SCAL
 9350 FORMAT(/23X,'Supplied scale factors are',8F6.2//32X,
     1 'For appropriate index groups see rescaling section below')
C===> OUTPUT UNIT CELL
      WRITE(NOUT,9360) (CX(J),J=1,6)
 9360 FORMAT(/12X,'Direct cell is A=',F8.3,' B=',F8.3,' C=',F8.3,
     1 ' Alpha=',F7.2,' Beta=',F7.2,' Gamma=',F7.2)
C===> OUTPUT TRANSFORMATION MATRICES
      IF(ITRANS.EQ.1) WRITE(NOUT,9410) TRANS
 9410 FORMAT(/37X,'The input indices are to be transformed by the mat
     1rices' // 3(1H ,46X,3F7.0,F12.0/))
C===> SORT OUT UNIT CELL CONTENTS AND GROUPS
      CALL ATREC
      ANAT=FLOAT(NAT)/(PTS*FLOAT((ICENT+1)*NSYM))
      WRITE(NOUT,370) ANAT
  370 FORMAT(/39X,'Number of atoms in asymmetric unit =',F6.2/)
      IF(NGP.NE.0)THEN
        DO 371 NJ=1,NGP
          NASSYM=NINT(ANAT)
          IF(NAG(NJ).GT.NASSYM)THEN
            WRITE(*,8001)
 8001       FORMAT(/'Stopping because number of atoms in input group '
     .                'greater than in asymmetric unit.')
            CALL EXIT
          ENDIF
  371   CONTINUE
      ENDIF
      IF(NNN.GT.0) WRITE(NOUT,9361) NNN,(IFORM(J),J=1,50)
 9361 FORMAT(/20X,'Input data with',I2,
     1 ' reflections per card - format ',50A1)
```

```
      NASU=INT(ANAT+0.5)
C===> WEIGHTED FOURIER
      REWIND NTAPEE
      WRITE(NTAPEE) ITLE,CX,NN,NO,NK,IAPX,BT,SC,NSYM,ICENT,
     1 LATT,LAT,NGP,NM,KSYS,PTS,NASU,ANAT,CVR,VVR,NTOT,ISYMP,NDIFF,
     2 IDIF,IVDIF,KARLE,IHVY,IGPFL
      WRITE(NTAPEE)(((TS(I,J),IS(K,I,J),K=1,2),I=1,3),J=1,NSYM)
      NF=NAG(1)
      WRITE(NTAPEE) NINF(1),NAG(1)
      WRITE(NTAPEE) (WZ(L),X(L),Y(L),Z(L),L=1,NF)
      WRITE(NTAPEE) SIGD,MAXH,IAD1,IAD2,LEVEL1,LEVEL2
      NRECS=678+(12*NSYM)+(4*NF)
  760 CALL DATAIN
C===> WEIGHTED FOURIER
      CALL RFAC(NRECS)
      REWIND NTAPEE
      WRITE (*,9362)
 9362 FORMAT(/' **** **** NORMAL COMPLETE **** **** '/)
C===> CLOSE SCRATCH FILE
      CLOSE(UNIT=NTAPEF,STATUS='DELETE')
      CALL PAUSE
      RETURN
      END
C
      SUBROUTINE ATREC
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFL,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INIT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEH
      COMMON/BLK1/IESN(24000),IX(1000),EX(1000),FX(1000),X(200),Y(200),
     1 Z(200),WZ(200),NCG1(200),NCG2(200),LLH(60),LLK(60),LLL(60),
     2 FO(60),ID(60),ED(60),EN(60),RHO(60),SIGMA(60),JUNK(11260)
      COMMON/BLK2/IS(2,3,24),TS(3,24),P(6),CX(9),NSYM,ICENT,NORI,SIGMAQ,
     1 NEX,LATT,LAT,PTS,KSYS,IAPX,NGP,NTOT,SIGD(20),IAD1(129),IAD2(129)
     2 ,IHE(24),IKE(24),ILE(24),MSH(24),KHL(6,128),IX(3),MKL(4),NASU,NRD
     3 ,JUNK3(206)
      COMMON/BLK3/STABLE(450),E(800),IFN(800),SCAL(8),FLGN(30),FLGD(30),
     1 AVR(30),DCV(50),VST(10,5),MST(5),ZT(25,5),EE(10),JUNK1(194)
      COMMON/BLK4/IOBUF(200),LEVEL1(129),LEVEL2(129),EDITF(100),
     1 IEDIT(100),TRANS(12),GIS(142),GIN(142),NINF(10),NAG(10),
     2 GPOP(10),GCELL(6,10),NCSF1(10),NCSF2(10),NASF(10),TAS(10),
     3 TAL(10),TBS(10),TBL(10),TCL(10),AL(10),AS(10),BL(10),BS(10),
     4 CL(10),NN(10),NO(10),F(10),KH(10),KK(10),KL(10),KD(10),SIG(10),
     5 NC1(10),NC2(10),NA(10),IFORM(68),CVR(10),VVR(10),TCVR(10),TVVR(10),
     6 MEWE(100),ENEW(100)
      COMMON/BLK5/NPC,NAT,NK,
     1 MAXDUP,EMAX,RHOMAX,NSF,LST,IK,NB,ISC,SLOPE,FLGK,BT,SC,EPS,
     2 DEL,KS,MULT,IND,NZR,TMUL,SINLMN,SINLMX,JUMP,NNN,NN,INEW
     3 ,NEDIT,ITRANS,MM,MZ,EN,ER,GARB
      COMMON/KEEP/N1(50),N2(50),ALT(50),AST(50),BLT(50),BST(50),CLT(50),KI(14)
     1 ,LTC(7),AVA(10),AVC(10),AVH(10),CPH(25),PCH(30),TABCVR(50),TABBVR(50),
     2 JFORM(68),JLATT
      DIMENSION       CR(9)
C===> PART 1- UNIT CELL CONTENTS,COVALENT RADII VAN DER WAALS RADII
      NAT=0
      DO 150 I=1,NK
      NO(I)=0
      NN(I)=NA(I)
C===> CHECK ATOM TYPE
```

```
      DO 120 J=1,50
      IF(NC1(I).NE.N1(J)) GOTO 120
      IF(NC2(I).NE.N2(J)) GOTO 120
C===> ATOM FOUND
      AS(I)=AST(J)
      AL(I)=ALT(J)
      BS(I)=BST(J)
      BL(I)=BLT(J)
      CL(I)=CLT(J)
C===> RADII
      CVR(I)=TABCVR(J)
      VVR(I)=TABBVR(J)
      GOTO 150
  120 CONTINUE
C===> ATOM IS NOT IN DEFAULT LIST SEARCH SFAC CARDS
      IF(NSF.EQ.0) CALL STOPP(20,1,0)
      DO 130 J=1,NSF
      IF(NC1(I).NE.NCSF1(J)) GOTO 130
      IF(NC2(I).NE.NCSF2(J)) GOTO 130
      AS(I)=TAS(J)
      AL(I)=TAL(J)
      BS(I)=TBS(J)
      BL(I)=TBL(J)
      CL(I)=TCL(J)
C===> RADII
      CVR(I)=TCVR(J)
      VVR(I)=TVVR(J)
      GOTO 150
  130 CONTINUE
C===> NO  MATCH FOUND
      CALL STOPP(9,1,0)
  150 CONTINUE
C===> NO OF ATOMS IN CELL
      DO 160 I=1,NK
      NO(I)=INT(AL(I)+BL(I)+CL(I)+0.5)
      IF(NO(I).NE.1) NAT=NAT+NA(I)
      IF(NO(I).GT.10)IHVY=IHVY+NA(I)
  160 CONTINUE
      WRITE(NOUT,9160)(NC1(I),NC2(I),NA(I),NO(I),AL(I),AS(I),
     1 BL(I),BS(I),CL(I),CVR(I),VVR(I),I=1,NK)
 9160 FORMAT(//51X,'Unit cell contents'//
     1 '  Atom',4X,'Number in cell',4X,'Atomic number',4X,
     2 'Scattering factor constants',3X,
     3 '(F = AA*EXP(-A*RHO) + BB*EXP(-B*RHO) + CC)',3X,5HRadii/
     4 (1H ,1X,2A1,I14,I17,5F15.3,5X,2F6.2))
C===> PART 2-THE GROUPS
      DO 180 I=1,142
  180 GIS(I)=0.0
      IF(NGP.EQ.0) GOTO 550
      WRITE(NOUT,190) NGP
  190 FORMAT(//42X,I5,2X,'Atomic groups were input')
      IMX=0
      II=1
      IST37=0
      DO 500 I=1,NGP
C===> NINF MAY BE 2, 3 OR 4 AND REFERS TO THE TYPE OF INFORMATION
C===> 2) RANDOM POSITION AND ORIENTATION
C===> 3) CORRECT ORIENTATION AND RANDOM POSITION
C===> 4) CORRECT POSITION AND ORIENTATION
      IF(IABS(NINF(I)).EQ.3.OR.IABS(NINF(I)).EQ.4) IMX=IMX+NAG(I)
```

```
      IF(IMX.LE.KUSE(11)) GOTO 375
C===> TOO MANY ATOMS FOR TYPE 3 OR 4 GROUPS
      CALL STOPP(37,0,1)
      IST37=IST37+1
      GOTO 500
  375 IF(GPOP(I).LE.ZERO) GPOP(I)=NSYM*(ICENT+1)*PTS
      WRITE(NOUT,9220) I,NINF(I),NAG(I),GPOP(I)
 9220 FORMAT(/14X,'Group no.',I3,' of type',I3,' containing',
     1 I3,' atoms and having a multiplicity of ',F5.1/
     2 /50X,'X',14X,'Y',14X,'Z'/)
      IF(NINF(I).EQ.6) NDIFF=1
      IF(NINF(I).EQ.5) NDIFF=2
      IF(NAG(I).LE.0) CALL STOPP(29,1,0)
      IF(NINF(I).EQ.5) KARLE=1
      DO 355 K=1,6
  355 CR(K)=GCELL(K,I)
      ISTART=II
      IFIN=ISTART+NAG(I)-1
      DO 350 K=ISTART,IFIN
      WRITE(NOUT,9380) NCG1(K),NCG2(K),X(K),Y(K),Z(K)
 9380 FORMAT(37X,2A1,3F15.4)
C===> IDENTIFY THE ATOM TYPE
      NI=0
  365 DO 360 L=1,NK
      IF(NCG1(K).NE.NC1(L)) GOTO 360
      IF(NCG2(K).NE.NC2(L)) GOTO 360
C===> FOUND
      NZ(K)=L
      GOTO 380
  360 CONTINUE
C===> UNIDENTIFIED ATOM ASSUME IT IS CARBON
      CALL STOPP(11,0,0)
      NCG1(K)=IH(21)
      NCG2(K)=IH(45)
C===> TRY AGAIN
      GOTO 365
C===> SUBTRACT GROUP FROM RANDOM TOTAL
  380 IF(NINF(I).LE.2.OR.NINF(I).EQ.6) NA(L)=NA(L)-INT(GPOP(I)+0.5)
  350 CONTINUE
      DO 200 J=1,6
  200 CR(J)=GCELL(J,I)
C===> CALCULATE SPHERICALLY AVERAGED MOLECULAR SCATTERING FACTOR
      IF(IABS(NINF(I)).EQ.2) CALL DEBYE(ISTART,IFIN,CR,I)
      II=IFIN+1
  500 CONTINUE
C===> CALCULATE NILSON (GIN) AND DEBYE (GIS) SCATTERING FACTORS
  550 NGP=NGP-IST37
      DO 600 I=1,142
      T=0.01*FLOAT(I-1)
      TT=T*T
      GIN(I)=0.0
      DO 590 J=1,NK
C===> BEWARE OF EXPONENTIAL OVERFLOWS
      X1=-BS(J)*TT
      IF(X1.LT.-100.0) X1=-100.0
      X2=-AS(J)*TT
      IF(X2.LT.-100.0) X2=-100.0
      FZ=AL(J)*EXP(X1)+BL(J)*EXP(X2)+CL(J)
      GIS(I)=GIS(I)+FZ*FZ*FLOAT(NA(J))
  590 GIN(I)=GIN(I)+FZ*FZ*FLOAT(NN(J))
  600 CONTINUE
```

```
C===> PART 3- THE SIGMA AND SIGD PARAMETERS
      DO 740 J=1,6
      SIG(J)=0.0
      DO 740 K=1,NK
  740 SIG(J)=SIG(J)+FLOAT(NW(K)/NN)*FLOAT(NO(K))**J
C===> RATIOS FOR TRIPLETS,QUARTET AND QUINTET DISTRIBUTIONS
      SIGD(1)=2.0*SIG(3)/     SIG(2)**1.5
      SIGD(2)=(3.0*SIG(3)**2-SIG(2)*SIG(4))/SIG(2)**3
      SIGD(5)=2.0*(15.0*SIG(3)**3-10.0*SIG(2)*SIG(3)*SIG(4)+
     1 SIG(2)*SIG(2)*SIG(5))/SIG(2)**4.5
      SIGD(6)=SIGD(1)*0.5
      SIGD(9)=SIG(3)*SIG(3)/SIG(2)**3
      SIGD(14)=2.0*SIG(3)**3/SIG(2)**4.5
      SIGD(15)=2.0*SIG(3)*(3.0*SIG(3)*SIG(3)-SIG(2)*SIG(4))/SIG(2)**4.5
      SIGD(16)=2.0*(SIG(4)**1.5)/(SIG(2)**3)
      SIGD(19)=2.0*SIG(4)/SIG(2)**2
      SIGD(20)=0.5*SIGD(19)
      RETURN
      END
C
      SUBROUTINE MENU1
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFL,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEH
      CALL CLSCN
      WRITE(*,9002)
 9002 FORMAT(' The following commands are now available:'/
     #'   RECYCLE  <     Fourier        >')
      WRITE(*,9004)
 9004 FORMAT(
     #'   LIST     <Print a full E-list >  SYMM   <Symmetry operation>'/
     2'   CELL     <A,B,C,Angles or cos >  LATTICE<A/C  P/A/B/C/I/F/R>'/
     3'   CONTENTS <n1,type,n2,type etc.>  SFAC   <Scattering factor >'/
     4'   LIMITS   <Sin max,sin min,emax>  BSCL   <B,scale factor(s) >'/
     5'   NEWE     <  H,K,L new E-value >  EDIT   <    H,K,L (F)     >'/
     6'   TRANS    <    Trans matrix    >  GROUP  <  Type,pop,cell   >'/
     7'   ATOM     < Atom label,x,y,z   >  NOCHECK< Do not check data>'/
     8'   DATA     <No of reflections per card   format of data      >'/
     9'              N.B.  this must be last instruction to NORMAL'/ )
      WRITE(*,9007)
 9007 FORMAT(
     2 /'In addition the commands: TITLE,END,MENU,LEVEL,NOPRINT,PRINT'/
     3 '                           DEFAULT,HARD,VERY HARD,MODEL,SHOW,X'/
     4'may be issued at any time.'/'Only the first 4 characters of a
     5ny command are required.'/'<CR> or a call to another module term
     6inates current input.'/'(N.B. Use upper case characters when typing
     7 commands.)'/)
      RETURN
      END
C
      SUBROUTINE STOPP(J,K,L)
C===> ERROR STOP PROGRAMME
C===> J IS ERROR CODE NUMBER; K=0/1 FOR NON-FATAL/FATAL ERROR;
C===> L=0/1 FOR OUTPUT ON UNITS 3 AND 5/OUTPUT ON UNIT 3 ONLY
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFL,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
```

```
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEH
       IF(K.GT.0) GOTO 200
C===> WARNING ONLY
       IF(L.EQ.0) WRITE(NOUT,9005) J
 9005 FORMAT(/30X,'******* Warning code no.',I3,' *******')
       WRITE(*,9006)  J
 9006 FORMAT(' ******* Warning code no.',I3,' *******')
       RETURN
  200 IF(L.EQ.0) WRITE(NOUT,9001) J
 9001 FORMAT(/30X,'******** Error code' ,I3,' ********')
       WRITE(*,9003) J
 9003 FORMAT(' ******* Error code' ,I3,' *******')
       IF(L.EQ.0) WRITE(NOUT,9002)
 9002 FORMAT(/30X,'****** This is a fatal error ******')
       WRITE(*,9004)
 9004 FORMAT('****** This is a fatal error ******'/)
       CALL EXIT
       END
C
       SUBROUTINE DATAIN
C===> READS INTENSITY DATA
       COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
      1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
      2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFL,ZERO,LK(32),
      3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
      4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEH
       COMMON/BLK1/IESN(24000),IX(1000),EX(1000),FX(1000),X(200),Y(200),
      1 Z(200),NZ(200),NCG1(200),NCG2(200),LLH(60),LLK(60),LLL(60),
      2 FO(60),ID (60),ED(60),EW(60),RHO(60),SIGMA(60),KHKL(60),
      3 RJUNK(60),JUNK(11140)
       COMMON/BLK2/IS(2,3,24),TS(3,24),P(6),CX(9),NSYM,ICENT,NORI,SIGMAQ,
      1 NEX,LATT,LAT,PTS,KSYS,IAPX,NGP,NTOT,SIGD(20),IAD1(129),IAD2(129)
      2 ,IHE(24),IKE(24),ILE(24),NSH(24),KHL(6,128),II(3),HKL(4),NASU,NRD
      3 ,JUNK3(206)
       COMMON/BLK3/STABLE(450),E(800),IFN(800),SCAL(8),FLGW(30),FLGD(30),
      1 AVR(30),DCV(50),VST(10,5),NST(5),ZT(25,5),EE(10),JUNK1(194)
       COMMON/BLK4/IOBUF(200),LEVEL1(129),LEVEL2(129),EDITF(100),
      1 IEDIT(100),TRANS(12),GIS(142),GIW(142),NINF(10),NAG(10),
      2 GPOP(10),GCELL(6,10),NCSF1(10),NCSF2(10),NASF(10),TAS(10),
      3 TAL(10),TBS(10),TBL(10),TCL(10),AL(10),AS(10),BL(10),BS(10),
      4 CL(10),NH(10),NO(10),F(10),KH(10),KK(10),KL(10),KD(10),SIG(10),
      5 NC1(10),NC2(10),NA(10),IFORM(68),CVR(10),VVR(10),TCVR(10),TVVR(10),
      6 NEWE(100),ENEW(100)
       COMMON/BLK5/NPC,NAT,NK,
      1 MAXDUP,EMAX,RHOMAX,NSF,LST,IK,NB,ISC,SLOPE,FLGK,BT,SC,EPS,
      2 DEL,KS,MULT,IND,NZR,TMUL,SINLMN,SINLMX,JUMP,NNN,    SE,INEW
      3 ,NEDIT,ITRANS,MM,MZ,EN,ER,GARB
       COMMON/BLK6/ IPHAZ,FILLER(5399)
       COMMON/KEEP/N1(50),N2(50),ALT(50),AST(50),BLT(50),BST(50),CLT(50),KX(14)
      1 ,LTC(7),AVA(10),AVC(10),AVH(10),CPH(25),PCH(30),TABCVR(50),TABBVR(50),
      2 JFORM(68),JLATT
       DIMENSION IED(60),BUFIO(200)
       EQUIVALENCE(LK7,LK(7)),(LK8,LK(8)),(LK(16),LK16),
      1 (LK(18),LK18),(LK(9),LK9),(LK(10),LK10),(LK24,LK(24))
       EQUIVALENCE(IED(1),ED(1)),(IOBUF(1),BUFIO(1)),(MAXIND,KUSE(25)),
      1 (KUSE5,KUSE(5)),(KUSE17,KUSE(17)),(KUSE1,KUSE(1))
       REWIND NTAPEF
       REWIND NTAPEG
C===> DATA STATEMENT IS AVOIDED ONCE AGAIN SO THAT CODE CAN BE RE-ENTERED
       NED=0
```

```
      NLM=0
      NHRT=0
      NREJ=0
      IFLAG=0
      NDUP=0
      NABS=0
      NBIG=0
      K=1
C===> OTHER INITIALISATIONS
   50 NRED=0
      MAXOVF=-999
      IEOF=0
      IPHAZ=0
      IF(IK.EQ.-2) NNN=1
      ICARD=NNN
      NN=ICARD
      IF(ICARD.EQ.0) NN=1
      IF(ICARD) 70,70,60
C
C===> WARNING-------MACHINE  SPECIFIC---------
C
C===> AT GLASGOW THE DATA FILE CONTAINS :
C===> WORD 1:  BATCH NO.*256**3 + (H+128)*256*256 + (K+128)*256 + L+128
C===> WORD 2:  NOT USED BY MITHRIL
C===> WORD 3:  (FOBS)**2
C===> WORD 4:  SIGMA(FOBS**2)
  70  READ(NTAPEG,END=35) KHKL(1),RJUNK(1),FO(1),SIG(1)
      KH(1)=MOD(KHKL(1),LK24)/LK16-LK7
      KK(1)=(MOD(KHKL(1),LK16)/LK8)-LK7
      KL(1)=MOD(KHKL(1),LK8)-LK7
      FO(1)=SQRT(FO(1))
      IF(FO(1).GT.ZERO) SIG(1)=0.5*SIG(1)/FO(1)
      IF(FO(1).LE.ZERO) SIG(1)=1.0
C===> THE GLASGOW VERSION TESTS TO SEE IF F IS OBSERVED (2*SIGMA)
C===> KD = 0/1 FOR OBSERVED/UNOBSERVED.  YOU MAY HAVE STORED THIS FLAG
C===> ON YOUR DATA FILE, IN WHICH CASE SIGMA MAY BE DISPENSED WITH
C===> SINCE IT IS NOT USED ANYWHERE ELSE IN THE PACKAGE (YET!)
      KD(1)=0
      IF(FO(1).LT.2.0*SIG(1)) KD(1)=1
C
C===> WARNING-------END OF INSTALLATION SPECIFIC COMMANDS-------
C
      GOTO 80
C===> DATA IN FORMATTED FORM ON FILE NTAPEG.
   60 IF(IK.EQ.-2) GOTO 65
C
C---- ---- STRIDE HAS PROBLEMS WITH THE FOLLOWING READ STATEMENT -----
C---- ---- IN THE MEANTIME USE A FORMAT TYPE SPECIFIC TO STRUCTURES --
C---- ---- FROM THE DIRECT METHODS TEST FILE -------------------------
C
C      READ(NTAPEG,IFORM,END=35)(KH(I),KK(I),KL(I),FO(I),KD(I),I=1,ICARD)
      READ(NTAPEG,8001,END=35)(KH(I),KK(I),KL(I),FO(I),KD(I),I=1,ICARD)
 8001 FORMAT(3I4,F8.2,I2)
C===> KD=0/1 FOR OBSERVED/UNOBSERVED
      GOTO 80
C===> END OF FILE ON INTENSITY DATA FILE
   35 FO(1)=-99.9
      GOTO 80
C===> READ E'S AND PHASE ANGLES FROM DIRECT METHODS TEST FILE
   65 READ(NTAPEG,9003,END=31) KH(1),KK(1),KL(1),FO(1),IPHAZ
```

```
 9003 FORMAT(3I4,36X,F8.3,I6  )
      GOTO 80
C===> END OF FILE MARKER
   31 IEOF=1
   80 DO 100 J=1,NN
      IF(FO(J).LT.0.0.OR.IEOF.EQ.1) GOTO 30
      IF(KH(J).EQ.0.AND.KK(J).EQ.0.AND.KL(J).EQ.0) GOTO 30
C===> COMPUTE THETA AND CHECK LIMITS
      NRED=NRED+1
      RH=KH(J)*KH(J)*P(1)+KK(J)*KK(J)*P(2)+KL(J)*KL(J)*P(3)+
     1 KH(J)*KK(J)*P(4)+KH(J)*KL(J)*P(5)+KK(J)*KL(J)*P(6)
      IF(RH.GT.0.99.OR.RH.GT.SINLMX.OR.RH.LT.SINLMN)GOTO 101
      RHOMAX=AMAX1(RHOMAX,RH)
C===> CHECK IF IT IS IN DELETION LIST
      IF(NEDIT.EQ.0) GOTO 110
      IF(NEDIT.EQ.NED) GOTO 110
      I=(KH(J)+LK7)*LK16+(KK(J)+LK7)*LK8+KL(J)+LK7
      DO 120 L=1,NEDIT
      IF(I.NE.IEDIT(L)) GOTO 120
C===> FOUND IN LIST
      IF(MAXOVF.GT.0)WRITE(*,9081) KH(J),KK(J),KL(J)
 9081 FORMAT(/'Reflection ',3I4,' has been removed/modified')
      NED=NED+1
      IF(EDITF(L).EQ.0.0) GOTO 100
      FO(J)=EDITF(L)
      GOTO 110
  120 CONTINUE
C===> CHECK IF INPUT INDICES ARE TO BE TRANSFORMED
  110 IF(ITRANS.EQ.0) GOTO 140
C===> TRANSFORM
      KHH=INT(TRANS(1)*KH(J)+TRANS(2)*KK(J)+TRANS(3)*KL(J)+TRANS(4))
      KKK=INT(TRANS(5)*KH(J)+TRANS(6)*KK(J)+TRANS(7)*KL(J)+TRANS(8))
      KLL=INT(TRANS(9)*KH(J)+TRANS(10)*KK(J)+TRANS(11)*KL(J)+TRANS(12))
      KH(J)=KHH
      KK(J)=KKK
      KL(J)=KLL
C===> H MUST BE > OR = 0
  140 IF(KH(J).GE.0) GOTO 150
      KH(J)=-KH(J)
      KK(J)=-KK(J)
      KL(J)=-KL(J)
      IF(IK.EQ.-2) IPHAZ=360-IPHAZ
  150 SIG(J)=FLOAT(1-2*KD(J))
C===> CHECK TO SEE IF THIS REFLECTION IS A SYSTEMATIC ABSENCE
      IF(ICHK.EQ.1) GOTO 155
      CALL ABSENT(KH(J),KK(J),KL(J),IABST)
      IF(IABST.EQ.0) GOTO 155
C===> ABSENT REFLECTION
      NABS=NABS+1
      IF(NABS.EQ.MAXDUP) WRITE(*,9082)
 9082 FORMAT(' etc.............................')
      IF(NABS.GE.MAXDUP) GOTO 100
      WRITE(*,9080) KH(J),KK(J),KL(J),FO(J)
 9080 FORMAT(' Reflection ',3I4,' (with F=',F9.4,
     1 ') is a systematic absence-rejected')
      GOTO 100
 155  CALL EXPAND(KH(J),KK(J),KL(J),1)
  240 KN=NEX+6+K
      IF(KN.LE.KUSE5) GOTO 230
C===> THIS REFLECTION FILLS THE BUFFER SO DUMP BUFFER TO DISC FIRST
```

```
      WRITE(NTAPEF) K,(IOBUF(L),L=2,K)
      NWRT=NWRT+1
      K=1
      GOTO 240
C===> LOOP THROUGH EXPANDED REFLECTIONS
  230 K=K+1
      IOBUF(K)=NEX
      DO 210 JJ=1,NEX
      IHM=IABS(IHE(JJ))
      IF(IHM.GT.MAXIND) GOTO 220
      IKM=IABS(IKE(JJ))
      IF(IKM.GT.MAXIND) GOTO 220
      ILM=IABS(ILE(JJ))
      IF(ILM.GT.MAXIND) GOTO 220
C===> INDICES O.K.
      MAXH=MAX0(IHM,MAXH)
      IAH=IHM+1
C===> ADDRESS CODE CALCULATIONS FOR FUTURE USE
      LEVEL1(IAH)=MAX0(LEVEL1(IAH),IKM)
      LEVEL2(IAH)=MAX0(LEVEL2(IAH),ILM)
C===> PACK INDICES INTO ONE WORD
      K=K+1
  210 IOBUF(K)=(IHE(JJ)+LK7)*LK18+(IKE(JJ)+LK7)*LK9+ILE(JJ)+LK7
      K=K+1
      BUFIO(K)=SIG(J)
      K=K+1
      IOBUF(K)=MULT
      K=K+1
C===> IF IK=-2 , PACK PHASE ANGLE WITH F
      IF(IK.EQ.-2) FO(J)=(IFIX(FO(J)*100.0 )*1000+IPHAZ)*0.00001
      BUFIO(K)=FO(J)
      K=K+1
      BUFIO(K)=EPS
      K=K+1
      BUFIO(K)=RH
      IF(IFLAG.EQ.0) GOTO 100
      WRITE(NTAPEF) K,(IOBUF(L),L=2,K)
      NWRT=NWRT+1
      GOTO 250
C===> INDEX TOO LARGE
 220  IF(NREJ.LT.MAXDUP) WRITE(*,9001) IHE(1),IKE(1),ILE(1),MAXIND
 9001 FORMAT(' Reflection',3I4,' has indices exceeding',I4,'-rejected')
      IF(NREJ.EQ.MAXDUP) WRITE(*,9002)
      NREJ=NREJ+1
      GOTO 100
C===> ALL DATA IN
  30  IFLAG=1
      IHE(J)=0
      IKE(J)=0
      ILE(J)=0
      NEX=1
      FO(J)=-100.0
      GOTO 240
  101 NLM=NLM+1
  100 CONTINUE
      IF(ICARD) 70,70,60
  250 ENDFILE NTAPEF
      REWIND NTAPEF
      IF(NWRT.EQ.0) CALL STOPP(27,1,1)
      IF(NREJ.GT.0) WRITE(NOUT,9002) NREJ
```

```
 9002 FORMAT('A total of ',I4,' reflections were rejected because ',
     1 'they had large indices')
C===> NOW THE ADDRESS CODES
      DO 400 J=1,KUSE1
C===> 1023975=999*LK10+999--JUST THOUGHT YOU WOULD LIKE TO KNOW !
  400 IESN(J)=1023975
      M=MAXH+1
      IAD1(1)=0
      DO 410 J=2,M
  410 IAD1(J)=IAD1(J-1)+(LEVEL1(J-1)*2+1)*(LEVEL2(J-1)*2+1)
      DO 420 J=1,M
      IAD1(J)=IAD1(J)+LEVEL2(J)*(2*LEVEL1(J)+1)+LEVEL1(J)+1
  420 IAD2(J)=2*LEVEL1(J)+1
      KJ=0
  560 READ(NTAPEF,END=590) N,(IOBUF(J),J=2,N)
      K=2
  570 NEX=IOBUF(K)
      IF(BUFIO(K+NEX+3).LT.0.0) GOTO 590
      DO 510 J=1,NEX
      K=K+1
      IHM=IOBUF(K)/LK18-LK7
      IKM=MOD(IOBUF(K),LK18)/LK9-LK7
      ILM=MOD(IOBUF(K),LK9)-LK7
      IF(IHM.EQ.0.AND.IKM.EQ.0.AND.ILM.EQ.0) GOTO 590
      IF(J.GT.1) GOTO 521
      IHMM=IHM
      IKMM=IKM
      ILMM=ILM
      IF(ICHK.EQ.1) GOTO 510
 521  IAH=IHM+1
      IAD=IAD1(IAH)+IAD2(IAH)*ILM+IKM
      L=MOD(IAD,2)
      IAD=IAD/2
      IF(IAD.GT.KUSE1) GOTO 530
      I=IESN(IAD)
      IF(L.EQ.0) GOTO 520
C===> L IS ODD  BITS 0-9
      III=MOD(I,LK10)
      IF(III.NE.999) GOTO 545
      IESN(IAD)=(I/LK10)*LK10
      GOTO 510
C===> L IS EVEN BITS 10-19
 520  III=I/LK10
      IF(III.NE.999) GOTO 545
      IESN(IAD)=MOD(I,LK10)
      GOTO 510
  545 JJ=J
      GOTO 540
  530 NBIG=NBIG+1
      MAXOVF=MAX0(IABS(IHM),IABS(IKM),IABS(ILM),MAXOVF)
      IF(NBIG.LT.MAXDUP)WRITE(*,9010) IHM,IKM,ILM,KUSE1
 9010 FORMAT(' Reflection ',3I4,' overflows array IESN(',I5,
     1 ') - rejected')
      IF(NBIG.EQ.MAXDUP) WRITE(*,9082)
  510 CONTINUE
      KJ=KJ+1
      NREF=NREF+1
      LLH(KJ)=IHMM
      LLK(KJ)=IKMM
      LLL(KJ)=ILMM
      FO(KJ)=BUFIO(K+3)
```

```
      SIGMA(KJ)=BUFIO(K+1)
      RHO(KJ)=BUFIO(K+5)
      IED(KJ)=IOBUF(K+2)
      EW(KJ)=BUFIO(K+4)
      IF(KJ.NE.KUSE17) GOTO 580
      CALL FCAL
      KJ=0
  580 K=K+6
      IF(K-N) 570,560,560
C===> DUPLICATE
  540 NDUP=NDUP+1
      IF(NDUP.EQ.MAXDUP) WRITE(*,9082)
      IF(NDUP.LT.MAXDUP) WRITE(*,9020) IHM,IKM,ILM
 9020 FORMAT(' Reflection ',3I4,' is a duplicate-rejected')
      K=K+NEX-JJ
      GOTO 580
C===> END OF INPUT FROM NTAPEG
  590 IF(NBIG.EQ.0) GOTO 591
C===> REFLECTIONS FOUND WHICH MAY OVERFLOW IESN
C      REWIND NTAPEE
      REWIND NTAPEF
      REWIND NTAPEG
      MAXDUP=-999
      MAXIND=MAXOVF-1
      GOTO 50
C===> EVERYTHING O.K.
  591 KJ=KJ+1
      LLH(KJ)=0
      LLK(KJ)=0
      LLL(KJ)=0
      FO(KJ)=-10.0
      CALL FCAL
      CALL CLSCN
      WRITE(NOUT,9030) NDUP,NABS,NRED,NED,NLM
 9030 FORMAT(/5X,'There are ',I4,' duplicates and ',I4,
     1 ' systematic absences in a data set containing',I5,
     2 ' reflections in total'/ /20X,
     3 'A total of ',I4,' reflections were edited and ',I4,
     4 ' exceeded the theta limits')
      WRITE(*,9031)NDUP,NABS,NED,NLM
 9031 FORMAT(' There are ',I4,' duplicates and ',I4,' systematic absence
     1s'//I4,' reflections were edited and ',I4,' exceeded the theta
     2limits')
      ENDFILE NTAPEE
      REWIND NTAPEE
      RETURN
      END
C
      SUBROUTINE FCAL
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFL,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INIT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEH
      COMMON/BLK1/IESN(24000),IX(1000),EX(1000),FX(1000),X(200),Y(200),
     1 Z(200),NZ(200),NCG1(200),NCG2(200),LLH(60),LLK(60),LLL(60),
     2 FO(60),ID (60),ED(60),EW(60),RHO(60),SIGMA(60),JUNK(11260)
      COMMON/BLK2/IS(2,3,24),TS(3,24),P(6),CX(9),NSYM,ICENT,NORI,SIGMAQ,
     1 NEX,LATT,LAT,PTS,KSYS,IAPX,NGP,NTOT,SIGD(20),IAD1(129),IAD2(129)
     2 ,IHE(24),IKE(24),ILE(24),NSH(24),KHL(6,128),II(3),HKL(4),NASU,NRD,
```

```
     3 JUNK3(206)
      COMMON/BLK3/STABLE(450),E(800),IFN(800),SCAL(8),FLGH(30),FLGD(30),
     1 AVR(30),DCV(50),VST(10,5),NST(5),ZT(25,5),EE(10),JUNK1(194)
      COMMON/BLK4/IOBUF(200),LEVEL1(129),LEVEL2(129),EDITF(100),
     1 IEDIT(100),TRANS(12),GIS(142),GIN(142),NINF(10),NAG(10),
     2 GPOP(10),GCELL(6,10),NCSF1(10),NCSF2(10),NASF(10),TAS(10),
     3 TAL(10),TBS(10),TBL(10),TCL(10),AL(10),AS(10),BL(10),BS(10),
     4 CL(10),NH(10),NO(10),F(10),KH(10),KK(10),KL(10),KD(10),SIG(10),
     5 NC1(10),NC2(10),NA(10),IFORM(68),CVR(10),VVR(10),TCVR(10),TVVR(10),
     6 NEWE(100),ENEW(100)
      COMMON/BLK5/NPC,NAT,NK,
     1 MAXDUP,EMAX,RHOMAX,NSF,LST,IK,NB,ISC,SLOPE,FLGK,BT,SC,EPS,
     2 DEL,KS,MULT,IND,NZR,TMUL,SINLMN,SINLMX,JUMP,NNN,NN,INEW
     3 ,NEDIT,ITRANS,MM,MZ,EN,ER,GARB
      COMMON/KEEP/N1(50),N2(50),ALT(50),AST(50),BLT(50),BST(50),CLT(50),KX(14)
     1 ,LTC(7),AVA(10),AVC(10),AVH(10),CPH(25),PCH(30),TABCVR(50),TABBVR(50),
     2 JFORM(68),JLATT
      DIMENSION I1(3),I2(3),IED(60)
      EQUIVALENCE(IED(1),ED(1)),(KUSE17,KUSE(17))
      VEC(XX)=XX*(XX+0.4807)/((XX+0.8636)*XX+1.3943)
      NGA=0
      DO 300 I=1,KUSE17
      IF(FO(I).GE.0.0) GOTO 305
C===> END  OF DATA
      LLH(I)=-1000
      FO(I)=-999.9
      GOTO 310
  305 I1(1)=LLH(I)
      I1(2)=LLK(I)
      I1(3)=LLL(I)
      IF(FO(I).LT.0.0) GO TO 310
      FF=FO(I)*FO(I)/PTS
      MULT=IED(I)
      EPS=EN(I)
      ID(I)=NSYM/MULT
C===> DETERMINE INDEX GROUP (FOR RESCALING)
      IF(KSYS.GE.7) GO TO 90
      LG=MOD(IABS(LLL(I)),2)
      IF(KSYS.GE.5) GO TO 80
      KG=MOD(IABS(LLK(I)),2)
      JG=MOD(IABS(LLH(I)),2)
C===> TRICLINIC, MONCLINIC AND ORTHORHOMBIC
      IF(KSYS.LE.3) IG=JG+2*KG+4*LG
C===> TETRAGONAL
      IF(KSYS.EQ.4) IG=JG+KG+3*LG
      GO TO 100
C===> TRIGONAL, HEXAGONAL AND RHOMBOHEDRAL INDEXED ON HEXAGONAL AXES
   80 IG=3*LG
      IF(MOD(LLH(I),3).EQ.0) IG=IG+1
      IF(MOD(LLK(I),3).EQ.0.OR.MOD(LLH(I)+LLK(I),3).EQ.0) IG=IG+1
      GO TO 100
C===> CUBIC AND PRIMITIVE RHOMBOHEDRAL
   90 IG=3*MOD(IABS(LLH(I)+LLK(I)+LLL(I)),2)
      IF(MOD(LLH(I)-LLL(I),3).EQ.0) IG=IG+1
      IF(MOD(LLK(I)-LLL(I),3).EQ.0.OR.MOD(LLH(I)-LLK(I),3).EQ.0) IG=IG+1
C===> PACK SYMMETRY FUNCTIONS FOR LATER USE
  100 ID(I)=10000*ID(I)+100*INT(EPS+0.5)+IG+1
C===> LOOK UP SCATTERING FACTOR TABLES GENERATED BY ATREC
      SINTH=100.0*SQRT(RHO(I))
      IND=MAX0(2,INT(SINTH+1.5))
```

```
      FRAC=SINTH-FLOAT(IND-1)
      BF=0.5*(GIW(IND+1)-GIW(IND-1))
      AF=BF+GIW(IND-1)-GIW(IND)
      FORM=AF*FRAC*FRAC+BF*FRAC+GIW(IND)
C===> 'WILSON' STRUCTURE FACTOR
      FORM=EPS*FORM
      EW(I)=FF/FORM
      IF(NGP.EQ.0) GO TO 190
      BF=0.5*(GIS(IND+1)-GIS(IND-1))
      AF=BF+GIS(IND-1)-GIS(IND)
      FORM=EPS*(AF*FRAC*FRAC+BF*FRAC+GIS(IND))
      NS=1
      NF=NAG(1)
C===> SCATTERING FACTORS FOR GROUPS OF KNOWN ORIENTATION
      CENT=FLOAT(ICENT+1)
      FCL=0.0
      DO 180 IGP=1,NGP
      REL=0.0
      RIM=0.0
      NGA=NINF(IGP)
      IF(NGA.NE.-4.AND.NGA.NE.-3.AND.NGA.NE.6) GO TO 170
      IENT=0
      DO 160 J=1,NSYM
      T=1000.0
      DO 130 L=1,3
      IF(NGA.EQ.-4.OR.NGA.EQ.6) T=T+FLOAT(I1(L))*TS(L,J)
      I2(L)=0
  130 CONTINUE
      DO 150 L=1,3
      DO 140 K=1,2
      M=IABS(IS(K,L,J))
      IF(M.GT.0) I2(M)=I2(M)+I1(L)*ISIGN(1,IS(K,L,J))
  140 CONTINUE
  150 CONTINUE
      CALL SFAC(NS,NF,I2,T,RHO(I),AA,BB,IENT)
      IENT=1
      IF(NGA.EQ.-3) GO TO 155
      REL=REL+AA
      IF(ICENT.EQ.0) RIM=RIM+BB
      IF(ICENT.EQ.1) REL=REL+AA
      GO TO 160
  155 FCL=FCL+(AA*AA+BB*BB)*PTS*CENT
  160 CONTINUE
      IF(NGA.EQ.-4.OR.NGA.EQ.6) FCL=FCL+(REL*REL+RIM*RIM)*PTS
  170 IF(NGA.EQ.6) GO TO 200
      NS=NS+NAG(IGP)
      NF=NF+NAG(IGP)
  180 CONTINUE
      FORM=FORM+FCL
C===> 'DEBYE' STRUCTURE FACTOR
  190 ED(I)=FF/FORM
      GO TO 300
C===> SIM WEIGHTING SCHEME
  200 FCL=SQRT(FCL*PTS)
      SIGMA(I)=FCL*EXP(-BT*RHO(I))
      ARG=2.0*SC*FO(I)*SIGMA(I)*EPS/FORM
      IF(ICENT.EQ.0) WATE=VEC(ARG)
      IF(ICENT.EQ.1) WATE=VEC(ARG*0.5)
      ARG=SC*EXP(BT*RHO(I))*FO(I)*WATE/FCL
      EW(I)=REL*ARG
```

```
      ED(I)=RIM*ARG
      EN=EN+SIGMA(I)
      ER=ER+FO(I)
  300 CONTINUE
C===> CREATE FILE FOR WEIGHTED FOURIER
  310 WRITE(NTAPEE) LLH,LLK,LLL,EH,ED,FO,SIGMA
      RETURN
      END
C
      SUBROUTINE SFAC(NS,NF,L,T,RRHO,AA,BB,IENT)
C===> STRUCTURE FACTOR CALCULATION
      COMMON/BLK1/IESN(24000),IX(1000),EX(1000),FX(1000),X(200),Y(200),
     1 Z(200),NZ(200),JUNK(12200)
      COMMON/BLK2/IS(2,3,24),TS(3,24),P(6),CX(9),NSYM,ICENT,JUNK1(1367)
      COMMON/BLK3/STABLE(450),E(800),IFN(800),JUNK4(532)
      COMMON/BLK4/IOBUF(200),LEVEL1(129),LEVEL2(129),EDITF(100),
     1 IEDIT(100),TRANS(12),GIS(142),GIH(142),NINF(10),NAG(10),
     2 GPOP(10),GCELL(6,10),NCSF1(10),NCSF2(10),NASF(10),TAS(10),
     3 TAL(10),TBS(10),TBL(10),TCL(10),AL(10),AS(10),BL(10),BS(10),
     4 CL(10),NH(10),NO(10),F(10),KH(10),KK(10),KL(10),KD(10),SIG(10),
     5 NC1(10),NC2(10),NA(10),IFORM(68),CVR(10),VVR(10),TCVR(10),TVVR(10),
     6 NEWE(100),ENEW(100)
      COMMON/BLK5/NPC,NAT,NK,
     1 MAXDUP,EMAX,RHOMAX,NSF,LST,IK,NB,ISC,SLOPE,FLGK,BT,SC,EPS,
     2 DEL,KS,MULT,IND,NZR,TMUL,SINLMN,SINLMX,JUMP,NNN,NN,INEW
     3 ,NEDIT,ITRANS,MM,MZ,EN,ER,GARB
      COMMON/KEEP/N1(50),N2(50),ALT(50),AST(50),BLT(50),BST(50),CLT(50),KX(14
     1 ,LTC(7),AVA(10),AVC(10),AVH(10),CPH(25),PCH(30),TABCVR(50),TABBVR(50),
     2 JFORM(68),JLATT
      DIMENSION L(3),CTABLE(360)
      EQUIVALENCE(CTABLE(1),STABLE(91))
      AA=0.0
      BB=0.0
      IF(IENT.EQ.1) GO TO 30
      DO 10 I=1,NK
   10 F(I)=AL(I)*EXP(-AS(I)*RRHO)+BL(I)*EXP(-BS(I)*RRHO)+CL(I)
   30 HJ=FLOAT(L(1))
      HK=FLOAT(L(2))
      HL=FLOAT(L(3))
      DO 50 I=NS,NF
      N=NZ(I)
      ARG=AMOD(HJ*X(I)+HK*Y(I)+HL*Z(I)+T,1.0)
      IARG=INT(360.0*ARG+0.5)+1
      IF(IARG.EQ.361) IARG=1
      AA=AA+F(N)*CTABLE(IARG)
      IF(ICENT.EQ.0)BB=BB+F(N)*STABLE(IARG)
   50 CONTINUE
      RETURN
      END
      SUBROUTINE ABSENT(IIH,IIK,IIL,IABST)
C===> CHECKS TO SEE IF REFLECTION IIH,IIK,IIL IS A SYSTEMATIC ABSENCE
C===> IABST =0/1 FOR NO/YES IT IS A SYSTEMATIC ABSENCE
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFL,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEH
      COMMON/BLK2/IS(2,3,24),TS(3,24),P(6),CX(9),NSYM,ICENT,NORI,SIGMAQ,
     1 NEX,LATT,LAT,PTS,KSYS,IAPX,NGP,NTOT,SIGD(20),IAD1(129),IAD2(129),
     2 JUNK1(1079)
```

```
      DIMENSION T(3),RI(3)
      EQUIVALENCE(RI(1),RH),(RI(2),RK),(RI(3),RL)
      IF(ISYMP.EQ.1.AND.LAT.EQ.1) GOTO 91
      RH=FLOAT(IIH)
      RK=FLOAT(IIK)
      RL=FLOAT(IIL)
      U=RH+RK
      GOTO(78,76,73,77,74,75,72),LAT
   72 IF(0.1-ABS(AMOD(U+RH+RL,2.9999))) 83,78,78
   73 U=RH+RL
      GOTO 77
   74 U=U+RL
      GOTO 77
   75 IF(0.1-ABS(AMOD(U,1.9999))) 83,76,76
   76 U=RK+RL
   77 IF(0.1-ABS(AMOD(U,1.9999))) 83,78,78
   78 IF(ISYMP.EQ.1) GOTO 91
      U=-999999.0
      R=0.0
      DO 82 J=1,NSYM
      DO 130 L=1,3
  130 T(L)=0.0
      DO 150 L=1,3
      DO 140 K=1,2
      M=IABS(IS(K,L,J))
      IF(M.GT.0) T(M)=T(M)+RI(L)*ISIGN(1,IS(K,L,J))
  140 CONTINUE
  150 CONTINUE
      IF(49.5-AMAX1(ABS(T(1)),ABS(T(2)))) 91,91,79
   79 Q=T(1)+100.0*(T(2)+100.0*T(3))
      IF(ICENT.EQ.1) Q=ABS(Q)
      IF(Q-U+0.01) 82,80,80
   80 O=6.283185*(RH*TS(1,J)+RK*TS(2,J)+RL*TS(3,J))
      X=COS(O)
      IF(ICENT.EQ.0) X=X+100.0*SIN(O)
      O=X
      R=R+O
      IF(Q-U-0.050)82,81,81
   81 R=O
      U=Q
   82 CONTINUE
      IF(ABS(R)-0.01) 83,91,91
   83 IABST=1
      RETURN
   91 IABST=0
      RETURN
      END
      SUBROUTINE EXPAND(IIH,IIK,IIL,MODE)
C===> FOR ANY REFLECTION H,K,L IT EXPANDS IT OVER THE HEMISPHERE
C===> TO GIVE NEX EXPANDED REFLECTIONS
C===> IF MODE=0 GIVES PHASE SHIFTS,ROTATION SIGNS AND PHASE RESTRICTIONS
C===> IF MODE=1 GIVES EPSILON AND MULTIPLICITY
C===> IF MODE=2 EXPANSION ONLY
      INTEGER HKL
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IRR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFL,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEH
      COMMON/BLK2/IS(2,3,24),TS(3,24),P(6),CX(9),NSYM,ICENT,NORI,SIGMAQ,
```

```
     1 NEX, LATT, LAT, PTS, KSYS, IAPX, NGP, NTOT, SIGD(20), IAD1(129), IAD2(129),
     2 IHE(24), IKE(24), ILE(24), NSH(24), KHL(6,128), II(3), HKL(4), NASU, JUNK3(207)
      COMMON/BLK5/NPC, NAT, NK,
     1 MAXDUP, EMAX, RHOMAX, NSF, LST, IK, NB, ISC, SLOPE, FLGK, BT, SC, EPS,
     2 DEL, KS, MULT, IND, NZR, TMUL, SINLMN, SINLMX, JUMP, NNN, NN, NSIZE, INEW
     3 , NEDIT, ITRANS, MM, MZ, EN, ER
      COMMON/BLK6/IPHAZ, FILLER(5399)
      EQUIVALENCE(LK16, LK(16)), (LK8, LK(8)), (LK7, LK(7))
     1 , (LK10, LK(10)), (LK5, LK(5)), (LK1, LK(1))
      MKANG=13
      IF(ICENT.EQ.0) MKANG=1
      II(1)=IIH
      II(2)=IIK
      II(3)=IIL
      IF(MODE.NE.1) GOTO 60
      EPS=1.0
      MULT=1
      IF(NSYM.EQ.1.AND.LAT.EQ.1) GOTO 60
      K1=LK16*IIH+LK8*(IIK+LK7)+IIL+LK7
      IK1=LK16+LK8-K1
   60 MAXI=1
      DO 250 J=1,NSYM
      DO 200 I=1,4
      HKL(I)=0
  200 KHL(I,J)=0
      DO 220 I=1,3
      KHL(4,J)=KHL(4,J)-II(I)*IFIX(TS(I,J)*24.0+0.1)
      HKL(4)=HKL(4)+II(I)*IFIX(TS(I,J)*12.0+0.1)
      DO 210 K=1,2
      M=IABS(IS(K,I,J))
      IF(M.EQ.0) GOTO 210
      IF=ISIGN(1,IS(K,I,J))*II(I)
      KHL(M,J)=KHL(M,J)+IF
      HKL(M)=HKL(M)+IF
  210 CONTINUE
  220 CONTINUE
      IF(MODE.NE.1) GOTO 70
      IF(NSYM.EQ.1) GOTO 70
      IF(J.EQ.1) GOTO 70
      K2=LK16*HKL(1)+LK8*(HKL(2)+LK7)+HKL(3)+LK7
      IF(K2.EQ.K1) EPS=EPS+1.0
      IF(ICENT.NE.0.AND.K2.EQ.IK1) EPS=EPS+1.0
      IF(K2.EQ.K1.OR.K2.EQ.IK1) MULT=MULT+1
C===> FIRST NON-ZERO INDEX MUST BE POSITIVE
   70 DO 230 I=1,3
      IF(KHL(I,J)) 235,230,235
  230 CONTINUE
  235 KHL(5,J)=ISIGN(1,KHL(I,J))
      DO 240 I=1,4
  240 KHL(I,J)=KHL(I,J)*KHL(5,J)
      KHL(6,J)=250000*KHL(1,J)+500*(KHL(2,J)+250)+KHL(3,J)+250
      JMO=J-1
      IF(JMO.LT.1) GOTO 247
C===> ELIMINATE DUPLICATIONS
      DO 245 I=1,JMO
      IF(KHL(6,I).EQ.KHL(6,J)) KHL(6,J)=0
  245 CONTINUE
C===> LOOK FOR STANDARD REFLECTION (MAX VALUE OF PACKED INDICES)
  247 IF(KHL(6,J).GT.KHL(6,MAXI)) MAXI=J
      IF(MODE.NE.0) GOTO 250
C===> PHASE RESTRICTION CODE
```

```
      IF(ICENT.EQ.1) GOTO 250
      DO 2160 K=1,3
      IF(II(K)+HKL(K)) 250,2160,250
 2160 CONTINUE
C===> FRIEDEL PAIR GENERATED THEREFORE PHASE RESTRICTED
      MKANG=MOD(HKL(4),12)+1
      IF(MKANG.LE.1) MKANG=MKANG+12
  250 CONTINUE
C===> EXPRESS GENERATED REFLECTION IN TERMS OF STANDARD REFLECTION
      KHLS=KHL(5,MAXI)
      L=KHL(4,MAXI)*KHL(5,MAXI)
C===> PUT IN KNOWN PHASE ANGLE - IK=-2 ONLY
      IF(IK.EQ.-2) IPHAZ=MOD(IPHAZ*KHL(5,MAXI)+KHL(4,MAXI)*15+3600,360)
      DO 360 J=1,NSYM
      KHL(4,J)=MOD(KHL(4,J)-KHL(5,J)*L,24)
      IF(KHL(4,J).LT.0) KHL(4,J)=24+KHL(4,J)
  360 KHL(5,J)=KHL(5,J)*KHLS
C===> ALL DONE - LOAD INTO ARRAYS
      IHE(1)=KHL(1,MAXI)
      IKE(1)=KHL(2,MAXI)
      ILE(1)=KHL(3,MAXI)
      IF(MODE.NE.0) GOTO 380
      I=LK16
      IF(KHL(5,MAXI).LT.0) I=0
      NSH(1)=I+ KHL(4,MAXI)*LK5+MKANG
  380 KHL(6,MAXI)=0
      NEX=1
      DO 400 J=1,NSYM
      IF(KHL(6,J).EQ.0) GOTO 400
      NEX=NEX+1
      IHE(NEX)=KHL(1,J)
      IKE(NEX)=KHL(2,J)
      ILE(NEX)=KHL(3,J)
      IF(MODE.NE.0) GOTO 400
      I=LK16
      IF(KHL(5,J).LT.0) I=0
      NSH(NEX)=I+ KHL(4,J)*LK5+MKANG
  400 CONTINUE
      RETURN
      END
C
      SUBROUTINE RFAC(NRECS)
C===> R FACTOR FOR INPUT REFLECTIONS
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFL,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INIT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEH
      COMMON/BLK1/IESN(24000),IX(1000),EX(1000),FX(1000),X(200),Y(200),
     1 Z(200),NZ(200),NCG1(200),NCG2(200),LLH(60),LLK(60),LLL(60),
     2 FO(60),ID(60),ED(60),EW(60),RHO(60),SIGMA(60),JUNK(11260)
      COMMON/BLK5/NPC,NAT,NK,
     1 MAXDUP,EMAX,RHOMAX,NSF,LST,IK,NB,ISC,SLOPE,FLGK,BT,SC,EPS,
     2 DEL,KS,MULT,IND,NZR,TMUL,SINLMN,SINLMX,JUMP,NNN,NN,INEW
     3 ,NEDIT,ITRANS,MM,MZ,EN,ER,GARB
      EQUIVALENCE (KUSE17,KUSE(17))
      SCALE=EN/ER
      R=0.0
      IEOF=0
      REWIND NTAPEE
```

```
      READ(NTAPEE)(JUNK(J),J=1,NRECS)
  100 READ(NTAPEE,END=101) LLH,LLK,LLL,EW,ED,FO,SIGMA
      GOTO 102
  101 IEOF=1
  102 DO 200 I=1,KUSE17
      IF(FO(I).LT.0.0) GO TO 250
      R=R+ABS(SIGMA(I)-SCALE*FO(I))
  200 CONTINUE
      IF(IEOF.EQ.0)GO TO 100
  250 R=100.0*R/EN
      WRITE(NOUT,280) R,NREF
  280 FORMAT(/39X,'R-factor =',F7.2,'%  for',  I6,2X,'reflections')
      WRITE(*,285) R,NREF
  285 FORMAT(/8X'R-factor=',F7.2,' for',I6,' reflections')
      RETURN
      END
C
      SUBROUTINE DEBYE(NMEM,NNTOT,CR,N)
C===> SCATTERING FACTORS FOR TYPE 2 GROUPS
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFL,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),NTAPEH
      COMMON/BLK1/IESN(24000),IX(1000),EX(1000),FX(1000),X(200),Y(200),
     1 Z(200),NZ(200),NCG1(200),NCG2(200),LLH(60),LLK(60),LLL(60),
     2 FO(60),ID (60),ED(60),EW(60),RHO(60),SIGMA(60),JUNK(11260)
      COMMON/BLK2/IS(2,3,24),TS(3,24),P(6),CX(9),NSYM,ICENT,NORI,SIGMAQ,
     1 NEX,LATT,LAT,PTS,KSYS,IAPX,NGP,NTOT,SIGD(20),IAD1(129),IAD2(129)
     2 ,IHE(24),IKE(24),ILE(24),NSH(24),KHL(6,128),II(3),HKL(4),NASU,NRD,
     3 JUNK5(206)
      COMMON/BLK3/STABLE(450),E(800),IFN(800),SCAL(8),FLGH(30),FLGD(30),
     1 AVR(30),DCV(50),VST(10,5),NST(5),ZT(25,5),EE(10),JUNK1(194)
      COMMON/BLK4/IOBUF(200),LEVEL1(129),LEVEL2(129),EDITF(100),
     1 IEDIT(100),TRANS(12),GIS(142),GIH(142),NINF(10),NAG(10),
     2 GPOP(10),GCELL(6,10),NCSF1(10),NCSF2(10),NASF(10),TAS(10),
     3 TAL(10),TBS(10),TBL(10),TCL(10),AL(10),AS(10),BL(10),BS(10),
     4 CL(10),NW(10),NO(10),F(10),KH(10),KK(10),KL(10),KD(10),SIG(10),
     5 NC1(10),NC2(10),NA(10),IFORM(68),CVR(10),VVR(10),TCVR(10),TVVR(10),
     6 NEWE(100),ENEW(100)
      COMMON/BLK5/NPC,NAT,NK,
     1 MAXDUP,EMAX,RHOMAX,NSF,LST,IK,NB,ISC,SLOPE,FLGK,BT,SC,EPS,
     2 DEL,KS,MULT,IND,NZR,TMUL,SINLMN,SINLMX,JUMP,NNN,NN,INEW
     3 ,NEDIT,ITRANS,MM,MZ,EN,ER,GARB
      COMMON/KEEP/N1(50),N2(50),ALT(50),AST(50),BLT(50),BST(50),CLT(50),KX(14)
     1 ,LTC(7),AVA(10),AVC(10),AVH(10),CPH(25),PCH(30),TABCVR(50),TABBVR(50),
     2 JFORM(68),JLATT
      DIMENSION G(142),CR(9)
      EQUIVALENCE(IX(1),G(1))
      POP=GPOP(N)
      FP=4.0*PI
      IF(CR(1).LT.ZERO) GO TO 15
      WRITE(NOUT,5) (CR(I),I=1,6)
    5 FORMAT(/'Given coordinates refer to the cell',2X,' A =',F6.2,
     1 2X,'B =',F6.2,2X,'C =',F6.2,2X,'Alpha =',F6.1,2X,'Beta =',
     2 F6.1,2X,'Gamma =',F6.1)
C===> CONVERT TO ORTHOGONAL COORDINATES
      ARG=1.0
      DO 25 I=4,6
      CR(I+3)=SIN(DTOR*CR(I))
      CR(I)=COS(DTOR*CR(I))
```

```
   25 ARG=ARG-CR(I)*CR(I)
      V=SQRT(ARG+2.0*CR(4)*CR(5)*CR(6))
      R1=CR(3)*(CR(4)-CR(5)*CR(6))/CR(9)
      R2=CR(3)*V/CR(9)
      DO 10 I=NMEM,NNTOT
      X(I)=CR(1)*X(I)+CR(2)*CR(6)*Y(I)+CR(3)*CR(5)*Z(I)
      Y(I)=CR(2)*CR(9)*Y(I)+R1*Z(I)
   10 Z(I)=R2*Z(I)
   15 D=1.0
      DO 100 I=1,142
      G(I)=0.0
      T=0.01*FLOAT(I-1)
      TT=T*T
      DO 20 J=1,NK
   20 F(J)=AL(J)*EXP(-AS(J)*TT)+BL(J)*EXP(-BS(J)*TT)+CL(J)
C===> SUM OVER INTERATOMIC VECTORS
      DO 50 K=NMEM,NNTOT
      KT=NZ(K)
      NT=K+1
      G(I)=G(I)+F(KT)*F(KT)
      IF(K.EQ.NNTOT) GO TO 50
      DO 40 L=NT,NNTOT
      LT=NZ(L)
      IF(I.EQ.1) GO TO 30
      D=FP*T*SQRT((X(K)-X(L))**2+(Y(K)-Y(L))**2+(Z(K)-Z(L))**2)
      D=SIN(D)/D
   30 G(I)=G(I)+2.0*F(KT)*F(LT)*D
   40 CONTINUE
   50 CONTINUE
      G(I)=SQRT(G(I))
  100 CONTINUE
      DO 140 I=1,142
  140 GIS(I)=GIS(I)+G(I)*G(I)*POP
      RETURN
      END
C
      SUBROUTINE TITLE2
$INT2
C===> WRITE TITLE PAGE ON GRAPHICS SCREEN
      CALL INITIL
C===> BLACK BACKGROUND
      CALL COLOUR(63)
      CALL CLEARW
      CALL WTEXT(25,445,2,2,0,1,32,'WELCOME TO :')
      CALL WTEXT(25,390,3,3,0,1,32,'THE STRIDE FOURIER PACKAGE')
      CALL WTEXT(40,1,1,1,0,1,32,'( Authors: C.J.Gilmore & S.R.Brown
     1University of Glasgow 1986 )')
      CALL WTEXT(2,270,2,2,0,1,4,'# FOURIER RECYCLING OF MOLECULAR FRAGMENTS'
     1)
      CALL WTEXT(2,225,2,2,0,1,4,'# FURTHER RECYCLING OF ANY FRAGMENTS FOUND'
     1)
      CALL WTEXT(2,180,2,2,0,1,4,'# E-MAP CALCULATION AND INTERPRETATION')
      CALL WTEXT(2,135,2,2,0,1,4,'# CONTOURED SECTIONS OF E-MAPS')
      CALL WTEXT(2,90,2,2,0,1,4,'# MANIPULATION AND EDITING OF FRAGMENTS')
      CALL WTEXT(2,45,2,2,0,1,4,'# INSPECTION OF CALCULATED E-MAPS')
      CALL WTEXT(5,480,4,2,0,1,10,'====================')
      CALL WTEXT(5,330,4,2,0,1,10,'====================')
      CALL WTEXT(15,345,4,2,2,1,10,'======')
      CALL WTEXT(512,345,4,2,2,1,10,'======')
      RETURN
      END
```

```
      SUBROUTINE MAPS( NELSE, NSYMOP)
C===>
C===>      M    M     AAA     PPPP     SSSS
C===>      MM MM    A    A    P    P  S
C===>      M M M    AAAAA    PPPP    SSSSS
C===>      M    M    A    A    P           S
C===>      M    M    A    A    P       SSSS
C===>
C===> THIS IS AN AMALGAM OF TWO PROGRAMMES : EXFFT AND SEARCH
C===> PART1:
C===> FAST FOURIER TRANSFORM PROGRAM
C===>
C===> AUTHOR : C. J. GILMORE UNIVERSITY OF GLASGOW
C===>
C===> COPYRIGHT 1984 UNIVERSITY OF GLASGOW
C===>
C===> VERSION FOR STRIDE 460, DATED MAY 1985
C===>
      INTEGER*1 LMAP
      COMMON/IO/NIN, NOUT, NTAPEA, NTAPEB, NTAPEC, NTAPED, NTAPEE,
     1 NTAPEF, NTAPEG, IH(155), A(200), ICH1(10), ICH2(10), ITLE(68), IR(68),
     2 ICALL, NRC, NCH, NREAD, IEND, NREF, PI, DTOR, MAXH, ISPFLG, ZERO, LK(32),
     3 IDEF, ICHK, ISYMP, LEVEL, IFLOW(25), KUSE(40), INIT, IRDY, RTOD, NULL,
     4 IDIF, IVDIF, KARLE, IHVY, IGPFL, NDIFF, ISPCH(30), ICLCK, ICNT
      COMMON /BLK1/ X(30000), IHH(100), G(2,100),
     1 LH(60), LLK(60), LL(60), EW(60), ED(60), NW(10), NO(10), NK, CVR(10),
     2 VVR(10), IOBUF(200), BUFIO(200), LINE(50), PAD1(8909)
      COMMON /BLK2/ IS(2,3,24), TS(3,24), NSYM, ICENT, LAT, IAPX,
     1 CELL(6), T(3,3), DXMAX(3), TL(3,4), FS(3,3,24), MLAT, NEQV, MCON, INT,
     2 FOM, KFRAG(20), NFRAG, LFRAG(20), MFRAG, NOFRAG, NN, MFRN, IFMT(20),
     3 KMATCH(20), ITYPE, NOINT, MNPPF, NUMB, NNO, PAD2(495)
      COMMON /BLK3/ CFOM(2250), PAD3(332)
      COMMON /BLK4/ IONP(6,200), NAFRG(5), ICH(12), IB(6), LINX(60), NSEQ,
     1 PAD4(864)
      COMMON /BLK5/ NOJOIN, NOPROJ, TWOPI, FACTOR, RMAX, RMIN, DOUT,
     1 MH(3), GARB2, GARBAG, XLIM(3), IOPT, NPIC, NP(3), NEX, ANGMIN, ANGMAX,
     2 DMIN, DMAX, DMUT, DM, DFRG, ALN, NPATT, IVDU, NAT, NPC, SCALE, IPASS
      COMMON/BLK6/ INBOND(1000), IJFMT(120), JBND(30), IPSET(200),
     1 ISONP(6,200), PAD6(2850)
      COMMON/SINCOS/SINE(450), NYSEC
      COMMON/CONTR/LMAP(1000000)
      DIMENSION CLINE(12), IHKL(800), E(800), IPHAZ(800), NXVAL(100), NYVAL(100),
     1 NZVAL(100)
      CHARACTER*4 YESNO(2), ICH1, ICH2, ITLE
      CHARACTER*1 CC
      INTEGER BESET
      EQUIVALENCE (KUSE30, KUSE(30)), (IHKL(1), X(1)),
     1 (E(1), X(10001)), (IPHAZ(1), X(20001))
      YESNO(1)='   NO'
      YESNO(2)='  YES'
C===> DEFINE CONSTANTS
C===> KUSE30 = DIMENSION OF ARRAYS IHKL, E, IPHAZ, EP = MAXIMUM NUMBER
C===> OF INDEPENDENT REFLECTIONS WHICH CAN BE INPUT  (KUSE30 .GE. NUMB)
C===> DIMENSION OF X = 3 * KUSE30
      MORE=NELSE
      NSIZE = 3 * KUSE30 - 1
      TWOPI = 360.0 * DTOR
C===> SET UP SINE-COSINE LOOKUP TABLE
      DO 17 J=1,450
      ARG=(J-1)*DTOR
   17 SINE(J)=SIN(ARG)
      NTAPEI=17
```

```
C===> CLEAR SCREEN
      CALL CLSCN
C===> OPEN SCRATCH FILE
  101 OPEN(UNIT=NTAPEF,STATUS='SCRATCH',FORM='UNFORMATTED')
C===> NELSE=1 FOR E-MAP & NELSE=2 FOR FOURIER MAP
      DO 33 J=1,200
   33 IPSET(J)=0
      DO 34 K=1,6
   34 IONP (K,J)=0
      IPASS=0
      NUMBER=1
      MNPPF=4
      ITYPE=NELSE-1
      NOINT=3
      IHVY=0
      NSEQ=0
      FACTOR=0.400
      DO 31 J=1,3
      MH(J)=0
   31 XLIM(J)=1.0
C===> STEREOCHEMICAL CRITERIA - MAX AND MIN DISTANCES AND ANGLES
      ANGMIN=85.0
      ANGMAX=145.0
      DMIN=1.10
      DMAX=1.95
C===> ALL INTERPEAK DISTANCES LESS THAN DMUT ARE PRINTED OUT
      DMUT=2.4
C===> IF 2 E-MAP PEAKS ARE CLOSER THAN DM THE SMALLER IS ELIMINATED
      DM=0.85
C===> IF TWO CLUSTERS ARE CLOSER TOGETHER THAN DFRG INCLUDE MORE PEAKS
      DFRG=2.80
C===> INT = RELATIVE WEIGHT OF IDENTIFIED ATOMS TO UNIDENTIFIED
      INT=2
C===> ALN = NUMBER OF LINES PER INCH ON LINEPRINTER - USED IN PICTUR
      ALN=6.0
      IBL=0
      NA=0
      NOPROJ=0
      MFRN=0
      NPC=0
      NOJOIN=0
      KS=1
      LSTOP=0
      DO 80 J=1,10
      IF(J.LE.5) NAFRG(J)=0
   80 ICH(J)=IH(J)
      ICH(11)=IH(45)
      ICH(12)=ISPCH(2)
C===> SET COUNTER FOR THE CONTOURED SECTIONS
      NYSEC=0
      ICNT=0
C===> DEAL WITH THE CASE OF A STORED MAP FIRST
      IF(NELSE.EQ.3) THEN
        ICNT=1
        REWIND NTAPEB
        READ(NTAPEB,7000)CELL,NEQV,ICENT,LAT,IDIF,IVDIF,MLAT
 7000   FORMAT(6F9.4,6I5)
        DO 260 J=1,24
          DO 270 L=1,3
            READ(NTAPEB,7001)IS(1,L,J),IS(2,L,J),TS(L,J)
 7001       FORMAT(2I6,F9.5)
  270     CONTINUE
```

```
  260   CONTINUE
        READ(NTAPEB,7002)NX,NZ,NY
 7002   FORMAT(3I6)
C===>  NEED ARRAY FS TO FIND EQUIVALENT PEAKS
        DO 301 I=1,NEQV
          DO 302 J=1,3
            DO 303 K=1,3
              FS(K,J,I)=0.0
  303       CONTINUE
            DO 304 K=1,2
              M=IABS(IS(K,J,I))
              IF(M.GT.0)FS(M,J,I)=FLOAT(ISIGN(1,IS(K,J,I)))
  304       CONTINUE
  302     CONTINUE
  301   CONTINUE
C===>  GET MAP POINTS AND CONTOUR ON GRAPHICS SCREEN
        CALL GETMAP(LMAP,NX,NZ,NY)
        RETURN
      ELSE IF(NELSE.EQ.2) THEN
C===>  WEIGHTED FOURIER REQUESTED
        GOTO 200
      ENDIF
C===>  PHASE FILE ALL SET UP
   52 REWIND NTAPED
      READ(NTAPED,9010,END=5,ERR=6)ITLE,ICENT,LAT,NSYM
C===>  NLINES COUNTS THE NUMBER OF LINES READ ON NTAPED
      NLINES=2
 9010 FORMAT(68A1/3I5)
      DO 10 J=1,NSYM
      READ(NTAPED,9011,END=5,ERR=6)  (TS(I,J),(IS(K,I,J),K=1,2),I=1,3)
      NLINES=NLINES+1
 9011 FORMAT(3(F10.5,2I5))
   10 CONTINUE
C===>  NEXT RECORD
      READ(NTAPED,9015,END=5,ERR=6) MH,CELL,NAT,NUMB
 9015 FORMAT(3I5/6F10.5/2I5)
      NLINES=NLINES+3
C===>  GET THE E-MAGNITUDES
      IX=0
   20 READ(NTAPED,9012,END=5,ERR=6) (LINE(L),CLINE(L),L=1,3)
 9012 FORMAT(3(I12,F10.5))
      NLINES=NLINES+1
      DO 25 L=1,3
      IX=IX+1
      IF(IX.GT.NUMB) GOTO 30
      E(IX)=CLINE(L)
      IHKL(IX)=LINE(L)
   25 CONTINUE
      GOTO 20
C===>  ALL E'S READ
   30 READ(NTAPED,9013,END=5,ERR=6)NH,NO,CVR,VVR,NK,IAPX,ISYMP,IVDIF,IDIF,NSET
 9013 FORMAT(10I6/10I6/10F7.3/10F7.3/5I5/I6)
      NLINES=NLINES+6
C===>  GET THE CFOM'S
   47 READ(NTAPED,9054,END=5,ERR=6) I
 9054 FORMAT(12I6)
      IF(I.NE.-1) GOTO 47
C===>  FOUND THE CFOM'S
      READ(NTAPED,9055,END=5,ERR=6) (CFOM(L),L=1,NSET)
 9055 FORMAT(10F7.2)
C===>  GET NTAPED IN CORRECT PLACE
```

```
      REWIND NTAPED
      DO 46 L=1,NLINES
   46 READ(NTAPED,9056) I
 9056 FORMAT(I1)
C===> READY TO READ PHASE SETS.
      IF(IPASS.GT.0) CALL CLSCN
   35 WRITE(*,9001)
 9001 FORMAT(/' Select E-map option :'//'    0  for best map'/'    n  for set #n'
     1/'    R  for review phase sets'/'    Q  for quit. '//)
      CALL DEC(1)
      IOPT=INT(A(1))
      CALL FINDCH
      IF(NCH.EQ.0) GOTO 67
C===> CHARACTER INPUT DETECTED - CHECK FOR A QUIT
      IF(ICH1(1).EQ.'Q'.OR.ICH1(1).EQ.'q') GOTO 90
C===> NOT A QUIT - CHECK FOR REVIEW
      IF(ICH1(1).EQ.'R'.OR.ICH1(1).EQ.'r') THEN
C===> A REVIEW OF THE PHASE SETS
   45 CALL CLSCN
      WRITE(*,9003)
 9003 FORMAT('Set  Abs fom     Psi-zero    Resid     Nqest     Nqint     Cfom
     1   Examined' /)
C===> NLINE COUNTS THE NO OF PHASE SETS OUTPUT TO SCREEN
      NLINE=0
   50 READ(NTAPED,9050,END=60) NNO,ABS,PSI,RESID,RN4,RN5,CFM
 9050 FORMAT(I5,6F10.5)
C===> READ THE ASSOCIATED PHASE ANGLES
      IF(NNO.GT.0) READ(NTAPED,9059,END=5,ERR=6) (IPHAZ(L),L=1,NUMB)
      IF(NNO.LT.0) THEN
C===> ALL PHASE SETS OUTPUT
      WRITE(*,9052)
      READ(*,9053) CC
      GOTO 52
      ENDIF
C===> NOT YET THE END OF THE PHASE SETS
      I=1
      IF(CFOM(NNO).LT.0) I=2
      WRITE(*,9051) NNO,ABS,PSI,RESID,RN4,RN5,CFM,YESNO(I)
 9051 FORMAT(I4,F8.3,F10.3,F12.3,3F10.3,4X,A4)
      NLINE=NLINE+1
      IF(NLINE.GE.20) THEN
      WRITE(*,9052)
 9052 FORMAT(/'Press <CR> for more:'\)
      READ(*,9053) CC
 9053 FORMAT(A)
      GOTO 45
      ENDIF
      GOTO 50
      ENDIF
C===> NUMERICAL INPUT
   67 IF(IOPT.EQ.0) THEN
C===> FIND THE BEST SET
      BEST=0.0
      BESET=0
      DO 61 L=1,NSET
      IF(CFOM(L).LT.0.0) GOTO 61
      IF(CFOM(L).GT.BEST) THEN
      BEST=CFOM(L)
      BESET=L
      ENDIF
   61 CONTINUE
      IF(BESET.EQ.0) THEN
      WRITE(*,9061)
```

```
 9061 FORMAT(/'All phase sets have now been investigated-there is now no best
     1 set'/)
      GOTO 52
      ENDIF
      IOPT=BESET
      ENDIF
C===> SELECT SPECIFIED PHASE SET
   70 READ(NTAPED,9050,END=5,ERR=6) NNO,ABS,PSI,RESID,RN4,RN5,CFM
      IF(NNO.LT.0) GOTO 75
      READ(NTAPED,9059,END=5,ERR=6) (IPHAZ(L),L=1,NUMB)
 9059 FORMAT(12I6)
      IF(NNO.NE.IOPT) GOTO 70
C===> FOUND THE SET - GET CFOM UPDATED AND NTAPED IN CORRECT PLACE
      CFOM(IOPT)=-99.9
   73 READ(NTAPED,9050,END=5,ERR=6) NNO
      IF(NNO.GT.0) THEN
      READ(NTAPED,9059,END=5,ERR=6) (JUNK,L=1,NUMB)
      GOTO 73
      ENDIF
C===> END OF THE PHASE SETS
      WRITE(NTAPED,9055) (CFOM(L),L=1,NSET)
      ENDFILE NTAPED
      GOTO 200
C===> ALL PHASE SETS READ
   75 WRITE(*,9060) IOPT
 9060 FORMAT(/'Phase set number ',I4,' is not on the file'/)
      GOTO 52
C===>
C===> NOW READY FOR THE MENU
C===>
  200 CALL MENU10
  210 CALL DEC(0)
      IF(IEND.NE.0.OR.IDEF.EQ.1) GOTO 1010
      IF(NREAD.EQ.50) GOTO 200
C===> CONTOUR
      IF(NREAD.EQ.57)THEN
      ICNT=1
      GOTO 210
      ENDIF
      IF(NREAD.GT.147.OR.NREAD.LT.137) THEN
      WRITE(*,9064)
 9064 FORMAT(/'Unrecognised command'/)
      GOTO 210
      ENDIF
      IG=NREAD-136
      GOTO(300,320,340,360,380,400,420,440,460,480,210),IG
C===> GRID AND LIMITS
  300 FACTOR=A(1)
      IF(FACTOR.LT.ZERO) FACTOR=0.400
      XLIM(1)=A(2)
      IF(XLIM(1).LT.ZERO) XLIM(1)=1.0
      XLIM(2)=A(3)
      IF(XLIM(2).LT.ZERO) XLIM(2)=1.0
      XLIM(3)=A(4)
      IF(XLIM(3).LT.ZERO) XLIM(3)=1.0
      GOTO 210
C===> NO OF PEAKS
  320 NPC=INT(A(1))
      IF(NRC.GT.1) IHVY=IABS(INT(A(2)))
      MNPPF=INT(A(3))
      IF(MNPPF.EQ.0) MNPPF=4
      GOTO 210
```

```
C===> DON'T JOIN THE DOTS
  340 NOJOIN=1
      GOTO 210
C===> NUMBER OF PROJECTIONS
  360 NOPROJ=MIN0(INT(A(1)),3)
      IF(NRC.GT.1) NOINT=INT(A(2))
      GOTO 210
C===> MAXIMIM BOND LENGTH TO OUTPUT
  380 DOUT=A(1)
      IF(DOUT.LT.ZERO) DOUT=2.40
      GOTO 210
C===> MINIMUM ACCEPTABLE BOND LENGTH
  400 DMIN=A(1)
      IF(DMIN.LT.ZERO) DMIN=1.10
      GOTO 210
C===> MAXIMUM ACCEPTABLE BOND LENGTH
  420 DMAX=A(1)
      IF(DMAX.LT.ZERO) DMAX=1.95
      GOTO 210
C===> MINIMUM ACCEPTABLE ANGLE
  440 ANGMIN=A(1)
      IF(ANGMIN.LT.ZERO) ANGMIN=85.0
      GOTO 210
C===> MAXIMUM ACCEPTABLE ANGLE
  460 ANGMAX=A(1)
      IF(ANGMAX.LT.ZERO) ANGMAX=145.0
      GOTO 210
C===> MOLECULE INPUT
  480 MFRN=1
      NUM=0
      NADR=0
      ISL=0
      IP=0
      NAFRG(1)=0
   60 DO 1000 I=1,68
      DO 150 J=1,12
      IF(ICH(J).EQ.  IR(I)) GO TO 160
  150 CONTINUE
C===> IS THE CHARACTER AN "=" SIGN (LINE CONTINUATION SYMBOL)
      IF(IH(18).NE.IR(I)) GOTO 155
C===> YES IT IS
      READ(*,9002,END=1000,ERR=155) IR
 9002 FORMAT(4X,68A1)
      GOTO 60
C===> IS THE CHARACTER A "!" IF SO IGNORE REST OF THE CARD
  155 IF(IR(I).EQ.ISPCH(1)) GOTO 210
C===> UNRECOGNISED CHARACTER
      WRITE(*,9009) IR
 9009 FORMAT(' **** Error in molecule command-input statement reads:'/
     1  1H ,68A1/' MOLECULE command ignored')
      GOTO 210
  160 IF(J.EQ.11) GO TO 170
      IF(J.EQ.12) GO TO 180
      NUM=10*NUM+J-1
      GO TO 1000
  170 IF(NUM.EQ.0) GO TO 1000
      IF(IP.EQ.0) IAT=NUM
      IF(IP.GT.0) IB(IP)=NUM
      ISL=0
      NUM=0
      IP=IP+1
      GO TO 1000
```

```
  180 IF(IP.EQ.0) GO TO 630
      IF(NUM.EQ.0) GO TO 190
      IB(IP)=NUM
      IP=IP+1
      NUM=0
  190 IP=IP-1
      IF(IP.EQ.0) GO TO 1000
      DO 250 K=1,IP
      NAFRG(NSEQ+1)=MAX0(NAFRG(NSEQ+1),IAT,IB(K))
      L=0
  610 L=L+1
      IF(IONP(L,IAT).EQ.IB(K)) GO TO 250
      IF(IONP(L,IAT).GT.0) GO TO 610
      IONP(L,IAT)=IB(K)
      IX=IB(K)
      L=0
  620 L=L+1
      IF(IONP(L,IX).GT.0) GO TO 620
      IONP(L,IX)=IAT
  250 CONTINUE
      IP=0
      GO TO 1000
  630 IF(ISL.EQ.(-1)) GOTO 210
      ISL=-1
      NSEQ=NSEQ+1
      N=NAFRG(NSEQ)
      NX=-29
C===> OUTPUT FRAGMENT CONNECTIVITY
      IF(NSEQ.EQ.1) WRITE(NOUT,9008)
 9008 FORMAT(1H0,10X,'Molecules input by user:'/1H ,10X,24(1H-))
      WRITE(NOUT,720) NSEQ,N
  720 FORMAT(1H0,10X,15HFragment number ,I4,5X,17HNumber of atoms =,I4)
  730 NX=NX+30
      NPP=MIN0(N,NX+29)
      WRITE(NOUT,740) (J,J=NX,NPP)
  740 FORMAT(1H0,10X,5HAtoms,5X,30I3)
      DO 780 K=1,6
      L=0
      LNO=0
      DO 750 J=1,60
  750 LINX(J)=ICH(11)
      DO 760 J=NX,NPP
      L=L+2
      IF(IONP (K,J).EQ.0) GO TO 760
      LNO=1
      II=IONP (K,J)
      JJ=II/10
      IF(JJ.GT.0) LINX(L-1)=ICH(JJ+1)
      JJ=II-10*JJ
      LINX(L)=ICH(JJ+1)
      IONP (K,J)=0
      IF(II.LT.J) GO TO 760
      NADR=NADR+1
      INBOND(NADR)=512*J+II
  760 CONTINUE
      IF(LNO.EQ.0) GO TO 790
      WRITE(NOUT,770) (LINX(J),J=1,L)
  770 FORMAT(1H ,10X,8Hbonds TO,2X,30(1X,2A1))
  780 CONTINUE
  790 IF(NPP.LT.N) GO TO 730
      NADR=NADR+1
      INBOND(NADR)=-1000
```

```
      IF(NSEQ.GE.5) THEN
      WRITE(*,9071)
 9071 FORMAT(/'Too many input fragments - last ones(s) ignored'/)
      GOTO 210
      ENDIF
 1000 CONTINUE
      GOTO 210
C===> ALL INPUT NOW COMPLETE-KEEP COPIES OF ILEVEL,NAT,NPIC AND IONP(6,200)
 1010 LEVELS=ILEVEL
      NPICS=NPIC
      DO 1011 J=1,200
      DO 1011 K=1,6
 1011 ISONP(K,J)=IONP(K,J)
 1200 NOSET=IPSET(IPASS)
      CALL CLSCN
      IF(II.EQ.1) THEN
      WRITE(NOUT,9185)ITLE
 9185 FORMAT(//68A1///)
      WRITE(NOUT,9190) IOPT,ABS,PSI,RESID,RN4,RN5
 9190 FORMAT(/'E-Map to be calculated'/22(1H-)//13X,6HNumset,7X,7HAbs fom,11
     1X,8HPsi zero,11X,5HResid,12X,'Nqest',11X,'Nqint'/9X,I8,F16.4,F18.2,F17.2
     2,2F16.2)
        WRITE(*,9193) IOPT,ABS,PSI,RESID,RN4,RN5
 9193 FORMAT(/'E-Map no.',I4,' to be calculated:'//
     1              'Abs fom=',F7.2/'Psizero=',F7.2/'Residual=',F6.2/
     2'Nqest=',F6.3 /'Nqint=',F6.3/)
      ENDIF
      CALL P1
      NATS=NAT
      ICONT=0
      CALL PP1(X,NSIZE,NP(1),NP(2),NP(3),LMAP,NXVAL,NYVAL,NZVAL,ICONT)
      WRITE(*,9097)
 9097 FORMAT(/'Do you wish to proceed with peak search ? '\)
      CALL YN(III)
      IF(III.EQ.1) GOTO 53
C===> RESTORE ILEVEL,NAT,NPIC AND IONP()
      ILEVEL=LEVELS
      NAT=NATS
      NPIC=NPICS
      DO 1012 J=1,200
      DO 1012 K=1,6
 1012 IONP(K,J)=ISONP(K,J)
      CALL SEARCH(NSYMOP,LMAP,NXVAL,NYVAL,NZVAL,NP(1),NP(2),NP(3),ICONT,NELSE)
   53 IPASS=IPASS+1
C===> OPTION TO LOOK AT MORE E-MAPS
      IF(MORE.EQ.1)GOTO 52
 1001 WRITE(*,9030)
 9030 FORMAT('**** **** Fourier calculation(s) complete **** ****')
C===> CLOSE SCRATCH FILE
   90 CLOSE(UNIT=NTAPEF,STATUS='DELETE')
      RETURN
C===> Premature end on phase file
    5 WRITE(*,9072)
 9072 FORMAT(/'Premature end on phase file/')
      CALL EXIT
C===> Error on phase file
    6 WRITE(*,9073) NLINES
 9073 FORMAT(/'Error on phase file around line number'  ,I5)
      CALL EXIT
      END
C
      SUBROUTINE P1
```

```
C===>  EXPAND DATA TO A FULL HEMISPHERE
       COMMON/IO/NIN, NOUT, NTAPEA, NTAPEB, NTAPEC, NTAPED, NTAPEE,
      1 NTAPEF, NTAPEG, IH(155), AA(200), ICH1(10), ICH2(10), ITLE(68), IR(68),
      2 ICALL, NRC, NCH, NREAD, IEND, NREF, PI, DTOR, MAXH, ISPFLG, ZERO, LK(32),
      3 IDEF, ICHK, ISYMP, LEVEL, IFLOH(25), KUSE(40), INIT, IRDY, RTOD, NULL,
      4 IDIF, IVDIF, KARLE, IHVY, IGPFL, NDIFF, ISPCH(30), ICLCK, ICNT
       COMMON /BLK1/ IHKL(10000), E(10000), IPHAZ(10000), IHH(100), G(2,100),
      1 LH(60), LLK(60), LL(60), EH(60), ED(60), NH(10), NO(10), NK, CVR(10),
      2 VVR(10), IOBUF(200), BUFIO(200), N(72), ICH(9), KX(10), PC(6), KIND(24),
      3 I1(3), I2(3), TAB(15), NAG(200), NINF(10), PAD1(8607)
       COMMON /BLK2/ IS(2,3,24), TS(3,24), NSYM, ICENT, LAT, IAPX,
      1 CELL(6), T(3,3), DXMAX(3), TL(3,4), FS(3,3,24), MLAT, NEQV, MCON, INT,
      2 FOM, KFRAG(20), NFRAG, LFRAG(20), MFRAG, NOFRAG, NN, MFRN, IFMT(20),
      3 KMATCH(20), ITYPE, NOINT, MNPPF, NUMB, NNO , PAD2(495)
       COMMON /BLK5/ NOJOIN, NOPROJ, TWOPI, FACTOR, RMAX, RMIN, DOUT,
      1 MH(3), GARB2, GARBAG, XLIM(3), NOSET, NPIC, NP(3), NEX, ANGMIN, ANGMAX,
      2 DMIN, DMAX, DMUT, DM, DFRG, ALN, NPATT, IVDU, NAT, NPC, SCALE, IPASS
       COMMON/SINCOS/SINE(450), NYSEC
       DIMENSION CFOM(2250), EP(10000), CX(9), MAXHKL(3), FO(60), SIGMA(60)
       EQUIVALENCE (CFOM(1), IPHAZ(1), EP(1)), (LK7, LK(7)), (LK8, LK(8)),
      1 (LK9, LK(9)), (LK16, LK(16)), (LK18, LK(18)), (KUSE5, KUSE(5)),
      2 (KUSE30, KUSE(30)), (LK10, LK(10)), (KUSE17, KUSE(17))
C===>
       FT1=3.0
       FT2=0.0
       FT3=1.5
       FT4=0.0
       NPIC=0
C===>  ITYPE = 0/1 FOR EMAP/WEIGHTED FOURIER
       IF(ITYPE.EQ.0) GOTO 10
C===>  WEIGHTED FOURIER
       REWIND NTAPEE
       READ(NTAPEE) ITLE, CX, NH, NO, NK, IAPX, B, SC, NSYM, ICENT, LATT, LAT, NGP,
      1 NN, KSYS, PTS, NAT, ANAT, CVR, VVR, NTOT, ISYMP, NDIFF, IDIF, IVDIF, KARLE, I,
      2 IGPFL
C===>  NOT WORKING YET - CHECK THAT CELL/CX ARE WHAT WE WANT
       READ(NTAPEE) (((TS(I,J), IS(K,I,J), K=1,2), I=1,3), J=1,NSYM)
       READ(NTAPEE) NINF(1), NAG(1)
C===>  ATOMS
C===>  REMEMBER NO DUMMY READS ON THE STRIDE
       NF=NAG(1)
       NREC=4*NF
       READ(NTAPEE)(JUNK, J=1,NREC)
       READ(NTAPEE)(JUNK, J=1,537)
       WRITE(NOUT,9001)
 9001  FORMAT(/////' Calculation of a weighted Fourier map'/1H ,37(1H-))
       WRITE(*,9002)
 9002  FORMAT(/' Weighted Fourier to be calculated')
       NUMB=0
C----  -----  READ FO AND SIGMA OFF NTAPEE EVEN THOUGH THEY WILL NOT BE USED ---
   22  READ(NTAPEE) LH, LLK, LL, EH, ED, FO, SIGMA
       DO 24 I=1,KUSE17
       IF(LH(I).LT.(-999)) GO TO 200
       NUMB=NUMB+1
       IHKL(NUMB)=LH(I)*LK16+(LLK(I)+LK7)*LK8+LL(I)+LK7
       MH(1)=MAX0(MH(1),IABS(LH(I)))
       MH(2)=MAX0(MH(2),IABS(LLK(I)))
       MH(3)=MAX0(MH(3),IABS(LL(I)))
       E(NUMB)=EH(I)
       EP(NUMB)=ED(I)
       IF(NUMB.EQ.KUSE30) GO TO 130
   24  CONTINUE
```

```
      GO TO  22
  130 WRITE(NOUT,140) KUSE30
  140 FORMAT(//25X,5HFirst,I6,  ' reflections accepted')
       WRITE(*,9141) KUSE30
 9141 FORMAT(/10X,'Only the first',I6,' Reflections were accepted')
      GO TO 200
C===>
C===> START E-MAP CALCULATIONS
C===>
   10 DO 185 I=1,NUMB
      E(I)=(FLOAT(MOD(IPHAZ(I),LK10))/100.0)*E(I)
      PHASE=DTOR*FLOAT(IPHAZ(I)/LK10)
      EP(I)=E(I)*SIN(PHASE)
  185 E(I)=E(I)*COS(PHASE)
  200 IF(ITYPE.EQ.1) THEN
      DO 205 I=1,6
  205 CELL(I)=CX(I)
      ENDIF
C===> CALCULATE NUMBER OF GRID POINTS FOR FOURIER MAP
      DO 280 I=1,3
      NP(I)=CELL(I)/FACTOR+0.5
      IF(I.EQ.3)  NP(I)=NP(I)+MOD(NP(I),2)
  250 NTEST=NP(I)
      DO 270 J=2,5
  260 IF(NTEST.NE.(NTEST/J)*J) GO TO 270
      NTEST=NTEST/J
      IF(NTEST.EQ.1) GO TO 280
      GO TO 260
  270 CONTINUE
      NP(I)=NP(I)+1
      IF(I.EQ.3)  NP(I)=NP(I)+1
      GO TO 250
  280 CONTINUE
      WRITE (NOUT,300) CELL,NP
  300 FORMAT (// 30X,1HA,9X,1HB,9X,1HC,6X,5HAlpha,5X,4HBeta,4X,5HGamma/
     1    3X,20HUnit cell parameters,3F10.3,3F9.1//
     2                                          2X,21HNumber of grid points,
     3  6X,2HNX,8X,2HNY,8X,2HNZ/ 6X,15Hin Fourier map ,3I10)
      WRITE(NOUT,1004) FACTOR
 1004 FORMAT(/6X,' The grid spacing is approximately',F6.3,' Angstroms')
      IF(IPNT.EQ.0)WRITE(NOUT,1006)
 1006 FORMAT(/  9X,' The map will not be printed')
      IF(IPNT.GT.0) WRITE(NOUT,1007) XLIM,ILEVEL
 1007 FORMAT(/9X,' The map will be printed from X=0.0 TO X=',F5.3,/,38X,
     1' Y=0.0 TO Y=',F5.3,/38X,' Z=0.0 TO Z=',F5.3/,
     29X' Points greater than ',I4,' will be underlined'/)
      IF (NP(1) .GT. 200) THEN
      WRITE (NOUT,320)
  320 FORMAT (1H0,22X,76HNX greater than 200. Increase array inut in SUB
     1ROUTINE OUTPUT of FFT program)
      STOP
      ENDIF
C===> GENERATE ONE COMPLETE HEMISPHERE OF REFLEXIONS
      DO 399 I=1,15
  399 TAB(I)=SIN(FLOAT(30*I)*DTOR)
      REWIND NTAPEE
      NEX = 0
      FSUM=0.0
      FT1=2.0*FT1
      IP5=LK8*(LK18+LK9+1)
      NBATCH=0
      DO 800 JN=1,NUMB
```

```
      A=E(JN)
      B=EP(JN)
      I1(1)=IHKL(JN)/LK16
      I1(2)=MOD(IHKL(JN),LK16)/LK8-LK7
      I1(3)=MOD(IHKL(JN),LK8)-LK7
      DO 700 J=1,NSYM
      DO 540 I=1,3
  540 I2(I)=0
      DO 560 I=1,3
      DO 550 K=1,2
      M=IABS(IS(K,I,J))
      IF(M.NE.0) I2(M)=I2(M)+I1(I)*ISIGN(1,IS(K,I,J))
  550 CONTINUE
  560 CONTINUE
      B1=1.0
      JS=1
      IF(I2(3)) 590,570,600
  570 IF(I2(1)) 590,580,600
  580 IF(I2(2)) 590,600,600
  590 B1=-1.0
      JS=-1
  600 KIND(J)=IP5+JS*(LK18*I2(1)+LK9*I2(2)+I2(3))
      DO 610 I=1,3
      IF(IABS(I2(I)).GE.NP(I)/2) GO TO 700
      IF(I2(I).GT.MH(I)) MH(I)=I2(I)
  610 CONTINUE
      IF(J.EQ.1) GO TO 630
      ND=J-1
      DO 620 I=1,ND
      IF(KIND(J).EQ.KIND(I)) GO TO 700
  620 CONTINUE
  630 NBATCH=NBATCH+1
      NEX=NEX+1
      NU=0
      DO 640 I=1,3
  640 NU=NU-I1(I)*INT(TS(I,J)*12.0+0.1)
      NU=MOD(NU,12)
      IF(NU.LE.0) NU=NU+12
      XC=TAB(NU+3)
      XS=TAB(NU)
      GR=XC*A-XS*B
      GI=(XS*A+XC*B)*B1
      FSUM=FSUM+SQRT(GR*GR+GI*GI)
      G(1,NBATCH)=GR
      G(2,NBATCH)=GI
      IHH(NBATCH)=KIND(J)
      IF(NBATCH.LT.100) GO TO 700
      WRITE(NTAPEE) IHH,G
      NBATCH=0
  700 CONTINUE
  800 CONTINUE
      IF(NBATCH.NE.0) WRITE(NTAPEE) IHH,G
      ENDFILE NTAPEE
      SCALE=3000.0/FSUM
      JN=JN-1
      WRITE(NOUT,901)MH,JN,NEX
  901 FORMAT( /30X,1HH,9X,1HK,9X,1HL/ 5X,16HMaximum indices ,3I10//10X,
     1I6,' reflections have been expanded to',I6,' reflections'/ )
      WRITE(*,9010) JN,NEX
 9010 FORMAT(/'A total of ',I6,' reflections were expanded to',I6,
     1' reflections'//'Fourier calculation now commencing')
      RETURN
```

```
      END
C
      SUBROUTINE YN(I)
C===> SCANS AN INPUT LINE. IF THE FIRST NON-BLANK CHARACTER IS A "Y"
C===> I IS SET TO 0 ; IF IT IS A "N" ,I IS SET TO 1
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),ICLCK,ICNT
      CHARACTER*4 IR,IH
      EQUIVALENCE (KUSE13,KUSE(13))
      I=0
  200 READ(*,9001) IR
 9001 FORMAT(68A1)
      DO 100 J=1,KUSE13
      IF(IR(J).EQ.IH(45)) GOTO 100
      IF(IR(J).EQ.'Y'.OR.IR(J).EQ.'y') RETURN
      IF(IR(J).EQ.'N'.OR.IR(J).EQ.'n') GOTO 110
      WRITE(*,9002)
 9002 FORMAT( /'Input command in error - type either Y or N'/)
      GOTO 200
  100 CONTINUE
      RETURN
  110 I=1
      RETURN
      END
C
      SUBROUTINE PP1 (X, NSIZE, NX, NY, NZ ,LMAP,NXVAL,NYVAL,NZVAL,ICONT)
      INTEGER*1 LMAP
      REAL X(NSIZE)
      DIMENSION LMAP(NZ,NX,NY),NXVAL(100),NYVAL(100),NZVAL(100)
      INTEGER  P1, P2, R, SKIP, RECS,  D(5)
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),ICLCK,ICNT
      COMMON /BLK2/ IS(2,3,24),TS(3,24),NSYM,ICENT,LAT,IAPX,
     1 CELL(6),T(3,3),DXMAX(3),TL(3,4),FS(3,3,24),NLAT,NEQV,NCON,INT,
     2 FOM,KFRAG(20),NFRAG,LFRAG(20),MFRAG,NOFRAG,NN,NFRM,IFMT(20),
     3 KMATCH(20),ITYPE,NOINT,MMPPF,PAD2(497)
      COMMON /BLK5/ NOJOIN,NOPROJ,TWOPI,FACTOR,RMAX,RMIN,DOUT,
     1 MH(3),GARB2,GARBAG,XLIM(3),NOSET,NPIC,NP(3),NEX,ANGMIN,ANGMAX,
     2 DMIN,DMAX,DMUT,DM,DFRG,ALN,NPATT,IVDU,NAT,NPC,SCALE,IPASS
      COMMON/SINCOS/SINE(450),NYSEC
       RMIN=0.
      RMAX=0.
      NSEC2=(NX+2)*NZ
      P1 = NSIZE/(2*NY*(MH(3) + 1))
      P2 = NSIZE/(NX*NZ)
      IF (P1 .LE. 0 .OR. P2 .LE. 0) GO TO 300
C===> PASS ONE. TRANSFORM ON X AND WRITE INTERMEDIATE RESULTS ON NTAPEF
      REWIND NTAPEF
      R = -MH(1)
  100 IF (R + P1 .GT. MH(1)) P1 = MH(1) + 1 - R
      CALL READ XL(X, NY, MH(3)+1, P1, R)
      D(1) = 2*(NY*P1*(MH(3)+1))
      D(2) = 2
      D(3) = D(1)
      D(4) = D(1)
      D(5) = 2*NY
```

```
      CALL CMPLFT (X(1), X(2), NSIZE, NY, D )
      CALL WRITE Y (X, NY, MH(3)+1, P1, P2)
      R = R + P1
      IF (R .LE. MH(1)) GO TO 100
C===> END OF PASS 1.  PASS 2 READS BACK THE INTERMEDIATE RESULTS,
C===> CALCULATES THE TRANSFORMS ON H AND L, AND WRITES OUT THE FINAL
C===> Y SECTIONS.
      REWIND NTAPEE
      WRITE(NTAPEE)CELL,NSYM,ICENT,LAT,TS,IS,IDIF,IVDIF
      WRITE(NTAPEE) NX,NZ,NY
      REWIND NTAPEF
      SKIP = 0
      R = 0
      P1 = NSIZE/(2*NY*(MH(3) + 1))
      RECS = (NY - 1)/P2
  200 IF (R + P2 .GT. NY) P2 = NY - R
      CALL READ HL (X, NX, NZ/2, P2, MH(1), MH(3), P1, SKIP, RECS)
      IF (R + P2 .LT. NY) REWIND NTAPEF
      SKIP = SKIP + 1
C===> X NOW CONTAINS INTERMEDIATE RESULTS STORED WITH L DOWN THE
C===> COLUMNS AND H ACROSS THE ROWS.
      D(1) = NX*NZ*P2
      D(2) = NZ
      D(3) = NZ*NX
      D(4) = 2*(MH(3) + 1)
      D(5) = 2
      CALL CMPLFT (X(1), X(2), NSIZE, NX, D)
      D(2) = 2
      D(3) = D(1)
      D(4) = D(1)
      D(5) = NZ
      CALL HERMFT (X(1), X(2), NSIZE, NZ/2, D)
C===> CONTOUR
      LIM=R+P2
      ICONT=0
      IF(LIM.GE.NY)ICONT=1
C===> CHECK FOR OVERFLOW OF ARRAY X( )
      NUMPT=P2*NX*NZ
      IF (NUMPT.GT.30000) THEN
        WRITE(*,9010)
 9010   FORMAT(/'Array X( ) in SUBROUTINE PP1 has overflowed.')
        CALL EXIT
      ENDIF
      CALL OUTY (X,LMAP, NZ, NX, P2, R, NY,ICONT,NIVAL,NYVAL,NZVAL)
      R = R + P2
      IF (R .LT. NY) GO TO 200
      WRITE(NTAPEE)RMIN,RMAX
      WRITE(NOUT,201)RMIN,RMAX
  201 FORMAT(/10X,' Function values in the map range from',F10.2,
     1' to ',F10.2)
      WRITE(*,9011) RMIN,RMAX
 9011 FORMAT(/'Fourier points range from ',F8.2,' to',F9.2)
      RETURN
  300 P1 = 2*NY*(MH(3)+1)
      P2 = NX * NZ
      WRITE (NOUT,350) NSIZE, P1, P2
  350 FORMAT(1H0,8X,37HRequest too large. Space available is,I10,
     1 33H and space requested is larger of,I10,4H and,I10)
      WRITE(*,9012)
 9012 FORMAT(/'Not enough room for Fourier calculations')
      CALL EXIT
      END
C
```

```
      SUBROUTINE READ KL( X,  NY,  NZ,  NX,  HS)
      INTEGER HS,  H,  HM,  HL
      REAL    X( 2, NY, NZ, NX)
      COMMON/IO/NIN, NOUT, NTAPEA, NTAPEB, NTAPEC, NTAPED, NTAPEE,
     1 NTAPEF, NTAPEG, IH( 155), A( 200), ICH1( 10), ICH2( 10), ITLE( 68), IR( 68),
     2 ICALL, NRC, NCH, NREAD, IEND, NREF, PI, DTOR, MAXH, ISPFLG, ZERO, LK( 32),
     3 IDEF, ICHK, ISYMP, LEVEL, IFLOW( 25), KUSE( 40), INIT, IRDY, RTOD, NULL,
     4 IDIF, IVDIF, KARLE, IHVY, IGPFL, NDIFF, ISPCH( 30), ICLCK, ICNT
      COMMON /BLK1/ XDM( 30000), IMAP( 8000), IHH( 100), G( 2, 100), INUT( 200),
     1 PAD1( 1500)
      COMMON /BLK5/ NOJOIN, NOPROJ, THOPI, FACTOR, RMAX, RMIN, DOUT,
     1 MH( 3), GARB2, GARBAG, XLIM( 3), NOSET, NPIC, NP( 3), NEX, ANGMIN, ANGMAX,
     2 DMIN, DMAX, DMUT, DM, DFRG, ALN, NPATT, IVDU, NAT, NPC, SCALE, IPASS
      EQUIVALENCE ( LK18, LK( 18)), ( LK9, LK( 9)), ( LK8, LK( 8))
      REWIND NTAPEE
      NBLOCK = NEX/100+1
      HM=LK18*( HS+NX-1+LK8)+( LK9+1)*( 2*LK8-1)
      HL=LK18*( HS+LK8)
      DO 100 H = 1,  NX
      DO 100 L = 1,  NZ
      DO 100 K = 1,  NY
      X( 1, K, L, H)=0. 0
  100 X( 2, K, L, H)=0. 0
      M=100
      DO 500 IB=1, NBLOCK
      IF( IB. EQ. NBLOCK)  M=MOD( NEX, 100)
      IF( M. EQ. 0)  GO TO 500
      READ( NTAPEE)  IHH, G
      DO 300 J = 1,  M
      IHJ=IHH( J)
      IF( IHJ. GT. HM. OR. IHJ. LT. HL)  GO TO 300
      H=IHJ/LK18-LK8
      L=IHJ-LK18*( H+LK8)
      K=L/LK9-LK8
      L=L-LK9*( K+LK8)-LK8
      NOKO = 0
      IF ( H. EQ. 0. AND. L. EQ. 0. AND. K. NE. 0)  NOKO = NY-K+1
      H = H - HS + 1
      IF( K. LT. 0) K=NY+K
      K = K + 1
      L = L + 1
      X( 1, K, L, H)=G( 1, J)
      X( 2, K, L, H)=G( 2, J)
      IF ( NOKO. EQ. 0)  GOTO 300
      X( 1, NOKO, L, H)=X( 1, K, L, H)
      X( 2, NOKO, L, H)=-X( 2, K, L, H)
  300 CONTINUE
  500 CONTINUE
      RETURN
      END
C
      SUBROUTINE WRITE Y ( X,  NY,  NZ,  NX,  SIZE)
      INTEGER SIZE,  H,  P,  Q,  R, RECSIZ
      REAL    X( 2, NY, NZ, NX)
      COMMON/IO/NIN, NOUT, NTAPEA, NTAPEB, NTAPEC, NTAPED, NTAPEE,
     1 NTAPEF, NTAPEG, IH( 155), A( 200), ICH1( 10), ICH2( 10), ITLE( 68), IR( 68),
     2 ICALL, NRC, NCH, NREAD, IEND, NREF, PI, DTOR, MAXH, ISPFLG, ZERO, LK( 32),
     3 IDEF, ICHK, ISYMP, LEVEL, IFLOW( 25), KUSE( 40), INIT, IRDY, RTOD, NULL,
     4 IDIF, IVDIF, KARLE, IHVY, IGPFL, NDIFF, ISPCH( 30), ICLCK, ICNT
      P = SIZE
      Q = 0
  100 R = Q + 1
      IF ( Q + P . GT.  NY)  P = NY - Q
```

```
      Q = Q + P
C===> Calculate how large the record will be
      RECSIZ=NX*(Q-R+1)*NZ*2
      WRITE (NTAPEF) RECSIZ,(((X(1,K,L,H),X(2,K,L,H), H = 1, NX),
     1  K = R, Q),  L=1,NZ)
      IF (Q .LT. NY) GO TO 100
      RETURN
      END
C
      SUBROUTINE READ HL (X, NX, NZ, NY, HMAX, LMAX, SIZE, SKIP, RECS)
C===> READS IN INTERMEDIATE RESULTS FOR ALL H AND L FOR A BLOCK OF K.
C===> THE INPUT IS STORED WITH L DOWN THE COLUMNS AND H ON THE ROWS.
      INTEGER HMAX, SIZE, SKIP ,RECS, H, HL, HU, P, Q,RECSIZ
      REAL    X(2,NZ,NX,NY)
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INIT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),ICLCK,ICNT
      LM = LMAX + 1
      P = SIZE
      HU = NX - HMAX
      IF (SKIP .LE. 0) GO TO 200
      DO 100 Q = 1, SKIP
  100 READ (NTAPEF)RECSIZ,(RUBISH,L=1,RECSIZ)
C===> READ DATA FOR NEGATIVE H
  200 IF (HU + P .GT. NX) GO TO 400
      HL = HU + 1
      HU = HU + P
      READ (NTAPEF)RECSIZ, (((X(1,L,H,K),X(2,L,H,K), H = HL, HU), K = 1, NY),
     1L=1,LM)
      IF (RECS .LE. 0) GO TO 200
      DO 300 Q = 1, RECS
  300 READ (NTAPEF)RECSIZ,(RUBISH,L=1,RECSIZ)
      GO TO 200
C===> GO PICK UP RECORD WHICH SPANS H = 0 IF NECESSARY
  400 IF (HU .NE. NX) GO TO 700
      HU = 0
C===> READ RECORDS FOR POSITIVE H
  500 IF (HU + P .GT. HMAX + 1) P = HMAX + 1 - HU
      HL = HU + 1
       HU = HU + P
      READ (NTAPEF)RECSIZ, (((X(1,L,H,K),X(2,L,H,K), H = HL, HU),
     1 K = 1, NY),  L=1,LM)
  550 IF (HU .EQ. HMAX + 1) GO TO 800
      IF (RECS .LE. 0) GO TO 500
      DO 600 Q = 1, RECS
  600 READ (NTAPEF)RECSIZ,(RUBISH,L=1,RECSIZ)
      GO TO 500
C===> SECTION TO READ A RECORD SPANNING H = 0
  700 HL = HU + 1
      HU = HU + P - NX
      IF (HU .GT. HMAX + 1) HU = HMAX + 1
      READ (NTAPEF) RECSIZ,(((X(1,L,H,K),X(2,L,H,K), H = HL, NX),
     1  (X(1,L,H,K),  X(2,L,H,K),H=1,HU),K=1,NY),L=1,LM)
      GO TO 550
C===> PAD THE ARRAY WITH ZERO WHERE THERE ARE NO DATA, AND FILL IN THE
C===> -H K 0 RESULTS BY SYMMETRY.
  800 DO 900 H = 2, HU
      HL = NX + 2 - H
      DO 900 K = 1, NY
      X(1,1,HL,K)=X(1,1,H,K)
      X(2,1,HL,K)=-X(2,1,H,K)
```

```
  900 CONTINUE
      HL = HMAX + 2
      HU = NX - HMAX
      IF (HU .LT. HL) GO TO 920
      DO 910 L = 1, LM
      DO 910 K = 1, NY
      DO 910 H = HL, HU
      X(1,L,H,K)=0.0
      X(2,L,H,K)=0.0
  910 CONTINUE
  920 IF (LM .GE. NZ) GO TO 940
      P = LM + 1
      DO 930 K = 1, NY
      DO 930 L = P, NZ
      DO 930 H = 1, NX
      X(1,L,H,K)=0.0
      X(2,L,H,K)=0.0
  930 CONTINUE
  940 RETURN
      END
C
      SUBROUTINE OUTY (X,LMAP, NZ, NX, NY, Y, NYT,ICONT,NXVAL,NYVAL,NZVAL)
      INTEGER*1 LMAP
      DIMENSION X(NZ,NX,NY),LMAP(NZ,NX,NYT),NXVAL(100),NYVAL(100),NZVAL(100)
      INTEGER STAR,Y,SEC,XL,XU
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),ICLCK,ICNT
      COMMON /BLK1/ZZ(30000),IMAP(8000),IHH(100),G(2,100),PAD1(1700)
      COMMON /BLK2/ IS(2,3,24),TS(3,24),NSYM,ICENT,LAT,IAPX,
     1 CELL(6),T(3,3),DXMAX(3),TL(3,4),FS(3,3,24),MLAT,NEQV,MCON,INT,
     2 FOM,KFRAG(20),NFRAG,LFRAG(20),MFRAG,NOFRAG,NN,MFRN,IPMT(20),
     3 PAD2(520)
      COMMON /BLK5/ NOJOIN,NOPROJ,TWOPI,FACTOR,RMAX,RMIN,DOUT,
     1 MH(3),GARB2,GARBAG,XLIM(3),NOSET,MPIC,MP(3),NEX,ANGMIN,ANGMAX,
     2 DMIN,DMAX,DMUT,DM,DFRG,ALN,NPATT,IVDU,NAT,NPC,SCALE,IPASS
      COMMON/SINCOS/SINE(450),NYSEC
C===> CHECK ON SIZE OF THE CONTOUR MAP
      NPOINT=NX*NYT*NZ
      IF(NPOINT.GT.1000000.OR.NX.GT.150.OR.NYT.GT.150.OR.NZ.GT.150)THEN
      ICNT=0
      IF(ICONT.EQ.1)THEN
      WRITE(*,9000)
 9000 FORMAT(/'****** UNABLE TO DO CONTOURED SECTIONS - MAP IS TOO BIG ******'
     1)
      ENDIF
      ENDIF
C===> PUT OUT SCALED Y-SECTIONS
   20 IF(IPNT.EQ.0.AND.ICNT.EQ.0) GO TO 250
      NXM=MIN0(NX,INT(XLIM(1)*FLOAT(NX))+1)
      NYM=MIN0(NYT,INT(XLIM(2)*FLOAT(NYT))+1)
      NZM=MIN0(NZ,INT(XLIM(3)*FLOAT(NZ))+1)
      ASC=1000.0/FLOAT(NX)
C===> GET X VALUES
      DO 30 II=1,NX
      NXVAL(II)=FLOAT((II-1))*ASC+0.5
   30 CONTINUE
  250 DO 900 K = 1, NY
      NYSEC=NYSEC+1
      SEC = Y + K - 1
      DO 350 J=1,NZ
```

```
      KK=(J-1)*(NX+2)
      DO 330 I=1,NX
      XX=X(J,I,K)*SCALE
      RMIN=AMIN1(RMIN,XX)

      RMAX=AMAX1(RMAX,XX)
      KK=KK+1
      IMAP(KK)=0
      IF(XX.NE.0.0) IMAP(KK)=XX+SIGN(0.5,XX)
  330 CONTINUE
      IMAP(KK+1)=IMAP(KK+1-NX)
      IMAP(KK+2)=IMAP(KK+2-NX)
  350 CONTINUE
      KK=KK+2
      WRITE(NTAPEE)(IMAP(I),I=1,KK)
      IF(ICNT.EQ.0) GOTO 900
      NYVAL(NYSEC)=1000.*(FLOAT(SEC))/(FLOAT(NYT))+0.5
      DO 700 I = 1,NZM
      NZVAL(I)=FLOAT((I-1))*1000./FLOAT(NZ)+0.5
      DO 600 J =1,NXM
      XX=X(I,J,K)*SCALE
C===> SCALE ALL POINTS TO BE OF MAGNITUDE BETWEEN 0 AND 1000
      LMAP(I,J,NYSEC)=INT((100.0/RMAX)*XX+SIGN(0.5,XX))
  600 CONTINUE
  700 CONTINUE
  900 CONTINUE
      RETURN
      END
C
      SUBROUTINE CMPLFT (X, Y, NSIZE, N, D )
C===> COMPLEX FINITE DISCRETE FOURIER TRANSFORM
C===> TRANSFORMS ONE DIMENSION OF MULTI-DIMENSIONAL DATA
C===> MODIFIED BY L. F. TEN EYCK FROM A ONE-DIMENSIONAL VERSION WRITTEN
C===> BY G. T. SANDE, 1969.
C===> THIS SUBROUTINE CALCULATES THE TRANSFORM
C===>           (X(T) + I*Y(T))*(COS(2*PI*T/N) - I*SIN(2*PI*T/N))
      REAL X(NSIZE),Y(NSIZE)
      INTEGER D(5),PMAX,PSYM,TWOGRP,FACTOR(15),SYM(15),UNSYM(15)
      P MAX=5
      TWO GRP = 4
      CALL S R FP (N, P MAX, TWO GRP, FACTOR, SYM, P SYM, UN SYM)
      CALL MDFTKD (N, FACTOR, D, X, Y, NSIZE)
      CALL DIPRP (N, SYM, P SYM, UN SYM, D, X, Y, NSIZE)
      RETURN
      END
C
      SUBROUTINE S R FP (PTS,PMAX,TWO GRP,FACTOR,SYM,P SYM,UN SYM)
C===> SYMMETRIZED REORDERING FACTORING SUBROUTINE
      INTEGER PTS,PMAX,TWO GRP,P SYM, FACTOR (10), SYM (10), UN SYM (10)
      INTEGER PP(14), QQ (7), F,P,P TWO,Q,R
      NEST=14
      N=PTS
      P SYM=1
      F=2
      P=0
      Q=0
  100 IF (N.LE.1) GO TO 500
      DO 200 J=F,PMAX
      IF (N.EQ.(N/J)*J) GO TO 300
  200 CONTINUE
      WRITE(8,9001)
 9001 FORMAT(/'Error in SUBROUTINE SRFP. This is a programming problem'/)
```

```
      STOP
  300 F=J
      N=N/F
      IF (N.EQ.(N/F)*F) GO TO 400
      Q=Q+1
      QQ(Q)=F
      GO TO 100
  400 N=N/F
      P=P+1
      PP(P)=F
      P SYM=P SYM*F
      GO TO 100
  500 R=1
      IF (Q.EQ.0) R=0
      IF (P.LT.1) GO TO 700
      DO 600 J=1,P
      JJ=P+1-J
      SYM(J)=PP(JJ)
      FACTOR(J)=PP(JJ)
      JJ=P+Q+J
      FACTOR(JJ)=PP(J)
      JJ=P+R+J
  600 SYM(JJ)=PP(J)
  700 IF (Q.LT.1) GO TO 900
      DO 800 J=1,Q
      JJ=P+J
      UN SYM(J)=QQ(J)
  800 FACTOR(JJ)=QQ(J)
      SYM(P+1)=PTS/(PSYM*PSYM)
  900 JJ=2*P+Q
      FACTOR(JJ+1)=0
      P TWO=1
      J=0
 1000 J=J+1
      IF (FACTOR(J).EQ.0) GO TO 1200
      IF (FACTOR(J).NE.2) GO TO 1000
      P TWO=P TWO*2
      FACTOR(J)=1
      IF (P TWO.GE.TWO GRP) GO TO 1100
      IF (FACTOR(J+1).EQ.2) GO TO 1000
 1100 FACTOR(J)=P TWO
      P TWO=1
      GO TO 1000
 1200 IF (P.EQ.0) R=0
      JJ=2*P+R
      SYM(JJ+1)=0
      IF (Q.LE.1) Q=0
      UN SYM(Q+1)=0
      RETURN
      END
C
      SUBROUTINE DIPRP (PTS, SYM, P SYM, UN SYM, DIM, X, Y, NSIZE)
C===> DOUBLE IN PLACE REORDERING PROGRAM
      REAL X(NSIZE), Y(NSIZE)
      INTEGER SYM(10), UN SYM(10), DIM(5), PTS, PSYM, DK, P UN SYM, TEST
      LOGICAL ONE MOD
      INTEGER SEP, DELTA, P, P0, P1, P2, P3, P4, P5, SIZE
      INTEGER V(14), MODULO(14), S(14), U(14)
      NEST=14
      NT = DIM(1)
      SEP = DIM(2)
      P2 = DIM(3)
```

```
      SIZE = DIM(4) - 1
      P4 = DIM(5)
      IF (SYM(1).EQ.0) GO TO 500
      DO 100 J=1,NEST
      U(J)=1
  100 S(J)=1
      N=PTS
      DO 200 J=1,NEST
      IF (SYM(J).EQ.0) GO TO 300
      JJ=NEST+1-J
      U(JJ)=N
      N=N/SYM(J)
  200 S(JJ)=N
  300 JJ=0
      L=1
      V(1)=1
  310 L=L+1
      V(L)=V(L-1)
  320 IF(L.LT.NEST) GO TO 310
      N=V(NEST)
      JJ=JJ+1
      IF (JJ.GE.N) GO TO 400
      DELTA = (N-JJ)*SEP
      P1 = (JJ-1)*SEP + 1
      DO 350 P0 = P1, NT, P2
      P3 = P0 + SIZE
      DO 350 P = P0, P3, P4
      P5 = P + DELTA
      T = X(P)
      X(P) = X(P5)
      X(P5) = T
      T = Y(P)
      Y(P) = Y(P5)
      Y(P5) = T
  350 CONTINUE
  400 V(L)=V(L)+S(L)
      IF(V(L).LE.U(L)) GO TO 320
      L=L-1
      IF(L.NE.0) GO TO 400
  500 IF (UN SYM(1).EQ.0) GO TO 1900
       P UN SYM=PTS/(PSYM*PSYM)
       MULT=P UN SYM/UN SYM(1)
       TEST=(UN SYM(1)*UN SYM(2)-1)*MULT*P SYM
       LLK=MULT
       DK=MULT
       DO 600 K=2,NEST
       IF (UN SYM(K).EQ.0) GO TO 700
       LLK=LLK*UN SYM(K-1)
       III=UN SYM(K)
       JJJ=DK
       IK=JJJ/III
       U(K)=(LLK-IK)*P SYM
       DK=IK
  600  MODS=K
  700 ONE MOD=MODS.LT.3
      IF (ONE MOD) GO TO 900
      DO 800 J=3,MODS
      JJ=MODS+3-J
  800 MODULO(JJ)=U(J)
  900 MODULO(2)=U(2)
      JL=(P UN SYM-3)*P SYM
      MS=P UN SYM*P SYM
```

```
      DO 1800 J=P SYM, JL, P SYM
      K=J
 1000 K=K*MULT
      IF (ONE MOD) GO TO 1200
      DO 1100 I=3, MODS
      K=K-(K/MODULO(I))*MODULO(I)
 1100 CONTINUE
 1200 IF (K.GE.TEST) GO TO 1300
      K=K-(K/MODULO(2))*MODULO(2)
      GO TO 1400
 1300 K=K-(K/MODULO(2))*MODULO(2)+MODULO(2)
 1400 IF (K.LT.J) GO TO 1000
      IF(K.EQ.J) GOTO 1800
      DELTA = (K-J)*SEP
      DO 1600 L=1, P SYM
      DO 1500 M=L, PTS, MS
      P1 = (M+J-1)*SEP + 1
      DO 1500 P0 = P1, NT, P2
      P3 = P0 + SIZE
      DO 1500 JJ = P0, P3, P4
      KK = JJ + DELTA
      T=X(JJ)
      X(JJ)=X(KK)
      X(KK)=T
      T=Y(JJ)
      Y(JJ)=Y(KK)
      Y(KK)=T
 1500 CONTINUE
 1600 CONTINUE
 1800 CONTINUE
 1900 RETURN
      END
C
      SUBROUTINE MDFTKD (N, FACTOR, DIM, X, Y, NSIZE)
C===> MULTI-DIMENSIONAL COMPLEX FOURIER TRANSFORM KERNEL DRIVER
      INTEGER FACTOR(10), DIM(5), F, P, R, S
      REAL X(NSIZE), Y(NSIZE)
      S = DIM(2)
      F = 0
      M = N
  100 F = F + 1
      P = FACTOR(F)
      IF (P.EQ.0) RETURN
      M = M/P
      R = M*S
      GO TO (100, 200, 300, 400, 500), P
  200 CALL R2 CFTK (N, M, X(1), Y(1), X(R+1), Y(R+1), DIM)
      GO TO 100
  300 CALL R3 CFTK (N, M, X(1), Y(1), X(R+1), Y(R+1), X(2*R+1), Y(2*R+1)
     1, DIM)
      GO TO 100
  400 CALL R4 CFTK (N, M, X(1), Y(1), X(R+1), Y(R+1), X(2*R+1), Y(2*R+1)
     1, X(3*R+1), Y(3*R+1), DIM)
      GO TO 100
  500 CALL R5 CFTK (N, M, X(1), Y(1), X(R+1), Y(R+1), X(2*R+1), Y(2*R+1)
     1, X(3*R+1), Y(3*R+1), X(4*R+1), Y(4*R+1), DIM)
      GO TO 100
      END
C
      SUBROUTINE R2 CFTK (N, M, X0, Y0, X1, Y1, DIM)
C===> RADIX 2 MULTI-DIMENSIONAL COMPLEX FOURIER TRANSFORM KERNEL
      INTEGER DIM(5), SIZE, SEP
```

```
      REAL X0(10), Y0(10), X1(10), Y1(10), IS, IU
      LOGICAL FOLD,ZERO
      COMMON /BLK5/ NOJOIN,NOPROJ,TWOPI,FACTOR,RMAX,RMIN,DOUT,
     1 MH(3),GARB2,GARBAG,XLIM(3),NOSET,NPIC,NP(3),NEX,ANGMIN,ANGMAX,
     2 DMIN,DMAX,DMUT,DM,DFRG,ALN,NPATT,IVDU,NAT,NPC,SCALE,IPASS
      COMMON/SINCOS/SINE(450),NYSEC
      NT = DIM(1)
      SEP = DIM(2)
      L1 = DIM(3)
      SIZE = DIM(4) - 1
      K2 = DIM(5)
      NS = N*SEP
      M2=M*2
      FM2 = FLOAT(M2)
      RECFM2=1.0/FM2
      M OVER 2=M/2+1
      MM2 = SEP*M2
      FJM1 = -1.0
      DO 600 J=1,M OVER 2
      FOLD=J.GT.1 .AND. 2*J.LT.M+2
      K0 = (J-1)*SEP + 1
      FJM1 = FJM1 + 1.0
      ANGLE = FJM1*RECFM2*360.0
      ZERO=ANGLE.EQ.0.0
      IF (ZERO) GO TO 200
      LCS=INT(ANGLE)+1
C===> USE A SINE/COSINE LOOK-UP TABLE
  710 IF(LCS.LE.360) GOTO 700
      LCS=LCS-360
      GOTO 710
  700 IF(LCS.GE.1) GOTO 720
      LCS=LCS+360
      GOTO 700
  720 S=SINE(LCS)
      C=SINE(LCS+90)
      GO TO 200
  100 FOLD=.FALSE.
      K0 = (M+1-J)*SEP + 1
      C=-C
  200 DO 500 KK = K0, NS, MM2
      DO 440 L = KK, NT, L1
      K1 = L + SIZE
      DO 420 K = L, K1, K2
      RS=X0(K)+X1(K)
      IS=Y0(K)+Y1(K)
      RU=X0(K)-X1(K)
      IU=Y0(K)-Y1(K)
      X0(K)=RS
      Y0(K)=IS
      IF (ZERO) GO TO 300
      X1(K)=RU*C+IU*S
      Y1(K)=IU*C-RU*S
      GO TO 420
  300 X1(K)=RU
      Y1(K)=IU
  420 CONTINUE
  440 CONTINUE
  500 CONTINUE
      IF (FOLD) GO TO 100
  600 CONTINUE
      RETURN
      END
```

```
C
      SUBROUTINE R3 CFTK (N, M, X0, Y0, X1, Y1, X2, Y2, DIM)
C===> RADIX 3 MULTI-DIMENSIONAL COMPLEX FOURIER TRANSFORM KERNEL
      INTEGER DIM(5), SIZE, SEP
      REAL X0(10),Y0(10),X1(10),Y1(10),X2(10),Y2(10), I0,I1,I2,IA,IB,IS
      LOGICAL FOLD,ZERO
      COMMON /BLK5/ NOJOIN,NOPROJ,TWOPI,FACTOR,RMAX,RMIN,DOUT,
     1 MH(3),GARB2,GARBAG,XLIM(3),NOSET,NPIC,NP(3),NEX,ANGMIN,ANGMAX,
     2 DMIN,DMAX,DMUT,DM,DFRG,ALN,NPATT,IVDU,NAT,NPC,SCALE,IPASS
      COMMON/SINCOS/SINE(450),NYSEC
      A=-.5
      B=0.8660254
      NT = DIM(1)
      SEP = DIM(2)
      L1 = DIM(3)
      SIZE = DIM(4) - 1
      K2 = DIM(5)
      NS = N*SEP
      M3=M*3
      FM3 = FLOAT(M3)
      RECFM3=1.0/FM3
      MM3 = SEP*M3
      M OVER 2=M/2+1
      FJM1 = -1.0
      DO 600 J=1,M OVER 2
      FOLD=J.GT.1 .AND. 2*J.LT.M+2
      K0 = (J-1)*SEP + 1
      FJM1 = FJM1 + 1.0
      ANGLE = FJM1*RECFM3*360.0
      ZERO=ANGLE.EQ.0.0
      IF (ZERO) GO TO 200
      LCS=INT(ANGLE)+1
C===> USE A SINE/COSINE LOOK-UP TABLE
  710 IF(LCS.LE.360) GOTO 700
      LCS=LCS-360
      GOTO 710
  700 IF(LCS.GE.1) GOTO 720
      LCS=LCS+360
      GOTO 700
  720 S1=SINE(LCS)
      C1=SINE(LCS+90)
      C2=C1*C1-S1*S1
      S2=S1*C1+C1*S1
      GO TO 200
  100 FOLD=.FALSE.
      K0 = (M+1-J)*SEP + 1
      T=C1*A+S1*B
      S1=C1*B-S1*A
      C1=T
      T=C2*A-S2*B
      S2=-C2*B-S2*A
      C2=T
  200 DO 500 KK = K0, NS, MM3
      DO 440 L = KK, NT, L1
      K1 = L + SIZE
      DO 420 K = L, K1, K2
      R0=X0(K)
      I0=Y0(K)
      RS=X1(K)+X2(K)
      IS=Y1(K)+Y2(K)
      X0(K)=R0+RS
      Y0(K)=I0+IS
```

```
      RA=R0+RS*A
      IA=I0+IS*A
      RB=(X1(K)-X2(K))*B
      IB=(Y1(K)-Y2(K))*B
      IF (ZERO) GO TO 300
      R1=RA+IB
      I1=IA-RB
      R2=RA-IB
      I2=IA+RB
      X1(K)=R1*C1+I1*S1
      Y1(K)=I1*C1-R1*S1
      X2(K)=R2*C2+I2*S2
      Y2(K)=I2*C2-R2*S2
      GO TO 420
  300 X1(K)=RA+IB
      Y1(K)=IA-RB
      X2(K)=RA-IB
      Y2(K)=IA+RB
  420 CONTINUE
  440 CONTINUE
  500 CONTINUE
      IF (FOLD) GO TO 100
  600 CONTINUE
      RETURN
      END
C
      SUBROUTINE R4 CFTK (N, M, X0, Y0, X1, Y1, X2, Y2, X3, Y3, DIM)
C===> RADIX 4 MULTI-DIMENSIONAL COMPLEX FOURIER TRANSFORM KERNEL
      INTEGER DIM(5), SIZE, SEP
      REAL X0(10),Y0(10),X1(10),Y1(10),X2(10),Y2(10),X3(10),Y3(10)
      LOGICAL FOLD,ZERO
      REAL I1,I2,I3,IS0,IS1,IU0,IU1
      COMMON /BLK5/ NOJOIN,NOPROJ,TWOPI,FACTOR,RMAX,RMIN,DOUT,
     1 MH(3),GARB2,GARBAG,XLIM(3),NOSET,NPIC,NP(3),NEX,ANGMIN,ANGMAX,
     2 DMIN,DMAX,DMUT,DM,DFRG,ALN,NPATT,IVDU,NAT,NPC,SCALE,IPASS
      COMMON/SINCOS/SINE(450),NYSEC
      NT = DIM(1)
      SEP = DIM(2)
      L1 = DIM(3)
      SIZE = DIM(4) - 1
      K2 = DIM(5)
      NS = N*SEP
      M4=M*4
      FM4 = FLOAT(M4)
      RECFM4=1.0/FM4
      MM4 = SEP*M4
      M OVER 2=M/2+1
      FJM1 = -1.0
      DO 600 J=1,M OVER 2
      FOLD=J.GT.1 .AND. 2*J.LT.M+2
      K0 = (J-1)*SEP + 1
      FJM1 = FJM1 + 1.0
      ANGLE = FJM1*RECFM4*360.0
      ZERO=ANGLE.EQ.0.0
      IF (ZERO) GO TO 200
      LCS=INT(ANGLE)+1
C===> USE A SINE/COSINE LOOK-UP TABLE
  710 IF(LCS.LE.360) GOTO 700
      LCS=LCS-360
      GOTO 710
  700 IF(LCS.GE.1) GOTO 720
      LCS=LCS+360
      GOTO 700
```

```
  720 S1=SINE(LCS)
      C1=SINE(LCS+90)
      C2=C1*C1-S1*S1
      S2=S1*C1+C1*S1
      C3=C2*C1-S2*S1
      S3=S2*C1+C2*S1
      GO TO 200
  100 FOLD=.FALSE.
      K0 = (M+1-J)*SEP + 1
      T=C1
      C1=S1
      S1=T
      C2=-C2
      T=C3
      C3=-S3
      S3=-T
  200 DO 500 KK = K0, NS, MM4
      DO 440 L = KK, NT, L1
      K1 = L + SIZE
      DO 420 K = L, K1, K2
      RS0=X0(K)+X2(K)
      IS0=Y0(K)+Y2(K)
      RU0=X0(K)-X2(K)
      IU0=Y0(K)-Y2(K)
      RS1=X1(K)+X3(K)
      IS1=Y1(K)+Y3(K)
      RU1=X1(K)-X3(K)
      IU1=Y1(K)-Y3(K)
      X0(K)=RS0+RS1
      Y0(K)=IS0+IS1
      IF (ZERO) GO TO 300
      R1=RU0+IU1
      I1=IU0-RU1
      R2=RS0-RS1
      I2=IS0-IS1
      R3=RU0-IU1
      I3=IU0+RU1
      X2(K)=R1*C1+I1*S1
      Y2(K)=I1*C1-R1*S1
      X1(K)=R2*C2+I2*S2
      Y1(K)=I2*C2-R2*S2
      X3(K)=R3*C3+I3*S3
      Y3(K)=I3*C3-R3*S3
      GO TO 420
  300 X2(K)=RU0+IU1
      Y2(K)=IU0-RU1
      X1(K)=RS0-RS1
      Y1(K)=IS0-IS1
      X3(K)=RU0-IU1
      Y3(K)=IU0+RU1
  420 CONTINUE
  440 CONTINUE
  500 CONTINUE
      IF (FOLD) GO TO 100
  600 CONTINUE
      RETURN
      END
C
      SUBROUTINE R5 CFTK(N,M,X0, Y0, X1, Y1, X2, Y2, X3, Y3, X4, Y4,DIM)
C===> RADIX 5 MULTI-DIMENSIONAL COMPLEX FOURIER TRANSFORM KERNEL
      INTEGER DIM(5), SIZE, SEP
      REAL X0(10),Y0(10),X1(10),Y1(10),X2(10),Y2(10),X3(10),Y3(10),
```

```
     1 X4(10),Y4(10),I0,I1,I2,I3,I4,IA1,IA2,IB1,IB2,IS1,IS2,IU1,IU2
       LOGICAL FOLD,ZERO
       COMMON /BLK5/ NOJOIN,NOPROJ,THOPI,FACTOR,RMAX,RMIN,DOUT,
     1 MH(3),GARB2,GARBAG,XLIM(3),NOSET,NPIC,NP(3),NEX,ANGMIN,ANGMAX,
     2 DMIN,DMAX,DMUT,DM,DFRG,ALN,NPATT,IVDU,NAT,NPC,SCALE,IPASS
       COMMON/SINCOS/SINE(450),NYSEC
       A1=0.30901699
       B1=0.95105652
       A2=-.80901699
       B2=0.58778525
       NT = DIM(1)
       SEP = DIM(2)
       L1 = DIM(3)
       SIZE = DIM(4) - 1
       K2 = DIM(5)
       NS = N*SEP
       M5=M*5
       FM5 = FLOAT(M5)
       RECFM5=1.0/FM5
       MM5 = SEP*M5
       M OVER 2=M/2+1
       FJM1 = -1.0
       DO 600 J=1,M OVER 2
       FOLD=J.GT.1 .AND. 2*J.LT.M+2
       K0 = (J-1)*SEP + 1
       FJM1 = FJM1 + 1.0
       ANGLE = FJM1*RECFM5*360.0
       ZERO=ANGLE.EQ.0.0
       IF (ZERO) GO TO 200
       LCS=INT(ANGLE)+1
C===>  USE A SINE/COSINE LOOK-UP TABLE
   710 IF(LCS.LE.360) GOTO 700
       LCS=LCS-360
       GOTO 710
   700 IF(LCS.GE.1) GOTO 720
       LCS=LCS+360
       GOTO 700
   720 S1=SINE(LCS)
       C1=SINE(LCS+90)
       C2=C1*C1-S1*S1
       S2=S1*C1+C1*S1
       C3=C2*C1-S2*S1
       S3=S2*C1+C2*S1
       C4=C2*C2-S2*S2
       S4=S2*C2+C2*S2
       GO TO 200
   100 FOLD=.FALSE.
       K0 = (M+1-J)*SEP + 1
       T=C1*A1+S1*B1
       S1=C1*B1-S1*A1
       C1=T
       T=C2*A2+S2*B2
       S2=C2*B2-S2*A2
       C2=T
       T=C3*A2-S3*B2
       S3=-C3*B2-S3*A2
       C3=T
       T=C4*A1-S4*B1
       S4=-C4*B1-S4*A1
       C4=T
   200 DO 500 KK = K0, NS, MM5
       DO 440 L = KK, NT, L1
```

```
      K1 = L + SIZE
      DO 420 K = L, K1, K2
      R0=X0(K)
      I0=Y0(K)
      RS1=X1(K)+X4(K)
      IS1=Y1(K)+Y4(K)
      RU1=X1(K)-X4(K)
      IU1=Y1(K)-Y4(K)
      RS2=X2(K)+X3(K)
      IS2=Y2(K)+Y3(K)
      RU2=X2(K)-X3(K)
      IU2=Y2(K)-Y3(K)
      X0(K)=R0+RS1+RS2
      Y0(K)=I0+IS1+IS2
      RA1=R0+RS1*A1+RS2*A2
      IA1=I0+IS1*A1+IS2*A2
      RA2=R0+RS1*A2+RS2*A1
      IA2=I0+IS1*A2+IS2*A1
      RB1=RU1*B1+RU2*B2
      IB1=IU1*B1+IU2*B2
      RB2=RU1*B2-RU2*B1
      IB2=IU1*B2-IU2*B1
      IF (ZERO) GO TO 300
      R1=RA1+IB1
      I1=IA1-RB1
      R2=RA2+IB2
      I2=IA2-RB2
      R3=RA2-IB2
      I3=IA2+RB2
      R4=RA1-IB1
      I4=IA1+RB1
      X1(K)=R1*C1+I1*S1
      Y1(K)=I1*C1-R1*S1
      X2(K)=R2*C2+I2*S2
      Y2(K)=I2*C2-R2*S2
      X3(K)=R3*C3+I3*S3
      Y3(K)=I3*C3-R3*S3
      X4(K)=R4*C4+I4*S4
      Y4(K)=I4*C4-R4*S4
      GO TO 420
  300 X1(K)=RA1+IB1
      Y1(K)=IA1-RB1
      X2(K)=RA2+IB2
      Y2(K)=IA2-RB2
      X3(K)=RA2-IB2
      Y3(K)=IA2+RB2
      X4(K)=RA1-IB1
      Y4(K)=IA1+RB1
  420 CONTINUE
  440 CONTINUE
  500 CONTINUE
      IF (FOLD) GO TO 100
  600 CONTINUE
      RETURN
      END
C
      SUBROUTINE HERMFT (X, Y, NSIZE, N, DIM )
C===> HERMITIAN SYMMETRIC FOURIER TRANSFORM
      REAL X(NSIZE), Y(NSIZE)
      INTEGER DIM(5), D2, D3, D4, D5
      COMMON /BLK5/ NOJOIN, NOPROJ, TWOPI, FACTOR, RMAX, RMIN, DOUT,
     1 MH(3), GARB2, GARBAG, XLIM(3), NOSET, NPIC, NP(3), NEX, ANGMIN, ANGMAX,
```

```
     2 DMIN, DMAX, DMUT, DM, DFRG, ALN, NPATT, IVDU, NAT, NPC, SCALE, IPASS
       COMMON/SINCOS/SINE(450), NYSEC
       TWO N = FLOAT(2*N)
       RETWON=1.0/TWO N

       A2=-.80901699
       B2=0.58778525
       NT = DIM(1)
       SEP = DIM(2)
       L1 = DIM(3)
       SIZE = DIM(4) - 1
       K2 = DIM(5)
       NS = N*SEP
       M5=M*5
       FM5 = FLOAT(M5)
       RECFM5=1.0/FM5
       MM5 = SEP*M5
       M OVER 2=M/2+1
       FJM1 = -1.0
       DO 600 J=1, M OVER 2
       FOLD=J.GT.1 .AND. 2*J.LT.M+2
       K0 = (J-1)*SEP + 1
       FJM1 = FJM1 + 1.0
       ANGLE = FJM1*RECFM5*360.0
       ZERO=ANGLE.EQ.0.0
       IF (ZERO) GO TO 200
       LCS=INT(ANGLE)+1
C===>  USE A SINE/COSINE LOOK-UP TABLE
   710 IF(LCS.LE.360) GOTO 700
       LCS=LCS-360
       GOTO 710
   700 IF(LCS.GE.1) GOTO 720
       LCS=LCS+360
       GOTO 700
   720 S1=SINE(LCS)
       C1=SINE(LCS+90)
       C2=C1*C1-S1*S1
       S2=S1*C1+C1*S1
       C3=C2*C1-S2*S1
       S3=S2*C1+C2*S1
       C4=C2*C2-S2*S2
       S4=S2*C2+C2*S2
       GO TO 200
   100 FOLD=.FALSE.
       K0 = (M+1-J)*SEP + 1
       T=C1*A1+S1*B1
       S1=C1*B1-S1*A1
       C1=T
       T=C2*A2+S2*B2
       S2=C2*B2-S2*A2
       C2=T
       T=C3*A2-S3*B2
       S3=-C3*B2-S3*A2
       C3=T
       T=C4*A1-S4*B1
       S4=-C4*B1-S4*A1
       C4=T
   200 DO 500 KK = K0, NS, MM5
       DO 440 L = KK, NT, L1
       K1 = L + SIZE
       DO 420 K = L, K1, K2
       R0=X0(K)
       I0=Y0(K)
```

```
      RS1=X1(K)+X4(K)
      IS1=Y1(K)+Y4(K)
      RU1=X1(K)-X4(K)
      IU1=Y1(K)-Y4(K)
      RS2=X2(K)+X3(K)
      IS2=Y2(K)+Y3(K)
      RU2=X2(K)-X3(K)
      IU2=Y2(K)-Y3(K)
      X0(K)=R0+RS1+RS2
      Y0(K)=I0+IS1+IS2
      RA1=R0+RS1*A1+RS2*A2
      IA1=I0+IS1*A1+IS2*A2
      RA2=R0+RS1*A2+RS2*A1
      IA2=I0+IS1*A2+IS2*A1
      RB1=RU1*B1+RU2*B2
      IB1=IU1*B1+IU2*B2
      RB2=RU1*B2-RU2*B1
      IB2=IU1*B2-IU2*B1
      IF (ZERO) GO TO 300
      R1=RA1+IB1
      I1=IA1-RB1
      R2=RA2+IB2
      I2=IA2-RB2
      R3=RA2-IB2
      I3=IA2+RB2
      R4=RA1-IB1
      I4=IA1+RB1
      X1(K)=R1*C1+I1*S1
      Y1(K)=I1*C1-R1*S1
      X2(K)=R2*C2+I2*S2
      Y2(K)=I2*C2-R2*S2
      X3(K)=R3*C3+I3*S3
      Y3(K)=I3*C3-R3*S3
      X4(K)=R4*C4+I4*S4
      Y4(K)=I4*C4-R4*S4
      GO TO 420
  300 X1(K)=RA1+IB1
      Y1(K)=IA1-RB1
      X2(K)=RA2+IB2
      Y2(K)=IA2-RB2
      X3(K)=RA2-IB2
      Y3(K)=IA2+RB2
      X4(K)=RA1-IB1
      Y4(K)=IA1+RB1
  420 CONTINUE
  440 CONTINUE
  500 CONTINUE
      IF (FOLD) GO TO 100
  600 CONTINUE
      RETURN
      END
C
      SUBROUTINE HERMFT (X, Y, NSIZE, N, DIM )
C===> HERMITIAN SYMMETRIC FOURIER TRANSFORM
      REAL X(NSIZE), Y(NSIZE)
      INTEGER DIM(5), D2, D3, D4, D5
      COMMON /BLK5/ NOJOIN,NOPROJ,TWOPI,FACTOR,RMAX,RMIN,DOUT,
     1 MH(3),GARB2,GARBAG,XLIM(3),NOSET,NPIC,NP(3),NEX,ANGMIN,ANGMAX,
     2 DMIN,DMAX,DMUT,DM,DFRG,ALN,NPATT,IVDU,NAT,NPC,SCALE,IPASS
      COMMON/SINCOS/SINE(450),NYSEC
      TWO N = FLOAT(2*N)
      RETWON=1.0/TWO N
      NT = DIM(1)
```

```
      D2 = DIM(2)
      D3 = DIM(3)
      D4 = DIM(4) - 1
      D5 = DIM(5)
      DO 100 I0 = 1, NT, D3
      I1 = I0 + D4
      DO 100 I = I0, I1, D5
      A = X(I)
      B = Y(I)
      X(I) = A + B
      Y(I) = A - B
  100 CONTINUE
      N OVER 2 = N/2 + 1
      IF (N OVER 2 .LT. 2) RETURN
      DO 400 I0 = 2, N OVER 2
      ANGLE = FLOAT(I0-1)*RETWON*360.0
      LCS=INT(ANGLE)+1
C===> USE A SINE/COSINE LOOK-UP TABLE
  710 IF(LCS.LE.360) GOTO 700
      LCS=LCS-360
      GOTO 710
  700 IF(LCS.GE.1) GOTO 720
      LCS=LCS+360
      GOTO 700
  720 SI=SINE(LCS)
      CO=SINE(LCS+90)
      K = (N + 2 - 2*I0)*D2
      K1 = (I0 - 1)*D2 + 1
      DO 300 I1 = K1, NT, D3
      I2 = I1 + D4
      DO 200 I = I1, I2, D5
      J = I + K
      A = X(I) + X(J)
      B = X(I) - X(J)
      C = Y(I) + Y(J)
      D = Y(I) - Y(J)
      E = B*CO + C*SI
      F = B*SI - C*CO
      X(I) = A + F
      X(J) = A - F
      Y(I) = E + D
  200 Y(J) = E - D
  300 CONTINUE
  400 CONTINUE
      CALL CMPLFT (X, Y, NSIZE, N, DIM )
      RETURN
      END
C
      SUBROUTINE MENU10
      CALL CLSCN
      WRITE(*,9001)
 9001 FORMAT(' Now select commands fom the following menu:'//
     1'   GRID    < Resolution (A) , X Limit , Y Limit , Z Limit      >'/
     2'           < Defaults:   0.400 A / 1.0 / 1.0 / 1.0             >'/
     3'   PEAKS   < Maximum No. of peaks, Number of heavy atoms present,>'/
     4'           < Minimum No. of peaks for plotable fragment (4)    >'/
     5'   NOJOIN  < Do not carry out any connectivity calculations    >'/
     6'   PROJECT < No. of projections, max. no. of interpretations   >'/
     7'   DOUT    < The maximum bond length to output  (Default 2.4 ) >'/
     8'   DMIN    < The minimum acceptable bond length (Default 1.1 ) >'/
     9'   DMAX    < The maximum acceptable bond length (Default 1.95) >'/
     #'   AMIN    < The minimum acceptable bond angle  (Default 85  ) >'/
```

```
     #'   AMAX     < The maximum acceptable bond angle  (Default 145 ) >'/
     #'   MOLE     < Inputs a set of connectivities to define molecule >'/
     #'   CONT     < Investigate contoured sections of the Fourier map >'/
     #)
      WRITE(*,9002)
 9002 FORMAT(
     2 /'In addition the commands:  TITLE, END, MENU, LEVEL, NOPRINT, PRINT'/
     3 '                            DEFAULT, HARD, VERY HARD, MODEL, SHOW, X'/
     4'may be issued at any time.'/'Only the first 4 characters of a
     5ny command are required.'/'<CR> or a call to another module term
     6inates current input.'/)
      RETURN
      END
C
      SUBROUTINE GETMAP(LMAP,NX,NZ,NY)
$INT2
C===> READS THE MAP OFF A FILE AND PREPARES FOR CONTOURING
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),ICLCK,ICNT
      INTEGER*1 LMAP
      INTEGER*4 NX,NZ,NY,NAT,NSYM,NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,
     1 NTAPED,NTAPEE,NTAPEF,NTAPEG,NTAPEH,IH,ICH1,ICH2,ITLE,IR,ICALL,
     2 NRC,NCH,NREAD,IEND,NREF,MAXH,ISPFLG,LK,IDEF,ICHK,ISYMP,LEVEL,
     3 IFLOW,KUSE,INXT,IRDY,NULL,IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,
     4 ISPCH,ICLCK,ICNT,NXVAL,NYVAL,NZVAL
      DIMENSION LMAP(NZ,NX,NY),X(4,200),NXVAL(150),NYVAL(150),NZVAL(150)
      DO 300 K=1,NY
        DO 400 L=1,NZ
          READ(NTAPEB,9001)(LMAP(L,J,K),J=1,NX)
 9001     FORMAT(16I4)
  400   CONTINUE
  300 CONTINUE
      READ(NTAPEB,9002)RMIN,RMAX
 9002 FORMAT(2F9.2)
      READ(NTAPEB,9003)NAT,NSYM
 9003 FORMAT(2I4)
      DO 515 JJ=1,NAT
        READ(NTAPEB,9004)(X(L,JJ),L=1,4)
 9004   FORMAT(4F10.4)
  515 CONTINUE
C===> GET NXVAL,NYVAL,NZVAL THE SECTION NUMBERS
      DO 600 J=1,NX
        NXVAL(J)=FLOAT((J-1))*1000.0/FLOAT(NX)+0.5
  600 CONTINUE
      DO 601 J=1,NY
        NYVAL(J)=FLOAT((J-1))*1000.0/FLOAT(NY)+0.5
  601 CONTINUE
      DO 602 J=1,NZ
        NZVAL(J)=FLOAT((J-1))*1000.0/FLOAT(NZ)+0.5
  602 CONTINUE
C===> NOW READY TO CONTOUR THE MAP
      CALL CONTOR(LMAP,NXVAL,NYVAL,NZVAL,NX,NY,NZ,X,NAT,NSYM)
      RETURN
      END
C
      SUBROUTINE CONTOR(LMAP,NXVAL,NYVAL,NZVAL,NUMX,NUMY,NUMZ,X,NAT,NSYM)
$INT2
C===>
C===>      CCCC    000    N    N  TTTTT    000    U    U  RRRR
```

```
C===>     C       O   O  NN  N    T    O   O  U   U  R   R
C===>     C       O   O  N N N    T    O   O  U   U  RRRR
C===>     C       O   O  N  NN    T    O   O  U   U  R   R
C===>      CCCC    OOO   N   N    T     OOO    UUU   R   R
C===>
C===> SUBROUTINE TO DRAW CONTOURS OF FOURIER MAPS
C===> USES BOB DIAMOND'S CONTOURING ALGORITHM
C===>
C===> AUTHOR:  S. R. BROWN
C===> UNIVERSITY OF GLASGOW, 1986
C===>
      LOGICAL BOO
      CHARACTER*1 DIRECT, DIRS(3)
      INTEGER PEAKS
      INTEGER*1 LMAP
      INTEGER*4 NUMX, NUMY, NUMZ, NXVAL, NYVAL, NZVAL, NUM, NAT, NSYM, JK
      COMMON/KONTUR/ GRID(150,150), BOO(150,150), NUM, ICOLD, JCOLD,
     1 IXSBIG, JYSBIG, IXSMIN, JYSMIN, ISEP, IXPEN, IYPEN, ICOLOR
      COMMON/GRAPH/IX(2,200), LCON(200,4), NTYPE(200), NFRG, NUMPNT, KL(200,4)
     1 , NPAGE, NFRAG, SCALE, IIX(2,200), NAMPNT, XMIN, YMIN, JKIND, JPRINT
      COMMON/DRAW/NZOOM, INEXT, ILAST, RLOW, RMARG, NHIGH, NLONG, N1, N2, N3,
     1 N1VAL(150), N2VAL(150), N3VAL(150), MAPTYP, N1ORIG, N2ORIG
     2 , ISMALL, JSMALL, IBIG, JBIG, RINDEX(3), IGEN, ISUPER, ISTRT1, ISTRT2, IE1, IE2
      COMMON/PEAK/PEAKS(5,1000), NPEAKS
      DIMENSION LMAP(NUMZ, NUMX, NUMY), NXVAL(150), NYVAL(150), NZVAL(150),
     1 NCOLOR(6), X(4,200)
C===> NCOLOR CONTAINS THE COLOURS BLUE, GREEN, MAGENTA, CYAN, YELLOW & RED
      DATA NCOLOR/15,51,12,3,48,60/
      DATA DIRS, DIRECT/'X','Y','Z','Y'/
C===> BIGGEST MAP WILL BE 150 x 150 POINTS
      NPAGE=0
      JKIND=1
      JPRINT=1
      RMARG=25.0
      RLOW=25.0
      NZOOM=0
      ILAST=0
      IGEN=0
      INEXT=0
      ISUPER=0
      NUMBER=0
C===> PREPARE INTERPRETED PEAKS TO BE PLOTTED ON CONTOURED SECTIONS
      NPEAKS=(NSYM+1)*NAT
      IF(NPEAKS.LE.1000) THEN
        NSYMEQ=0
        JK=0
        CALL GENER(X, NSYMEQ, JK, LMAP, NUMX, NUMY, NUMZ, NAT, NUMBER)
C===> NOW THE SYMMETRY EQUIVALENTS
        NSYMEQ=1
        DO 2 JK=1, NSYM
          CALL GENER(X, NSYMEQ, JK, LMAP, NUMX, NUMY, NUMZ, NAT, NUMBER)
    2   CONTINUE
      ENDIF
C===> CLEAR SCREEN & WRITE TITLE ON CONSOLE
      CALL CLSCN
      WRITE(*,9001)
 9001 FORMAT('*************************************************************'/
     $       '*                                                           *'/
     1       '*          WELCOME TO THE CONTOURING PROGRAM                *'/
     $       '*         ---------------------------------                 *'/
     2       '*************************************************************'/
     3)
```

```
C===> NOW WRITE TITLE ON THE GRAPHICS SCREEN
      CALL INITIL
      CALL SETPAG(0)
      CALL WTEXT(75,430,8,8,0,1,3,'CONTOUR')
      CALL WTEXT(75,415,4,2,0,1,60,'==============')
      CALL WTEXT(30,280,3,3,0,1,27,'A ROUTINE TO INVESTIGATE')
      CALL WTEXT(30,250,3,3,0,1,27,'CONTOURED SECTIONS FROM')
      CALL WTEXT(30,220,3,3,0,1,27,'THE FOURIER MAP.')
      CALL WTEXT(22,325,3,2,0,1,32,'<<<<<<<<<<<<<>>>>>>>>>>>>>')
      CALL WTEXT(22,195,3,2,0,1,32,'<<<<<<<<<<<<<>>>>>>>>>>>>>')
      CALL WTEXT(70,80,2,2,0,1,3,'*')
      CALL WTEXT(90,80,2,2,0,1,3,'Use light-pen to start')
      CALL WTEXT(70,10,1,1,0,1,32,'( AUTHOR:  S.R.BROWN   UNIVERSITY OF GLAS
     1GOW, 1986 )')
C===> READ LIGHTPEN
    4 CALL RDPEN(LX,LY)
      IF(LX.LT.60.OR.LX.GT.80) GOTO 4
      IF(LY.LT.70.OR.LY.GT.90) GOTO 4
      CALL INITIL
      IF(NPEAKS.GT.1000)THEN
        WRITE(*,5)
    5   FORMAT(/10X,'**** Warning - too many peaks to be labelled ****')
      ENDIF
C===>
C===>    MAPS ARE VIEWED IN SECTIONS AS FOLLOWS:
C===>    MAPTYP=0:   X-SECTIONS  N1=NUMY  N2=NUMZ
C===>    MAPTYP=1:   Y-SECTIONS  N1=NUMX  N2=NUMZ
C===>    MAPTYP=0:   Z-SECTIONS  N1=NUMX  N2=NUMY
C===>
C===> INITIALLY LOOK DOWN SHORTEST AXIS
      CALL AXIS(DIRECT,DIRS,NUMX,NUMY,NUMZ)
      CALL TYPE(DIRECT,DIRS,NUMX,NUMY,NUMZ,NXVAL,NYVAL,NZVAL)
C===> FILL UP GRID OF CONTOUR POINTS ACCORDING TO THE TYPE OF SECTIONS
C===> (X, Y OR Z) TO BE VIEWED
    8 CALL FILLER(DIRECT,DIRS)
C===> LOOP THROUGH SECTIONS
C
      DO 1000 L=1,N3
C===> DRAW APPROPRIATE CONTOUR MAP
 500    CALL DRWCON(NUMX,NUMY,NUMZ,LMAP,L,NXVAL,NYVAL,NZVAL,NCOLOR,X,DIRECT,
     1  DIRS)
C===> FIND OUT WHETHER USER WISHES TO SEE MENU
        CALL WTEXT(250,5,2,2,0,1,3,'* MENU')
        IF(ISUPER.EQ.1) THEN
          CALL WTEXT(345,5,2,2,0,1,3,'* CONTINUE')
        ELSE
          CALL WTEXT(345,5,2,2,0,1,3,'* NEXT SECTION')
          IF(L.EQ.N3)CALL WTEXT(5,495,2,2,0,1,48,'LAST SECTION')
        ENDIF
  11    CALL RDPEN(KX,KY)
        IF(KY.LT.0.OR.KY.GT.15) GOTO 11
        IF(KX.LT.240.OR.KX.GT.360) GOTO 11
        IF(KX.GT.260.AND.KX.LT.340) GOTO 11
C===> GO ON AND DRAW NEXT MAP
        IF(KX.LE.360.AND.KX.GE.340)THEN
          CALL WTEXT(250,5,2,2,0,0,0,'* MENU')
C===> RESET VARIABLES TO ALLOW SUPERIMPOSING OF SECTIONS
          IF(ISUPER.EQ.1)THEN
            CALL WTEXT(345,5,2,2,0,0,0,'* CONTINUE')
            IGEN=0
            NZOOM=0
            ISUPER=0
```

```
          ELSE
            CALL WTEXT(345,5,2,2,0,0,0,'* NEXT SECTION')
C===> REMOVE LAST SECTION MESSAGE
            IF (L.EQ.N3)THEN
              L=0
              CALL WTEXT(5,495,2,2,0,0,0,'LAST SECTION')
            ENDIF
          ENDIF
          GOTO 1000
        ENDIF
C===> DISPLAY MENU
        IF(KX.LE.260.AND.KX.GE.240) THEN
  12      CALL HPAGE
          CALL CMENU(1)
          CALL PAGE
C===> INTERPRET MENU
   10     CALL RDPEN(MX,MY)
          IF(MX.GT.20.OR.MX.LT.0)GOTO 10
          IF(MY.GT.420.OR.MY.LT.0) GOTO 10
C===> OPTION #1: EXIT FROM CONTOURS
          IF(MY.LE.420.AND.MY.GE.400) THEN
            CALL CONT
            RETURN
C===> OPTION #2: SUPERIMPOSE PREVIOUS PLANE
          ELSE IF (MY.LE.385.AND.MY.GE.365)THEN
C===> FIRST CHECK THAT CURRENT DISPLAY IS NOT A GENERAL PLANE OR AN ENLARGED
C===> SECTION - SUPERIMPOSING IS NOT AVAILABLE IN THESE CASES
            IF(ISUPER.EQ.1) THEN
              WRITE(*,14)
   14         FORMAT(/'Superimpose is not available for the current display.'/
     1                       ' Please select another option.')
              GOTO 10
            ELSE
              ILAST=1
              IF(L.EQ.1)THEN
                L=N3
              ELSE
                L=L-1
              ENDIF
            ENDIF
C===> OPTION #3: SUPERIMPOSE NEXT PLANE
          ELSE IF (MY.LE.350.AND.MY.GE.330) THEN
            IF(ISUPER.EQ.1) THEN
              WRITE(*,15)
   15         FORMAT(/'Superimpose is not available for the current display.'/
     1                       ' Please select another option.')
              GOTO 10
            ELSE
              INEXT=1
              IF(L.EQ.N3)THEN
                L=1
              ELSE
                L=L+1
              ENDIF
            ENDIF
C===> OPTION #4: CHOOSE SECTION TO BE DRAWN
          ELSE IF (MY.LE.315.AND.MY.GE.295) THEN
C===> REMOVE MENU
            CALL CMENU(0)
            ISUPER=0
            IGEN=0
            IF(NZOOM.EQ.1)CALL RESTOR(DIRECT,DIRS)
```

```
              WRITE(*,24) N3, DIRECT
    24        FORMAT(/' There are',I3,' sections along the ',A1,' direction.'/
      1             ' Which one do you wish to look at ? '\)
    21        READ *, NEWSEC
              IF(NEWSEC.LT.1.OR.NEWSEC.GT.N3) THEN
                WRITE(*,23)
    23          FORMAT(/' Invalid section number. Try again : '\)
                GOTO 21
              ELSE
                L=NEWSEC
              ENDIF
C===> OPTION #5: ZOOM
            ELSE IF (MY.LE.280.AND.MY.GE.260) THEN
              CALL ZOOM
              IF(NZOOM.EQ.1) THEN
                ISUPER=1
                CALL DRWCON(NUMX,NUMY,NUMZ,LMAP,L,WXVAL,WYVAL,WZVAL,WCOLOR,X,
     1DIRECT,DIRS)
                CALL WTEXT(250,5,2,2,0,1,3,'* MENU')
                CALL WTEXT(345,5,2,2,0,1,3,'* CONTINUE')
C===> MAKE SELECTION
  107           CALL RDPEN(LX,LY)
                IF(LY.LT.0.OR.LY.GT.15)GOTO 107
                IF(LX.LT.240.OR.LX.GT.360)GOTO 107
                IF(LX.LT.340.AND.LX.GT.260)GOTO 107
                CALL WTEXT(250,5,2,2,0,0,0,'* MENU')
                CALL WTEXT(345,5,2,2,0,0,0,'* CONTINUE')
                IF(LX.GE.340.AND.LX.LE.360)THEN
C===> RESTORE VARIABLES
                  CALL RESTOR(DIRECT,DIRS)
                ELSE IF(LX.GE.240.AND.LX.LE.260)THEN
                  GOTO 12
                ENDIF
C===> UNABLE TO DO AN ENLARGED SECTION - GO ON AND DRAW NEXT SECTION
              ELSE IF (NZOOM.EQ.0) THEN
                GOTO 500
              ENDIF
C===> OPTION #6: CHANGE CONTOURING MARGIN
            ELSE IF(MY.LE.245.AND.MY.GE.225) THEN
              CALL CMENU(0)
              WRITE(*,25)RMARG
    25        FORMAT(/' Contouring interval is at present',F7.2,' Please enter
     1new value : '\)
    27        READ *,RMARG
              IF(RMARG.LT.1)THEN
                WRITE(*,26)
    26          FORMAT(/' Invalid entry. Please try again : '\)
                GOTO 27
              ENDIF
C===> OPTION #7: CHANGE LOWER LIMIT FOR CONTOURING
            ELSE IF (MY.LE.210.AND.MY.GE.190)THEN
              CALL CMENU(0)
              WRITE(*,30)RLOW
    30        FORMAT(/' Lower limit is at present',F6.2,' Please enter new value
     1 : '\)
    32        READ *,RLOW
C===> CHECK FOR "SENSIBLE" INPUT
              IF(RLOW.LT.1.0.OR.RLOW.GT.100.0) THEN
                WRITE(*,31)
    31          FORMAT(/' Invalid entry. Please try again : '\)
                GOTO 32
              ENDIF
C===> OPTION #8: SCREEN DUMP
```

```
          ELSE IF (MY.LE.175.AND.MY.GE.155) THEN
        CALL CMENU(0)
        CALL DUMP
        CALL FLICK
        CALL DMENU1(0)
        CALL CMENU(1)
C===> GET NEXT OPTION
            GOTO 10
C===> OPTION #9: DIRECTION TO BE VIEWED
          ELSE IF(MY.LE.140.AND.MY.GE.120)THEN
C===> REMOVE MENU
   34       CALL CMENU(0)
            IF(IGEN.EQ.1) THEN
              WRITE(*,33)DIRECT
   33         FORMAT(/'This plane was calculated by solving for ',A1,/
     1              'Type in new variable (X, Y or Z) : '\)
            ELSE
              IF(NZOOM.EQ.1)CALL RESTOR(DIRECT,DIRS)
              WRITE(*,35)DIRECT
   35         FORMAT(/'Sections are currently being viewed along the ',A1,'
     1 direction.'/'Type in new direction (X, Y or Z) : '\)
            ENDIF
            READ(*,36)DIRECT
   36       FORMAT(A)
            IF(DIRECT.NE.'X'.AND.DIRECT.NE.'x'.AND.DIRECT.NE.'Y'.AND.
     1 DIRECT.NE.'y'.AND.DIRECT.NE.'Z'.AND.DIRECT.NE.'z')THEN
              WRITE(*,37)
   37         FORMAT(/'Invalid choice of input. Please try again.')
              GOTO 34
            ENDIF
            CALL TYPE(DIRECT,DIRS,NUMX,NUMY,NUMZ,NXVAL,NYVAL,NZVAL)
C===> FILL UP GRID OF CONTOUR POINTS ACCORDINGLY
            GOTO 8
C===> OPTION #10: VIEW A GENERAL PLANE
          ELSE IF(MY.LE.105.AND.MY.GE.85)THEN
            IGEN=1
            ISUPER=1
C===> CLEAR MENU
            CALL CMENU(0)
   68       WRITE(*,70)
   70       FORMAT(/'Input the Miller Indices of the plane to be viewed : '\)
            READ *,RINDEX
C===> TRAP FOR RUBBISH INPUT
            NILS=0
            DO 72 J=1,3
              IF(RINDEX(J).EQ.0.0) NILS=NILS+1
              IF(RINDEX(J).GT.15.OR.RINDEX(J).LT.0.OR.NILS.EQ.3) THEN
                WRITE(*,73)
   73           FORMAT(/'Invalid choice of Indices. Please try again.')
                GOTO 68
              ENDIF
   72       CONTINUE
C===> CALCULATE PLANE IN THE WAY WHICH WILL GIVE MOST GRID POINTS
            IF(RINDEX(1).NE.0.0.AND.RINDEX(2).NE.0.0.AND.
     1                                      RINDEX(3).NE.0.0)THEN
              CALL AXIS(DIRECT,DIRS,NUMX,NUMY,NUMZ)
            ELSE
              CALL PICK(DIRECT,DIRS,1,2,3,NUMY,NUMZ)
              CALL PICK(DIRECT,DIRS,2,1,3,NUMX,NUMZ)
              CALL PICK(DIRECT,DIRS,3,1,2,NUMX,NUMY)
            ENDIF
            CALL TYPE(DIRECT,DIRS,NUMX,NUMY,NUMZ,NXVAL,NYVAL,NZVAL)
            CALL FILLER(DIRECT,DIRS)
```

```
C===> DRAW GENERAL PLANE
            CALL DRWCON(NUMX,NUMY,NUMZ,LMAP,L,NXVAL,NYVAL,NZVAL,NCOLOR,X,
     1DIRECT,DIRS)
            CALL WTEXT(250,5,2,2,0,1,3,'* MENU')
            CALL WTEXT(345,5,2,2,0,1,3,'* CONTINUE')
  108       CALL RDPEN(LX,LY)
            IF(LY.LT.0.OR.LY.GT.15)GOTO 108
            IF(LX.LT.240.OR.LX.GT.360)GOTO 108
            IF(LX.LT.340.AND.LX.GT.260)GOTO 108
            CALL WTEXT(250,5,2,2,0,0,0,'* MENU')
            CALL WTEXT(345,5,2,2,0,0,0,'* CONTINUE')
            IF(LX.GE.340.AND.LX.LE.360)THEN
              NZOOM=0
              ISUPER=0
              IGEN=0
            ELSE IF(LX.GE.240.AND.LX.LE.260)THEN
              GOTO 12
            ENDIF
C===> OPTION #11: CONTINUE
          ELSE IF(MY.LE.70.AND.MY.GE.50)THEN
            CALL CMENU(0)
            INEXT=0
            ILAST=0
            IF(NZOOM.EQ.1)CALL RESTOR(DIRECT,DIRS)
C===> OPTION #12: QUIT
          ELSE IF (MY.LE.35.AND.MY.GE.15)THEN
            CALL EXIT
C===> RE-READ MENU
          ELSE
            GOTO 10
          ENDIF
        ENDIF
C===> REDRAW MAP WITH ANY CHANGES ABOVE INSERTED
        GOTO 500
 1000 CONTINUE
      RETURN
      END
C
      SUBROUTINE FILLER(DIRECT,DIRS)
$INT2
C===> FILL UP GRID OF POINTS TO BE CONTOURED ACCORDING TO THE REQUIRED
C===> DIRECTION OF VIEW
      CHARACTER*1 DIRECT,DIRS(3)
      LOGICAL BOO
      INTEGER*4 NUM
      COMMON/KONTUR/ GRID(150,150),BOO(150,150),NUM,ICOLD,JCOLD,
     1 IXSBIG,JYSBIG,IXSMIN,JYSMIN,ISEP,IXPEN,IYPEN,ICOLOR
      COMMON/GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
     1 ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
      COMMON/DRAW/NZOOM,INEXT,ILAST,RLOW,RMARG,NHIGH,NLONG,N1,N2,N3,
     1 N1VAL(150),N2VAL(150),N3VAL(150),MAPTYP,N1ORIG,N2ORIG
     2 ,ISMALL,JSMALL,IBIG,JBIG,RINDEX(3),IGEN,ISUPER,ISTRT1,ISTRT2,IE1,IE2
C===> GET NO OF POINTS IN X AND Z DIRECTIONS
      IF(N2.GT.N1)THEN
        NUM=N2
```

```
      ELSE
        NUM=N1
      ENDIF
C===> MAKE UP SQUARE GRID BY FILLING 'MISSING POINTS' WITH A VALUE
C===> THAT WON'T SHOW UP ON THE CONTOUR DIAGRAM
      NN1=N1
      NN2=N2
C===> SPECIAL CASE: ZOOMED SECTION
      IF(NZOOM.EQ.1)THEN
        NN1=N1ORIG
        NN2=N2ORIG
      ENDIF
      IF(N2.LT.N1)THEN
        DO 7 J=N2+1,N1
          DO 8 I=1,N1
            GRID(I,J)=-100.0
  8       CONTINUE
  7     CONTINUE
      ELSE IF (N1.LT.N2)THEN
        DO 17 J=1,N2
          DO 18 I=N1+1,N2
            IF(NZOOM.EQ.1)THEN
              GRID(I,J)=-100.0
            ELSE
              GRID(J,I)=-100.0
            ENDIF
  18      CONTINUE
  17    CONTINUE
      ENDIF
C===> CALCULATE SEPARATION BETWEEN GRID POINTS
      ISEPX=490/N1
      ISEPZ=490/N2
      IF(ISEPX.LT.ISEPZ)THEN
        ISEP=ISEPX
      ELSE
        ISEP=ISEPZ
      ENDIF
C===> ORIGIN IS THE POINT (IXSMIN,JYSMIN)
      IF(NN1.GE.NN2)THEN
        NHIGH=ISEP*N2
        NLONG=ISEP*N1
      ELSE
        NHIGH=ISEP*N1
        NLONG=ISEP*N2
      ENDIF
      JYSMIN=(510-NHIGH)/2+12
      IXSMIN=(500-NLONG)/2+12
      RETURN
      END
C
      SUBROUTINE GENER(X,NSYMEQ,JK,LMAP,NUMX,NUMY,NUMZ,NAT,NUMBER)
$INT2
C===> PREPARE INTERPRETED PEAKS AND ALL THEIR SYMMETRY EQUIVALENTS TO
C===> BE PLOTTED ON CONTOURED SECTIONS
      INTEGER PEAKS
      INTEGER*1 LMAP
      INTEGER*4 NUMX,NUMY,NUMZ,NAT,IPRIN,JK
      DIMENSION XN(3),PK(3),NUMS(3),NUMBR(3),LMAP(NUMZ,NUMX,NUMY)
     1 ,X(4,200)
      COMMON/PEAK/PEAKS(5,1000),NPEAKS
      IPRIN=1
      NUMBR(1)=NUMX
      NUMBR(2)=NUMY
```

```
      NUMBR(3)=NUMZ
C===> LOOP THROUGH THE PEAKS
      DO 3 JJ=1,NAT
        NUMBER=NUMBER+1
C===> PUT THE COORDINATES OF THE PEAK ( OR SYMMETRY EQUIVALENT) IN XN
        IF(NSYMEQ.EQ.0)THEN
          DO 20 I=1,3
            XN(I)=X(I,JJ)
   20     CONTINUE
        ELSE
          CALL OPER(JK,XN,X(1,JJ),X(2,JJ),X(3,JJ),IPRIN)
        ENDIF
C===> LOOP THROUGH THE COORDINATES OF A PEAK
        DO 2 IJ=1,3
          PK(IJ)=XN(IJ)
          IF(PK(IJ).GT.1.0)THEN
            PK(IJ)=PK(IJ)-1.0
          ELSE IF(PK(IJ).LE.0.0)THEN
            PK(IJ)=PK(IJ)+1.0
          ENDIF
C===> CALCULATE GRID COORDINATES OF THE PEAK
          NUMS(IJ)=INT(PK(IJ)*FLOAT(NUMBR(IJ)))+1
    5     IF(NUMS(IJ).LE.0)NUMS(IJ)=NUMS(IJ)+NUMBR(IJ)
          IF(NUMS(IJ).GT.NUMBR(IJ))NUMS(IJ)=NUMS(IJ)-NUMBR(IJ)
          IF(NUMS(IJ).LE.0.OR.NUMS(IJ).GT.NUMBR(IJ))GOTO 5
   2    CONTINUE
C===> STORE THE GRID COORDINATES, PEAK NUMBER AND PEAK HEIGHT IN ARRAY PEAKS
        DO 40 KK=1,3
          PEAKS(KK,NUMBER)=NUMS(KK)
   40   CONTINUE
        PEAKS(5,NUMBER)=JJ
        PEAKS(4,NUMBER)=NSYMEQ
   3  CONTINUE
      RETURN
      END
C
      SUBROUTINE CMENU(I)
$INT2
C===> DISPLAY CONTOURING OPTIONS AVAILABLE
      DO 10 J=1,12
        NYVAL=445-(35*J)
        CALL WTEXT(10,NYVAL,2,2,0,I,27,'*')
  10  CONTINUE
      CALL WTEXT(10,480,2,2,0,I,56,'THE FOLLOWING OPTIONS ARE NOW AVAILABLE :'
     1)
      CALL WTEXT(35,410,2,2,0,I,35,'EXIT FROM CONTOURING PART OF PROGRAM')
      CALL WTEXT(35,375,2,2,0,I,35,'SUPER-IMPOSE PREVIOUS PLANE')
      CALL WTEXT(35,340,2,2,0,I,35,'SUPER-IMPOSE NEXT PLANE')
      CALL WTEXT(35,305,2,2,0,I,35,'CHOOSE SECTION TO BE DRAWN')
      CALL WTEXT(35,270,2,2,0,I,35,'ZOOM IN ON A SECTION USING THE MOUSE')
      CALL WTEXT(35,235,2,2,0,I,35,'CHANGE VALUE OF THE CONTOURS')
      CALL WTEXT(35,200,2,2,0,I,35,'CHANGE LOWER LIMIT FOR CONTOURING')
      CALL WTEXT(35,165,2,2,0,I,35,'DUMP CONTOUR MAP TO PRINTER')
      CALL WTEXT(35,130,2,2,0,I,35,'CHOOSE DIRECTION TO BE VIEWED')
      CALL WTEXT(35,95,2,2,0,I,35,'VIEW GENERAL PLANE')
      CALL WTEXT(35,60,2,2,0,I,35,'CONTINUE')
      CALL WTEXT(35,25,2,2,0,I,35,'QUIT - PROGRAM ABORT')
      RETURN
      END
C
      SUBROUTINE DRWCON(NUMX,NUMY,NUMZ,LMAP,L,NXVAL,NYVAL,NZVAL,NCOLOR,X
     1,DIRECT,DIRS)
$INT2
```

```
C===> DRAW CONTOURS
      INTEGER*1 LMAP
      INTEGER*4 NUMX,NUMY,NUMZ,IGRID,NUM,NXVAL,NYVAL,NZVAL
      INTEGER*2 TWO56,NTWO56,COLOUR
      CHARACTER*1 DIRECT,DIRS(3)
      LOGICAL BOO
      COMMON/KONTUR/GRID(150,150),BOO(150,150),NUM,ICOLD,JCOLD,
     1 IXSBIG,JYSBIG,IXSMIN,JYSMIN,ISEP,IXPEN,IYPEN,ICOLOR
      COMMON/GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
     1,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
      COMMON/DRAW/NZOOM,INEXT,ILAST,RLOW,RMARG,NHIGH,NLONG,N1,N2,N3,
     1 N1VAL(150),N2VAL(150),N3VAL(150),MAPTYP,N1ORIG,N2ORIG
     2 ,ISMALL,JSMALL,IBIG,JBIG,RINDEX(3),IGEN,ISUPER,ISTRT1,ISTRT2,IE1,IE2
      DIMENSION LMAP(NUMZ,NUMX,NUMY),NXVAL(150),NYVAL(150),NZVAL(150),
     1 NCOLOR(6),NUMCHR(4),NPOINT(2,50),NUMPK(50),NEQV(50),X(4,200)
      TWO56=256
      NTWO56=-256
C===> INITIALISE ARRAYS
      DO 5 J=1,50
        DO 6 I=1,2
          NPOINT(I,J)=0
    6   CONTINUE
        NEQV(J)=0
        NUMPK(J)=0
    5 CONTINUE
      JJ=0
C===> PREPARE FOR CALCULATING GENERAL SECTION
      IF(IGEN.EQ.1)THEN
        LORIG=L
C===> SOLVE THE EQUATION OF THE PLANE FOR X
        IF(MAPTYP.EQ.0)THEN
          RNUM1=REAL(NUMY)
          RNUM2=REAL(NUMZ)
          RNUM3=REAL(NUMX)
          R1=RINDEX(2)
          R2=RINDEX(3)
          R3=RINDEX(1)
C===> SOLVE THE EQUATION OF THE PLANE FOR Y
        ELSE IF(MAPTYP.EQ.1)THEN
          RNUM1=REAL(NUMX)
          RNUM2=REAL(NUMZ)
          RNUM3=REAL(NUMY)
          R1=RINDEX(1)
          R2=RINDEX(3)
          R3=RINDEX(2)
C===> SOLVE THE EQUATION OF THE PLANE FOR Z
        ELSE IF(MAPTYP.EQ.2)THEN
          RNUM1=REAL(NUMX)
          RNUM2=REAL(NUMY)
          RNUM3=REAL(NUMZ)
          R1=RINDEX(1)
          R2=RINDEX(2)
          R3=RINDEX(3)
        ENDIF
      ENDIF
      MM=N3+1
C===> CHECK FOR INTERPRETED PEAKS (INCLUDING SYMMETRY EQUIVALENTS) IN THIS
C===> SECTION
      IF(NZOOM.NE.1) THEN
        DO 20 I=1,N1
          IF(IGEN.EQ.1) THEN
            XX=REAL(I)/RNUM1
          ENDIF
```

```
          DO 10 K=1,N2
C===> POINT USED DEPENDS ON WHICH KIND OF SECTIONS (X,Y,Z OR GENERAL PLANE)
C===> ARE REQUIRED
C===> DEAL WITH GENERAL PLANE FIRST
              IF(IGEN.EQ.1) THEN
                YY=REAL(K)/RNUM2
                ZZ=RNUM3*((1.0-R1*XX-R2*YY)/R3)
                INZ=NINT(ZZ)+1
C  11             IF(INZ.LT.1)INZ=INZ+N3
C                 IF(INZ.GT.N3)INZ=INZ-N3
                IF(INZ.LE.1.OR.INZ.GT.MM) THEN
                  L=-1
C===> POINT FOR GENERAL PLANE
                ELSE
                  IF(INZ.EQ.MM)INZ=1
                  L=INZ
                ENDIF
              ENDIF
              IF(L.EQ.-1) THEN
                LP=-99
              ELSE
C===> X-SECTIONS
                IF(MAPTYP.EQ.0)THEN
                  LP=LMAP(K,L,I)
C===> Y-SECTIONS
                ELSE IF(MAPTYP.EQ.1) THEN
                  LP=LMAP(K,I,L)
C===> Z-SECTIONS
                ELSE IF(MAPTYP.EQ.2)THEN
                  LP=LMAP(L,I,K)
                ENDIF
              ENDIF
C===> GET GRID OF POINTS FOR CONTOURING
              IF (N1.GE.N2) THEN
                GRID(I,K)=FLOAT(LP)
              ELSE
                GRID(K,I)=FLOAT(LP)
              ENDIF
   10       CONTINUE
   20     CONTINUE
        ENDIF
        CALL POINT(IXSMIN,JYSMIN)
C===> SET THE INITIAL PEN POSITION AT THE GIVEN ORIGIN
        IYPEN=JYSMIN
        IXPEN=IXSMIN
C===> CALL CONTOURING ROUTINE
        IF(INEXT.EQ.0.AND.ILAST.EQ.0)THEN
          CALL HPAGE
        ELSE
C===> DRAW OVER EXISTING MAP FOR SUPERIMPOSING
          IF(NPAGE.EQ.0)NUM1=1
          IF(NPAGE.EQ.1)NUM1=0
          NPAGE=NUM1
          CALL SETPAG(NPAGE)
        ENDIF
C===> ORDINARY MAP - CONTOUR IN GREEN
        IF(INEXT.EQ.0.AND.ILAST.EQ.0) ICOLOR=NCOLOR(2)
C===> NEXT MAP - RED
        IF(INEXT.EQ.1) ICOLOR=NCOLOR(6)
C===> PREVIOUS MAP - BLUE
        IF(ILAST.EQ.1) ICOLOR=NCOLOR(1)
        UPPER=5.0*RMARG
        DO 30 CON=RLOW,UPPER,RMARG
```

```
          CALL CONTUR(CON)
    30 CONTINUE
C===> DRAW PEAKS ON CONTOURED SECTION
        IF(IGEN.EQ.0)CALL FINDPK(L)
        IF(ILAST.EQ.0.AND.INEXT.EQ.0) THEN
C===> DRAW WINDOW
          ICOLOR=56
          NN1=N1
          NN2=N2
          IF(NZOOM.EQ.1) THEN
            NN1=N1ORIG
            NN2=N2ORIG
          ENDIF
C===> GET NO. OF SEGMENTS
          IF(NN1.GE.NN2) THEN
            NBROAD=N1-1
            NUP=N2-1
          ELSE
            NBROAD=N2-1
            NUP=N1-1
          ENDIF
C===> CALCULATE BOUNDARY OF MAP
          IXSBIG=IXSMIN+(NBROAD*ISEP)
          JYSBIG=JYSMIN+(NUP*ISEP)
          CALL POINT(IXSMIN,JYSMIN)
          DO 41 J=1,NBROAD
            CALL RELVEC(ISEP,0,0,1,ICOLOR)
    41    CONTINUE
          DO 42 J=1,NUP
            CALL RELVEC(0,ISEP,0,1,ICOLOR)
    42    CONTINUE
          CALL RELVEC(0,2,0,1,ICOLOR)
          DO 43 J=1,NBROAD
            CALL RELVEC(-ISEP,0,0,1,ICOLOR)
    43    CONTINUE
          DO 44 J=1,NUP
            CALL RELVEC(0,-ISEP,0,1,ICOLOR)
    44    CONTINUE
          CALL RELVEC(0,-2,0,1,ICOLOR)
C===> LABEL HORIZONTAL AXIS
          IF((MAPTYP.EQ.0.AND.NN1.GE.NN2).OR.(MAPTYP.EQ.2.AND.NN1.LT.NN2))THEN
            CALL WTEXT(230,JYSMIN-20,1,1,0,1,0,'----> Y')
          ELSE IF((MAPTYP.EQ.1.AND.NN1.GE.NN2).OR.(MAPTYP.EQ.2.AND.NN1.GE.NN2))
     .          THEN
            CALL WTEXT(230,JYSMIN-20,1,1,0,1,0,'----> X')
          ELSE IF((MAPTYP.EQ.0.AND.NN1.LT.NN2).OR.(MAPTYP.EQ.1.AND.NN1.LT.NN2))
     .          THEN
            CALL WTEXT(230,JYSMIN-20,1,1,0,1,0,'----> Z')
          ENDIF
C===> LABEL FOR GENERAL PLANE
          IF(IGEN.EQ.1) THEN
            CALL WTEXT(2,495,2,2,0,1,3,'GENERAL PLANE')
C===> DISPLAY IT'S MILLER INDICES
            CALL LABEL
          ELSE
C===> DISPLAY WHICH SECTION
            CALL WTEXT(145,5,2,2,0,1,3,DIRECT)
            CALL WTEXT(20,5,2,2,0,1,3,'SECTION      =')
            NUMBER=N3VAL(L)
            CALL NUMB(NUMBER,NUMCHR)
            CALL WTEXT(180,5,2,2,0,1,3,NUMCHR)
C===> SECTION NUMBER
            CALL NUMB(L,NUMCHR)
```

```
          CALL NTEXT(110,5,2,2,0,1,3,NUMCHR)
        ENDIF
        IF(NZOOM.NE.1)THEN
C===>  PUT X & Z VALUES ON THE RESPECTIVE AXES
          CALL NTEXT(IXSMIN,(JYSMIN-12),1,1,0,1,0,'0')
          CALL NTEXT((IXSMIN-6),JYSMIN,1,1,2,1,0,'0')
C===>  FIRST THE VALUES ALONG THE X-AXIS
          DO 50 J=1,4
            NVAL=(N1*J)/4
            NC=N1VAL(NVAL)
            CALL NUMB(NC,NUMCHR)
            IF(N1.GE.N2)THEN
              IXVAL=IXSMIN-10+J*((NLONG-ISEP)/4)
              CALL NTEXT(IXVAL,JYSMIN-12,1,1,0,1,0,NUMCHR)
            ELSE
              IYVAL=JYSMIN-10+J*((NHIGH-ISEP)/4)
              CALL NTEXT(IXSMIN-6,IYVAL,1,1,2,1,0,NUMCHR)
            ENDIF
   50     CONTINUE
C===>  NOW THOSE ON THE Z-AXIS
          DO 55 K=1,4
            NVAL=(N2*K)/4
            NC=N2VAL(NVAL)
            CALL NUMB(NC,NUMCHR)
            IF(N1.GE.N2)THEN
              IYVAL=JYSMIN-10+K*((NHIGH-ISEP)/4)
              CALL NTEXT(IXSMIN-6,IYVAL,1,1,2,1,0,NUMCHR)
            ELSE
              IXVAL=IXSMIN-10+K*((NLONG-ISEP)/4)
              CALL NTEXT(IXVAL,JYSMIN-12,1,1,0,1,0,NUMCHR)
            ENDIF
   55     CONTINUE
        ENDIF
C===>  BRING HIDDEN PAGE INTO VIEW
        CALL PAGE
      ELSE IF (ILAST.EQ.1)  THEN
        ILAST=0
        IF(L.EQ.N3)  THEN
          L=1
        ELSE
          L=L+1
        ENDIF
      ELSE
        INEXT=0
        IF(L.EQ.1)THEN
          L=N3
        ELSE
          L=L-1
        ENDIF
      ENDIF
      IF(IGEN.EQ.1)THEN
        L=LORIG
      ENDIF
      RETURN
      END
C
      SUBROUTINE NUMB(NUMBER,NUMCHR)
$INT2
      DIMENSION NUMCHR(4)
      DATA NMARK/' '/
      IF(NUMBER.GE.100)THEN
        NUMCHR(1)=NUMBER/100
```

```
        NUMCHR(2)=(NUMBER-(NUMCHR(1)*100))/10
        NUMCHR(3)=NUMBER-(NUMCHR(1)*100+NUMCHR(2)*10)
        NUMCHR(4)=NMARK
      ELSE IF(NUMBER.GE.10) THEN
        NUMCHR(1)=NUMBER/10
        NUMCHR(2)=NUMBER-(NUMCHR(1)*10)
        NUMCHR(3)=NMARK
      ELSE
        NUMCHR(1)=NUMBER
        NUMCHR(2)=NMARK
      ENDIF
      RETURN
      END
C
      SUBROUTINE TYPE(DIRECT,DIRS,NUMX,NUMY,NUMZ,NXVAL,NYVAL,NZVAL)
$INT2
C===> PREPARE FOR DRAWING X-, Y- OR Z-SECTIONS
      LOGICAL BOO
      CHARACTER*1 DIRECT,DIRS(3)
      INTEGER*4 NUMX,NUMY,NUMZ,NXVAL,NYVAL,NZVAL
      COMMON/DRAW/NZOOM,INEXT,ILAST,RLOW,RMARG,NHIGH,NLONG,N1,N2,N3,
     1 N1VAL(150),N2VAL(150),N3VAL(150),MAPTYP,N1ORIG,N2ORIG
     2 ,ISMALL,JSMALL,IBIG,JBIG,RINDEX(3),IGEN,ISUPER,ISTRT1,ISTRT2,IE1,IE2
      DIMENSION NXVAL(150),NYVAL(150),NZVAL(150)
C===> X-SECTIONS
      IF(DIRECT.EQ.'X'.OR.DIRECT.EQ.'x')THEN
        DIRECT=DIRS(1)
        MAPTYP=0
        N1=NUMY
        N2=NUMZ
        N3=NUMX
        DO 40 J=1,150
          N1VAL(J)=NYVAL(J)
          N2VAL(J)=NZVAL(J)
          N3VAL(J)=NXVAL(J)
   40   CONTINUE
C===> Y-SECTIONS
      ELSE IF (DIRECT.EQ.'Y'.OR.DIRECT.EQ.'y') THEN
        DIRECT=DIRS(2)
        MAPTYP=1
        N1=NUMX
        N2=NUMZ
        N3=NUMY
        DO 50 J=1,150
          N1VAL(J)=NXVAL(J)
          N2VAL(J)=NZVAL(J)
          N3VAL(J)=NYVAL(J)
   50   CONTINUE
C===> Z-SECTIONS
      ELSE IF (DIRECT.EQ.'Z'.OR.DIRECT.EQ.'z') THEN
        DIRECT=DIRS(3)
        MAPTYP=2
        N1=NUMX
        N2=NUMY
        N3=NUMZ
        DO 60 J=1,150
          N1VAL(J)=NXVAL(J)
          N2VAL(J)=NYVAL(J)
          N3VAL(J)=NZVAL(J)
   60   CONTINUE
      ENDIF
      RETURN
      END
```

```
C
      SUBROUTINE AXIS(DIRECT,DIRS,NUMX,NUMY,NUMZ)
$INT2
C===> CALCULATES THE SHORTEST AXIS AND STORES IT IN DIRECT
      CHARACTER*1 DIRECT,DIRS(3)
      INTEGER*4 NUMX,NUMY,NUMZ
      IF(NUMZ.LE.NUMX.AND.NUMZ.LE.NUMY) THEN
C===> Z-SECTIONS
        DIRECT=DIRS(3)
      ELSE IF(NUMY.LE.NUMX.AND.NUMY.LT.NUMY) THEN
C===> Y-SECTIONS
        DIRECT=DIRS(2)
      ELSE IF(NUMX.LT.NUMZ.AND.NUMY.LT.NUMY) THEN
C===> X-SECTIONS
        DIRECT=DIRS(1)
      ENDIF
      RETURN
      END
C
      SUBROUTINE PICK(DIRECT,DIRS,I,J,K,NN2,NN3)
$INT2
C===> PICKS WHICH OF X,Y OR Z TO SOLVE FOR WHEN CALCULATING A GENERAL PLANE
      CHARACTER*1 DIRECT,DIRS(3)
      INTEGER*4 NN2,NN3
      COMMON/DRAW/NZOOM,INEXT,ILAST,RLOW,RMARG,WHIGH,WLONG,N1,N2,N3,
     1 N1VAL(150),N2VAL(150),N3VAL(150),MAPTYP,N1ORIG,N2ORIG
     2 ,ISMALL,JSMALL,IBIG,JBIG,RINDEX(3),IGEN,ISUPER,ISTRT1,ISTRT2,IE1,IE2
      IF(RINDEX(I).EQ.0.0) THEN
        IF(RINDEX(J).EQ.0.0.AND.RINDEX(K).NE.0.0) THEN
          DIRECT=DIRS(K)
        ELSE IF(RINDEX(K).EQ.0.0.AND.RINDEX(J).NE.0.0) THEN
          DIRECT=DIRS(J)
        ELSE
          IF(NN2.LE.NN3) THEN
            DIRECT=DIRS(J)
          ELSE
            DIRECT=DIRS(K)
          ENDIF
        ENDIF
      ENDIF
      RETURN
      END
C
      SUBROUTINE FINDPK(NUMSEC)
$INT2
C===> FINDS PEAKS AND WRITES THEM ONTO CONTOURED SECTIONS
      INTEGER PEAKS,COLOUR
      INTEGER*4 NUM
      LOGICAL BOO
      DIMENSION NUMCHR(4)
      COMMON/PEAK/PEAKS(5,1000),NPEAKS
      COMMON/DRAW/NZOOM,INEXT,ILAST,RLOW,RMARG,WHIGH,WLONG,N1,N2,N3,
     1 N1VAL(150),N2VAL(150),N3VAL(150),MAPTYP,N1ORIG,N2ORIG
     2 ,ISMALL,JSMALL,IBIG,JBIG,RINDEX(3),IGEN,ISUPER,ISTRT1,ISTRT2,IE1,IE2
      COMMON/KONTUR/ GRID(150,150),BOO(150,150),NUM,ICOLD,JCOLD,
     1 IXSBIG,JYSBIG,IXSMIN,JYSMIN,ISEP,IXPEN,IYPEN,ICOLOR
C===> X-SECTIONS
      IF(MAPTYP.EQ.0)THEN
        LL=1
        JJ=2
        KK=3
C===> Y-SECTIONS
```

```
      ELSE IF(MAPTYP.EQ.1)THEN
        LL=2
        JJ=1
        KK=3
C===> Z-SECTIONS
      ELSE IF(MAPTYP.EQ.2)THEN
        LL=3
        JJ=1
        KK=2
      ENDIF
C===> LOOP THROUGH PEAKS
      DO 100 J=1,NPEAKS
        IF(PEAKS(LL,J).EQ.NUMSEC)THEN
C===> ESTABLISH THE SCREEN COORDINATES OF THE PEAK
C===> FIRST DEAL WITH THE SPECIAL CASE OF A "ZOOMED" MAP
          IF(NZOOM.EQ.1) THEN
            IF(N1ORIG.GE.N2ORIG)THEN
              LX=PEAKS(JJ,J)
              LY=PEAKS(KK,J)
            ELSE
              LY=PEAKS(JJ,J)
              LX=PEAKS(KK,J)
            ENDIF
C===> CHECK THAT PEAK WAS IN THE ENLARGED REGION
            IF(LX.LT.ISTRT1.OR.LX.GT.IE1.OR.LY.GT.IE2.OR.LY.LT.ISTRT2)GOTO 100
            NXCORD=(LX-ISTRT1-1)*ISEP+IXSMIN
            NYCORD=(LY-ISTRT2-1)*ISEP+JYSMIN
          ELSE
C===> ORDINARY SECTION
            IF(N1.GE.N2)THEN
              NXCORD=(PEAKS(JJ,J)-1)*ISEP+IXSMIN
              NYCORD=(PEAKS(KK,J)-1)*ISEP+JYSMIN
            ELSE
              NXCORD=(PEAKS(KK,J)-1)*ISEP+IXSMIN
              NYCORD=(PEAKS(JJ,J)-1)*ISEP+JYSMIN
            ENDIF
          ENDIF
C===> INTERPRETED PEAK OR SYMMETRY EQUIVALENT ?
          COLOUR=48
          IF(PEAKS(4,J).EQ.1)COLOUR=12
C===> DRAW PEAK AND NUMBER IT
          CALL WTEXT(NXCORD-2,NYCORD-2,2,2,0,1,COLOUR,'+')
          CALL NUMB(PEAKS(5,J),NUMCHR)
          CALL WTEXT(NXCORD+5,NYCORD+5,1,1,0,1,3,NUMCHR)
        ENDIF
  100 CONTINUE
      RETURN
      END
C
      SUBROUTINE LABEL
$INT2
C===> WRITE MILLER INDICES ON TO A GENERAL PLANE
      DIMENSION NUMCHR(4),INDEX(3)
      COMMON/DRAW/NZOOM,INEXT,ILAST,RLOW,RMARG,WHIGH,NLONG,N1,N2,N3,
     1 N1VAL(150),N2VAL(150),N3VAL(150),MAPTYP,N1ORIG,N2ORIG
     2 ,ISMALL,JSMALL,IBIG,JBIG,RINDEX(3),IGEN,ISUPER,ISTRT1,ISTRT2,IE1,IE2
      CALL WTEXT(2,5,2,2,0,1,3,'INDICES:')
C===> CONVERT INDICES TO INTEGER FORM
      DO 10 J=1,3
        INDEX(J)=NINT(RINDEX(J))
   10 CONTINUE
C===> WRITE INDICES ON SCREEN
      DO 20 K=1,3
```

```
        CALL NUMB(INDEX(K),NUMCHR)
        IX=90+(K*20)
        CALL NTEXT(IX,5,2,2,0,1,3,NUMCHR)
   20 CONTINUE
      RETURN
      END
C
      SUBROUTINE RESTOR(DIRECT,DIRS)
$INT2
C===> RESTORES VARIABLES AFTER EXAMINING A ZOOMED SECTION
      CHARACTER*1 DIRECT,DIRS(3)
      COMMON/DRAW/NZOOM,INEXT,ILAST,RLOW,RMARG,NHIGH,NLONG,N1,N2,N3,
     1 N1VAL(150),N2VAL(150),N3VAL(150),MAPTYP,N1ORIG,N2ORIG
     2 ,ISMALL,JSMALL,IBIG,JBIG,RINDEX(3),IGEN,ISUPER,ISTRT1,ISTRT2,IE1,IE2
      IF(NZOOM.EQ.1)THEN
        NZOOM=0
        ISUPER=0
C===> RESET NO. OF POINTS IN EITHER DIRECTION AND FILL IN "BLANKS" TO MAKE UP
C===> A SQUARE GRID ACCORDINGLY
        N1=N1ORIG
        N2=N2ORIG
        CALL FILLER(DIRECT,DIRS)
      ENDIF
      RETURN
      END
C
      SUBROUTINE CONTUR(ISOVAL)
C
C   -- Bob Diamond's Contouring Algorithm --
C   -- Contouring Around NUM*NUM Grid Points --
C   -- Grid Point Values are in array GRID --
C   -- Contours represent value given as ISOVAL --
C   -- IXSMIN,JYSMIN is lower left point of grid (in screen units) --
C   -- ISEP = Separation between Grid Points (in screen units) --
C
      LOGICAL BOO
      REAL ISOVAL
      INTEGER*4 NUM
      COMMON/KONTUR/ GRID(150,150),BOO(150,150),NUM,ICOLD,JCOLD,
     1 IXSBIG,JYSBIG,IXSMIN,JYSMIN,ISEP,IXPEN,IYPEN,ICOLOR
      ICOLD=1
      JCOLD=1
      DO 5 I=1,NUM
      DO 5 J=1,NUM
        BOO(I,J)=.FALSE.
5     CONTINUE
C
C   -- Spiral in Towards the Middle --
C
      DO 10 I=1,NUM
10    CALL CHASE(I,1,1,ISOVAL)
      L=0
      DO 20 K=NUM,(NUM/2)+1,-1
        L=L+1
C
C   -- Up --
C
        DO 15 J=L,K
15      CALL CHASE(K,J,4,ISOVAL)
C
C   -- Left --
C
        DO 16 I=K,L,-1
```

```
16       CALL CHASE(I,K,3,ISOVAL)
C
C    -- Down --
C
         DO 17 J=K,L+1,-1
17       CALL CHASE(L,J,2,ISOVAL)
C
C    -- Right --
C
         DO 18 I=L,K-1
18       CALL CHASE(I,L+1,1,ISOVAL)
20     CONTINUE
       RETURN
       END
C
       SUBROUTINE CHASE(I,J,ID,ISOVAL)
       LOGICAL BOO,INCHAR,LRFLAG,OK,EDGTST
       REAL ISOVAL
       INTEGER*4 NUM
       COMMON/KONTUR/ GRID(150,150),BOO(150,150),NUM,ICOLD,JCOLD,
      1 IXSBIG,JYSBIG,IXSMIN,JYSMIN,ISEP,IXPEN,IYPEN,ICOLOR
       IC=I
       JC=J
       IF(BOO(IC,JC)) GO TO 20
       SIGN=(GRID(IC,JC)-ISOVAL)*(GRID(ICOLD,JCOLD)-ISOVAL)
       IF(SIGN.GE.0) GO TO 20
C
C    -- Boundary Crossed --
C
       IA=0
       IF(GRID(ICOLD,JCOLD)-ISOVAL.GT.0) IA=1
       ITEST=ID+IA+ICOLD+JCOLD
       IODD=ITEST-2*(ITEST/2)
       EDGTST=.FALSE.
       IF(IODD.EQ.0) THEN
         LRFLAG=.FALSE.
       ELSE
         LRFLAG=.TRUE.
       ENDIF
       ICSTO=IC
       JCSTO=JC
       ICSTLD=ICOLD
       JCSTLD=JCOLD
       IDELTA=((GRID(ICOLD,JCOLD)-ISOVAL)/
      1(GRID(ICOLD,JCOLD)-GRID(IC,JC)))*ISEP
       IF(IC.EQ.ICOLD) GO TO 2
       IY=(JC-1)*ISEP+JYSMIN
       IF(IC.GT.ICOLD) GO TO 1
       IX=(ICOLD-1)*ISEP+IXSMIN-IDELTA
       GO TO 3
1      IX=(ICOLD-1)*ISEP+IXSMIN+IDELTA
       GO TO 3
2      IX=(IC-1)*ISEP+IXSMIN
       IF(JC.GT.JCOLD) GO TO 5
       IY=(JCOLD-1)*ISEP+JYSMIN-IDELTA
       GO TO 3
5      IY=(JCOLD-1)*ISEP+JYSMIN+IDELTA
3      IXSTO=IX
       IYSTO=IY
C===>  MOVE PEN TO THE POINT (IX,IY) WITHOUT DRAWING
       CALL POINT(IX,IY)
       IXPEN=IX
       IYPEN=IY
```

```
      GO TO 13
4     EDGTST=.TRUE.
      SIGN=(GRID(IC,JC)-ISOVAL)*(GRID(ICOLD,JCOLD)-ISOVAL)
      IF(SIGN.GE.0) GO TO 10
      IF(IC.NE.ICSTO) GO TO 12
      IF(JC.NE.JCSTO) GO TO 12
      IF(ICOLD.NE.ICSTLD) GO TO 12
      IF(JCOLD.NE.JCSTLD) GO TO 12
      GO TO 25
12    BOO(ICOLD,JCOLD)=.TRUE.
      BOO(IC,JC)=.TRUE.
      IDELTA=((GRID(ICOLD,JCOLD)-ISOVAL)/
     1(GRID(ICOLD,JCOLD)-GRID(IC,JC)))*ISEP
      IF(IC.EQ.ICOLD) GO TO 7
      IY=(JC-1)*ISEP+JYSMIN
      IF(IC.GT.ICOLD) GO TO 6
      IX=(ICOLD-1)*ISEP+IXSMIN-IDELTA
      GO TO 9
6     IX=(ICOLD-1)*ISEP+IXSMIN+IDELTA
      GO TO 9
7     IX=(IC-1)*ISEP+IXSMIN
      IF(JC.GT.JCOLD) GO TO 8
      IY=(JCOLD-1)*ISEP+JYSMIN-IDELTA
      GO TO 9
8     IY=(JCOLD-1)*ISEP+JYSMIN+IDELTA
9     CONTINUE
13    CONTINUE
C===> DRAW TO THE POINT (IX,IY)
      IX1=IX-IXPEN
      IY1=IY-IYPEN
      CALL RELVEC(IX1,IY1,0,1,ICOLOR)
      IXPEN=IX
      IYPEN=IY
14    LRFLAG=.NOT.LRFLAG
10    OK=.TRUE.
      IF(LRFLAG) GO TO 15
      IC1=IC
      JC1=JC
      IC1OLD=ICOLD
      JC1OLD=JCOLD
      CALL LEFT(IC,JC,OK)
      IF(OK) GO TO 4
      IF(EDGTST) GO TO 24
      IC=ICSTO
      JC=JCSTO
      ICOLD=ICSTLD
      JCOLD=JCSTLD
      GO TO 14
15    IC1=IC
      JC1=JC
      IC1OLD=ICOLD
      JC1OLD=JCOLD
      CALL RIGHT(IC,JC,OK)
      IF(OK) GO TO 4
      IF(EDGTST) GO TO 24
      IC=ICSTO
      JC=JCSTO
      ICOLD=ICSTLD
      JCOLD=JCSTLD
      GO TO 14
20    ICOLD=IC
      JCOLD=JC
```

```
      GO TO 26
C===> DRAW TO THE POINT (IXSTO,IYSTO)
25    IX1=IXSTO-IXPEN
      IY1=IYSTO-IYPEN
      CALL RELVEC(IX1,IY1,0,1,ICOLOR)
      IXPEN=IXSTO
      IYPEN=IYSTO
      GO TO 27
 24      CONTINUE
27    ICOLD=ICSTO
      JCOLD=JCSTO
26    CONTINUE
      RETURN
      END
C
      SUBROUTINE LEFT(IC,JC,OK)
      LOGICAL BOO,OK
      REAL ISOVAL
      INTEGER*4 NUM
      COMMON/KONTUR/ GRID(150,150),BOO(150,150),NUM,ICOLD,JCOLD,
     1 IXSBIG,JYSBIG,IXSMIN,JYSMIN,ISEP,IXPEN,IYPEN,ICOLOR
      IF(JC.EQ.JCOLD) GO TO 10
C
C   -- From Down To Right --
C
      ICOLD=IC
      IF(JC.GT.JCOLD) GO TO 5
      JCOLD=JC
      IC=IC+1
      IF(IC.GT.NUM) GO TO 20
      RETURN
C
C   -- From Up To Left --
C
5     JCOLD=JC
      IC=IC-1
      IF(IC.LE.0) GO TO 20
      RETURN
C
C   -- From Left To Down --
C
10    JCOLD=JC
      IF(IC.GT.ICOLD) GO TO 15
      ICOLD=IC
      JC=JC-1
      IF(JC.LE.0) GO TO 20
      RETURN
C
C   -- From Right To Up --
C
15    ICOLD=IC
      JC=JC+1
      IF(JC.GT.NUM) GO TO 20
      RETURN
20    OK=.FALSE.
      RETURN
      END
C
      SUBROUTINE RIGHT(IC,JC,OK)
      LOGICAL BOO,OK
      REAL ISOVAL
      INTEGER*4 NUM
```

```
      COMMON/KONTUR/ GRID(150,150),BOO(150,150),NUM,ICOLD,JCOLD,
     1 IXSBIG,JYSBIG,IXSMIN,JYSMIN,ISEP,IXPEN,IYPEN,ICOLOR
      IF(JC.EQ.JCOLD) GO TO 10
C
C    -- From Down To Left --
C
      ICOLD=IC
      IF(JC.GT.JCOLD) GO TO 5
      JCOLD=JC
      IC=IC-1
      IF(IC.LE.0) GO TO 20
      RETURN
C
C    -- From Up To Right --
C
5     JCOLD=JC
      IC=IC+1
      IF(IC.GT.NUM) GO TO 20
      RETURN
C
C    -- From Left  To Up --
C
10    JCOLD=JC
      IF(IC.GT.ICOLD) GO TO 15
      ICOLD=IC
      JC=JC+1
      IF(JC.GT.NUM) GO TO 20
      RETURN
C
C    -- From Right To Down --
C
15    ICOLD=IC
      JC=JC-1
      IF(JC.LE.0) GO TO 20
      RETURN
20    OK=.FALSE.
      RETURN
      END
C
C
      SUBROUTINE DUMP
$INT2
C===> HARDCOPY TO LINE PRINTER
      CHARACTER*1 ANSWER
      CHARACTER*50 TEXT
      LOGICAL*2 SWL,SWM,SWR
      COMMON/GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
     1 ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
      IX=256
      IY=256
C===> DISPLAY MENU OPTIONS
      IF(JKIND.EQ.1)THEN
        CALL DMENU1(1)
      ELSE
        CALL DMENU2(1)
      ENDIF
   10 CALL RDPEN(MX,MY)
      IF(MX.GT.20.OR.MX.LT.0)GOTO 10
      IF(MY.GT.450.OR.MY.LT.310)GOTO 10
C===> OPTION#1 - SELECT PRINTER
      IF(MY.LE.450.AND.MY.GE.430)THEN
        IF(JKIND.EQ.1)THEN
```

```
          CALL DMENU1(0)
        ELSE
          CALL DMENU2(0)
        ENDIF
   65   IF(JPRINT.EQ.1)THEN
          WRITE(*,70)
   70     FORMAT(/' Printer is presently set to NEC. Do you wish to change to'
     .             ' Epson ? (y/n) : '\)
        ELSE
          WRITE(*,72)
   72     FORMAT(/' Printer is presently set to Epson. Do you wish to change '
     .             'to NEC ? (y/n) : '\)
        ENDIF
        READ(*,71)ANSWER
   71   FORMAT(A)
        IF(ANSWER.NE.'Y'.AND.ANSWER.NE.'y'.AND.ANSWER.NE.'N'.
     .                                          AND.ANSWER.NE.'n') GOTO 65
        IF(ANSWER.EQ.'Y'.OR.ANSWER.EQ.'y')THEN
          IF(JPRINT.EQ.1)NPRINT=2
          IF(JPRINT.EQ.2)NPRINT=1
          JPRINT=NPRINT
        ENDIF
        IF(JKIND.EQ.1)THEN
          CALL DMENU1(1)
        ELSE
          CALL DMENU2(1)
        ENDIF
C===> TITLE - USE MOUSE TO POSITION TEXT ON SCREEN
       ELSE IF(MY.LE.410.AND.MY.GE.390)THEN
          CALL CLSCN
          CALL CLEAR(0)
          WRITE(*,75)
   75     FORMAT(/' Use the cursor to locate the point where text will start.'/
     .             'Press the middle key of the mouse at that point.')
          CALL STARTM
  100     CALL RMNON
          CALL WTEXT(IX,IY,1,1,0,1,48,'+')
          CALL MOUSE(IDX,IDY,SWL,SWM,SWR,IER)
          IF(SWM)THEN
            WRITE(*,101)
  101       FORMAT(/'Enter title (maximum of 50 characters)')
            READ(*,102)TEXT
  102       FORMAT(A)
C===> ERASE CURSOR BEFORE WRITING TEXT ON DIAGRAM
            CALL WTEXT(IX,IY,1,1,0,1,48,'+')
            CALL WTEXT(IX,IY,1,1,0,1,0,TEXT)
            CALL PAUSE
            CALL RMNOFF
            CALL CLEAR(1)
          ELSE
            CALL WTEXT(IX,IY,1,1,0,1,48,'+')
            IX=IX+2*IDX
            IY=IY+2*IDY
C===> KEEP CURSOR ON SCREEN
            IF(IX.GT.510)IX=510
            IF(IX.LT.0)IX=0
            IF(IY.GT.510)IY=510
            IF(IY.LT.0)IY=0
            GOTO 100
          ENDIF
C===> PREPARE FOR SCREEN DUMP OR RETURN
       ELSE IF((MY.LE.370.AND.MY.GE.350).OR.(MY.LE.330.AND.MY.GE.310))THEN
          CALL CLEAR(0)
```

```
C===>  HARDCOPY
         IF(MY.LE.370.AND.MY.GE.350)THEN
           IF(JPRINT.EQ.1)THEN
             CALL SCDUMP
           ELSE
             CALL HARDCP
           ENDIF
         ENDIF
         RETURN
       ENDIF
       GOTO 10
       END
C
       SUBROUTINE DMENU1(I)
$INT2
C===>  HARDCOPY OPTIONS AVAILABLE IN CONTOUR
       DO 10 J=1,4
         NYVAL=480-(40*J)
         CALL WTEXT(10,NYVAL,2,2,0,I,27,'*')
   10  CONTINUE
       CALL WTEXT(100,490,2,2,0,I,60,'HARDCOPY OPTIONS:')
       CALL WTEXT(35,440,2,2,0,I,35,'SELECT PRINTER (NEC/EPSON)')
       CALL WTEXT(35,400,2,2,0,I,35,'ENTER TEXT ON DIAGRAM')
       CALL WTEXT(35,360,2,2,0,I,35,'EXECUTE HARDCOPY')
       CALL WTEXT(35,320,2,2,0,I,35,'RETURN TO MAIN MENU')
       RETURN
       END
C
       SUBROUTINE DMENU2(I)
$INT2
C===>  MENU OPTIONS AVAILABLE FOR HARDCOPY IN MITHGRAPH
       CALL WTEXT(420,490,1,1,0,I,0,'HARDCOPY')
       CALL WTEXT(10,440,1,1,0,I,0,'PRINTER')
       CALL WTEXT(10,400,1,1,0,I,0,'TITLE')
       CALL WTEXT(10,360,1,1,0,I,0,'EXECUTE')
       CALL WTEXT(10,320,1,1,0,I,0,'RETURN')
       RETURN
       END
C
       SUBROUTINE FLICK
$INT2
       COMMON/GRAPH/IX(2,200),LCON(200,4),WTYPE(200),NFRG,NUMPNT,KL(200,4)
      1  ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
C===>  RETURNS PREVIOUS SCREEN INTACT
       IF(NPAGE.EQ.0)NUM1=1
       IF(NPAGE.EQ.1)NUM1=0
       NPAGE=NUM1
       CALL SETPAG(NPAGE)
       RETURN
       END
C
       SUBROUTINE CLEAR(I)
$INT2
C===>  I=0:  CLEAR HARDCOPY MENU FROM SCREEN
C===>  I=1:  PUT HARDCOPY MENU BACK ON SCREEN
       COMMON/GRAPH/IX(2,200),LCON(200,4),WTYPE(200),NFRG,NUMPNT,KL(200,4)
      1  ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
       COMMON/DRAW/NZOOM,INEXT,ILAST,RLOW,RMARG,WHIGH,WLONG,W1,W2,W3,
      1  W1VAL(150),W2VAL(150),W3VAL(150),MAPTYP,W1ORIG,W2ORIG
      2  ,ISMALL,JSMALL,IBIG,JBIG,RINDEX(3),IGEN,ISUPER,ISTRT1,ISTRT2,IE1,IE2
C===>  CALLED FROM CONTOURING ROUTINE
       IF(JKIND.EQ.1)THEN
```

```
        CALL FLICK
C===> TIDY UP SCREEN
        IF(I.EQ.0.AND.ISUPER.EQ.0)THEN
          CALL WTEXT(250,5,2,2,0,0,0,'* MENU')
          CALL WTEXT(345,5,2,2,0,0,0,'* NEXT SECTION')
        ENDIF
C===> CALLED FROM MITHGRAPH
      ELSE IF(JKIND.EQ.2) THEN
        CALL DMENU2(I)
      ENDIF
      RETURN
      END
C
      SUBROUTINE STARTM
$INT2
C===> START MOUSE
      LOGICAL*2 SWL,SWM,SWR
      WRITE(*,5)
    5 FORMAT(/'Place the mouse in the centre of the pad and press any of its
     . keys.')
      CALL IMC
   10 CALL MOUSE(IDX,IDY,SWL,SWM,SWR,IER)
C===> INTERROGATE SWITCHES
      IF(SWL)RETURN
      IF(SWM)RETURN
      IF(SWR)RETURN
      GOTO 10
      END
C
      SUBROUTINE ZOOM
$INT2
C===> GET AREA FOR ZOOMING
      CHARACTER*1 DIRECT,DIRS(3)
      LOGICAL BOO
      INTEGER*4 NUM
      DIMENSION RGRID(150,150)
      COMMON/KONTUR/ GRID(150,150),BOO(150,150),NUM,ICOLD,JCOLD,
     1 IXSBIG,JYSBIG,IXSMIN,JYSMIN,ISEP,IXPEN,IYPEN,ICOLOR
      COMMON/GRAPH/IXX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
     1,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
      COMMON/DRAW/NZOOM,INEXT,ILAST,RLOW,RMARG,NHIGH,NLONG,N1,N2,N3,
     1 N1VAL(150),N2VAL(150),N3VAL(150),MAPTYP,N1ORIG,N2ORIG
     2 ,ISMALL,JSMALL,IBIG,JBIG,RINDEX(3),IGEN,ISUPER,ISTRT1,ISTRT2,IE1,IE2
      NZOOM=1
      N1ORIG=N1
      N2ORIG=N2
C===> FLICK MAP BACK INTO VIEW
      IF(NPAGE.EQ.0)NUM1=1
      IF(NPAGE.EQ.1)NUM1=0
      NPAGE=NUM1
      CALL SETPAG(NPAGE)
C===> REMOVE WRITING
      CALL WTEXT(250,5,2,2,0,0,0,'* MENU')
      CALL WTEXT(345,5,2,2,0,0,0,'* NEXT SECTION')
C===> DISPLAY INSTRUCTIONS FOR USING THE MOUSE
      CALL CLSCN
      WRITE(*,20)
   20 FORMAT(/'The switches on the mouse operate as follows:'///'Left   -
     1 keep this button depressed to outline the area to be enlarged'//'
     2Middle - press this button to change cursor colour (yellow/cyan)'//'
     3Right  - return to main program to see contouring of enlarged section'
     4///)
```

```
      CALL MOUSEP
C===> IS THE AREA DEFINED BY THE USER "SENSIBLE" ?
      IF(IBIG.LE.ISMALL.OR.JBIG.LE.JSMALL)THEN
        WRITE(*,25)
   25   FORMAT(/'Improperly defined area - unable to contour.')
        NZOOM=0
        CALL CLSCN
        RETURN
      ENDIF
C===> GET NO OF POINTS IN EACH DIRECTION
      NO1=(IBIG-ISMALL)/ISEP
      NO2=(JBIG-JSMALL)/ISEP
      NPTS=NO1*NO2
C===> CHECK THERE ARE ENOUGH POINTS TO CONTOUR
      IF(NPTS.LT.50)THEN
        WRITE(*,45) NPTS
   45   FORMAT(/'Unable to enlarge this section as there are only',I4,' grid
     1 points for contouring.')
C===> RETURN TO CONTOURING MENU
        NZOOM=0
        CALL CLSCN
        RETURN
      ENDIF
C===> GET GRID OF POINTS FOR CONTOURING
      IF(N1ORIG.GE.N2ORIG)THEN
        N1=NO1
        N2=NO2
      ELSE
        N1=NO2
        N2=NO1
      ENDIF
      ISTRT1=(ISMALL-IXSMIN)/ISEP
      ISTRT2=(JSMALL-JYSMIN)/ISEP
      IE1=ISTRT1+NO1
      IE2=ISTRT2+NO2
      DO 30 I=1,N1
        DO 40 J=1,N2
          NX=ISTRT1+I
          NY=ISTRT2+J
          IF(N1ORIG.GE.N2ORIG)THEN
            GRID(I,J)=GRID(NX,NY)
          ELSE
            GRID(I,J)=GRID(NY,NX)
          ENDIF
   40   CONTINUE
   30 CONTINUE
C===> MAKE UP TO SQUARE GRID WITH "BLANK" POINTS, CALCULATE NEW ORIGIN AND
C===> GRID SEPARATION
      CALL FILLER(DIRECT,DIRS)
      CALL CLSCN
      RETURN
      END
C
      SUBROUTINE MOUSEP
$INT2
C===> RETURNS THE AREA TO BE ENLARGED IN THE CONTOURING ROUTINE. THE AREA IS
C===> THE RECTANGLE DEFINED BY THE POINTS (ISMALL,JSMALL), (IBIG,JSMALL),
C===> (IBIG,JBIG) AND (ISMALL,JBIG)
      LOGICAL BOO
      LOGICAL*2 SWL,SWM,SWR
      INTEGER COLOR1,COLOR2
      INTEGER*4 NUM
      COMMON/KONTUR/ GRID(150,150),BOO(150,150),NUM,ICOLD,JCOLD,
```

```
     1 IXSBIG,JYSBIG,IXSMIN,JYSMIN,ISEP,IXPEN,IYPEN,ICOLOR
       COMMON/DRAW/NZOOM,INEXT,ILAST,RLOW,RMARG,NHIGH,NLONG,N1,N2,N3,
     1 N1VAL(150),N2VAL(150),N3VAL(150),MAPTYP,N1ORIG,N2ORIG
     2 ,ISMALL,JSMALL,IBIG,JBIG,RINDEX(3),IGEN,ISUPER,ISTRT1,ISTRT2,IE1,IE2
       ILS=0
       IBIG=0
       JBIG=0
       ISMALL=512
       JSMALL=512
C===> INITIAL CURSOR POSITION IS IN THE MIDDLE OF THE SCREEN.
       IX=256
       IY=256
       NPOINT=0
C===> Yellow
       COLOR1=48
C===> INITIALISE THE MOUSE.
       WRITE(*,9001)
 9001 FORMAT(/'Place the mouse on the centre of the pad and press any of
     1its keys'/)
       CALL IMC
  100 CALL MOUSE(IDX,IDY,SWL,SWM,SWR,IER)
       IF(IER.EQ.1) CALL EXIT
C===> WAIT FOR A SWITCH TO BE PRESSED
       IF(SWL) GOTO 10
       IF(SWM) GOTO 10
       IF(SWR) GOTO 10
       GOTO 100
C===> DRAW THE CURSOR
   10 CALL RMWON
       CALL WTEXT(IX,IY,1,1,0,1,COLOR1,'+')
C===> READ THE MOUSE
       CALL MOUSE(IDX,IDY,SWL,SWM,SWR,IER)
       IF(IER.EQ.1) CALL EXIT
C===> INTERROGATE THE SWITCHES
C===> RIGHT HAND SWITCH - RETURN TO THE CONTOURING ROUTINE
       IF(SWR)THEN
         CALL RMWOFF
         RETURN
       ENDIF
C===> MIDDLE SWITCH - CHANGE CURSOR COLOUR
       IF(SWM) THEN
         IF(COLOR1.EQ.48)THEN
           COLOR2=3
         ELSE
           COLOR2=48
         ENDIF
         COLOR1=COLOR2
       ENDIF
C===> NOW THE LH SWITCH - USED TO DEFINE THE AREA TO BE ENLARGED
  15   IF(SWL) THEN
         ILS=1
       ELSE
C===> DELETE OLD CURSOR POSITION FROM THE SCREEN.
         CALL WTEXT(IX,IY,1,1,0,1,COLOR1,'+')
       ENDIF
C===> NOW SORT OUT THE CURSOR SHIFTS.
   20 IX=IX+2*IDX
       IY=IY+2*IDY
C===> KEEP CURSOR WITHIN THE CONTOURED SECTION
       IF(IX.GT.IXSBIG) IX=IXSBIG
       IF(IY.GT.JYSBIG) IY=JYSBIG
       IF(IX.LT.IXSMIN) IX=IXSMIN
       IF(IY.LT.JYSMIN) IY=JYSMIN
```

```
C===>  GET THE COORDINATES OF THE AREA TO BE ENLARGED
       IF(ILS.EQ.1)THEN
         IF(ISMALL.GT.IX)ISMALL=IX
         IF(IBIG.LT.IX)IBIG=IX
         IF(JSMALL.GT.IY)JSMALL=IY
         IF(JBIG.LT.IY)JBIG=IY
       ENDIF
       ILS=0
       GOTO 10
       END
C
       SUBROUTINE MOUSE (IDX,IDY,SL,SM,SR,IER)
$INT2
C===>
C===>  THIS IS THE MOUSE READING ROUTINE.  THE PARAMETERS ARE AS FOLLOWS:
C===>
C===>  IDX    THE INCREMENT ON THE X-TRANSLATION.
C===>  IDY    THE INCREMENT ON THE Y-TRANSLATION
C===>  SL     TRUE IF THE LEFT-HAND SWITCH IS CLOSED OTHERWISE FALSE.
C===>  SM     TRUE IF THE CENTRE SWITCH IS CLOSED OTHERWISE FALSE.
C===>  SR     TRUE IF THE RIGHT-HAND SWITCH IS CLOSED OTHERWISE FALSE.
C===>  IER    = 0 / 1 FOR NO ERROR / ERROR ON READING THE MOUSE.
C===>
C===>  CALLS THE ASSEMBLER ROUTINE IGETBY
C===>
       INTEGER*1 I,IMIN,IX,IY
       INTEGER*4 J
       LOGICAL*2 SL,SM,SR
       CHARACTER*1 YN
       IER=0
       IMIN=-121
C===>  MAXIMUM NUMBER OF TRIES TO READ THE MOUSE IS 50.
       MAXTRY=50
  10   NTRIES=0
C===>  READ CONTROL WORD FROM THE MOUSE.
  20   I=IGETBY()
       IF(I.GT.IMIN) THEN
C===>  BAD CONTROL BYTE.
         NTRIES=NTRIES+1
         IF(NTRIES.GE.MAXTRY) THEN
           WRITE(*,9001)
 9001      FORMAT(/' **** Problem in reading the mouse.  Do you want to try
     1 again ? - '\)
           READ(*,9002) YN
 9002      FORMAT(A)
           IF(YN.EQ.'Y'.OR. YN.EQ.'y') THEN
             GOTO 10
           ELSE
             WRITE(*,9003)
 9003        FORMAT(/' ****  Stopping because of mouse reading problems  ****'/)
             IER=1
             RETURN
           ENDIF
C===>  CARRY ON TRYING.
         ELSE
           GOTO 20
         ENDIF
C===>  CHECK BITS ON FIRST BYTE ARE O.K.
       ELSE
C===>  INCREMENT X AND Y SHIFTS.
         IX=IGETBY()
         IY=IGETBY()
         IX=IX+IGETBY()
```

```
        IY=IY+IGETBY()
C===>  GET SHIFTS IN TWO BYTE FORM
        IDX=IX
        IDY=IY
C===>  SORT OUT THE SWITCHES.
        SL=.FALSE.
        SM=.FALSE.
        SR=.FALSE.
        J=I
        J=IABS(J)-121
        IF(J.EQ.0) RETURN
        GOTO (101,102,103,104,105,106,107),J
  101   SR=.TRUE.
        RETURN
  102   SM=.TRUE.
        RETURN
  103   SR=.TRUE.
        SM=.TRUE.
        RETURN
  104   SL=.TRUE.
        RETURN
  105   SL=.TRUE.
        SR=.TRUE.
        RETURN
  106   SL=.TRUE.
        SM=.TRUE.
        RETURN
  107   SL=.TRUE.
        SM=.TRUE.
        SR=.TRUE.
        RETURN
C===>  ALL SWITCHES AND INCREMENTS READ SUCESSFULLY.
      ENDIF
      END
```

```
      SUBROUTINE SEARCH( NSYMOP, LMAP, NXVAL, NYVAL, NZVAL, NX, NYT, NZ, ICONT, NELSE)
C===>
C===>       SSSS   EEEEE    AAA    RRRR     CCCC   H    H
C===>      S       E       A   A   R   R   C       H    H
C===>       SSS    EEEEE   AAAAA   RRRR    C       HHHHH
C===>          S   E       A   A   R  R    C       H    H
C===>      SSSS    EEEEE   A   A   R   R    CCCC   H    H
C===>
C===>  SUBROUTINE FOR THE AUTOMATIC SEARCH AND INTERPRETATION OF MAPS
C===>  STRIDE 460 VERSION
       INTEGER*1 LMAP
       COMMON/IO/NIN, NOUT, NTAPEA, NTAPEB, NTAPEC, NTAPED, NTAPEE,
      1  NTAPEF, NTAPEG, IH( 155), A( 200), ICH1( 10), ICH2( 10), ITLE( 68), IR( 68),
      2  ICALL, NRC, NCH, NREAD, IEND, NREF, PI, DTOR, MAXH, ISPFLG, ZERO, LK( 32),
      3  IDEF, ICHK, ISYMP, LEVEL, IFLOW( 25), KUSE( 40), INXT, IRDY, RTOD, NULL,
      4  IDIF, IVDIF, KARLE, IHVY, IGPFL, NDIFF, ISPCH( 30), ICLCK, ICNT
       COMMON /BLK1/ NSTORE( 25100), XA( 3, 200), XB( 4, 200), ISYM( 100),
      1  IFRAG( 200), ICON( 200), IBOND( 2000), JFRAG( 200),
      2  JCON( 200), JBOND( 2000), IUSE( 200), IONA( 6, 200), MO( 200),
      3  MU( 200), NO( 200), NU( 200), KEYST( 200), LH( 200), LR( 200), LX( 200),
      4  JK( 200), IK( 200), KEY( 200), IPL( 200), CAN( 200), COM( 200), LL( 200),
      5  IA( 6, 200), LS( 200), X( 4, 200)
       COMMON /BLK2/ IS( 2, 3, 24), TS( 3, 24), NSYM, ICENT, LAT, IAPX,
      1  CELL( 6), T( 3, 3), DXMAX( 3), TL( 3, 4), FS( 3, 3, 24), MLAT, NEQV, MCON, INT,
      2  FOM, KFRAG( 20), NFRAG, LFRAG( 20), MFRAG, NOFRAG, NN, MFRN, IFMT( 20),
      3  KMATCH( 20), ITYPE, NOINT, MNPPF
       COMMON /BLK4/ IONP( 6, 200), NAFRG( 5), ICH( 12), IB( 6), LINX( 60), NSEQ
       COMMON /BLK5/ NOJOIN, NOPROJ, TWOPI, FACTOR, RMAX, RMIN, DOUT,
      1  MH( 3), ILEVEL, IPNT, XLIM( 3), NOSET, NPIC, MP( 3), NEX, ANGMIN, ANGMAX,
      2  DMIN, DMAX, DMUT, DM, DFRG, ALN, NPATT, IVDU, NAT, NPC, SCALE, IPASS
       COMMON/BLK6/ INBOND( 1000), RUBISH( 150)
       DIMENSION NXVAL( 100), NYVAL( 100), NZVAL( 100), LMAP( NX, NYT, NZ)
       EQUIVALENCE ( KUSE31, KUSE( 31)), ( KUSE32, KUSE( 32))
       NTAPEI=17
       NADR=0
       ITIMES=0
C===>  INPUT DATA FROM FFT FILE
       OPEN( FILE=' SRCH. FIL' , UNIT=NTAPEE, FORM=' UNFORMATTED' , STATUS=' OLD' )
       REWIND NTAPEE
       READ ( NTAPEE) CELL, NEQV, ICENT, LAT, TS, IS, IDIF, IVDIF
       IF ( NPC . LE. 0) GO TO 860
       NAT = NPC
       NPIC = NPC
       GO TO 800
   860 NAT = MIN0(( 11*NAT +13)/9+10*IDIF+10*IVDIF+10*IAPX, KUSE31-20)
       NPIC = MIN0(( 3*NAT +1)/2, KUSE31-20)
   800 CONTINUE
       WRITE( NOUT, 9051)
  9051 FORMAT( /////' Peak search and interpretation' /1H , 30( 1H-))
       WRITE( *, 9001)
  9001 FORMAT( /' Peak search now begins' )
       WRITE ( NOUT, 883)
   883 FORMAT ( /)
C===>  HALVE Y-AXIS BY C. OF. S. OR LATTICE TYPE IF POSSIBLE
   801 IHALF=0
       IF ( ICENT. EQ. 1) IHALF=-1
       IF ( IHALF. NE. 0) GOTO 888
       GOTO ( 888, 886, 888, 886, 886, 886, 888), LAT
   886 IHALF = 1
   888 CALL CENTRE ( 1, 1)
```

```
C===> EXPAND TO FULL SPACE GROUP SYMMETRY
C===> PUT CENTRING TRANSLATIONS IN TL(I,J)
      GO TO (940,890,890,890,890,910,920),LAT
  890 CALL CENTRE(2,LAT)
      GO TO 940
  910 CALL CENTRE(2,2)
      CALL CENTRE(3,3)
      CALL CENTRE(4,4)
      GO TO 940
  920 CALL CENTRE(2,6)
      CALL CENTRE(3,7)
  940 DO 1000 I=1,NEQV
      DO 970 J=1,3
      DO 950 K=1,3
  950 FS(K,J,I)=0.0
      DO 960 K=1,2
      M=IABS(IS(K,J,I))
      IF(M.GT.0) FS(M,J,I)=FLOAT(ISIGN(1,IS(K,J,I)))
  960 CONTINUE
  970 CONTINUE
      IF (I.EQ.1.OR.IHALF.NE.0) GOTO 1000
C===> HALVE Y-AXIS BY SYMMETRY OP. IF POSSIBLE
      DO 980 J=1,3
      IF (IABS(IS(1,J,I)).NE.J.OR.IS(2,J,I).NE.0) GOTO 1000
  980 CONTINUE
      IF(IS(1,2,I).EQ.(-2).AND.ABS(TS(2,I)).LT.1E-6) IHALF= -1
      IF(IS(1,2,I).EQ.2.AND.ABS(TS(2,I)-0.5).LT.1E-6) IHALF=1
 1000 CONTINUE
      NSYM=NEQV*(ICENT+1)*MLAT
      IF(IPASS.GT.1) GOTO 1061
      IF(NPATT.EQ.1) GOTO 410
C===> OUTPUT STEREOCHEMICAL CRITERIA
      WRITE(NOUT,9075)       NAT,IHVY
 9075 FORMAT(31X,'Number of atoms to be found is',I4//23X,
     1 'Number of heavy atoms in the asymmetric unit IS ',I2)
      WRITE(NOUT,9076) DMIN,DMAX,ANGMIN,ANGMAX,DMUT
 9076 FORMAT(/16X,'Stereochemical criteria:',
     1'Minimum bonding distance is',F6.2/40X,'Maximum bonding distance i
     2s'   ,F6.2/40X,'Mimimum bond angle is',F6.1/40X,'Maximum bond angle
     3 is',F6.1//11X,'Interatomic distances are output when less than',
     4 F6.2,' angstroms')
      IF(NOPROJ.LE.0) WRITE(NOUT,9077)
 9077 FORMAT(/11X,'Number of projections to be output is decided by the
     1 programme')
      IF(NOPROJ.GT.0) WRITE(NOUT,9078) NOPROJ
 9078 FORMAT(/11X,'Number of projections to be output for each cluster i
     1s',I3)
C===> SET UP CONSTANTS
  410 ANGMIN=COS(DTOR*ANGMIN)
      ANGMAX=COS(DTOR*ANGMAX)
      DMIN=DMIN*DMIN
      DMAX=DMAX*DMAX
      DMUT = DMUT * DMUT
      CA=COS(DTOR*CELL(4))
      CB=COS(DTOR*CELL(5))
      CC=COS(DTOR*CELL(6))
      V=CELL(1)*CELL(2)*CELL(3)*SQRT(1.0-CA*CA-CB*CB-CC*CC+2.0*CA*CB*CC)
      DXMAX(1)=DM*CELL(2)*CELL(3)*SIN(DTOR*CELL(4))/V
      DXMAX(2)=DM*CELL(3)*CELL(1)*SIN(DTOR*CELL(5))/V
      DXMAX(3)=DM*CELL(1)*CELL(2)*SIN(DTOR*CELL(6))/V
      DM=DM*DM
```

```
C===> SET UP MATRIX TO CALCULATE DISTANCES AND ANGLES
      DO 1020 I=4,6
      J=8/I
      K=15/I
      T(J,K)=COS(DTOR*CELL(I))
 1020 T(K,J)=T(J,K)
      DO 1060 I=1,3
      T(I,I)=1.0
      DO 1040 J=1,3
 1040 T(I,J)=T(I,J)*CELL(I)*CELL(J)
 1060 CONTINUE
C===> SEARCH MAP FOR PEAK POSITIONS AND RECORD PEAK HEIGHTS
 1061 CALL PKSRCH(NSTORE,KUSE31,KUSE32,IHALF,ISTOP)
      IF(ISTOP.EQ.1) RETURN
      CALL SORT10(X,KUSE31,NAT,4)
C===> OUTPUT PEAKS TO USER
      ITOP=MIN0(15,NAT)
      CALL CLSCN
      WRITE(*,9012) ITOP
 9012 FORMAT(' Here is a list of the top',I3,' peaks:'/10X,
     1' Number',7X,'X'9X,'Y',9X,'Z',6X,'Height'  )
      DO 1147 J=1,ITOP
 1147 WRITE(*,9002)J,(X(K,J),K=1,4)
 9002 FORMAT(11X,I4,F12.4,2F10.4,F10.1)
      IF(NPATT.EQ.1) RETURN
      IF(LEVEL.LE.2) GOTO 1148
      WRITE(*,9004)
 9004 FORMAT('Do you want to continue with this solution ? (y/n)')
      CALL YN(I)
      IF(I.EQ.1) RETURN
 1148 WRITE(*,9003)
 9003 FORMAT(/'Peak interpretation now begins')
 1145 NPIC=NAT
C===> ADJUST VALUES OF DXMAX FOR BOND LENGTH CALCULATION
      FACT=DFRG/SQRT(DM)
      DO 1160 I=1,3
 1160 DXMAX(I)=FACT*DXMAX(I)
      DFRG = DFRG * DFRG
C===> IDENTIFY SEPARATE CLUSTERS AND CALCULATE BOND LENGTHS AND ANGLES
      NN=0
      NOFRAG=0
      DO 1200 LOOP=1,KUSE31
      IFRAG(LOOP)=0
      IUSE(LOOP)=0
 1200 KEYST(LOOP)=0
      IF(NOJOIN.EQ.0) CALL CLSTRS(0)
      IF(NOJOIN.EQ.0.AND.MCON.GT.0) GO TO 1230
      WRITE(NOUT,1220)
 1220 FORMAT( /1H ,20X,34HNojoin specified or no bonds found//)
      IF(NOJOIN.EQ.0.AND.MCON.EQ.0) WRITE(*,1221)
 1221 FORMAT('No bonds found')
      GOTO 1650
C===> FOR EACH CLUSTER OF MORE THAN MNPPF PEAKS WHICH HAS BEEN FOUND,
C===> PLOT THE PEAKS IN PROJECTION, INTERPRET THEM IN TERMS OF
C===> BONDED PEAKS AND COMPARE THEM WITH THOSE ATOMS INPUT BY THE USER
 1230 WRITE(*,1222) NFRAG
 1222 FORMAT(/'Total no.of clusters=',I3)
      IF(NFRAG.LE.3) GOTO 1224
      WRITE(*,1223)
 1223 FORMAT('More than three clusters-do you wish to continue ? (y/n)'
     1)
```

```
      CALL YN(I)
      IF(I.EQ.1) RETURN
 1224 L=0
      DO 1600 NOFRG=1,NFRAG
      NOFRAG = NOFRG
      IF (KFRAG(NOFRAG).LE.MNPPF) GOTO 1600
      IW = 0
      NINT=1
      FOM=0
 1240 ISTART=0
C===> INTERPRETATION OF PEAKS BEGINS WITH PEAK ISTART
C===> INTERPRET CLUSTER IN TERMS OF BONDED PEAKS AND ELIMINATE
C===> PEAKS WHICH DO NOT FULFIL DISTANCE AND ANGLE CRITERIA
      CALL INTERP(ISTART,KUSE31,FOMF)
      IF(ISTART.EQ.0)GO TO 1600
C===> ACCEPT INTERPRETATION IF FRAGMENT 1 CONTAINS MORE THAN MNPPF PEAKS
      IF (ISTART .EQ. 1) GO TO 1400
      IF (LFRAG(1).LT.MNPPF+1) GOTO 1240
C===> IF MORE THAN 1 FRAGMENT,SHUFFLE THEM AROUND
 1400 IF (MFRAG.LE.1.OR.NN.EQ.0) GOTO 1450
      CALL SHFFLE(MAXFRG)
      IF (MAXFRG.EQ.0) GOTO 1450
C===> REDO CLUSTER - INITIALISE ARRAYS FOR CLSTRS
      MC=0
      M=1
      DO 1418 I=1,NAT
      IF(IFRAG(I).NE.0.AND.IFRAG(I).NE.-1000) GO TO 1412
      IUSE(I)=0
      LW(I)=0
      LR(I)=0
      GO TO 1418
 1412 IF(M.GT.MCON) GO TO 1418
      II=IBOND(M)/4194304
      IF(II-I) 1416,1414,1418
 1414 MC=MC+1
      IBOND(MC)=IBOND(M)
 1416 M=M+1
      GO TO 1412
 1418 CONTINUE
      MCON=MC
      CALL CLSTRS(1)
C===> REINTERPRET CLUSTER
      FOM=0.0
 1420 CALL INTERP(ISTART,KUSE31,FOMF)
C===> ACCEPT INTERPRETATION IF FRAGMENT 1 CONTAINS MORE
C===> THAN MNPPF PEAKS AND MORE THAN ONE NEW PEAK IS INCLUDED IN THE
C===> INTERPRETATION AND THE SIZE OF THE FRAGMENTS IS SIGNIFICANT.
      IF (ISTART.EQ.0) GOTO 1600
      IF (ISTART.EQ.1) GOTO 1450
      IF (LFRAG(1).GE.MNPPF+1.AND.FOMF*FLOAT(IWT).GT.FOM) GOTO 1450
      ISTART=0
      GOTO 1420
C===> PLOT THIS CLUSTER IN THE BEST PROJECTION(S)
 1450 WRITE(*,1226)
 1226 FORMAT(/'Press <CR> to continue'\)
 1451 IF (NOPROJ.GE.0.AND.NINT.EQ.1) CALL PICTUR(NUM)
C===> COMPARE THE FRAGMENTS FOUND WITH THOSE INPUT BY THE USER
      IF (MFRN.GT.0) CALL COMPAR
C===> OPTION TO DRAW CLUSTER
      IF (L.EQ.179) GOTO 1600
      IF(NOFRAG.GT.1) WRITE(*,1470)
```

```
 1470 FORMAT(/'Do you wish to continue examining clusters ? (y/n)'\)
      CALL YN(JJ)
      IF(JJ.EQ.0) GOTO 1480
      L=179
      GOTO 1600
C===> LOOK AT CONTOURED SECTIONS OF EITHER E-MAPS OR FOURIER MAPS
 1480 IF(ICONT.EQ.1.AND.ICNT.EQ.1.AND.ITIMES.EQ.0) THEN
      CALL CONTOR(LMAP,NXVAL,NYVAL,NZVAL,NX,NYT,NZ,X,NAT,NSYM)
      CALL PAUSE
      ITIMES=1
      ENDIF
C===> OUTPUT BONDING PATTERN AND IDENTIFIED ATOMS
      CALL OUTERP(NELSE,ISTART,FOMF,NUM,NSYMOP)
      IF(ISTART.GT.0) NINT=NINT+1
      IF (NINT.GT.NOINT) GOTO 1600
      ISTART=0
      GOTO 1420
 1600 CONTINUE
C===> OUTPUT INTERPEAK DISTANCES, PEAK HEIGHTS, AND BOND ANGLES
      CALL BONGLE
C===> CREATE PEAK FILE FOR RECYCLING. IUSE(I).LT.0 IF PEAK I IS GOOD
 1650 REWIND NTAPEI
      DO 1800 I=1,NAT
 1800 WRITE(NTAPEI,9700)       (X(J,I),J=1,3)
      ENDFILE NTAPEI
      REWIND NTAPEI
 9700 FORMAT( 'ATOM C     ',3F10.5)
      RETURN
      END
C-------------------------------------------------------------------------------
      SUBROUTINE PKSRCH(NR3D,KUSE31,KUSE32,IHALF,ISTOP)
C===> FIND POSITIONS OF PEAKS BY FITTING QUADRATIC FUNCTION TO 19 POINTS
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),DD(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),ICLCK,ICNT
      COMMON /BLK1/ NSTORE(25100),XA(3,200),XB(4,200),ISYM(100),
     1 IFRAG(200),ICON(200),IBOND(2000),JFRAG(200),
     2 JCON(200),JBOND(2000),IUSE(200),IONA(6,200),MO(200),
     3 MU(200),ZY(200),NU(200),KEYST(200),LH(200),LR(200),LX(200),
     4 JK(200),IK(200),KEY(200),IPL(200),CAN(200),CON(200),LL(200),
     5 IA(6,200),LS(200),X(4,200)
      COMMON /BLK2/ IS(2,3,24),TS(3,24),NSYM,ICENT,LAT,IAPX,
     1 CELL(6),T(3,3),DXMAX(3),TL(3,4),FS(3,3,24),MLAT,NEQV,MCON,INT,
     2 FOM,KFRAG(20),NFRAG,LFRAG(20),MFRAG,NOFRAG,NN,MFRN,IFMT(20),
     3 KMATCH(20),ITYPE,NOINT,MNPPF
      COMMON /BLK5/ NOJOIN,NOPROJ,THOPI,FACTOR,RMAX,RMIN,DOUT,
     1 MH(3),ILEVEL,IPNT,XLIM(3),NOSET,NPIC,NP(3),NEX,ANGMIN,ANGMAX,
     2 DMIN,DMAX,DMUT,DM,DFRG,ALN,NPATT,IVDU,NAT,NPC,SCALE,IPASS
      DIMENSION  NR3D(KUSE32), XS(3), X1(3), IDIFF(19), B(19)
C===> INPUT DIMENSIONS OF E-MAP AND CHECK ARRAY SIZE
      REWIND NTAPEE
C===> NO DUMMY READS ON THE STRIDE
      READ(NTAPEE)(RSIZE,J=1,227)
      READ (NTAPEE) NNX,NNZ,NNY
C===> HALVE Y-AXIS IF POSSIBLE
      NNYOLD=NNY
      IF (IHALF.NE.0) NNY=NNY-NNY/2
      NNXP2 = NNX + 2
      NXZ = NNXP2 * (NNZ + 2)
```

```
      NXZ3 = 3 * NXZ
      IF (NXZ3 .LE. KUSE32) GO TO 1020
      WRITE (NOUT,940) KUSE32, NXZ3
  940 FORMAT(//33H Not enough room in array NSTORE.,10X,18HSpace availab
     1le  =,I7,10X,17HSpace required  =,I7)
      WRITE(*,9410)
 9410 FORMAT('Not enough room for the map in array NSTORE')
      STOP
      RETURN
C===> INITIALISE CONSTANTS
 1020 DX = 1.0 / FLOAT(NNX)
      DY = 1.0/FLOAT(NNYOLD)
      DZ = 1.0/FLOAT(NNZ)
      ILEVEL = 0
      LIMIT = MIN0(KUSE31, 2*NAT)
C===> SET UP TABLE OF OFFSETS TO LOOK UP 19 POINTS IN MAP
 1100 IDIFF(1) = -NXZ - 1
      IDIFF(2) = -NXZ - NNXP2
      IDIFF(3) = -NXZ
      IDIFF(4) = -NXZ + NNXP2
      IDIFF(5) = -NXZ + 1
      IDIFF(6) = -NNXP2 - 1
      IDIFF(7) = -1
      IDIFF(8) = NNXP2 - 1
      IDIFF(9) = -NNXP2
      IDIFF(10) = 0
      DO 1120 I=1,9
      J=20-I
 1120 IDIFF(J) = -IDIFF(I)
C===> INITIALISE VARIABLES
      NO = 0
      IY = -1
      NY = 0
C===> REPOSITION FILE TO READ IN FIRST TWO SECTIONS AGAIN
 1200 REWIND NTAPEE
C===> NO DUMMY READS - READ FIRST TWO RECORDS IN ONE GO
      READ(NTAPEE)(RSIZE,J=1,230)
      IF (IY+2.EQ.NNYOLD) GOTO 1400
C===> READ IN FIRST SECTION (END OF LAST UNIT CELL)
      MAX=NXZ
      ISKIP=NNYOLD-1
      DO 1305 I=1,ISKIP
 1305 READ (NTAPEE)(RSIZE,J=1,2)
      CALL RDSECT(NR3D,MAX,NNXP2,NNZ,NXZ3,KUSE32)
      REWIND NTAPEE
C===> READ SECOND SECTION (START OF UNIT CELL)
      READ(NTAPEE)(RSIZE,J=1,230)
      CALL RDSECT(NR3D,MAX,NNXP2,NNZ,NXZ3,KUSE32)
C===> READ IN THE NEXT SECTION
 1400 MX = MAX - NXZ + NNX + 1
      CALL RDSECT(NR3D,MAX,NNXP2,NNZ,NXZ3,KUSE32)
      IY = IY + 1
      NY = MOD(NY+2, 3) - 1
      KK = NXZ3
      IF (NY) 1440,1460,1500
 1440 KK = -NXZ3
 1460 DO 1480 I=1,5
      IDIFF(I) = IDIFF(I) - KK
 1480 CONTINUE
      IF (NY .EQ. 0) GO TO 1540
 1500 DO 1520 I=15,19
```

```
 1520 IDIFF(I) = IDIFF(I) - KK
C===> SEARCH ONE SECTION OF THE E-MAP
 1540 DO 2000 IZ=1,NNZ
      MN = MX + 3
      MX = MX + NNXP2
      DO 1980 IX=MN,MX
      IF (NR3D(IX) .LT. ILEVEL) GO TO 1980
C===> LOCATE GRID POINT WITH MAXIMUM DENSITY
      DO 1560 I=1,9
      J = IDIFF(I) + IX
      IF (NR3D(IX) .LE. NR3D(J)) GO TO 1980
 1560 CONTINUE
      DO 1580 I=11,19
      J = IDIFF(I) + IX
      IF(NR3D(IX).LT.NR3D(J)) GOTO 1980
 1580 CONTINUE
C===> FITS A QUADRATIC FUNCTION TO 19 POINTS
C===> DENS=A+C*X+D*Y+E*Z-0.5*F*(X*X+Y*Y+Z*Z)
      DO 1600 I=1,19
      J=IDIFF(I)+IX
 1600 B(I) = NR3D(J)
      B1 = B(3) + B(7) + B(9) + B(11) + B(13) + B(17)
      B2 = B(1) + B(2) + B(4) + B(5) + B(6) + B(8) + B(12) + B(14) +
     1  B(15) + B(16) + B(18) + B(19)
      F = (30.0 * B(10) + 11.0 * B1 - 8.0 * B2) / 63.0
      C = (B(5)+B(12)+B(13)+B(14)+B(19)-B(1)-B(6)-B(7)-B(8)-B(15))/10.0
      DELTAX = C / F
      IF (ABS(DELTAX) .GT. 1.0) GO TO 1620
      D = (B(15)+B(16)+B(17)+B(18)+B(19)-B(1)-B(2)-B(3)-B(4)-B(5))/10.0
      DELTAY = D / F
      IF (ABS(DELTAY) .GT. 1.0) GO TO 1620
      E = (B(4)+B(8)+B(11)+B(14)+B(18)-B(2)-B(6)-B(9)-B(12)-B(16))/10.0
      DELTAZ = E / F
      IF (ABS(DELTAZ) .LE. 1.0) GO TO 1640
 1620 DELTAX = 0.0
      DELTAY = 0.0
      DELTAZ = 0.0
 1640 XX = (FLOAT(IX-MN+1) + DELTAX) * DX
      YY = (FLOAT(IY) + DELTAY) * DY
      ZZ = (FLOAT(IZ) + DELTAZ) * DZ
C===> PEAK HEIGHT INTERPOLATION - NOT SUITABLE IF PEAK SHAPE IS POOR
C===> A = (9.0 * B(10) + 4.0 * B1 - B2) / 21.0
C===> B(10) = AMAX1(A+0.5*(C*DELTAX+D*DELTAY+E*DELTAZ), B(10))
      NOP1 = NO+1
      X(1,NOP1) = XX
      X(2,NOP1) = YY
      X(3,NOP1) = ZZ
      X(4,NOP1) = B(10)
      IF (NO .EQ. 0) GO TO 1820
      IR=0
C===> IF TWO PEAKS ARE CLOSER THAN DM ANGSTROMS ELIMINATE THE SMALLER
      IPRIN=0
      DO 1800 K=1,NSYM
      CALL OPER(K,XS,XX,YY,ZZ,IPRIN)
      DO 1780 I=1,NO
      DO 1720 L=1,3
      X1(L) = X(L,I) - XS(L)
 1680 IF (ABS(X1(L)) .LE. 0.5) GO TO 1700
      X1(L) = X1(L) - SIGN(1.0, X1(L))
      GO TO 1680
 1700 IF (ABS(X1(L)) .GT. DXMAX(L)) GO TO 1780
```

```
 1720 CONTINUE
      IF (QUAD(X1,X1,T) .GT. DM) GO TO 1780
      IF(IR.GT.0) X(4,IR)=0.0
      IR=0
      IF (B(10) .LE. X(4,I)) GO TO 1980
      X(1,I) = XX
      X(2,I) = YY
      X(3,I) = ZZ
      X(4,I) = B(10)
      IR=I
 1780 CONTINUE
 1800 CONTINUE
      IF(IR.GT.0) GO TO 1980
 1820 NO = NOP1
      IF (NO .LT. LIMIT) GO TO 1980
      CALL SORT10(X,KUSE31,NO,4)
      NO = NPIC
      ILEVEL = X(4,NPIC) + 0.5
 1980 CONTINUE
 2000 CONTINUE
      IF (IY .GE. NNY) GO TO 2100
      IF (IY - NNYOLD + 2) 1400,1200,1400
C===> SORT PEAKS IN ORDER OF PEAK HEIGHT
 2100 CALL SORT10(X,KUSE31,NO,4)
      NNN = MIN0(NO,NPIC)
      IF (NNN .EQ. NPIC) RETURN
C===> INSUFFICIENT PEAKS FOUND - LOWER SCAN ILEVEL AND TRY AGAIN
      ILEVEL = ILEVEL - 100
      IF(ILEVEL.GE.(-200)) GO TO 1100
      NPIC=NO
      RETURN
      END
C-------------------------------------------------------------------------------
      SUBROUTINE RDSECT(NR3D,MAX,NNXP2,NNZ,NXZ3,KUSE32)
      DIMENSION  NR3D(KUSE32)
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),ICLCK,ICNT
      IF (MAX .GE. NXZ3) MAX = 0
      MIN=MAX+1
      MAX=MAX+NNXP2*NNZ
      READ(NTAPEE)(NR3D(IX),IX=MIN,MAX)
      MX=MIN+2*NNXP2-1
      DO 3 IX=MIN,MX
      MAX=MAX+1
    3 NR3D(MAX)=NR3D(IX)
      RETURN
      END
C-------------------------------------------------------------------------------
      SUBROUTINE CLSTRS(NPASS)
C===> IDENTIFY SEPARATE BONDED CLUSTERS AND SET UP BONDING ARRAY.
C===> NPASS DETERMINES PATH THROUGH SUBROUTINE.
C===> NPASS = 0 : SET UP ALL CLUSTERS ; NPASS =1 : REDO CLUSTER NOFRAG
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),ICLCK,ICNT
      COMMON /BLK1/ NSTORE(25100),XA(3,200),XB(4,200),ISYM(100),
```

```
     1 IFRAG(200),ICON(200),IBOND(2000),JFRAG(200),
     2 JCON(200),JBOND(2000),IUSE(200),IONA(6,200),MO(200),
     3 MU(200),NO(200),NU(200),KEYST(200),LW(200),LR(200),LX(200),
     4 JK(200),IK(200),KEY(200),IPL(200),CAN(200),CON(200),LL(200),
     5 IA(6,200),LS(200),X(4,200)
       COMMON /BLK2/ IS(2,3,24),TS(3,24),NSYM,ICENT,LAT,IAPX,
     1 CELL(6),T(3,3),DXMAX(3),TL(3,4),FS(3,3,24),MLAT,NEQV,MCON,IHT,
     2 FOM,KFRAG(20),NFRAG,LFRAG(20),MFRAG,NOFRAG,NN,MFRN,IFMT(20),
     3 KMATCH(20),ITYPE,NOINT,MNPPF
       COMMON /BLK4/ IONP(6,200),NAFRG(5),ICH(12),BB(6),LINX(60),NSEQ
       COMMON /BLK5/ NOJOIN,NOPROJ,THOPI,FACTOR,RMAX,RMIN,DOUT,
     1 MH(3),ILEVEL,IPNT,XLIM(3),NOSET,NPIC,NP(3),NEX,ANGMIN,ANGMAX,
     2 DMIN,DMAX,DMUT,DM,DFRG,ALN,NPATT,IVDU,NAT,NPC,SCALE,IPASS
       DIMENSION     IB(3),XS(3),X1(3),       BSTOR(3),XSTOR(3)
       ICHECK=0
       IF (NPASS.EQ.0) GOTO 1010
       ICHECK=1
C===>  REDUCE ISYM ARRAY TO EXCLUDE NOFRAG BONDINGS
       NNN=0
       DO 995 I=1,NN
       IF (ISYM(I)/250000.EQ.NOFRAG) GOTO 995
       NNN=NNN+1
       ISYM(NNN)=ISYM(I)
  995  CONTINUE
       NN=NNN
       GOTO 1025
 1000  NAT=NPIC
C===>  CLEAR ARRAYS
 1010  NN=0
       MCON=0
       DO 1020 I=1,NAT
       IFRAG(I)=0
       ICON(I)=0
       LW(I) = 0
 1020  LR(I) = 0
C===>  APPLY DISTANCE CRITERION TO PEAKS TO IDENTIFY SEPARATE CLUSTERS
C===>  PEAKS WITHIN DISTANCE DMAX ARE POSSIBLE BONDED PEAKS
       NFRAG=0
       NOFRAG=0
 1025  NATM1=NAT-1
C===>  START A NEW CLUSTER AT ATOM II
       DO 1180 II=1,NATM1
       IF (IFRAG(II).EQ.(-1000)) GOTO 1030
       IF (ICHECK.EQ.1.OR.IFRAG(II).NE.0) GOTO 1180
       NOFRAG=NOFRAG+1
       NFRAG=NFRAG+1
 1030  ICHECK=0
       IFRAG(II)=NOFRAG
C===>  KOUNT RECORDS NUMBER OF PEAKS IN PRESENT CLUSTER.
       KOUNT=1
       I=II
       IBEGIN=II
       IF (NPASS.EQ.1) IBEGIN=1
 1040  DO 1140 J=IBEGIN,NAT
C===>  DO NOT CONSIDER PEAK PAIRS MORE THAN ONCE
       IF (IFRAG(J).LT.0.AND.IFRAG(J).NE.(-1000)) GOTO 1140
       JMOVE = 0
       KSYM=0
       DO 1120 K=1,NSYM
C===>  OMIT THE IDENTITY OPERATION WHEN I & J ARE THE SAME
       IF (I.EQ.J.AND.K.EQ.1) GOTO 1120
```

```
C===> CALC K'TH SYMM POSITION OF J'TH PEAK & MOVE AS CLOSE
C===> TO I'TH PEAK AS POSSIBLE BY LATTICE TRANSLATIONS
      IF(JSYMM(I,J,K,IB,XS,X1).NE.0) GO TO 1120
C===> CALCULATE DISTANCE BETWEEN PEAKS I & J
      DIST=QUAD(X1,X1,T)
      IF (DIST.GT.DFRG) GOTO 1120
C===> PEAK J IS CLOSE TO CLUSTER NOFRAG
      LR(J)=1
C===> DO NOT CONSIDER PEAKS OUTSIDE BONDING DISTANCE
C===> OR PEAKS LYING ON SYMMETRY ELEMENTS
      IF (DIST.GT.DMAX) GOTO 1120
      IF (DIST.LT.0.04) GOTO 1120
C===> PEAKS I & J ARE WITHIN BONDING DISTANCE-ADD TO CLUSTER
      IF (IABS(IFRAG(J)).EQ.NOFRAG) GOTO 1080
      IF (IFRAG(J).EQ.(-1000)) GOTO 1060
C===> PEAK J CAN BE MOVED, SO MOVE IT & RECORD SYMM. OP.
      JMOVE=1
      KSYM=K
      DO 1050 L=1,3
      XSTOR(L)=XS(L)
 1050 BSTOR(L)=FLOAT(IB(L)-5)
C===> ADD J TO CLUSTER
 1060 IFRAG(J)=NOFRAG
      KOUNT=KOUNT+1
C===> IF PEAK MOVED, ADD TO BONDING ARRAY
      IF (KSYM.EQ.K) GOTO 1100
 1080 JBND=100000*IB(1)+10000*IB(2)+1000*IB(3)+K
      IF (JBND.EQ.555001) GOTO 1100
C===> CLUSTER NOFRAG BONDS TO ITSELF THROUGH I & J - RECORD IN ISYM
      NN=NN+1
      IF(I.LT.J) ISYM(NN)=250000*NOFRAG+500*I+J
      IF(I.GE.J) ISYM(NN)=250000*NOFRAG+500*J+I
      GOTO 1120
 1100 IF(I.EQ.J) GO TO 1120
      IDIST=1000.0*SQRT(DIST)+0.5
      MCON=MCON+1
      IBOND(MCON)=(512*I+J)*8192+IDIST
      MCON=MCON+1
      IBOND(MCON)=(512*J+I)*8192+IDIST
 1120 CONTINUE
      IF (JMOVE .EQ. 0) GO TO 1140
      DO 1125 L=1,3
 1125 X(L,J) = XSTOR(L)
C===> IF ONE PEAK IN A FRAGMENT MOVES, MOVE ALL THE REST
      IF (NPASS.NE.1.OR.JFRAG(J).EQ.0) GOTO 1140
      IPRIN=0
      DO 1135 LOOP=1,NAT
      IF (LOOP.EQ.J) GOTO 1135
      IF (IFRAG(LOOP).NE.0) GOTO 1135
      IF (JFRAG(LOOP).NE.JFRAG(J)) GOTO 1135
      CALL OPER(KSYM,XS,X(1,LOOP),X(2,LOOP),X(3,LOOP),IPRIN)
      DO 1130 L=1,3
 1130 X(L,LOOP)=XS(L)
 1135 CONTINUE
 1140 CONTINUE
C===> PEAK I HAS BEEN COMPLETELY DEALT WITH
      IFRAG(I)=-IFRAG(I)
C===> CHOOSE NEXT PEAK IN CLUSTER
      DO 1160 I=1,NAT
      IF(IFRAG(I).EQ.NOFRAG)GO TO 1040
 1160 CONTINUE
```

```
C===> CHECK FOR CLOSE APPROACHES BETWEEN CLUSTERS
      IF (NPASS.EQ.1) GOTO 1178
      IF (NAT .EQ. NPIC) GO TO 1178
      DO 1175 I=1,NAT
      IF (KOUNT.LE.4) GO TO 1170
      IF (IABS(IFRAG(I)) .NE. NOFRAG) GO TO 1165
      IF (LW(I) .EQ. 0) GO TO 1170
C===> CLUSTERS ARE CLOSE TOGETHER - INCREASE NUMBER OF PEAKS
      GO TO 1000
 1165 IF (LW(I) .EQ. 0) LW(I) = LR(I)
 1170 LR(I) = 0
 1175 CONTINUE
 1178 IF(KOUNT.EQ.1)GO TO 1179
      KFRAG(NOFRAG)=KOUNT
      GOTO 1180
C===> SINGLE PEAKS ARE GIVEN A CLUSTER NUMBER OF ZERO
 1179 IFRAG(II)=0
      NFRAG=NFRAG-1
      NOFRAG=NOFRAG-1
 1180 CONTINUE
      DO 1240 I=1,NAT
 1240 IFRAG(I)=IABS(IFRAG(I))
C===> PUT ENTRIES IN IBOND ARRAY IN ORDER OF PEAK HEIGHT
      IF(MCON.GT.0) CALL ISORT(IBOND,MCON)
      RETURN
      END
C-------------------------------------------------------------------------------
      SUBROUTINE INTERP(ISTART,KUSE31,FOMF)
C===> INTERPRET CLUSTER IN TERMS OF BONDED PEAKS AND ELIMINATE
C===> PEAKS WHICH DO NOT FULFIL DISTANCE AND ANGLE CRITERIA
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),ICLCK,ICNT
      COMMON /BLK1/ NSTORE(25100),XA(3,200),XB(4,200),ISYM(100),
     1 IFRAG(200),ICON(200),IBOND(2000),JFRAG(200),
     2 JCON(200),JBOND(2000),IUSE(200),IONA(6,200),MO(200),
     3 MU(200),NO(200),NU(200),KEYST(200),LW(200),LR(200),LX(200),
     4 JK(200),IK(200),KEY(200),IPL(200),CAN(200),CON(200),LL(200),
     5 IA(6,200),LS(200),X(4,200)
      COMMON /BLK2/ IS(2,3,24),TS(3,24),NSYM,ICENT,LAT,IAPX,
     1 CELL(6),T(3,3),DXMAX(3),TL(3,4),FS(3,3,24),MLAT,NEQV,MCON,INT,
     2 FOMN,KFRAG(20),NFRAG,LFRAG(20),MFRAG,NOFRAG,NN,MFRN,IFMT(20),
     3 KMATCH(20),ITYPE,NOINT,MNPPF
      COMMON /BLK4/ IONP(6,200),NAFRG(5),ICH(12),IB(6),LINX(60),NSEQ
      COMMON /BLK5/ NOJOIN,NOPROJ,THOPI,FACTOR,RMAX,RMIN,DOUT,
     1 MH(3),ILEVEL,IPNT,XLIM(3),NOSET,NPIC,NP(3),NEX,ANGMIN,ANGMAX,
     2 DMIN,DMAX,DMUT,DM,DFRG,ALN,NPATT,IVDU,NAT,NPC,SCALE,IPASS
      DIMENSION     X1(3),X2(3)
C===> COPY CLUSTER INTO NEW ARRAY
      M=1
      J=0
      DO 980 I=1,NAT
      JCON(I)=0
      JFRAG(I)=0
      ICON(I)=J
      IF (IFRAG(I) .NE. NOFRAG) GO TO 980
      IF(ISTART.EQ.0.AND.IUSE(I).EQ.0)ISTART=I
      IF (ISTART.LE.IHVY) ISTART=0
  950 IF(M.GT.MCON) GO TO 980
```

```
      II=IBOND(M)/4194304
      IF(II-I) 970,960,980
  960 JCON(I)=JCON(I)+1
      J=J+1
      JBOND(J)=MOD(IBOND(M),4194304)
      IF (I.EQ.ISTART.AND.JBOND(J)/8192.LE.IHVY) ISTART = 0
  970 M=M+1
      GO TO 950
  980 CONTINUE
      ICON(NAT+1)=J
      IF (ISTART.EQ.0) RETURN
      MFRAG=0
      IDMIN=1000.0*SQRT(DMIN)+0.5
C===> START INTERPRETATION OF NEW SET OF PEAKS AT PEAK II
      II=ISTART
      IF (IHVY.EQ.0) GOTO 1080
      DO 1020 I=1,IHVY
      IF (IFRAG(I).NE.NOFRAG) GOTO 1020
      JCON(I)=0
      KS=ICON(I)+1
      KL=ICON(I+1)
      IF(KS.GT.KL) GO TO 1010
      DO 1000 J=KS,KL
      L=JBOND(J)/8192
 1000 CALL ELIM(L,JFRAG(L),JCON,JBOND,ICON,KUSE31)
 1010 CALL ELIM(I,JFRAG(I),JCON,JBOND,ICON,KUSE31)
 1020 CONTINUE
      GOTO 1080
 1040 II=0
 1060 II=II+1
      IF(II.GT.NAT)GO TO 1620
      IF(JFRAG(II).NE.0)GO TO 1060
      IF (JCON(II) .LE. 0) GO TO 1060
 1080 MFRAG=MFRAG+1
      JFRAG(II)=MFRAG
      LFRAG(MFRAG)=1
      I=II
 1100 KS=ICON(I)+1
      KL=ICON(I+1)
C===> APPLY MINIMUM DISTANCE CRITERION TO PEAKS AROUND THE ITH PEAK
      DO 1140 J=KS,KL
      IF(JBOND(J).EQ.0) GO TO 1140
      IDIST=MOD(JBOND(J),8192)
      IF(IDIST.GE.IDMIN)GO TO 1140
      L=JBOND(J)/8192
C===> PEAKS I AND L ARE TOO CLOSE - ELIMINATE L
      CALL ELIM(L,JFRAG(L),JCON,JBOND,ICON,KUSE31)
 1140 CONTINUE
C===> APPLY ANGLE CRITERIA TO PEAKS AROUND THE ITH PEAK
      IF(KS.EQ.KL) GOTO 1480
C===> CONSIDER THE FRAGMENT - I - ANYTHING
      DO 1360 J=KS,KL
      IF(JBOND(J).EQ.0) GOTO 1360
      JJ=JBOND(J)/8192
C===> FIND A FRAGMENT PEAK
      IF(IABS(JFRAG(JJ)).NE.MFRAG) GOTO 1360
C===> SET UP VECTOR JJ - I FOR ANGLE CALCULATION
      DO 1300 L=1,3
 1300 X1(L) = X(L,JJ) - X(L,I)
      VEC1=0.001*FLOAT(MOD(JBOND(J),8192))
      DO 1340 K=KS,KL
```

```
      IF(JBOND(K).EQ.0) GO TO 1340
      KK=JBOND(K)/8192
      IF(JJ.EQ.KK)GO TO 1340
C===> CALCULATE COSINE OF ANGLE BETWEEN PEAKS JJ - I - KK
      DO 1320 L=1,3
 1320 X2(L)=X(L,KK)-X(L,I)
      VEC2=0.001*FLOAT(MOD(JBOND(K),8192))
      COSA=QUAD(X1,X2,T)/(VEC1*VEC2)
C===> IF ANGLE IS WITHIN LIMITS CONTINUE TESTING
      IF(COSA.GE.ANGMAX.AND.COSA.LE.ANGMIN)GO TO 1340
C===> ELIMINATE KK IF IT IS NOT IN THE FRAGMENT
      IF(IABS(JFRAG(KK)).NE.MFRAG)GO TO 1330
      LFRAG(MFRAG) = LFRAG(MFRAG) - 1
C===> KK IS IN THE FRAGMENT - HAS IT ALREADY BEEN CONSIDERED
      IF(JFRAG(KK).GT.0)GO TO 1330
C===> KK HAS ALREADY BEEN FULLY CONSIDERED FOR THE FRAGMENT-ELIMINATE I
      CALL ELIM(I,JFRAG(I),JCON,JBOND,ICON,KUSE31)
      GO TO 1500
 1330 CALL ELIM(KK,JFRAG(KK),JCON,JBOND,ICON,KUSE31)
 1340 CONTINUE
 1360 CONTINUE
C===> CONSIDER THE ANGLES NONFRAG - I - NONFRAG
      KLM1=KL-1
      DO 1460 J=KS,KLM1
      IF(JBOND(J).EQ.0) GO TO 1460
      JJ=JBOND(J)/8192
C===> FIND A NON-FRAGMENT PEAK
      IF(IABS(JFRAG(JJ)).EQ.MFRAG)GO TO 1460
C===> ADD PEAK TO FRAGMENT
      JFRAG(JJ)=MFRAG
      LFRAG(MFRAG)=LFRAG(MFRAG)+1
C===> SET UP VECTOR JJ - I FOR ANGLE CALCULATION
      DO 1400 L=1,3
 1400 X1(L)=X(L,JJ)-X(L,I)
      VEC1=0.001*FLOAT(MOD(JBOND(J),8192))
      JP1=J+1
      DO 1440 K=JP1,KL
      IF(JBOND(K).EQ.0) GO TO 1440
      KK = JBOND(K)/8192
C===> FIND A NON-FRAGMENT PEAK
      IF(IABS(JFRAG(KK)).EQ.MFRAG)GO TO 1440
C===> CALCULATE COSINE OF ANGLE BETWEEN PEAKS JJ - I - KK
      DO 1420 L=1,3
 1420 X2(L)=X(L,KK)-X(L,I)
      VEC2=0.001*FLOAT(MOD(JBOND(K),8192))
      COSA=QUAD(X1,X2,T)/(VEC1*VEC2)
C===> IF ANGLE IS OUTSIDE LIMITS ELIMINATE KK
      IF(COSA.LT.ANGMAX.OR.COSA.GT.ANGMIN)CALL ELIM(KK,JFRAG(KK),
     1   JCON,JBOND,ICON,KUSE31)
 1440 CONTINUE
 1460 CONTINUE
C===> SEE IF LAST PEAK CAN JOIN FRAGMENT
 1480 JJ=JBOND(KL)/8192
      IF(JCON(JJ).LE.0.OR.JFRAG(JJ).NE.0)GO TO 1490
      JFRAG(JJ)=MFRAG
      LFRAG(MFRAG)=LFRAG(MFRAG)+1
C===> PEAK I HAS NOW BEEN FULLY CONSIDERED
 1490 JFRAG(I)=-JFRAG(I)
C===> FIND LARGEST PEAK IN CLUSTER NOT YET DEALT WITH
 1500 DO 1520 I=1,NAT
      IF(JFRAG(I).EQ.MFRAG)GO TO 1100
```

```
 1520 CONTINUE
      IF(LFRAG(MFRAG).GT.1)GO TO 1600
C===> SINGLE PEAKS ARE GIVEN A FRAGMENT NUMBER OF ZERO
      JFRAG(II)=0
      MFRAG=MFRAG-1
 1600 IF(II-ISTART)1060,1040,1060
C===> INTERPRETATION COMPLETE
 1620 FOMF=0.0
      KKK=0
      DO 1640 I=1,NAT
      JFRAG(I)=IABS(JFRAG(I))
      J=JFRAG(I)
      IF(J.EQ.0.OR.LFRAG(J).LT.4) GO TO 1640
      FOMF=FOMF+ALOG10(FLOAT(LFRAG(J)))*X(4,I)
      IF(IUSE(I).EQ.0) KKK=1
 1640 CONTINUE
      IF(KKK.EQ.0) FOMF=0.0
      IF(IUSE(ISTART).EQ.0)IUSE(ISTART)=1
      RETURN
      END
C-------------------------------------------------------------------------------
      SUBROUTINE SHFFLE(MAXFRG)
C===> THIS ROUTINE SHUFFLES FRAGMENTS ACCORDING TO THE BONDINGS
C===> STORED IN ISYM, ALTERING FRAGMENT NUMBER AS WELL
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),ICLCK,ICNT
      COMMON /BLK1/ NSTORE(25100),XA(3,200),XB(4,200),ISYM(100),
     1 IFRAG(200),ICON(200),IBOND(2000),JFRAG(200),
     2 JCON(200),JBOND(2000),IUSE(200),IONA(6,200),MO(200),
     3 MU(200),NO(200),NU(200),KEYST(200),LH(200),LR(200),LX(200),
     4 JK(200),IK(200),KEY(200),IPL(200),CAN(200),CON(200),LL(200),
     5 IA(6,200),LS(200),X(4,200)
      COMMON /BLK2/ IS(2,3,24),TS(3,24),NSYM,ICENT,LAT,IAPX,
     1 CELL(6),T(3,3),DXMAX(3),TL(3,4),FS(3,3,24),MLAT,NEQV,MCON,IHT,
     2 FOM,KFRAG(20),NFRAG,LFRAG(20),MFRAG,NOFRAG,NN,MFRN,IFMT(20),
     3 KMATCH(20),ITYPE,NOINT,MNPPF
      COMMON /BLK4/ IONP(6,200),NAFRG(5),ICH(12),BB(6),LINX(60),NSEQ
      COMMON /BLK5/ NOJOIN,NOPROJ,THOPI,FACTOR,RMAX,RMIN,DOUT,
     1 MH(3),ILEVEL,IPNT,XLIM(3),NOSET,NPIC,NP(3),NEX,ANGMIN,ANGMAX,
     2 DMIN,DMAX,DMUT,DM,DFRG,ALN,NPATT,IVDU,NAT,NPC,SCALE,IPASS
      DIMENSION    XTEMP(3),TN(3),IB(3),XS(3),X1(3)
      MAXFRG=0
C===> CONSIDER EACH SYMM. OP. IN TURN
      DO 700 K1=1,NN
C===> OBTAIN PEAKS IN RELEVANT SYMM OPS AND CLUSTER
      ITEMP=ISYM(K1)/250000
      IF (ITEMP.NE.NOFRAG) GOTO 700
      I1=MOD(ISYM(K1),250000)/500
      I2=MOD(ISYM(K1),500)
C===> IF SAME FRAGMENT, DISREGARD SYMM. OP.
      IF(JFRAG(I1).EQ.JFRAG(I2)) GOTO 700
C===> DETERMINE WHICH FRAGMENT TO MOVE, AND OBTAIN CORRECT
C===> SYMMETRY OPERATION TO DO SO,ACCOUNTING FOR SINGLE PEAKS.
      JI1=JFRAG(I1)
      JI2=JFRAG(I2)
      IF (JI1.EQ.0) GOTO 150
      IF (JI2.EQ.0) GOTO 200
      IF(LFRAG(JI1).GE.LFRAG(JI2)) GOTO 200
```

```
  150 IAT=I1
      IOTHER=I2
      GOTO 300
  200 IAT=I2
      IOTHER=I1
  300 JIAT=JFRAG(IAT)
      JOTHER=JFRAG(IOTHER)
C===> IDENTIFY ALL PEAKS IN FRAGMENT TO BE MOVED, AND MOVE THEM
      DO 345 K=1,NSYM
      IF(JSYMM(IOTHER,IAT,K,IB,XS,X1).NE.0) GO TO 345
      IF (QUAD(X1,X1,T).LE.DMAX) GOTO 347
  345 CONTINUE
  347 DO 350 L=1,3
      X(L,IAT)=XS(L)
      TN(L)=FLOAT(IB(L)-5)
  350 CONTINUE
      JFRAG(IAT)=JOTHER
      IF (JIAT.EQ.0) GOTO 520
      IPRIN=0
      DO 500 K2=1,NAT
      IF (JFRAG(K2).NE.JIAT) GOTO 500
      CALL OPER(K,XTEMP,X(1,K2),X(2,K2),X(3,K2),IPRIN)
      DO 450 K5=1,3
  450 X(K5,K2)=XTEMP(K5)+TN(K5)
      JFRAG(K2)=JOTHER
  500 CONTINUE
C===> CORRECT LFRAG ARRAY
      LFRAG(JOTHER)=LFRAG(JOTHER)+LFRAG(JIAT)
      LFRAG(JIAT)=0
      GOTO 550
  520 LFRAG(JOTHER)=LFRAG(JOTHER)+1
  550 IF (LFRAG(JOTHER).GT.MAXFRG) MAXFRG=LFRAG(JOTHER)
C===> CLOSE UP LFRAG & JFRAG ARRAYS
      IF (JIAT.EQ.0) GOTO 700
      MFRAG=MFRAG-1
      DO 620 K7=1,MFRAG
      IF (LFRAG(K7).NE.0) GOTO 620
      LFRAG(K7)=LFRAG(K7+1)
      LFRAG(K7+1)=0
  620 CONTINUE
      DO 650 K8=1,NAT
      IF (JFRAG(K8).LT.JIAT) GOTO 650
      JFRAG(K8)=JFRAG(K8)-1
  650 CONTINUE
  700 CONTINUE
C===> RETURN IF NO SHUFFLING HAS TAKEN PLACE
      IF (MAXFRG.EQ.0) RETURN
C===> PUTS IFRAG -VE FOR ALL PEAKS IN CLUSTERS OTHER THAN NOFRAG,
C===> ZEROS IFRAG FOR MOVABLE PEAKS, AND PUTS IFRAG
C===> EQUAL TO -1000 FOR THE LARGEST COMBINED FRAGMENT.
      DO 900 LOOP=1,MFRAG
      IF (LFRAG(LOOP).NE.MAXFRG) GOTO 900
      GOTO 1000
  900 CONTINUE
 1000 JCOMB=LOOP
      DO 1300 LOOP=1,NAT
      IF (IFRAG(LOOP).EQ.NOFRAG) GOTO 1100
      IFRAG(LOOP)=-IFRAG(LOOP)
      GOTO 1300
 1100 IF (JFRAG(LOOP).EQ.JCOMB) GOTO 1200
      IFRAG(LOOP)=0
```

```
      GOTO 1300
 1200 IFRAG(LOOP)=-1000
 1300 CONTINUE
      RETURN
      END
C-------------------------------------------------------------------------------
      SUBROUTINE COMPAR
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),ICLCK,ICNT
      COMMON /BLK1/ NSTORE(25100),XA(3,200),XB(4,200),ISYM(100),
     1 IFRAG(200),ICON(200),IBOND(2000),JFRAG(200),
     2 JCON(200),JBOND(2000),IUSE(200),IONA(6,200),MO(200),
     3 MU(200),NO(200),NU(200),KEYST(200),LH(200),LR(200),LX(200),
     4 JK(200),IK(200),KEY(200),IPL(200),CAN(200),CON(200),LL(200),
     5 IA(6,200),LS(200),X(4,200)
      COMMON /BLK2/ IS(2,3,24),TS(3,24),NSYM,ICENT,LAT,IAPX,
     1 CELL(6),T(3,3),DXMAX(3),TL(3,4),FS(3,3,24),MLAT,NEQV,MCON,INT,
     2 FOM,KFRAG(20),NFRAG,LFRAG(20),MFRAG,NOFRAG,NN,MFRN,IFMT(20),
     3 KMATCH(20),ITYPE,NOINT,MNPPF
      COMMON /BLK4/ IONP(6,200),NAFRG(5),ICH(12),IB(6),LINX(60),NSEQ
      COMMON /BLK5/ NOJOIN,NOPROJ,TWOPI,FACTOR,RMAX,RMIN,DOUT,
     1 MH(3),ILEVEL,IPNT,XLIM(3),NOSET,NPIC,NP(3),NEX,ANGMIN,ANGMAX,
     2 DMIN,DMAX,DMUT,DM,DFRG,ALN,NPATT,IVDU,NAT,NPC,SCALE,IPASS
      COMMON/BLK6/ INBOND(1000),RUBISH(150)
      DO 2000 MOFRAG=1,MFRAG
      KMATCH(MOFRAG)=0
      IF(LFRAG(MOFRAG).LT.5) GO TO 2000
      M=0
      DO 100 I=1,NAT
      NU(I)=0
      IF(JFRAG(I).NE.MOFRAG) GO TO 100
      M=M+1
      NO(M)=I
      NU(I)=M
  100 CONTINUE
      DO 150 I=1,NAT
      IF(NU(I).EQ.0) GO TO 150
      KEYST(I)=0
      M=NU(I)
      DO 130 J=1,6
  130 IONP(J,M)=0
      K=0
      KS=ICON(I)+1
      KL=ICON(I+1)
      DO 140 J=KS,KL
      MC=JBOND(J)/8192
      IF(MC.LE.0) GO TO 140
      K=K+1
      IONP(K,M)=NU(MC)
  140 CONTINUE
  150 CONTINUE
      CALL CANON(M,MR)
      KM=0
      DO 200 I=1,M
      K=NU(I)
      MU(K)=NO(I)
      LH(I)=LX(I)
      IF(I.LE.6.OR.I.LE.MR) KM=KM+LH(I)
```

```
      DO 190 J=1,6
  190 IONA(J,I)=IONP(J,I)
  200 CONTINUE
      NADR=0
      MSTOR=1
      MERIT=1
      DO 1800 NX=1,NSEQ
      N=NAFRG(NX)
      IF(N.EQ.0) GOTO 2000
      DO 350 I=1,N
      LL(I)=0
      DO 340 J=1,6
  340 IONP(J,I)=0
  350 CONTINUE
  360 NADR=NADR+1
      IF(INBOND(NADR).LT.(-999)) GO TO 370
      I=INBOND(NADR)/512
      J=MOD(INBOND(NADR),512)
      K=LL(I)+1
      IONP(K,I)=J
      LL(I)=K
      K=LL(J)+1
      IONP(K,J)=I
      LL(J)=K
      GO TO 360
  370 CALL CANON(N,NR)
      KN=0
      DO 410 I=1,N
      IF(I.LE.6.OR.I.LE.NR) KN=KN+LX(I)
  410 CONTINUE
      IF(KN.GE.KM) CALL TELL(N,NR,M,MR,MERIT,1,IONA,IONP,LM)
      IF(KN.LT.KM) CALL TELL(M,MR,N,NR,MERIT,2,IONP,IONA,LX)
      IF(MERIT.LE.MSTOR) GO TO 1800
      MSTOR=MERIT+1
      DO 450 I=1,N
      K=NU(I)
  450 NO(K)=I
      DO 500 I=1,NAT
      IF(JFRAG(I).EQ.MOFRAG) KEYST(I)=0
  500 CONTINUE
      DO 1700 I=1,M
      IF(MO(I).EQ.0) GO TO 1700
      K=MO(I)
      J=MU(I)
      KEYST(J)=NO(K)
 1700 CONTINUE
      KMATCH(MOFRAG)=NX
 1800 CONTINUE
 2000 CONTINUE
      RETURN
      END
C-------------------------------------------------------------------------------
      SUBROUTINE PICTUR(NUM)
C===> PLOT PROJECTION OF CLUSTER FOUND ONTO LEAST SQUARES PLANE
C===> PLOT ALSO AN ORTHOGONAL  PROJECTION IF THE ATOMIC OVERLAP IS LARGE
      CHARACTER*4 IFMT,ISPCH,IBLANK,IEND
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IDUM,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),ICLCK,ICNT
```

```
      COMMON /BLK1/ NSTORE(25100),XA(3,200),XB(4,200),ISYM(100),
     1 IFRAG(200),ICON(200),IBOND(2000),JFRAG(200),
     2 JCON(200),JBOND(2000),IUSE(200),IONA(6,200),MO(200),
     3 MU(200),NO(200),NU(200),KEYST(200),LH(200),LR(200),LX(200),
     4 JK(200),IK(200),KEY(200),IPL(200),CAN(200),CON(200),LL(200),
     5 IA(6,200),LS(200),X(4,200)
      COMMON /BLK2/ IS(2,3,24),TS(3,24),NSYM,ICENT,LAT,IAPX,
     1 CELL(6),T(3,3),DXMAX(3),TL(3,4),FS(3,3,24),MLAT,NEQV,MCON,INT,
     2 FOM,KFRAG(20),NFRAG,LFRAG(20),MFRAG,NOFRAG,NN,MFRN,DUMY(20),
     3 KMATCH(20),ITYPE,NOINT,MNPPF
      COMMON /BLK4/ IONP(6,200),NAFRG(5),ICH(12),IB(6),LINX(60),NSEQ
      COMMON /BLK5/ NOJOIN,NOPROJ,THOPI,FACTOR,RMAX,RMIN,DOUT,
     1 MH(3),ILEVEL,IPNT,XLIM(3),NOSET,NPIC,NP(3),NEX,ANGMIN,ANGMAX,
     2 DMIN,DMAX,DMUT,DM,DFRG,ALN,NPATT,IVDU,NAT,NPC,SCALE,IPASS
      COMMON/BLK6/ INBOND(1000),IFMT(120),RUBISH(30)
      DIMENSION      B(3,3),V(3,3),IND(3),SUM(3),XMAX(3),XMIN(3)
      EQUIVALENCE (KUSE31,KUSE(31))
C===> DO NOT OUTPUT PICTURES WITH LESS THAN 4 PEAKS
      NUM = KFRAG(NOFRAG)
      IF (NUM .LT. 4) RETURN
      ANUM = NUM
C===> SET UP ORTHOGONALISATION MATRIX
      COSN=(COS(DTOR*CELL(5))-COS(DTOR*CELL(6))*COS(DTOR*CELL(4)))/(SIN(
     1DTOR*CELL(6))*SIN(DTOR*CELL(4)))
      SINN=SQRT(1.0-COSN**2)
      A11=CELL(1)*SINN*SIN(DTOR*CELL(6))
      A21=CELL(1)*COS(DTOR*CELL(6))
      A22=CELL(2)
      A23=CELL(3)*COS(DTOR*CELL(4))
      A31=CELL(1)*COSN*SIN(DTOR*CELL(6))
      A33 = CELL(3) * SIN(DTOR*CELL(4))
C===> INITIALISE ARRAYS
      DO 1020 J=1,3
      SUM(J)=0.0
      XMAX(J)=-10000.0
      XMIN(J)=10000.0
      DO 1000 I=1,3
 1000 B(I,J)=0.0
 1020 CONTINUE
C===> CONVERT PEAK COORDINATES TO AN ORTHOGONAL SYSTEM AND SET UP
C===> THE MATRIX FOR THE PLANE FITTING CALCULATIONS
      DO 1080 I=1,NAT
      IF(IFRAG(I).NE.NOFRAG)GO TO 1080
      XA(1,I)=X(1,I)*A11
      XA(2,I)=X(1,I)*A21+X(2,I)*A22+X(3,I)*A23
      XA(3,I)=X(1,I)*A31+X(3,I)*A33
      DO 1060 J=1,3
      SUM(J)=SUM(J)+XA(J,I)
      DO 1040 K=1,3
 1040 B(J,K)=B(J,K)+XA(J,I)*XA(K,I)
 1060 CONTINUE
 1080 CONTINUE
      DO 1120 J=1,3
      DO 1100 K=1,3
 1100 B(J,K)=B(J,K)-SUM(J)*SUM(K)/ANUM
 1120 CONTINUE
C===> CALCULATE EIGENVECTORS AND EIGENVALUES OF MATRIX
      CALL EIGEN(B,V,IND)
C===> CONVERT COORDINATES TO AXES GIVEN BY EIGENVECTORS
      K=0
      DO 1180 I=1,NAT
```

```
      IF(IFRAG(I).NE.NOFRAG) GOTO 1180
      K=K+1
      XB(4,K)=I
      DO 1140 J=1,3
      L=IND(J)
      XB(J,K)=XA(1,I)*V(1,L)+XA(2,I)*V(2,L)+XA(3,I)*V(3,L)
      XMAX(J)=AMAX1(XMAX(J),XB(J,K))
 1140 XMIN(J)=AMIN1(XMIN(J),XB(J,K))
 1180 CONTINUE
      NNN = 0
      N1 = 2
      N2 = 1
      IF (115.0/(XMAX(1)-XMIN(1)) .GE. 2.0/0.254) GO TO 1190
      N1 = 1
      N2 = 2
C===> CALCULATE SCALE FACTOR TO FILL PAGE OR GIVE 2.5 CMS/A
 1190 AMAX = AMAX1(XMAX(N2)-XMIN(N2), XMAX(N2+1)-XMIN(N2+1))
      SCALE = AMIN1(115.0/AMAX, 2.5/0.254)
      SCL = 0.254 * SCALE
C===> SORT PEAKS IN ORDER OF PLOTTING
 1200 CALL SORT10(XB,KUSE31,NUM,N1)
      RETURN
      END
C-------------------------------------------------------------------------------
      SUBROUTINE OUTERP(NELSE,ISTART,FOMF,NUM,NSYMOP)
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),ICLCK,ICNT
      COMMON /BLK1/ NSTORE(25100),XA(3,200),XB(4,200),ISYM(100),
     1 IFRAG(200),ICON(200),IBOND(2000),JFRAG(200),
     2 JCON(200),JBOND(2000),IUSE(200),IONA(6,200),MO(200),
     3 MU(200),NO(200),NU(200),KEYST(200),LW(200),LR(200),LX(200),
     4 JK(200),IK(200),KEY(200),IPL(200),CAN(200),CON(200),LL(200),
     5 IA(6,200),LS(200),X(4,200)
      COMMON /BLK2/ IS(2,3,24),TS(3,24),NSYM,ICENT,LAT,IAPX,
     1 CELL(6),T(3,3),DXMAX(3),TL(3,4),FS(3,3,24),MLAT,NEQV,MCON,INT,
     2 FOM,KFRAG(20),NFRAG,LFRAG(20),MFRAG,NOFRAG,NN,MFRN,IFMT(20),
     3 KMATCH(20),ITYPE,NOINT,MNPPF
      COMMON /BLK4/ IONP(6,200),NAFRG(5),JUNK(12),IB(6),LINX(60),NSEQ
      COMMON /BLK5/ NOJOIN,NOPROJ,THOPI,FACTOR,RMAX,RMIN,DOUT,
     1 MH(3),ILEVEL,IPNT,XLIM(3),NOSET,NPIC,NP(3),NEX,ANGMIN,ANGMAX,
     2 DMIN,DMAX,DMUT,DM,DFRG,ALN,NPATT,IVDU,NAT,NPC,SCALE,IPASS
      ISK=ISPCH(3)
      ICY=IH(43)
C===> DECIDE IF SET IS WORTH PRINTING
      ISTART=-ISTART
      FOMM=0.0
      DO 200 I=1,NAT
      J=JFRAG(I)
      IF(J.EQ.0.OR.LFRAG(J).LT.4) GOTO 200
      FOMM=FOMM+ALOG10(FLOAT(LFRAG(J)))*X(4,I)*FLOAT(INT)
  200 CONTINUE
      IDR=0
      IF(FOMM.GT.FOMF) IDR=1
      IF(FOM.GE.FOMM.AND.FOM.GE.FOMF) RETURN
      FOM=AMAX1(FOMM,FOMF)
      DO 240 I=1,NAT
      IF(IFRAG(I).NE.NOFRAG) GO TO 240
      IUSE(I)=IABS(IUSE(I))
```

```
      IF(JCON(I).LE.0) GO TO 240
      J=JFRAG(I)
      IF(LFRAG(J).LT.4) GO TO 240
      IF(IDR.EQ.0) IUSE(I)=-1
      IF(IDR.EQ.1.AND.KEYST(I).GT.0) IUSE(I)=-1
  240 CONTINUE
      ISTART=-ISTART
      WRITE(NOUT,245)
  245 FORMAT(///)
C===> OUTPUT FRAGMENT COMPARISON RESULTS
      IF(MFRN.EQ.0) GO TO 290
      DO 270 K=1,MFRAG
      IF(LFRAG(K).LT.5) GO TO 270
      NID=0
      DO 250 I=1,NAT
      IF(JFRAG(I).NE.K.OR.KEYST(I).LE.0) GO TO 250
      NID=NID+1
      IONP(1,NID)=I
      IONP(2,NID)=KEYST(I)
  250 CONTINUE
      IF(NID.GT.0) WRITE(NOUT,260) NID,K,KMATCH(K),((IONP(I,J),I=1,2),
     1 J=1,NID)
  260 FORMAT(/1H0,24X,I5,3X,17HPeaks of fragment,I5,3X,
     1 34Hmatch with atoms of input molecule,I5/1H ,24X,74(1H*)/
     2 1H ,35X,24HPeak-atom correspondence//(10(I6,2H -,I4)))
  270 CONTINUE
C===> CALL GRAPHICS ROUTINES
  290 CALL SCREEN(NELSE,XB,NUM,ICON,JBOND,NOFRAG,JCON,X,CELL,NSYMOP)
      RETURN
      END
C-------------------------------------------------------------------------------
      SUBROUTINE BONGLE
C===> CALCULATE AND OUTPUT PEAK POSITIONS, BOND LENGTHS, AND ANGLES.
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IDUM,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),ICLCK,ICNT
      COMMON /BLK1/ NSTORE(25100),XA(3,200),XB(4,200),ISYM(100),
     1 IFRAG(200),ICON(200),IBOND(2000),JFRAG(200),
     2 JCON(200),JBOND(2000),IUSE(200),IONA(6,200),MO(200),
     3 MU(200),NO(200),NU(200),KEYST(200),LW(200),LR(200),LX(200),
     4 JK(200),IK(200),KEY(200),IPL(200),CAN(200),CON(200),LL(200),
     5 IA(6,200),LS(200),X(4,200)
      COMMON /BLK2/ IS(2,3,24),TS(3,24),NSYM,ICENT,LAT,IAPX,
     1 CELL(6),T(3,3),DXMAX(3),TL(3,4),FS(3,3,24),MLAT,NEQV,MCON,IWT,
     2 FOM,KFRAG(20),NFRAG,LFRAG(20),MFRAG,NOFRAG,NN,MFRN,DUMY(20),
     3 KMATCH(20),ITYPE,NOINT,MNPPF
      COMMON /BLK4/ IONP(6,200),NAFRG(5),ICH(12),BB(6),LINX(60),NSEQ
      COMMON /BLK5/ NOJOIN,NOPROJ,TWOPI,FACTOR,RMAX,RMIN,DOUT,
     1 MH(3),ILEVEL,IPNT,XLIM(3),NOSET,NPIC,NP(3),NEX,ANGMIN,ANGMAX,
     2 DMIN,DMAX,DMUT,DM,DFRG,ALN,NPATT,IVDU,NAT,NPC,SCALE,IPASS
      COMMON/BLK6/ INBOND(1000),IFMT(120),JBND(30)
      DIMENSION IB(3),XS(3),X1(3),ENIL1(4,4),LINE1(4,6),ENIL(30)
C===> OUTPUT PEAK HEIGHTS, PEAK COORDINATES AND CLUSTER NUMBERS
      WRITE (NOUT,900)
  900 FORMAT('Interpreted peak list'/1H ,21(1H-)///
     1  1X,'   Peak        Height',13X,1HX,9X,1HY,9X,1HZ,9X,
     2 ' Cluster number')
      DO 940 I=1,NAT
      ICON(I+1)=0
```

```
      IFRAG(I)=IABS(IFRAG(I))
      N=MIN0(IFRAG(I)+1,15)
      K=X(4,I)+0.5
      WRITE(NOUT,1000)I,K,(X(J,I),J=1,3),IFRAG(I)
 1000 FORMAT(I7,I13,7X,3F10.4,I15)
  940 CONTINUE
      N = MIN0(NFRAG, 14)
      IW=0
      WRITE(NOUT,950)
  950 FORMAT(//'Cluster number     Peaks'/)
      DO 960 I=1,N
      WRITE(NOUT,970)I,KFRAG(I)
  970 FORMAT(I8,I15)
  960 CONTINUE
C===> MOVE PRINTER TO TOP OF NEXT PAGE
      MP=MOD((NAT+N+8),60)
      DO 200 K=1,MP
      WRITE(NOUT,100)
  100 FORMAT(/)
  200 CONTINUE
      WRITE(NOUT,1110)
 1110 FORMAT('Table of interpeak distances'/28(1H-)//2X'From',3X,7HTo Dist)
      ICON(1)=0
      DO 1115 I=1,MCON
      II=IBOND(I)/4194304
      ICON(II+1)=I
 1115 CONTINUE
      DO 1260 I=1,NAT
      IF(ICON(I+1).EQ.0) ICON(I+1)=ICON(I)
      N=0
      JUMP=0
      DO 1220 J=1,NAT
      IF(J.EQ.I) GO TO 1170
      DO 1120 K=1,3
      X1(K)=X(K,I)-X(K,J)
      IF(ABS(X1(K)).GT.DXMAX(K)) GO TO 1170
 1120 CONTINUE
      DIST=QUAD(X1,X1,T)
      IF(DIST.GT.DMUT) GO TO 1170
      N=N+1
      JBND(N)=J
      ENIL(N)=SQRT(DIST)
 1170 IF(N.LT.10.AND.J.NE.NAT) GO TO 1220
      IF(JUMP.EQ.0.AND.N.GT.0) WRITE(NOUT,1190) I,(JBND(L),ENIL(L),L=1,N
     1)
 1190 FORMAT(1H ,I5,10(I6,F5.2))
      IF(JUMP.NE.0.AND.N.GT.0) WRITE(NOUT,1210) (JBND(L),ENIL(L),L=1,N)
 1210 FORMAT(1H ,5X,10(I6,F5.2))
      N=0
      JUMP=1
 1220 CONTINUE
 1260 CONTINUE
      DMX=SQRT(DMAX)
      WRITE(NOUT,1300) DMX
 1300 FORMAT(/////81HTable of bond angles (in degrees) among peaks in
     1 above table closer together than,F5.2,1HA/87(1H-)/)
      N=0
      DO 1500 I=1,NAT
      KS=ICON(I)+1
      KL=ICON(I+1)
      IF(KS.GE.KL) GO TO 1500
```

```
      KLM1=KL-1
      DO 1490 J=KS,KLM1
      V1=0.001*FLOAT(MOD(IBOND(J),8192))
      IF(V1.GT.DMX.OR.V1.LT.0.2) GO TO 1490
      JJ=IBOND(J)/8192
      JJ=MOD(JJ,512)
      JP1=J+1
      DO 1480 K=JP1,KL
      V2=0.001*FLOAT(MOD(IBOND(K),8192))
      IF(V2.GT.DMX.OR.V2.LT.0.2) GO TO 1480
      KK=IBOND(K)/8192
      KK=MOD(KK,512)
      DO 1430 L=1,3
1430  X1(L)=X(L,KK)-X(L,JJ)
      DIST=QUAD(X1,X1,T)
      IF(DIST.LT.0.2) GO TO 1480
      COSA=0.5*(V1*V1+V2*V2-DIST)/(V1*V2)
      SINA=SQRT(AMAX1(1.0-COSA*COSA,0.001))
      N=N+1
      LINE1(1,N)=JJ
      LINE1(2,N)=I
      LINE1(3,N)=KK
      LINE1(4,N)=AMOD(ATAN2(SINA,COSA)*RTOD+180.0,180.0)+0.5
      IF(N.LT.6) GO TO 1480
      WRITE(NOUT,1470) ((LINE1(L,N),L=1,4),N=1,6)
1470  FORMAT(6(I6,2H -,I3,2H -,I3,I4))
      N=0
1480  CONTINUE
1490  CONTINUE
1500  CONTINUE
      IF(N.NE.0) WRITE(NOUT,1470) ((LINE1(L,J),L=1,4),J=1,N)
      N = -1
      NATM1=NAT-1
      DO 1800 I=1,NATM1
      DO 1780 J=I,NAT
      DO 1740 K=1,NSYM
      IF(I.EQ.J.AND.K.EQ.1) GO TO 1740
      IF(JSYMM(I,J,K,IB,XS,X1).NE.0) GO TO 1740
      IBJ=100*IB(1)+10*IB(2)+IB(3)
      IF(IBJ.EQ.555.AND.K.EQ.1) GO TO 1740
      DIST=QUAD(X1,X1,T)
      IF(DIST.GT.DMUT) GO TO 1740
      IF(DIST.LT.0.2.AND.I.EQ.J) ICON(I)=10000
      ISIM=MOD(K,NEQV)
      IF(ISIM.EQ.0) ISIM=NEQV
      IP=(K-1)/NEQV
      ILAT=MOD(IP,MLAT)+1
      IF(IP.GE.MLAT) ISIM=-ISIM
      N=N+1
      IF (N.EQ.0) WRITE(NOUT,1550)
1550  FORMAT(//// 54H Distances between atoms in different asymmetric un
     1its/1H ,53(1H-)//3(5X,35HFrom   To Symm  x     y     z     Dist))
      IF (N.EQ.0) N = 1
      LINE1(N,1)=I
      LINE1(N,2)=J
      LINE1(N,3)=ISIM
      ENIL1(N,1)=FLOAT(IB(1))+TL(1,ILAT)-5.0
      ENIL1(N,2)=FLOAT(IB(2))+TL(2,ILAT)-5.0
      ENIL1(N,3)=FLOAT(IB(3))+TL(3,ILAT)-5.0
      ENIL1(N,4)=SQRT(DIST)
      IF(N.LT.3) GO TO 1740
```

```
      WRITE(NOUT,1710) ((LINE1(N,L),L=1,3),(ENIL1(N,L),L=1,4),N=1,3)
 1710 FORMAT(3(I9,I5,1H(,I3,3F5.2,1H),F6.2))
      N=0
 1740 CONTINUE
 1780 CONTINUE
 1800 CONTINUE
      IF(N.GT.0) WRITE(NOUT,1710) ((LINE1(J,L),L=1,3),
     1  (ENIL1(J,L),L=1,4),J=1,N)
      DO 1850 I=1,NAT
      IF(ICON(I).GE.999) WRITE(NOUT,1840) I
 1840 FORMAT(/'Peak',I4,'  lies on a symmetry element'/)
 1850 CONTINUE
      IF(NN.LE.0) GOTO 1900
      IF(NN.GT.1) CALL ISORT(ISYM,NN)
      ISYM(NN+1)=0
      JJ=0
      K=0
      DO 1890 I=1,NN
      II=ISYM(I)/250000
      JJ=II
      K=K+2
      LW(K-1)=MOD(ISYM(I),250000)/500
      LW(K)=MOD(ISYM(I),500)
      IF(K.LT.12.AND.II.EQ.ISYM(I+1)/250000) GO TO 1890
      WRITE(NOUT,1870)II,(LW(J),J=1,K)
 1870 FORMAT(/'  Cluster',I3,' joins to itself through the peak pair(s)',5X,
     16(I7,1H,I3)/)
      K=0
 1890 CONTINUE
 1900 RETURN
      END
C-------------------------------------------------------------------------------
      SUBROUTINE CENTRE(J,L)
C===> RETRIEVE CENTRING TRANSLATIONS
      COMMON /BLK2/ IS(2,3,24),TS(3,24),NSYM,ICENT,LAT,IAPX,
     1 CELL(6),T(3,3),DXMAX(3),TL(3,4),FS(3,3,24),MLAT,NEQV,MCON,INT,
     2 FOM,KFRAG(20),NFRAG,LFRAG(20),MFRAG,NOFRAG,NN,MFRN,IFMT(20),
     3 KMATCH(20),ITYPE,NOINT,MNPPF
      DIMENSION TST(21)
      DO 10 JJ=1,21
   10 TST(JJ)=0.0
      DO 15 JJ=5,7
      TST(JJ)=0.5
      TST(JJ+4)=0.5
      TST(JJ+8)=0.5
   15 TST(JJ+12)=2.0/3.0
      TST(16)=1.0/3.0
      TST(20)=1.0/3.0
      TST(21)=1.0/3.0
      MLAT=J
      M=3*L-2
      TL(1,J)=TST(M)
      TL(2,J)=TST(M+1)
      TL(3,J)=TST(M+2)
      RETURN
      END
C-------------------------------------------------------------------------------
      SUBROUTINE EIGEN(B,V,IND)
C===> COMPUTE EIGENVECTORS AND EIGENVALUES OF REAL SYMMETRIC MATRIX
      DIMENSION B(3,3),V(3,3),IND(3)
      DO 1020 I=1,3
```

```
      DO 1000 J=1,3
 1000 V(I,J)=0.0
      V(I,I)=1.0
 1020 CONTINUE
 1040 KNT=0
      IND1=1
      IND3=1
      DO 1500 I=1,2
      IP1=I+1
      DO 1460 J=IP1,3
      IF (ABS(B(I,J)) .LT. 0.000001 * B(IND1,IND1)) KNT = KNT + 1
      BIJ=B(I,I)-B(J,J)
      IF(ABS(B(I,J)).LT.ABS(BIJ))GO TO 1100
      T=SIGN(1.0,B(I,J)*BIJ)
      GO TO 1180
 1100 T=B(I,J)/BIJ
 1180 G=T/(2.0+2.0*T*T)
      SN=2.0*G/(1.0+G*G)
      CS=1.0-G*SN
      DO 1200 K=1,3
      BIK=B(I,K)
      B(I,K)=CS*B(I,K)+SN*B(J,K)
 1200 B(J,K)=CS*B(J,K)-SN*BIK
      DO 1220 K=1,3
      BKI=B(K,I)
      B(K,I)=CS*B(K,I)+SN*B(K,J)
      B(K,J)=CS*B(K,J)-SN*BKI
      VKI=V(K,I)
      V(K,I)=CS*V(K,I)+SN*V(K,J)
 1220 V(K,J)=CS*V(K,J)-SN*VKI
 1460 CONTINUE
      IF(B(IP1,IP1).GT.B(IND1,IND1))IND1=IP1
      IF(B(IND3,IND3).GT.B(IP1,IP1))IND3=IP1
 1500 CONTINUE
      IF(KNT.LT.3)GO TO 1040
      IND(1)=IND1
      IND(2)=1
      IND(3)=IND3
      IF(IND1.EQ.IND(2).OR.IND3.EQ.IND(2))IND(2)=2
      IF(IND1.EQ.IND(2).OR.IND3.EQ.IND(2))IND(2)=3
C===> CHECK FOR RIGHT-HANDED SET OF VECTORS
      DET = V(1,1)*(V(2,2)*V(3,3)-V(2,3)*V(3,2))+V(1,2)*(V(2,3)*V(3,1)
     1  -V(2,1)*V(3,3))+V(1,3)*(V(2,1)*V(3,2)-V(2,2)*V(3,1))
      IF (DET .GT. 0.0) RETURN
      DO 1520 I=1,3
 1520 V(I,3) = -V(I,3)
      RETURN
      END
C-------------------------------------------------------------------------------
      SUBROUTINE SORT10(X,KUSE31,NAT,N)
C===> SORT PEAKS IN ORDER OF PEAK HEIGHT OR IN ORDER OF PLOTTING
      DIMENSION X(4,KUSE31),T(4)
      INT=2
 1000 INT=INT+INT
      IF(INT.LT.NAT)GO TO 1000
      INT=MIN0(NAT,(3*INT)/4-1)
 1020 INT=INT/2
      IFIN=NAT-INT
      DO 1200 II=1,IFIN
      I=II
      J=I+INT
```

```
      IF(X(N,I).GE.X(N,J))GO TO 1200
      DO 1060 K=1,4
 1060 T(K)=X(K,J)
 1080 DO 1100 K=1,4
 1100 X(K,J)=X(K,I)
      J=I
      I=I-INT
      IF(I)1140,1140,1120
 1120 IF(X(N,I).LT.T(N))GO TO 1080
 1140 DO 1160 K=1,4
 1160 X(K,J)=T(K)
 1200 CONTINUE
      IF(INT.NE.1)GO TO 1020
      RETURN
      END
C-------------------------------------------------------------------------------
      SUBROUTINE ISORT(N,M)
C===> GENERAL INTEGER SORT ROUTINE
      DIMENSION N(M)
      INT=2
 1000 INT=INT+INT
      IF(INT.LT.M) GO TO 1000
      INT=MIN0(M,(3*INT)/4-1)
 1020 INT=INT/2
      IFIN=M-INT
      DO 1200 II=1,IFIN
      I=II
      J=I+INT
      IF(N(I).LE.N(J)) GO TO 1200
      IT=N(J)
 1080 N(J)=N(I)
      J=I
      I=I-INT
      IF(I.GT.0.AND.N(I).GT.IT) GO TO 1080
      N(J)=IT
 1200 CONTINUE
      IF(INT.NE.1) GO TO 1020
      RETURN
      END
C-------------------------------------------------------------------------------
      SUBROUTINE ELIM(II,NNN,JCN,JBND,ICN,KUSE31)
C===> ELIMINATE THE IITH PEAK AND UPDATE THE BONDING ARRAY
      DIMENSION JCN(KUSE31), ICN(KUSE31), JBND(2000)
      JCN(II)=0
      NNN=0
      KS=ICN(II)+1
      KL=ICN(II+1)
      IF(KS.GT.KL) RETURN
      DO 20 I=KS,KL
      J=JBND(I)/8192
      JCN(J)=JCN(J)-1
      JBND(I)=0
      KB=ICN(J)+1
      KE=ICN(J+1)
      IF(KB.GT.KE) GO TO 20
      DO 10 K=KB,KE
      L=JBND(K)/8192
      IF(L.EQ.II) JBND(K)=0
   10 CONTINUE
   20 CONTINUE
      RETURN
```

```
      END
      FUNCTION QUAD(X1,X2,T)
C===> FUNCTION USED IN THE CALCULATION OF DISTANCES AND ANGLES
      DIMENSION X1(3),X2(3),T(3,3)
      QUAD=0.0
      DO 1000 I=1,3
 1000 QUAD = QUAD+X2(I)*(T(I,1)*X1(1)+T(I,2)*X1(2)+T(I,3)*X1(3))
      RETURN
      END
      FUNCTION JSYMM(I,J,K,IB,XS,X1)
C===> PERFORM THE KTH SYMMETRY OPERATION ON THE JTH PEAK
C===> AND MOVE IT AS CLOSE AS POSSIBLE TO PEAK I BY LATTICE TRANSLATIONS
      COMMON /BLK1/ NSTORE(25100),XA(3,200),XB(4,200),ISYM(100),
     1 IFRAG(200),ICON(200),IBOND(2000),JFRAG(200),
     2 JCON(200),JBOND(2000),IUSE(200),IONA(6,200),MO(200),
     3 MU(200),NO(200),NU(200),KEYST(200),LW(200),LR(200),LX(200),
     4 JK(200),IK(200),KEY(200),IPL(200),CAN(200),CON(200),LL(200),
     5 IA(6,200),LS(200),X(4,200)
      COMMON /BLK2/ IS(2,3,24),TS(3,24),NSYM,ICENT,LAT,IAPX,
     1 CELL(6),T(3,3),DXMAX(3),TL(3,4),FS(3,3,24),MLAT,NEQV,MCON,INT,
     2 FOM,KFRAG(20),NFRAG,LFRAG(20),MFRAG,NOFRAG,NN,MFRN,IFMT(20)
     3 ,JUNK(23)
      DIMENSION IB(3),XS(3),X1(3)
      JSYMM = 0
      IPRIN=0
      CALL OPER(K,XS,X(1,J),X(2,J),X(3,J),IPRIN)
      DO 1080 L=1,3
      IB(L)=5
 1060 X1(L)=X(L,I)-XS(L)
      IF(ABS(X1(L)).LE.0.5)GO TO 1070
      XS(L)=XS(L)+SIGN(1.0,X1(L))
      IB(L)=IB(L)+ISIGN(1,IFIX(2.0*X1(L)))
      GO TO 1060
 1070 IF (ABS(X1(L)) .GT. DXMAX(L)) GO TO 1100
 1080 CONTINUE
      RETURN
 1100 JSYMM = 1
      RETURN
      END
C-------------------------------------------------------------------------------
      SUBROUTINE OPER(J,XN,X,Y,Z,IPRIN)
C===> APPLY J'TH SYMMETRY ELEMENT TO X,Y,Z & PUT RESULT IN XN
      DIMENSION XN(3)
      COMMON /BLK2/ IS(2,3,24),TS( 72 ),NSYM,ICENT,LAT,IAPX,
     1 CELL(6),T(3,3),DXMAX(3),TL( 12),FS(  216 ),MLAT,NEQV,MCON,INT,
     2 FOM,KFRAG(20),NFRAG,LFRAG(20),MFRAG,NOFRAG,NN,MFRN,IFMT(20),
     3 KMATCH(20),ITYPE,NOINT,MNPPF
      ISYM=MOD(J,NEQV)
      IF(ISYM.EQ.0) ISYM=NEQV
      IP=(J-1)/NEQV
      ILAT=3*MOD(IP,MLAT)
      JT=3*(ISYM-1)
      JJ=9*ISYM-8
      DO 20 L=1,3
      IND1=JT+L
      IND2=ILAT+L
      XN(L)=X*FS(JJ)+Y*FS(JJ+1)+Z*FS(JJ+2)
      IF(IP.GE.MLAT) XN(L)=-XN(L)
      XN(L)=XN(L)+TS(IND1)+TL(IND2)
   20 JJ=JJ+3
      RETURN
```

```
      END
C-----------------------------------------------------------------------------
      SUBROUTINE TELL( N, NR, M, MR, MERIT, KOMP, INA, INP, LLH)
C===> POINT BY POINT FRAGMENT COMPARISON
      DIMENSION INA( 6, 200), INP( 6, 200), LLH( 200)
      COMMON/IO/NIN, NOUT, NTAPEA, NTAPEB, NTAPEC, NTAPED, NTAPEE,
     1 NTAPEF, NTAPEG, IH( 155), A( 200), ICH1( 10), ICH2( 10), ITLE( 68), IR( 68),
     2 ICALL, NRC, NCH, NREAD, IEND, NREF, PI, DTOR, MAXH, ISPFLG, ZERO, LK( 32),
     3 IDEF, ICHK, ISYMP, LEVEL, IFLOW( 25), KUSE( 40), INIT, IRDY, RTOD, NULL,
     4 IDIF, IVDIF, KARLE, IHVY, IGPFL, NDIFF, ISPCH( 30), ICLCK, ICNT
      COMMON /BLK1/ NSTORE( 25100), XA( 3, 200), XB( 4, 200), ISYM( 100),
     1 IFRAG( 200), ICON( 200), IBOND( 2000), JFRAG( 200),
     2 JCON( 200), JBOND( 2000), IUSE( 200), IONA( 6, 200), MO( 200),
     3 MU( 200), NO( 200), NU( 200), KEYST( 200), LW( 200), LR( 200), LX( 200),
     4 JK( 200), IK( 200), KEY( 200), IPL( 200), KAN( 200), CON( 200), LL( 200),
     5 IA( 6, 200), LS( 200), X( 4, 200)
      COMMON /BLK2/ IS( 2, 3, 24), TS( 3, 24), NSYM, ICENT, LAT, IAPX,
     1 CELL( 6), T( 3, 3), DXMAX( 3), TL( 3, 4), FS( 3, 3, 24), MLAT, NEQV, MCON, IHT,
     2 FOM, KFRAG( 20), NFRAG, LFRAG( 20), MFRAG, NOFRAG, NN, MFRN, IFMT( 20),
     3 KMATCH( 20), ITYPE, NOINT, MNPPF
      COMMON /BLK3/ JNP( 200), JNA( 200), LMP( 200), LMA( 200)
      COMMON /BLK4/ IONP( 6, 200), NAFRG( 5), ICH( 12), IB( 6), LINX( 60), NSEQ
      COMMON /BLK5/ NOJOIN, NOPROJ, THOPI, FACTOR, RMAX, RMIN, DOUT,
     1 MH( 3), ILEVEL, IPNT, XLIM( 3), NOSET, NPIC, NP( 3), NEX, ANGMIN, ANGMAX,
     2 DMIN, DMAX, DMUT, DM, DFRG, ALN, NPATT, IVDU, NAT, NPC, SCALE, IPASS
      ICOUNT=0
      NAX=MAX0( N, M)
      DO 20 I=1, NAX
      KEY( I) =0
      IPL( I) =0
   20 MO( I) =0
      KY=MAX0( NR, 6)
      LY=MAX0( MR, 6)
      L=1
      LMP( 1) =0
      LMA( 1) =0
      DO 50 I=1, NAX
      LMP( I+1) =LMP( I)
      LMA( I+1) =LMA( I)
      DO 40 J=1, 6
      IF( INA( J, I). GT. I. OR. INA( J, I). EQ. 0) GO TO 30
      K=LMA( I+1) +1
      JNA( K) =INA( J, I)
      LMA( I+1) =K
   30 IF( INP( J, I). EQ. 0) GO TO 40
      K=LMP( I+1) +1
      JNP( K) =INP( J, I)
      LMP( I+1) =K
   40 CONTINUE
   50 CONTINUE
      NF=0
      LR( M+1) =0
      DO 90 I=1, M
      MM=M+1-I
   90 LR( MM) =LR( MM+1) +LLH( MM)
  100 IK( L) =LMA( L)
      IF( L. LE. M) GO TO 200
  150 L=L-1
      IF( L. EQ. 0) GO TO 500
      IF( KEY( L). EQ. 0) GO TO 150
      KX=KEY( L)
```

```
      NF=NF-LLH(L)
      IPL(KX)=0
      IF(MERIT.GT.LR(1)) GO TO 150
  200 IF(NF.NE.0) GO TO 230
      K=KEY(L)+1
      IF(K.LE.N) GO TO 350
  220 KEY(L)=0
      L=L+1
      IF(NF+LR(L).GE.MERIT) GO TO 100
      GO TO 150
  230 IF(IK(L).NE.LMA(L).AND.JK(L).NE.KAN(L)) GO TO 260
  240 IK(L)=IK(L)+1
      IF(IK(L).GT.LMA(L+1)) GO TO 220
      K=IK(L)
      K=JNA(K)
      IF(KEY(K).EQ.0) GO TO 240
      K=KEY(K)
      JK(L)=LMP(K)
      KAN(L)=LMP(K+1)
  260 JK(L)=JK(L)+1
      K=JK(L)
      K=JNP(K)
      IF(IPL(K).NE.0) GO TO 230
      LP=LMP(K+1)
      LA=LMA(L+1)
      IF(L.GT.LY) GO TO 290
      LJ=LMA(L)
  270 LJ=LJ+1
      IF(LJ.GT.LA) GO TO 290
      J=JNA(LJ)
      IF(KEY(J).EQ.0) GO TO 270
      LB=LMP(K)+1
  280 IF(LB.GT.LP) GO TO 230
      IF(JNP(LB).EQ.KEY(J)) GO TO 270
      LB=LB+1
      GO TO 280
  290 IF(K.GT.KY) GO TO 350
      LJ=LMP(K)
  300 LJ=LJ+1
      IF(LJ.GT.LP) GO TO 350
      J=JNP(LJ)
      IF(IPL(J).EQ.0) GO TO 300
      LB=LMA(L)+1
  310 IF(LB.GT.LA) GO TO 230
      IF(JNA(LB).EQ.IPL(J)) GO TO 300
      LB=LB+1
      GO TO 310
  350 KEY(L)=K
      IPL(K)=L
      NF=NF+LLH(L)
      ICOUNT=ICOUNT+1
      IF(NF.LT.MERIT) GO TO 400
      MERIT=NF+1
      DO 380 I=1,NAX
      IF(KOMP.EQ.1) MO(I)=KEY(I)
      IF(KOMP.EQ.2) MO(I)=IPL(I)
  380 CONTINUE
  400 L=L+1
      IF(ICOUNT.LT.100000) GO TO 100
      WRITE(NOUT,450)
  450 FORMAT(1H0,30X,57HWarning - fragment comparison too complicated fo
```

```
     1r program)
  500 RETURN
      END
C-------------------------------------------------------------------------------
      SUBROUTINE CANON(M,MR)
C===> CANONICAL DESCRIPTION & SORT FOR FRAGMENT COMPARISON
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),Z(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),ICLCK,ICNT
      COMMON /BLK1/ NSTORE(25100),XA(3,200),XB(4,200),ISYM(100),
     1 IFRAG(200),ICON(200),IBOND(2000),JFRAG(200),
     2 JCON(200),JBOND(2000),IUSE(200),IONA(6,200),MO(200),
     3 MU(200),NO(200),NU(200),KEYST(200),LW(200),LR(200),LX(200),
     4 JK(200),IK(200),KEY(200),IPL(200),CAN(200),CON(200),LL(200),
     5 IA(6,200),LS(200),X(4,200)
      COMMON /BLK2/ IS(2,3,24),TS(3,24),NSYM,ICENT,LAT,IAPX,
     1 CELL(6),T(3,3),DXMAX(3),TL(3,4),FS(3,3,24),MLAT,NEQV,MCON,INT,
     2 FOM,KFRAG(20),NFRAG,LFRAG(20),MFRAG,NOFRAG,NN,MFRN,IFMT(20),
     3 KMATCH(20),ITYPE,NOINT,MNPPF
      COMMON /BLK4/ IONP(6,200),NAFRG(5),ICH(12),IB(6),LINX(60),NSEQ
      COMMON /BLK5/ NOJOIN,NOPROJ,THOPI,FACTOR,RMAX,RMIN,DOUT,
     1 MH(3),ILEVEL,IPNT,XLIM(3),NOSET,NPIC,NP(3),NEX,ANGMIN,ANGMAX,
     2 DMIN,DMAX,DMUT,DM,DFRG,ALN,NPATT,IVDU,NAT,NPC,SCALE,IPASS
      DO 20 I=1,M
      CON(I)=1.0
      CAN(I)=0.0
      LL(I)=0
      NU(I)=0
      DO 10 J=1,6
      IF(IONP(J,I).GT.0) LL(I)=LL(I)+1
   10 CONTINUE
   20 CONTINUE
      AM=0.0
      DO 110 I=1,M
      DO 80 J=1,M
      DO 70 K=1,6
      L=IONP(K,J)
      IF(L.GT.0) CAN(J)=CAN(J)+CON(L)
   70 CONTINUE
   80 CONTINUE
      DO 90 J=1,M
      CON(J)=CAN(J)
      CAN(J)=0.0
      IF(I.EQ.M) AM=AMAX1(AM,CON(J))
   90 CONTINUE
  110 CONTINUE
      DO 130 I=1,M
      A=CON(I)/AM
      A=10000.0*A*A
      LX(I)=A+20000.5
      IF(LX(I).EQ.20000) LX(I)=20001
  130 CONTINUE
      DO 255 IEN=1,10
      DO 250 I=1,M
      IF(LL(I).NE.1) GO TO 250
      DO 240 K=1,M
      IF(LL(K).LE.1) GO TO 240
      DO 230 J=1,6
      IF(IONP(J,K).EQ.I) LL(K)=LL(K)-1
```

```
  230 CONTINUE
  240 CONTINUE
      LL(I)=0
      LX(I)=LX(I)-20000
  250 CONTINUE
  255 CONTINUE
      DO 300 I=1,M
      LM=0
      DO 280 J=1,M
      IF(LX(J).LE.LM) GO TO 280
      IF(I.EQ.1) GO TO 270
      DO 260 K=1,6
      L=IONP(K,J)
      IF(L.EQ.0) GO TO 280
      IF(NU(L).NE.0) GO TO 270
  260 CONTINUE
      GO TO 280
  270 LM=LX(J)
      IX=J
  280 CONTINUE
      NU(IX)=I
      DO 290 J=1,6
  290 IA(J,I)=IONP(J,IX)
      LS(I)=LX(IX)
      LX(IX)=0
  300 CONTINUE
      MR=0
      DO 350 I=1,M
      LX(I)=LS(I)
      IF(LL(I).GT.1) MR=MR+1
      DO 330 J=1,6
      IONP(J,I)=0
      IF(IA(J,I).EQ.0) GO TO 330
      MIN=1000
      DO 320 K=1,6
      IF(IA(K,I).LE.0) GO TO 320
      IX=IA(K,I)
      IF(MIN.LT.NU(IX)) GO TO 320
      MIN=NU(IX)
      KM=K
  320 CONTINUE
      IONP(J,I)=MIN
      IA(KM,I)=-1
  330 CONTINUE
  350 CONTINUE
      RETURN
      END
C
      SUBROUTINE GEOM1
C===> CALCULATE AND OUTPUT BOND LENGTHS AND ANGLES.
C===> N.B. THIS ROUTINE MUST BE PLACED BEFORE THE 'MITHGRAPH' ROUTINES
C===> AS THEY ARE ALL INTEGER 2 WHEREAS THIS IS INTEGER 4
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IDUM,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),ICLCK,ICNT
      COMMON /BLK1/ NSTORE(25100),XA(3,200),XB(4,200),ISYM(100),
     1 IFRAG(200),ICON(200),IBOND(2000),JFRAG(200),
     2 JCON(200),JBOND(2000),IUSE(200),IONA(6,200),MO(200),
     3 MU(200),NO(200),NU(200),KEYST(200),LW(200),LR(200),LX(200),
```

```
     4 JK(200),IK(200),KEY(200),IPL(200),CAN(200),CON(200),LL(200),
     5 IA(6,200),LS(200),X(4,200)
      COMMON /BLK2/ IS(2,3,24),TS(3,24),NSYM,ICENT,LAT,IAPX,
     1 CELL(6),T(3,3),DXMAX(3),TL(3,4),FS(3,3,24),MLAT,NEQV,MCON,IHT,
     2 FOM,KFRAG(20),NFRAG,LFRAG(20),MFRAG,NOFRAG,NN,MFRN,DUMY(20),
     3 KMATCH(20),ITYPE,NOINT,MNPPF
      COMMON /BLK4/ IONP(6,200),NAFRG(5),ICH(12),BB(6),LINX(60),NSEQ
      COMMON /BLK5/ NOJOIN,NOPROJ,THOPI,FACTOR,RMAX,RMIN,DOUT,
     1 MH(3),ILEVEL,IPNT,XLIM(3),NOSET,NPIC,NP(3),NEX,ANGMIN,ANGMAX,
     2 DMIN,DMAX,DMUT,DM,DFRG,ALN,NPATT,IVDU,NAT,NPC,SCALE,IPASS
      COMMON/BLK6/ INBOND(1000),IFMT(120),JBND(30)
      DIMENSION IB(3),XS(3),X1(3),LINE1(4,4),ENIL(30)
C===> INITIALISE ARRAY
      DO 10 I=1,NAT
        ICON(I+1)=0
   10 CONTINUE
C===> CLEAR CONSOLE SCREEN
      CALL CLSCN
      WRITE(*,1110)
 1110 FORMAT(10X,'Table of interpeak distances'/10X,28(1H-)//2X'From',3X,7HTo
     1Dist)
C===> COUNT LINE NUMBERS FOR OUTPUTTING "PAGES" OF DATA ON CONSOLE
      LN=4
      ICON(1)=0
      DO 15 I=1,MCON
        II=IBOND(I)/4194304
        ICON(II+1)=I
   15 CONTINUE
      DO 20 I=1,NAT
        IF(ICON(I+1).EQ.0) ICON(I+1)=ICON(I)
        N=0
        JUMP=0
        DO 30 J=1,NAT
          IF(J.EQ.I) GO TO 40
          DO 50 K=1,3
            X1(K)=X(K,I)-X(K,J)
            IF(ABS(X1(K)).GT.DXMAX(K)) GO TO 40
   50     CONTINUE
C===> QUAD IS A FUNCTION SUBROUTINE
          DIST=QUAD(X1,X1,T)
          IF(DIST.GT.DMUT) GO TO 40
          N=N+1
          JBND(N)=J
          ENIL(N)=SQRT(DIST)
   40     IF(N.LT.5.AND.J.NE.NAT) GO TO 30
C===> INCREMENT LINE COUNTER
          LN=LN+1
          IF(LN.EQ.23)THEN
            WRITE(*,1120)
 1120       FORMAT(50X,'Press <CR> to continue')
            READ(*,1125)CR
 1125       FORMAT(A)
            LN=0
          ENDIF
C===> WRITE OUT DISTANCES WHEN ALL THOSE TO A PARTICULAR PEAK HAVE BEEN
C===> FOUND OR WHEN THERE ARE ENOUGH TO FILL A ROW ON THE SCREEN
          IF(JUMP.EQ.0.AND.N.GT.0)
     1            WRITE(*,1190) I,(JBND(L),ENIL(L),L=1,N)
 1190     FORMAT(1H ,I5,5(I6,F5.2))
          IF(JUMP.NE.0.AND.N.GT.0)WRITE(*,1210)(JBND(L),ENIL(L),L=1,N)
 1210     FORMAT(1H ,5X,5(I6,F5.2))
```

```
          N=0
          JUMP=1
   30    CONTINUE
   20 CONTINUE
C===> PUT ANGLES ON 'NEW PAGE'
      WRITE(*,1120)
      READ(*,1125)CR
      CALL CLSCN
      LN=3
C===> GET MAX DISTANCE
      DMX=SQRT(DMAX)
      WRITE(*,1300) DMX
 1300 FORMAT(7X,'Table of bond angles among peaks closer together than',F5.2,1
     1HA/7X,58(1H-)/)
      N=0
      DO 60 I=1,NAT
        KS=ICON(I)+1
        KL=ICON(I+1)
        IF(KS.GE.KL) GO TO 60
        KLM1=KL-1
        DO 70 J=KS,KLM1
          V1=0.001*FLOAT(MOD(IBOND(J),8192))
          IF(V1.GT.DMX.OR.V1.LT.0.2) GO TO 70
          JJ=IBOND(J)/8192
          JJ=MOD(JJ,512)
          JP1=J+1
          DO 80 K=JP1,KL
            V2=0.001*FLOAT(MOD(IBOND(K),8192))
            IF(V2.GT.DMX.OR.V2.LT.0.2) GO TO 80
            KK=IBOND(K)/8192
            KK=MOD(KK,512)
            DO 90 L=1,3
   90         X1(L)=X(L,KK)-X(L,JJ)
            DIST=QUAD(X1,X1,T)
            IF(DIST.LT.0.2) GO TO 80
            COSA=0.5*(V1*V1+V2*V2-DIST)/(V1*V2)
            SINA=SQRT(AMAX1(1.0-COSA*COSA,0.001))
            N=N+1
            LINE1(1,N)=JJ
            LINE1(2,N)=I
            LINE1(3,N)=KK
            LINE1(4,N)=AMOD(ATAN2(SINA,COSA)*RTOD+180.0,180.0)+0.5
            IF(N.LT.4) GO TO 80
            LN=LN+1
            IF(LN.EQ.23)THEN
              WRITE(*,1120)
              READ(*,1125)CR
              LN=0
            ENDIF
C===> WRITE WHEN THERE ARE ENOUGH BOND ANGLES TO FILL A ROW ON SCREEN
            WRITE(*,1470) ((LINE1(L,N),L=1,4),N=1,4)
 1470       FORMAT(4(I6,2H -,I3,2H -,I3,I4))
            N=0
   80     CONTINUE
   70   CONTINUE
   60 CONTINUE
      IF(N.NE.0) WRITE(*,1470) ((LINE1(L,J),L=1,4),J=1,N)
      WRITE(*,8000)
 8000 FORMAT(//16X,'---- Geometry calculations complete ----'/)
      RETURN
      END
```

```
C
      SUBROUTINE SCREEN( NELSE, XB, NAT, ICON, JBOND, NOFRAG, JCON, X, CELL, NSYMOP)
$INT2
C===>
C===>  M   M  III  TTTTT  H   H  GGGGG  RRRR   AA   PPPP  H   H
C===>  MM MM   I     T    H   H  G      R  R  A  A  P  P  H   H
C===>  M M M   I     T    HHHHH  G  GG  RRRR  AAAA  PPPP  HHHHH
C===>  M   M   I     T    H   H  G   G  R R   A  A  P     H   H
C===>  M   M  III    T    H   H  GGGGG  R  R  A  A  P     H   H
C===>
C===>  A SERIES OF MOLECULAR MODELLING ROUTINES TO AID THE
C===>  DETERMINATION OF CRYSTAL STRUCTURES
C===>
C===>  AUTHOR:  S. R. BROWN
C===>
C===>  UNIVERSITY OF GLASGOW  1986
C===>
C===>
      INTEGER*4 NIN, NOUT, NTAPEA, NTAPEB, NTAPEC, NTAPED, NTAPEE, NTAPEF, IK,
     1 NTAPEG, IH, ICH1, ICH2, ITLE, IR, ICALL, NRC, NCH, NREAD, IEND, NREF, MAXH,
     2 ISPFL, LK, IDEF, ICHK, ISYMP, LEVEL, IFLOW, KUSE, INXT, IRDY, NULL, IDIF,
     3 IVDIF, KARLE, IHVY, IGPFL, NDIFF, ISPCH, ICLCK, ICNT, NAT, ICON, JBOND, NOFRAG
     4 , JCON, NSYMOP, NELSE
      COMMON/IO/NIN, NOUT, NTAPEA, NTAPEB, NTAPEC, NTAPED, NTAPEE,
     1 NTAPEF, NTAPEG, IH(155), A(200), ICH1(10), ICH2(10), ITLE(68), IR(68),
     2 ICALL, NRC, NCH, NREAD, IEND, NREF, PI, DTOR, MAXH, ISPFL, ZERO, LK(32),
     3 IDEF, ICHK, ISYMP, LEVEL, IFLOW(25), KUSE(40), INXT, IRDY, RTOD, NULL,
     4 IDIF, IVDIF, KARLE, IHVY, IGPFL, NDIFF, ISPCH(30), ICLCK, ICNT
      COMMON/GRAPH/IX(2,200), LCON(200,4), NTYPE(200), NFRG, NUMPNT, KL(200,4)
     1 , NPAGE, NFRAG, SCALE, IIX(2,200), NAMPNT, XMIN, YMIN, JKIND, JPRINT
      DIMENSION XB(4,200), ICON(200), JBOND(2000), NUMCHR(4),
     1 XXB(2,200), JCON(200), X(4,200), CELL(6), OLDXB(3,200)
      DATA NMARK/' '/
      NFRAG=NOFRAG
      NPAGE=0
      NFRG=NAT
      JKIND=2
C===> MAXIMUM OF 200 ATOMS
C===> SWITCH FOR NUMBER/DON'T NUMBER ATOMS
      NUMPNT=0
      NFILE=0
C===> AUTO SCALING: -MAKE X & Y INTEGERS SUCH THAT  100 < X < 500
C===> AND  50 < Y < 450
      XMAX=-1000000
      XMIN=1000000
      YMAX=-1000000
      YMIN=1000000
      DO 50 J=1,200
C===> ASSUME (FOR NOW) ALL ATOMS ARE C sp3 - THERE IS AN OPTION TO
C===> CHANGE ATOM TYPES LATER IN THE PROGRAM
        NTYPE(J)=1
        DO 60 K=1,4
          LCON(J,K)=0
          KL(J,K)=0
   60   CONTINUE
   50 CONTINUE
C===> FIND MAX & MIN X & Y COORDINATES
      DO 100 I=1,NAT
        XMIN=AMIN1(XB(1,I),XMIN)
        XMAX=AMAX1(XB(1,I),XMAX)
        YMAX=AMAX1(XB(2,I),YMAX)
```

```
        YMIN=AMIN1(XB(2,I),YMIN)
  100 CONTINUE
C===> MAKE ALL COORDINATES POSITIVE
      XMAX=XMAX-XMIN
      YMAX=YMAX-YMIN
      DO 200 I=1,NAT
        XXB(1,I)=XB(1,I)-XMIN
        XXB(2,I)=XB(2,I)-YMIN
  200 CONTINUE
C===> SCALE COORDINATES
      IF(XMAX.GE.YMAX) THEN
        SCALE=400/XMAX
      ELSE
        SCALE=400/YMAX
      ENDIF
      OSCALE=SCALE
C===> USE ARRAYS OLDXB( ) & IIX( ) TO STORE THE ORIGINAL VALUES OF XB( ) &
C===> IX( ) - THESE GET ALTERED IN SOME OF THE LATER ROUTINES
      DO 400 I=1,NAT
        XXB(1,I)=XXB(1,I)*SCALE
        XXB(2,I)=XXB(2,I)*SCALE
        IX(1,I)=IFIX(XXB(1,I))+100
        IX(2,I)=IFIX(XXB(2,I))+50
        IIX(1,I)=IX(1,I)
        IIX(2,I)=IX(2,I)
        DO 410 J=1,3
          OLDXB(J,I)=XB(J,I)
  410   CONTINUE
  400 CONTINUE
C===> NOW JOIN POINTS
      DO 1947 IT=1,NAT
        IP=IFIX(XB(4,IT))
        LIM1=ICON(IP)+1
        LIM2=ICON(IP+1)
        IF(LIM2.LT.LIM1) GOTO 1947
        DO 1950 IQ=LIM1,LIM2
          JDIV=(JBOND(IQ))/8192
          IF(JDIV.EQ.0) GOTO 1950
          IF(IP.GT.JDIV) GOTO 1950
          IL=0
          DO 1949 MG=1,NAT
            IF(IFIX(XB(4,MG)).NE.JDIV) GOTO 1949
            IL=MG
            GOTO 1951
 1949     CONTINUE
 1951     IF(IL.EQ.0) GOTO 1950
          IP1=IFIX(XB(4,IL))
C===> SET UP CONNECTIVITY TABLE
          DO 1955 L=1,4
            IF(LCON(IL,L).GT.0) THEN
              GOTO 1955
            ELSE
              LCON(IL,L)=IT
              GOTO 1956
            END IF
 1955     CONTINUE
 1956     DO 1957 K=1,4
            IF(LCON(IT,K).GT.0)THEN
              GOTO 1957
            ELSE
              LCON(IT,K)=IL
```

```
              GOTO 1960
            END IF
 1957     CONTINUE
C===> DIFFERENT FORM OF CONNECTIVITY TABLE
 1960     DO 1961 LL=1,4
            IF(KL(IP1,LL).GT.0) THEN
              GOTO 1961
            ELSE
              KL(IP1,LL)=IP
              GOTO 1962
            ENDIF
 1961     CONTINUE
 1962     DO 1963 KK=1,4
            IF(KL(IP,KK).GT.0) THEN
              GOTO 1963
            ELSE
              KL(IP,KK)=IP1
              GOTO 1950
            ENDIF
 1963     CONTINUE
 1950   CONTINUE
 1947 CONTINUE
C===> NOW READY TO USE GRAPHICS
      IF(NTITLE.EQ.1)GOTO 15
C===> PRINT TITLE PAGE
      CALL INITIL
      CALL SETPAG(0)
C===> BLACK BACKGROUND
      CALL COLOUR(63)
      CALL CLEARW
      CALL WTEXT(55,295,2,2,0,1,0,'A program to manipulate and edit')
      CALL WTEXT(40,265,2,2,0,1,0,'molecular fragments found by Mithril')
      CALL SETDAT(60)
      CALL WTEXT(65,345,7,2,0,1,0,'=========')
C===> SET PEN TO WRITE BLUE
      CALL SETDAT(2)
      CALL WTEXT(65,360,7,7,0,1,0,'MITHGRAPH')
      CALL WTEXT(80,105,2,2,0,1,0,'*  (USE LIGHT-PEN TO START)')
      CALL WTEXT(90,5,1,1,0,1,0,'( Author: S.R.Brown  University of Glasgow 1
     1986 )')
C===> NOW DRAW WINDOW
      CALL SETDAT(27)
      CALL WTEXT(25,435,4,2,0,1,0,'==================')
      CALL WTEXT(25,230,4,2,0,1,0,'==================')
      CALL WTEXT(35,235,4,2,2,1,0,'=========')
      CALL WTEXT(484,235,4,2,2,1,0,'=========')
      NTITLE=1
C===> READ LIGHTPEN
    4 CALL RDPEN(LX,LY)
      IF(LX.LT.70.OR.LX.GT.90) GOTO 4
      IF(LY.LT.95.OR.LY.GT.115) GOTO 4
      CALL INITIL
C===> DISPLAY FRAGMENT
   15 CALL SHWFRG(NAT,XB)
C===> PUT MENU ON SCREEN
      CALL GMENU1(1)
C===> READ LIGHTPEN
   10 CALL RDPEN(NXLP,NYLP)
      IF(NXLP.GT.20.OR.NXLP.LT.0) GOTO 10
C===> SCREEN DUMP
      IF(NYLP.LE.240.AND.NYLP.GE.220) THEN
```

```
          CALL GMENU1(0)
          CALL WINDOW(0)
          CALL DUMP
          CALL GMENU1(1)
          GOTO 10
C===>   CONTINUE
        ELSE IF (NYLP.LE.415.AND.NYLP.GE.395) THEN
          CALL CONT
          GOTO 20
C===>   PUT WINDOW ROUND FRAGMENT
        ELSE IF (NYLP.LE.345.AND.NYLP.GE.325) THEN
          CALL WINDOW(1)
          GOTO 10
C===>   QUIT
        ELSE IF (NYLP.LE.450.AND.NYLP.GE.430) THEN
          CALL EXIT
C===>   NUMBER ATOMS
        ELSE IF (NYLP.LE.310.AND.NYLP.GE.290) THEN
C===>   ENSURE NO NAMES ARE PRESENT
          IF(MOD(NAMPNT,2).NE.0)THEN
            NAMPNT=NAMPNT+1
            CALL NAMES(XB,NAT)
          ENDIF
          NUMPNT=NUMPNT+1
          CALL NUMBER(XB,NAT)
          GOTO 10
C===>   FILES
        ELSE IF (NYLP.LE.275.AND.NYLP.GE.255) THEN
          CALL FILES(X,XB,NAT,CELL,JCON,NSYMOP)
          GOTO 10
C===>   GO INTO EDIT MODE
        ELSE IF (NYLP.LE.380.AND.NYLP.GE.360) THEN
          CALL EDIT(NAT,XB,X,JCON,NSYMOP,NELSE)
C===>   TRANSLATE MOLECULE
        ELSE IF (NYLP.LE.205.AND.NYLP.GE.185) THEN
          CALL GMENU1(0)
          CALL WINDOW(0)
          CALL TRANS(NAT,XB)
          CALL GMENU2(0)
C===>   RESTORE FRAGMENT TO ITS ORIGINAL POSITION
        ELSE IF (NYLP.LE.170.AND.NYLP.GE.150) THEN
          CALL REST(NAT,XB,OLDXB,OSCALE)
C===>   INCREASE/REDUCE SIZE OF THE FRAGMENT
        ELSE IF (NYLP.LE.135.AND.NYLP.GE.115) THEN
          CALL GMENU1(0)
          CALL ENLRGE(NAT,XB)
C===>   SPACE-FILLED MODEL
        ELSE IF(NYLP.LE.100.AND.NYLP.GE.80) THEN
          CALL SPACE(XB,NAT)
C===>   NAME ATOMS
        ELSE IF(NYLP.GE.45.AND.NYLP.LE.65)THEN
C===>   FIRST ENSURE THERE ARE NO NUMBERS PRESENT
          IF(MOD(NUMPNT,2).NE.0)THEN
            NUMPNT=NUMPNT+1
            CALL NUMBER(XB,NAT)
          ENDIF
          NAMPNT=NAMPNT+1
          CALL NAMES(XB,NAT)
          GOTO 10
C===>   ROTATE FRAGMENT
        ELSE IF(NYLP.LE.30.AND.NYLP.GE.10) THEN
```

```
        CALL GMENU1(0)
        CALL ROTATE(NAT,XB)
      ELSE
C===> GO BACK AND READ LIGHTPEN AGAIN
        GOTO 10
      END IF
      GOTO 15
   20 RETURN
      END
C
      SUBROUTINE NUMBER(XB,NAT)
$INT2
C===> INSERTS OR REMOVES ATOM NUMBERS
C===> NUMBERS MUST BE CONVERTED INTO CHARACTER STRINGS
      DIMENSION NUMCHR(4),XB(4,200)
      INTEGER*4 NAT
      COMMON/GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
     1 ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
      DATA NMARK/' '/
      NC=0
      NP=1
      IF(MOD(NUMPNT,2).EQ.0)NP=0
      DO 100 N=1,NAT
        IF(LCON(N,1).EQ.0)GOTO 100
        NC=IFIX(XB(4,N))
        IF(NC.GE.100) THEN
          NUMCHR(1)=NC/100
          NUMCHR(2)=(NC-(NUMCHR(1)*100))/10
          NUMCHR(3)=NC-(NUMCHR(1)*100+NUMCHR(2)*10)
          NUMCHR(4)=NMARK
        ELSEIF(NC.GE.10)THEN
          NUMCHR(1)=NC/10
          NUMCHR(2)=NC-(NUMCHR(1)*10)
          NUMCHR(3)=NMARK
        ELSE
          NUMCHR(1)=NC
          NUMCHR(2)=NMARK
        ENDIF
        CALL NTEXT(IX(1,N)+3,IX(2,N)+3,1,1,0,NP,0,NUMCHR)
  100 CONTINUE
      RETURN
      END
C
      SUBROUTINE GMENU1(I)
$INT2
C===> I=1: DISPLAY MENU    I=0: REMOVE MENU
      DIMENSION NUMCHR(2)
      COMMON /GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
     1 ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
      DATA NMARK/' '/
      CALL WTEXT(10,480,1,1,0,I,32,'CLUSTER NO.')
      NUMCHR(1)=NFRAG
      NUMCHR(2)=NMARK
      CALL WTEXT(80,480,1,1,0,I,32,NUMCHR)
      CALL WTEXT(10,440,1,1,0,I,0,'QUIT')
      CALL WTEXT(10,405,1,1,0,I,0,'CONT')
      CALL WTEXT(10,370,1,1,0,I,0,'EDIT')
      CALL WTEXT(10,335,1,1,0,I,0,'WINDOW')
      CALL WTEXT(10,300,1,1,0,I,0,'NUMBER')
      CALL WTEXT(10,265,1,1,0,I,0,'FILES')
      CALL WTEXT(10,230,1,1,0,I,0,'DUMP')
```

```
      CALL WTEXT(10,195,1,1,0,I,0,'TRANSLATE')
      CALL WTEXT(10,160,1,1,0,I,0,'RESTORE')
      CALL WTEXT(10,125,1,1,0,I,0,'ENLARGE')
      CALL WTEXT(10,90,1,1,0,I,0,'SPACEFILL')
      CALL WTEXT(10,55,1,1,0,I,0,'NAMES')
      CALL WTEXT(10,20,1,1,0,I,0,'ROTATE')
      RETURN
      END
C
      SUBROUTINE WINDOW(I)
$INT2
C===> DRAW OR REMOVE WINDOW AROUND FRAGMENT
      CALL LOADXY(90,40)
      DO 100 L=1,20
        IF(L.LE.5) THEN
          J=84
          K=0
        ELSE IF(L.GT.5.AND.L.LE.10) THEN
          J=0
          K=84
        ELSE IF(L.GT.10.AND.L.LE.15) THEN
          J=-84
          K=0
        ELSE
          J=0
          K=-84
        END IF
        CALL RELVEC(J,K,0,I,56)
  100 CONTINUE
      RETURN
      END
C
      SUBROUTINE EXIT
$INT2
C===> EXIT FROM PROGRAM
      INTEGER*4 NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,NTAPEF,
     1 NTAPEG,IH,ICH1,ICH2,ITLE,IR,ICALL,NRC,NCH,NREAD,IEND,NREF,MAXH,
     2 ISPFL,LK,IDEF,ICHK,ISYMP,LEVEL,IFLOW,KUSE,INXT,IRDY,NULL,IDIF,
     3 IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH,ICLCK,ICNT
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFL,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),ICLCK,ICNT
      LOGICAL*4 FILE
      CALL INITL1
      CALL WTEXT(40,250,5,5,0,1,35,'END OF PROGRAM')
C===> ERASE SEARCH DATA FILE
      INQUIRE(FILE='SRCH.FIL',EXIST=FILE)
      IF(FILE)THEN
C===> FIRST HAVE TO OPEN IT
        OPEN(FILE='SRCH.FIL',UNIT=NTAPEE,FORM='UNFORMATTED')
        CLOSE(UNIT=NTAPEE,STATUS='DELETE')
      ENDIF
      STOP
      END
C
      SUBROUTINE CHEMOD(NAT,JCON,X,CELL)
$INT2
C===> ROUTINE TO WRITE A FILE THAT CAN BE PASSED TO THE CHEMMOD
C===> SYSTEM OF MOLECULAR MODELLING ROUTINES
```

```
      INTEGER*4 NIN, NOUT, NTAPEA, NTAPEB, NTAPEC, NTAPED, NTAPEE, NTAPEF,
     1 NTAPEG, IH, ICH1, ICH2, ITLE, IR, ICALL, NRC, NCH, NREAD, IEND, NREF, MAXH,
     2 ISPFL, LK, IDEF, ICHK, ISYMP, LEVEL, IFLOW, KUSE, INXT, IRDY, NULL, IDIF,
     3 IVDIF, KARLE, IHVY, IGPFL, NDIFF, ISPCH, ICLCK, ICNT, NAT, JCON
      COMMON/IO/NIN, NOUT, NTAPEA, NTAPEB, NTAPEC, NTAPED, NTAPEE,
     1 NTAPEF, NTAPEG, IH(155), A(200), ICH1(10), ICH2(10), ITLE(68), IR(68),
     2 ICALL, NRC, NCH, NREAD, IEND, NREF, PI, DTOR, MAXH, ISPFL, ZERO, LK(32),
     3 IDEF, ICHK, ISYMP, LEVEL, IFLOW(25), KUSE(40), INXT, IRDY, RTOD, NULL,
     4 IDIF, IVDIF, KARLE, IHVY, IGPFL, NDIFF, ISPCH(30), ICLCK, ICNT
      COMMON/GRAPH/IX(2,200), LCON(200,4), NTYPE(200), NFRG, NUMPNT, KL(200,4)
     1 , NPAGE, NFRAG, SCALE, IIX(2,200), NAMPNT, XMIN, YMIN, JKIND, JPRINT
      DIMENSION ZRO(200), JCON(200), X(4,200), CELL(6), NEW(200), NCON(200,4)
      CHARACTER*10 CFILE
      LOGICAL*4 FILE
      NCHEM=16
      NIL1=0
      NIL2=0
      NATF=0
C===> INITIALISE ARRAYS
      DO 50 I=1,NAT
        ZRO(I)=0.0
        NEW(I)=0
        DO 45 JJ=1,4
          NCON(I,JJ)=0
   45   CONTINUE
   50 CONTINUE
C===> FIND OUT HOW MANY ATOMS ARE IN THE FRAGMENT
      DO 100 L=1,200
        IF(KL(L,1).EQ.0)GOTO 100
        NATF=NATF+1
        NEW(L)=NATF
  100 CONTINUE
      IF(NATF.LE.4)RETURN
C===> GET FILE
      WRITE(*,9000)
 9000 FORMAT(/'Enter the name of the CHEMMOD data file -'\)
      READ(*,9001)CFILE
 9001 FORMAT(A)
      INQUIRE(FILE=CFILE,EXIST=FILE)
      IF(FILE)THEN
        OPEN(FILE=CFILE,UNIT=NCHEM)
      ELSE
C===>   FILE DOES NOT EXIST
        OPEN(FILE=CFILE,UNIT=NCHEM,STATUS='NEW',FORM='FORMATTED')
      END IF
      WRITE(NCHEM,110)ITLE
  110 FORMAT(68A1)
      WRITE(NCHEM,120)CELL
  120 FORMAT(6F9.3)
      WRITE(NCHEM,130)NATF
  130 FORMAT(I4)
      DO 150 L=1,NAT
        IF(KL(L,1).EQ.0)GOTO 150
        WRITE(NCHEM,140)NTYPE(L),(X(J,L),J=1,3),ZRO(L)
  140   FORMAT('C',I5,4X,3F9.4,F11.5)
  150 CONTINUE
      WRITE(NCHEM,160)NATF
  160 FORMAT(I4)
C===> ALTER ATOM NUMBERS SO THAT THEY ARE NUMBERED CONSECUTIVELY
C===> & CHANGE CONNECTIVITY TABLE ACCORDINGLY
      NN=0
```

```
      DO 60 L=1,NAT
        IF(KL(L,1).EQ.0) GOTO 60
        NN=NN+1
        DO 70 J=1,4
          IF(KL(L,J).EQ.0)GOTO 60
          NCON(NN,J)=NEW(KL(L,J))
   70   CONTINUE
   60 CONTINUE
C===> CONNECTIVITY TABLE
      DO 170 K=1,NATF
        WRITE(NCHEM,165)K,(NCON(K,L),L=1,4),NIL1,NIL2
  165   FORMAT(I2,6I4)
  170 CONTINUE
      RETURN
      END
C
      SUBROUTINE SHWFRG(NAT,XB)
$INT2
C===> DISPLAYS FRAGMENT GIVEN ATOM CONNECTIVITIES
      DIMENSION XB(4,200)
      COMMON/GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
     1 ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
      INTEGER*4 NAT
C===> DRAW TO HIDDEN PAGE
      CALL HPAGE
      DO 100 J=1,NAT
        IF(LCON(J,1).EQ.0)GOTO 100
        DO 200 L=1,4
          IF(LCON(J,L).EQ.0)GOTO 200
          MP=LCON(J,L)
          CALL POINT(IX(1,MP),IX(2,MP))
C===> GET COLOURS CORRESPONDING TO DIFFERENT ATOM TYPES
          N1=IFIX(XB(4,MP))
          N2=IFIX(XB(4,J))
          CALL COLOR(N1,NCLR1)
          CALL COLOR(N2,NCLR2)
          IDX=IX(1,J)-IX(1,MP)
          IDY=IX(2,J)-IX(2,MP)
          CALL RELVEC(IDX/2,IDY/2,0,1,NCLR1)
          CALL RELVEC(IDX/2,IDY/2,0,1,NCLR2)
  200   CONTINUE
  100 CONTINUE
C===> BRING HIDDEN PAGE INTO VIEW
      CALL PAGE
      RETURN
      END
C
      SUBROUTINE EDIT(NAT,XB,X,JCON,NSYMOP,NELSE)
$INT2
C===> DISPLAY EDIT MENU
      CHARACTER*1 CH
      DIMENSION XB(4,200),X(4,200)
      COMMON/GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
     1 ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
      INTEGER*4 NAT,JCON(200),NSYMOP,NELSE
      CALL GMENU1(0)
C===> WAIT BEFORE DISPLAYING EDIT OPTIONS TO ALLOW USER TIME TO MOVE LIGHT PEN
      CALL PAUSE
    5 CALL EMENU(1)
C===> NUMBER MOLECULE
C===> FIRST GET RID OF NAMES (IF PRESENT)
```

```
      IF( MOD( NAMPNT, 2) . NE. 0) THEN
        NAMPNT=NAMPNT+1
        CALL NAMES( XB, NAT)
      ENDIF
      IF( MOD( NUMPNT, 2) . EQ. 0) NUMPNT=NUMPNT+1
      CALL NUMBER( XB, NAT)
C===> READ LIGHT PEN
   10 CALL RDPEN( IXLP, IYLP)
      IF( IXLP. GT. 20. OR. IXLP. LT. 0) GOTO 10
C===> RETURN TO MASTER MENU
      IF( IYLP. LE. 300. AND. IYLP. GE. 280) THEN
   20   CALL EMENU( 0)
        CALL GMENU1( 1)
        GOTO 30
C===> CHANGE ATOM TYPE
      ELSE IF ( IYLP. LE. 460. AND. IYLP. GE. 440) THEN
        CALL EMENU( 0)
        CALL ATMTYP( JCON, XB, NAT)
        CALL SHWFRG( NAT, XB)
C===> EXIT FROM PROGRAM
      ELSE IF ( IYLP. LE. 500. AND. IYLP. GE. 480) THEN
        CALL EXIT
C===> REMOVE AN ATOM FROM THE FRAGMENT
      ELSE IF ( IYLP. LE. 420. AND. IYLP. GE. 400) THEN
        CALL EMENU( 0)
        CALL REMOVE( XB, NAT, JCON)
C===> GEOM
      ELSE IF ( IYLP. LE. 380. AND. IYLP. GE. 340) THEN
        CALL EMENU( 0)
        CALL GEOM1
C===> UPDATE MODEL FILE FOR FOURIER RECYCLING OF FRAGMENT
      ELSE IF( IYLP. LE. 340. AND. IYLP. GE. 320) THEN
        CALL EMENU( 0)
   25   WRITE( *, 9000)
 9000   FORMAT( ///' This option will store the atoms in this fragment as a
     . group in a model file'/' for use in Fourier recycling. Proceed ? (y/n)
     . - '\)
C===> GET OPTION
        READ( *, 9001) CH
 9001   FORMAT( A)
C===> REVERT TO EDIT
        IF( CH. EQ. 'N'. OR. CH. EQ. 'n') THEN
          GOTO 5
C===> UPDATE MODEL FILE
        ELSE IF( CH. EQ. 'Y'. OR. CH. EQ. 'y') THEN
          CALL UPDATE( XB, X, NAT, NSYMOP, JCON, NELSE)
        ELSE
          WRITE( *, 9002)
 9002     FORMAT( /' Invalid option - try again')
          GOTO 25
        ENDIF
      ENDIF
      GOTO 5
   30 RETURN
      END
C
      SUBROUTINE EMENU( I)
$INT2
      CALL WTEXT( 450, 490, 1, 1, 0, I, 0, ' EDIT')
      CALL WTEXT( 10, 490, 1, 1, 0, I, 0, ' QUIT')
      CALL WTEXT( 10, 450, 1, 1, 0, I, 0, ' TYPE')
```

```
      CALL WTEXT(10,410,1,1,0,I,0,'REMOVE')
      CALL WTEXT(10,370,1,1,0,I,0,'GEOM')
      CALL WTEXT(10,330,1,1,0,I,0,'UPDATE')
      CALL WTEXT(10,290,1,1,0,I,0,'RETURN')
      RETURN
      END
C
      SUBROUTINE ATMTYP(JCON,XB,NAT)
$INT2
C===> ROUTINE TO DISPLAY TYPES OF ATOMS - USER THEN PICKS A NEW ATOM TYPE
      INTEGER*4 JCON(200),NAT
      CHARACTER*13 NAMES(20)
      DIMENSION NVAL(20),XB(4,200)
      COMMON/GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
     1 ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
      DATA NAMES/'C sp3        ','C sp2        ','C Aromatic   ','C sp
     1 ','N (+)R(1)R(2)','N Tetrahedral','N Aromatic   ','N (-)Aromatic',
     2 'N Trigonal   ','O Single bond','O Double bond','S Digonal    ',
     3 'S Tetrahedral','Phosphorus   ','Hydrogen     ','Fluorine     ',
     4 'Chlorine     ','Bromine      ','Iodine       ','Other        '/
C===> ARRAY NVAL( ) STORES THE ATOMIC VALENCIES
      DATA NVAL/4,3,3,2,4,4,3,3,3,2,1,2,4,5,1,1,1,1,1,5/
C===> DISPLAY OPTIONS
      CALL WTEXT(10,460,2,1,0,1,0,'* Atom type')
      CALL WTEXT(10,420,2,1,0,1,0,'* Single atom(s)')
C===> READ LIGHT-PEN
    5 CALL RDPEN(MX,MY)
      IF(MX.GT.20.OR.MX.LT.0)GOTO 5
      IF(MY.GT.470.OR.MY.LT.410)GOTO 5
C===> CHANGE ALL ATOMS OF ONE TYPE OR INDIVIDUAL ATOMS
      IF(MY.LE.430.AND.MY.GE.410) THEN
        NSING=1
      ELSE IF (MY.LE.470.AND.MY.GE.450)THEN
        NSING=2
      ELSE
        GOTO 5
      ENDIF
C===> GET USER TO ENTER NUMBER OF ATOM TO BE CHANGED
 10   IF (NSING.EQ.1) THEN
        WRITE(*,9000)
 9000   FORMAT(/'Type in number of atom to be changed   '\)
        READ(*,9001)NATM
 9001   FORMAT(I2)
        IF(NATM.EQ.0)GOTO 400
        NTP=NTYPE(NATM)
      ENDIF
C===> WRITE TO HIDDEN PAGE
      CALL HPAGE
      IF(NSING.EQ.1)THEN
C===> SHOW THE PRESENT ATOM TYPE
        CALL WTEXT(50,480,2,2,0,1,60,'ATOM IS PRESENTLY:')
        CALL WTEXT(280,480,2,2,0,1,60,NAMES(NTP))
        CALL WTEXT(50,440,2,2,0,1,60,'PICK NEW ATOM TYPE FROM THE LIST BELOW')
      ELSE IF (NSING.EQ.2) THEN
        CALL WTEXT(75,480,2,2,0,1,60,' CHOOSE ATOM TYPE TO BE CHANGED')
        CALL WTEXT(115,440,2,2,0,1,60,' FROM THE LIST BELOW')
      ENDIF
      CALL DISMEN(NAMES)
C===> BRING HIDDEN PAGE INTO VIEW
      CALL PAGE
C===> RE-SET PEN TO DRAW WHITE
```

```
      CALL SETDAT(0)
      CALL RDMEN(ITYPE)
      IF(NSING.EQ.1)THEN
C===> CHECK VALENCY BEFORE CHANGING
        IF(JCON(NATM).GT.NVAL(ITYPE))THEN
          WRITE(*,9002)
 9002     FORMAT(/'Bond type not allowed - valency too great')
        ELSE
          NTYPE(NATM)=ITYPE
        ENDIF
        CALL SHWFRG(NAT,XB)
        CALL NUMBER(XB,NAT)
C===> CHANGE MORE ATOM TYPES ?
        CALL WTEXT(50,50,2,2,0,1,2,'Change more atoms ?')
        CALL WTEXT(80,20,2,2,0,1,2,'* Yes')
        CALL WTEXT(180,20,2,2,0,1,2,'* No')
C===> READ LIGHT-PEN
  220   CALL RDPEN(LX,LY)
        IF(LY.LT.10.OR.LY.GT.30) GOTO 220
        IF(LX.LT.70.OR.LX.GT.190) GOTO 220
        IF(LX.GT.90.AND.LX.LT.170) GOTO 220
C===> ERASE SCREEN PROMPT
        CALL WTEXT(50,50,2,2,0,0,0,'Change more atoms ?')
        CALL WTEXT(80,20,2,2,0,0,0,'* Yes')
        CALL WTEXT(180,20,2,2,0,0,0,'* No')
C===> YES
        IF(LX.LE.90.AND.LX.GE.70) THEN
          GOTO 10
C===> NO
        ELSE IF(LX.LE.190.AND.LX.GE.170) THEN
          RETURN
        ENDIF
      ELSE IF(NSING.EQ.2) THEN
        CALL WTEXT(75,480,2,2,0,0,0,' CHOOSE ATOM TYPE TO BE CHANGED')
        CALL WTEXT(115,440,2,2,0,0,0,' FROM THE LIST BELOW')
C===> GIVE USER TIME TO MOVE LIGHTPEN
        CALL PAUSE
C===> DISPLAY MENU OF ATOM TYPES AGAIN
        CALL HPAGE
        CALL WTEXT(50,460,2,2,0,1,60,'PICK NEW ATOM TYPE FROM THE LIST BELOW
     . :')
        CALL DISMEN(NAMES)
C===> BRING HIDDEN PAGE INTO VIEW
        CALL PAGE
        CALL SETDAT(0)
C===> READ NEW ATOM TYPE FROM THE MENU
        CALL RDMEN(JTYPE)
        DO 300 I=1,200
          IF(NTYPE(I).EQ.ITYPE.AND.JCON(I).LE.NVAL(JTYPE))NTYPE(I)=JTYPE
  300   CONTINUE
      ENDIF
 400  RETURN
      END
C
      SUBROUTINE DISMEN(NAMES)
$INT2
C===> 20 DIFFERENT ATOM TYPES AVAILABLE
      CHARACTER*13 NAMES(20)
C===> DISPLAY MENU OF ATOM TYPES AVAILABLE
      DO 100 J=1,10
        L=415-35*J
```

```
        CALL WTEXT(60,L,2,2,0,1,3,NAMES(J))
        CALL WTEXT(310,L,2,2,0,1,3,NAMES(J+10))
        CALL WTEXT(30,L,2,2,0,1,3,'*')
        CALL WTEXT(280,L,2,2,0,1,3,'*')
  100 CONTINUE
      RETURN
      END
C
      SUBROUTINE RDMEN(J)
$INT2
C===> READ ATOM TYPE OFF A MENU
  200 CALL RDPEN(LX,LY)
      IF(LX.LT.20.OR.LX.GT.290)GOTO 200
      IF(LX.LT.270.AND.LX.GT.40)GOTO 200
C===> GET NEW ATOM TYPE
      DO 210 J=1,10
        NYC=420-35*J
        NUY=NYC+10
        NDY=NYC-10
C===> RIGHT SIDE OF MENU
        IF(LX.GT.270)GOTO 205
        IF(LY.LT.NUY.AND.LY.GT.NDY) RETURN
  205   IF(LY.LT.NUY.AND.LY.GT.NDY) THEN
          J=J+10
          RETURN
        ENDIF
 210  CONTINUE
      RETURN
      END
C
      SUBROUTINE RDPEN(LX,LY)
$INT2
C===> READS LIGHT PEN & RETURNS SCREEN COORDINATES OF THE POSITION WHERE
C===> GOOD READ OCCURRED
      LOGICAL PEN
   10 PEN=.FALSE.
      CALL RDLTPN(NX,NY,PEN)
C===> CHECK FOR GOOD READ
      IF(.NOT.PEN) GOTO 10
C===> ADJUST NX & NY TO BE IN THE RANGE 0 TO 512
      LX=(NX-10)*2
      LY=(NY-1)*2
      RETURN
      END
C
      SUBROUTINE TRANS(NAT,XB)
$INT2
C===> MOVES FRAGMENT IN X OR Y DIRECTIONS
      INTEGER*4 NAT
      LOGICAL PEN
      DIMENSION XB(4,200)
      COMMON/GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
     1 ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
      CALL WTEXT(10,490,2,1,0,1,0,'Choose + or - first')
      CALL GMENU2(1)
   10 CALL RDPEN(LX,LY)
      IF(LX.LT.230.OR.LX.GT.350) GOTO 10
      IF(LX.GT.250.AND.LX.LT.330) GOTO 10
      IF(LY.GT.25.OR.LY.LT.5) GOTO 10
C===> NEGATIVE TRANSLATION
      IF(LX.LE.250.AND.LX.GE.230) THEN
```

```
          ADDX=-0.05
          ADDY=-0.05
C===>   POSITIVE TRANSLATION
        ELSE IF(LX.GE.330.AND.LX.LE.350)THEN
          ADDX=0.05
          ADDY=0.05
        ENDIF
        CALL WTEXT(10,490,2,1,0,0,0,'Choose + or - first')
        CALL WTEXT(10,490,2,1,0,1,0,'Now choose direction of translation')
    20  CALL RDPEN(MX,MY)
        IF(MY.GT.45)GOTO 20
        IF(MX.LT.70.OR.MX.GT.290)GOTO 20
        IF(MX.GT.90.AND.MX.LT.170)GOTO 20
        IF(MX.GT.190.AND.MX.LT.270)GOTO 20
C===>   X
        IF(MX.LT.90)THEN
          ADDY=0.0
C===>   Y
        ELSE IF(MX.GT.90.AND.MX.LT.190)THEN
          ADDX=0.0
C===>   STOP
        ELSE
          GOTO 200
        END IF
   50   DO 100 J=1,NAT
          XB(1,J)=XB(1,J)+ADDX
          XB(2,J)=XB(2,J)+ADDY
          IX(1,J)=IFIX((XB(1,J)-XMIN)*SCALE)+100
          IX(2,J)=IFIX((XB(2,J)-YMIN)*SCALE)+50
  100   CONTINUE
        CALL SHWFRG(NAT,XB)
        CALL GMENU2(1)
C===>   READ LIGHTPEN
        PEN=.FALSE.
        CALL RDLTPN(NX,NY,PEN)
        IF(.NOT.PEN)GOTO 50
C===>   ADJUST NX & NY TO BE IN THE RANGE 0 TO 512
        LX=(NX-10)*2
        LY=(NY-1)*2
        IF(LY.GT.45.OR.LY.LT.5)GOTO 50
        IF(LY.LE.45.AND.LY.GT.25)GOTO 150
        IF(LX.LT.30.OR.LX.GT.350)GOTO 50
C===>   NEGATIVE TRANSLATION
        IF(LX.LE.250.AND.LX.GE.230)THEN
          ADDX=-ABS(ADDX)
          ADDY=-ABS(ADDY)
C===>   POSITIVE TRANSLATION
        ELSE IF(LX.GE.330.AND.LX.LE.350)THEN
          ADDX=ABS(ADDX)
          ADDY=ABS(ADDY)
C===>   QUICKEN
        ELSE IF (LX.GE.30.AND.LX.LE.50) THEN
          IF(ADDX.EQ.0.0)THEN
            IF(ADDY.LT.0.AND.ADDY.GT.-0.25) ADDY=ADDY-0.005
            IF(ADDY.GT.0.AND.ADDY.LT.0.25) ADDY=ADDY+0.005
          ELSE
            IF(ADDX.LT.0.AND.ADDX.GT.-0.25) ADDX=ADDX-0.005
            IF(ADDX.GT.0.AND.ADDX.LT.0.25) ADDX=ADDX+0.005
          ENDIF
C===>   SLOW DOWN
        ELSE IF (LX.GE.130.AND.LX.LE.150) THEN
```

```
        IF(ADDX.EQ.0.0)THEN
          IF(ADDY.LT.-0.02) ADDY=ADDY+0.005
          IF(ADDY.GT.0.02) ADDY=ADDY-0.005
        ELSE
          IF(ADDX.LT.-0.02) ADDX=ADDX+0.005
          IF(ADDX.GT.0.02) ADDX=ADDX-0.005
        ENDIF
      ENDIF
      GOTO 50
  150 IF(LX.LT.70.OR.LX.GT.290)GOTO 50
      IF(LX.GT.90.AND.LX.LT.170)GOTO 50
      IF(LX.GT.190.AND.LX.LT.270)GOTO 50
C===> X DIRECTION
      IF(LX.LE.90.AND.LX.GE.70)THEN
        IF(ADDX.EQ.0.0)THEN
          ADDX=ADDY
          IF(ADDY.EQ.0.0) ADDX=0.05
          ADDY=0.0
        ENDIF
C===> Y DIRECTION
      ELSE IF(LX.LE.190.AND.LX.GE.170)THEN
        IF(ADDY.EQ.0.0)THEN
          ADDY=ADDX
          IF(ADDX.EQ.0.0) ADDY=0.05
          ADDX=0.0
        ENDIF
C===> STOP
      ELSE IF(LX.LE.290.AND.LX.GE.270)THEN
        GOTO 200
      ENDIF
      GOTO 50
  200 RETURN
      END
C
      SUBROUTINE GMENU2(I)
$INT2
      CALL WTEXT(80,35,2,2,0,I,3,'* X')
      CALL WTEXT(180,35,2,2,0,I,3,'* Y')
      CALL WTEXT(280,35,2,2,0,I,3,'* STOP')
      CALL WTEXT(40,15,2,2,0,I,3,'* Q')
      CALL WTEXT(140,15,2,2,0,I,3,'* S')
      CALL WTEXT(240,15,2,2,0,I,3,'* -')
      CALL WTEXT(340,15,2,2,0,I,3,'* +')
      RETURN
      END
C
      SUBROUTINE REST(NAT,XB,OLDXB,OSCALE)
C===> RESTORES MOLECULE TO ITS ORIGINAL POSITION
$INT2
      INTEGER*4 NAT
      COMMON/GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
     1 ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
      DIMENSION XB(4,200),OLDXB(3,200)
C===> PUT SCREEN & CELL COORDINATES BACK TO THEIR ORIGINAL VALUES
      DO 100 J=1,NAT
        IX(1,J)=IIX(1,J)
        IX(2,J)=IIX(2,J)
        DO 150 K=1,3
          XB(K,J)=OLDXB(K,J)
  150   CONTINUE
  100 CONTINUE
```

```
C===>  RESTORE SCALE FACTOR TO IT'S ORIGINAL VALUE
       SCALE=OSCALE
       RETURN
       END
C
       SUBROUTINE REMOVE(XB,NAT,JCON)
C===>  REMOVES AN ATOM FROM THE FRAGMENT
$INT2
       INTEGER*4 NAT,JCON(200)
       DIMENSION XB(4,200)
       COMMON /GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
      1 ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
     5 CALL WTEXT(20,20,2,2,0,1,32,'Remove which atom ?  ---->')
     6 WRITE(*,8)
     8 FORMAT(/' Atom number ? '\)
       READ(*,10)NATM
    10 FORMAT(I3)
C===>  CHECK WHETHER ATOM EXISTS
       IF(NATM.LT.1.OR.NATM.GT.200)THEN
         WRITE(*,12)
    12   FORMAT(/'Invalid atom number - try again')
         GOTO 6
       ENDIF
C===>  IS ATOM BONDED OR NON-BONDED ?
       IF(KL(NATM,1).EQ.0)THEN
         WRITE(*,14)NATM
    14   FORMAT('Atom',I4,' is not in fragment  -  try again')
         GOTO 6
       ENDIF
C===>  REMOVE PROMPT FOR ATOM NUMBER
       CALL WTEXT(20,20,2,2,0,0,0,'Remove which atom ?  ---->')
       DO 15 I=1,NAT
         IF(IFIX(XB(4,I)).EQ.NATM) GOTO 16
    15 CONTINUE
       GOTO 5
    16 IATM=I
       DO 20 J=1,4
         LCON(IATM,J)=0
         KL(NATM,J)=0
    20 CONTINUE
C===>  REMOVE ATOM FROM CONNECTIVITY TABLE
       DO 30 I=1,NAT
         DO 40 K=1,4
           IF(KL(I,K).NE.NATM)GOTO 40
           GOTO (21,22,23,24)K
    21     KL(I,1)=KL(I,2)
    22     KL(I,2)=KL(I,3)
    23     KL(I,3)=KL(I,4)
    24     KL(I,4)=0
C===>  REDUCE VALENCY NUMBER OF ANY ATOM ATTACHED TO THE ONE BEING REMOVED
           JCON(I)=JCON(I)-1
           GOTO 30
    40   CONTINUE
    30 CONTINUE
       DO 50 II=1,NAT
         DO 60 KK=1,4
           IF(LCON(II,KK).NE.IATM)GOTO 60
           GOTO (31,32,33,34)KK
    31     LCON(II,1)=LCON(II,2)
    32     LCON(II,2)=LCON(II,3)
    33     LCON(II,3)=LCON(II,4)
```

```
   34      LCON(II,4)=0
           GOTO 50
   60    CONTINUE
   50 CONTINUE
   65   CALL SHWFRG(NAT,XB)
        CALL EMENU(1)
C===>  REMOVE MORE ATOMS?
        CALL WTEXT(50,50,2,2,0,1,2,'Remove more atoms ?')
        CALL WTEXT(80,20,2,2,0,1,2,'* Yes')
        CALL WTEXT(180,20,2,2,0,1,2,'* No')
   70   CALL RDPEN(LX,LY)
        IF(LY.LT.10.OR.LY.GT.30)GOTO 70
        IF(LX.LT.70.OR.LX.GT.190)GOTO 70
        IF(LX.GT.90.AND.LX.LT.170)GOTO 70
C===>  YES
        IF(LX.LE.90.AND.LX.GE.70)  THEN
C===>  ERASE SCREEN PROMPT
          CALL WTEXT(50,50,2,2,0,0,0,'Remove more atoms ?')
          CALL WTEXT(80,20,2,2,0,0,0,'* Yes')
          CALL WTEXT(180,20,2,2,0,0,0,'* No')
          CALL NUMBER(XB,NAT)
          GOTO 5
C===>  NO
        ELSE IF(LX.LE.190.AND.LX.GE.170)  THEN
          CALL SHWFRG(NAT,XB)
          CALL EMENU(1)
        END IF
        RETURN
        END
C
        SUBROUTINE ENLRGE(NAT,XB)
$INT2
C===>  MAKES THE FRAGMENT BIGGER OR SMALLER AS REQUESTED
        LOGICAL PEN
        INTEGER*4 NAT
        DIMENSION XB(4,200)
        COMMON/GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
       1 ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
C===>  SET MAGNIFYING FACTOR
        RMAG=1.0
C===>  DISPLAY OPTIONS
        CALL GMENU3(1)
C===>  GET THE INITIAL INSTRUCTION
    45  CALL RDPEN(LX,LY)
        IF(LY.LT.10.OR.LY.GT.30)GOTO 45
        IF(LX.LT.30.OR.LX.GT.370)GOTO 45
        IF(LX.GT.50.AND.LX.LT.190)GOTO 45
        IF(LX.GT.210.AND.LX.LT.350)GOTO 45
C===>  STOP
        IF(LX.GT.350.AND.LX.LT.370)THEN
          GOTO 200
C===>  ENLARGE
        ELSE IF(LX.GT.30.AND.LX.LT.50)THEN
          RMAG=1.005
C===>  REDUCE
        ELSE IF(LX.GT.190.AND.LX.LT.210)THEN
          RMAG=0.995
        ENDIF
C===>  MAIN LOOP TO ADJUST SCREEN COORDINATES
   150  SCALE =SCALE*RMAG
        DO 160 I=1,NAT
```

```
        IX(1,I)=IFIX((XB(1,I)-XMIN)*SCALE)+100
        IX(2,I)=IFIX((XB(2,I)-YMIN)*SCALE)+50
  160 CONTINUE
      CALL SHNFRG(NAT,XB)
      CALL GMENU3(1)
C===> LOOK FOR FURTHER INSTRUCTION
      CALL RDLTPN(NX,NY,PEN)
      IF(.NOT.PEN)GOTO 150
C===> ADJUST NX & NY TO BE IN THE RANGE 0 TO 512
      MX=(NX-10)*2
      MY=(NY-1)*2
      IF(MY.LT.10.OR.MY.GT.30)GOTO 150
      IF(MX.LT.30.OR.MX.GT.370)GOTO 150
      IF(MX.GT.50.AND.MX.LT.190)GOTO 150
      IF(MX.GT.210.AND.MX.LT.350)GOTO 150
C===> STOP
      IF(MX.GT.350.AND.MX.LT.370)THEN
        GOTO 200
C===> ENLARGE
      ELSE IF(MX.GT.30.AND.MX.LT.50)THEN
        RMAG=1.005
C===> REDUCE
      ELSE IF(MX.GT.190.AND.MX.LT.210)THEN
        RMAG=0.995
      ENDIF
      GOTO 150
  200 CALL GMENU3(0)
  310 RETURN
      END
C
      SUBROUTINE GMENU3(I)
$INT2
C===> DISPLAYS MENU IN ENLARGE/REDUCE OPTION
      CALL WTEXT(40,20,2,2,0,I,0,'* Enlarge')
      CALL WTEXT(200,20,2,2,0,I,0,'* Reduce')
      CALL WTEXT(360,20,2,2,0,I,0,'* Halt')
      RETURN
      END
C
      SUBROUTINE COLOR(N,NCOLOR)
$INT2
C===> USE DIFFERENT COLOURS FOR DIFFERENT ATOM TYPES
      COMMON/GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
     1 ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
C===> CARBON - GREEN
      IF(NTYPE(N).GT.0.AND.NTYPE(N).LE.4) NCOLOR=51
C===> NITROGEN - BLUE
      IF(NTYPE(N).GT.4.AND.NTYPE(N).LE.9) NCOLOR=15
C===> OXYGEN - RED
      IF(NTYPE(N).GT.9.AND.NTYPE(N).LE.11) NCOLOR=60
C===> SULPHUR - YELLOW
      IF(NTYPE(N).GT.11.AND.NTYPE(N).LE.13) NCOLOR=48
C===> PHOSPHORUS - MAGENTA
      IF(NTYPE(N).EQ.14) NCOLOR=12
C===> HYDROGEN - WHITE
      IF(NTYPE(N).EQ.15) NCOLOR=64
C===> HALIDE - CYAN
      IF(NTYPE(N).GT.15.AND.NTYPE(N).LE.19) NCOLOR=3
C===> OTHER - ORANGE
      IF(NTYPE(N).EQ.20) NCOLOR=56
      RETURN
```

```
      END
C
      SUBROUTINE SPACE(XB,NAT)
$INT2
C===> DRAWS SPACE-FILLED MODEL OF FRAGMENT
      DIMENSION ATMRAD(20),RATMRD(20),XB(4,200),NORDER(200)
      INTEGER*4 NAT
      COMMON/GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
     1 ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
      DATA ATMRAD/0.77,0.77,0.77,0.77,0.74,0.74,0.74,0.74,0.74,0.74,
     1 0.74,1.04,1.04,1.10,0.37,0.72,0.99,1.14,1.33,0.75/
C===> ADJUST THE RADII TO THE SCALE BEING USED
      DO 20 I=1,20
        RATMRD(I)=ATMRAD(I)*SCALE
   20 CONTINUE
C===> GET ATOMS IN RIGHT ORDER FOR DRAWING
      CALL SORTZ(XB,NAT,NORDER)
C===> DRAW TO HIDDEN PAGE
      CALL HPAGE
      CALL VTYPE(0)
      DO 100 K=1,NAT
C===> CHECK IF ATOM IS BONDED
        J=NORDER(K)
        IF(LCON(J,1).EQ.0)GOTO 100
        JJ=IFIX(XB(4,J))
        NTYP=NTYPE(JJ)
C===> GET THE ATOMIC RADIUS
        NRAD=IFIX(RATMRD(NTYP))
        CALL SETDAT(0)
        CALL CIRCLE(IX(1,J),IX(2,J),NRAD+1,0)
C===> USE COLOUR APPROPRIATE TO THE ATOM TYPE
        CALL COLOR(JJ,NCLR)
        CALL SETDAT(NCLR)
        CALL SIRCLE(IX(1,J),IX(2,J),NRAD,1)
C===> DRAW SMALL WHITE DOT TO HIGHLIGHT CIRCLE
        NNRAD=NRAD/3
        CALL SETDAT(0)
        CALL SIRCLE(IX(1,J)+NNRAD,IX(2,J)+NNRAD,3,1)
  100 CONTINUE
C===> BRING HIDDEN PAGE INTO VIEW AND DISPLAY MENU
      CALL PAGE
      CALL GMENU4(1)
C===> SELECT RELEVANT OPTION
  200 CALL RDPEN(LX,LY)
      IF(LX.LT.0.OR.LX.GT.20) GOTO 200
      IF(LY.GT.500.OR.LY.LT.400) GOTO 200
C===> QUIT
      IF(LY.LE.500.AND.LY.GE.480)THEN
        CALL EXIT
C===> SCREEN DUMP
      ELSE IF(LY.LE.460.AND.LY.GE.440)THEN
        CALL GMENU4(0)
        CALL DUMP
        CALL GMENU4(1)
        GOTO 200
C===> RETURN TO MAIN MENU
      ELSE IF(LY.LE.420.AND.LY.GE.400) THEN
        CALL GMENU4(0)
        RETURN
      ENDIF
      GOTO 200
```

```
      END
C
      SUBROUTINE GMENU4(I)
$INT2
C===> MENU FOR SPACEFILLING OPTIONS
      CALL WTEXT(10,490,1,1,0,I,0,'* QUIT')
      CALL WTEXT(10,450,1,1,0,I,0,'* DUMP')
      CALL WTEXT(10,410,1,1,0,I,0,'* RETURN')
      RETURN
      END
C
      SUBROUTINE NAMES(XB,NAT)
$INT2
C===> INSERTS/REMOVES ATOM NAMES
      DIMENSION XB(4,200)
      CHARACTER*2 NAM,LIST(11)
      INTEGER*4 NAT
      COMMON/GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
     1 ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
      DATA LIST/'C ','N ','O ','S ','P ','H ','F ','Cl','Br','I ','  '/
      NP=1
      IF(MOD(NAMPNT,2).EQ.0) NP=0
      CALL SETDAT(0)
C===> GET ATOM NAMES
      DO 100 N=1,NAT
        IF(LCON(N,1).EQ.0)GOTO 100
        NN=IFIX(XB(4,N))
        IF(NTYPE(NN).GT.0.AND.NTYPE(NN).LE.4)NAM=LIST(1)
        IF(NTYPE(NN).GT.4.AND.NTYPE(NN).LE.9)NAM=LIST(2)
        IF(NTYPE(NN).GT.9.AND.NTYPE(NN).LE.11)NAM=LIST(3)
        IF(NTYPE(NN).GT.11.AND.NTYPE(NN).LE.13)NAM=LIST(4)
        IF(NTYPE(NN).EQ.14)NAM=LIST(5)
        IF(NTYPE(NN).EQ.15)NAM=LIST(6)
        IF(NTYPE(NN).EQ.16)NAM=LIST(7)
        IF(NTYPE(NN).EQ.17)NAM=LIST(8)
        IF(NTYPE(NN).EQ.18)NAM=LIST(9)
        IF(NTYPE(NN).EQ.19)NAM=LIST(10)
        IF(NTYPE(NN).EQ.20)NAM=LIST(11)
        CALL WTEXT(IX(1,N)+3,IX(2,N)+3,1,1,0,NP,0,NAM)
  100 CONTINUE
      RETURN
      END
C
      SUBROUTINE CONT
$INT2
C===> DISPLAY MESSAGE TO LET USER KNOW THAT THE PROGRAM IS CONTINUING
C===> AND THAT THE GRAPHICS OPTIONS ARE NO LONGER AVAILABLE
      COMMON/GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
     1 ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
C===> DRAW TO HIDDEN PAGE
      CALL HPAGE
      CALL WTEXT(45,250,4,4,0,1,35,'PROGRAM CONTINUES')
C===> SET PEN TO DRAW WHITE AGAIN
      CALL SETDAT(0)
C===> BRING HIDDEN PAGE INTO VIEW
      CALL PAGE
C===> CLEAR CONSOLE SCREEN
      CALL CLSCN
      RETURN
      END
C
```

```
      SUBROUTINE HPAGE
C===> DRAWS TO HIDDEN PAGE
      COMMON/GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
     1 ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
      IF(NPAGE.EQ.0)NUM1=2
      IF(NPAGE.EQ.1)NUM1=3
      NPAGE=NUM1
      CALL CLRPLN(NPAGE)
      CALL SETPAG(NPAGE)
      RETURN
      END
C
      SUBROUTINE PAGE
C===> SWAPS SCREEN PAGES OVER - BRINGS HIDDEN PLANE INTO VIEW
      COMMON/GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
     1 ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
      IF(NPAGE.EQ.2)NPAGE=1
      IF(NPAGE.EQ.3)NPAGE=0
      CALL SETPAG(NPAGE)
      RETURN
      END
C
      SUBROUTINE ROTATE(NAT,XB)
$INT2
      INTEGER*4 NAT
      LOGICAL LPEN
      DIMENSION ROTMAT(3,3),QVRG(3),XB(4,200),XXB(2,200)
      COMMON/GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
     1 ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
C===> GET 'CELL COORDINATES' UP TO DATE
      DO 3 J=1,NAT
        XB(1,J)=FLOAT(IX(1,J)-100)/SCALE+XMIN
        XB(2,J)=FLOAT(IX(2,J)-50)/SCALE+YMIN
    3 CONTINUE
C===> GET CENTRE OF FRAGMENT
      DO 5 I=1,3
        QVRG(I)=0.0
    5 CONTINUE
      NATM=0
      DO 8 I=1,NAT
        IF(LCON(I,1).EQ.0)GOTO 8
        NATM=NATM+1
        DO 10 J=1,3
          QVRG(J)=QVRG(J)+XB(J,I)
   10   CONTINUE
    8 CONTINUE
      DO 12 I=1,3
        QVRG(I)=QVRG(I)/FLOAT(NATM)
   12 CONTINUE
      XCENT=QVRG(1)
      YCENT=QVRG(2)
      ZCENT=QVRG(3)
C===> DISPLAY MENU
      CALL RMENU(1)
C===> GET DIRECTION OF ROTATION
      CALL WTEXT(40,500,2,1,0,1,48,'Choose + or - first')
   14 CALL RDPEN(LX,LY)
      IF(LY.GT.20)GOTO 14
C===> +VE ROTATION
      IF(LX.LE.270.AND.LX.GE.250) THEN
        THETA=1.0
```

```
C===> -VE ROTATION
      ELSE IF(LX.LE.370.AND.LX.GE.350) THEN
        THETA=-1.0
      ELSE
        GOTO 14
      ENDIF
      ANGL=THETA/57.2957795
C===> REMOVE MESSAGE FROM SCREEN
      CALL WTEXT(40,500,2,1,0,0,0,'Choose + or - first')
      CALL WTEXT(40,480,2,1,0,1,48,'Now choose axis for rotation')
   25 CALL RDPEN(NX,NY)
      IF(NY.GT.40.OR.NY.LT.20)GOTO 25
C===> STOP
      IF(NX.LE.350.AND.NX.GE.330) THEN
        CALL WTEXT(40,480,2,1,0,0,0,'Now choose axis for rotation')
        CALL RMENU(0)
        RETURN
C===> Z
      ELSE IF(NX.LE.250.AND.NX.GE.230) THEN
        XAXIS=0.0
        YAXIS=0.0
        ZAXIS=1.0
C===> Y
      ELSE IF(NX.LE.150.AND.NX.GE.130) THEN
        XAXIS=0.0
        YAXIS=1.0
        ZAXIS=0.0
C===> X
      ELSE IF(NX.LE.50.AND.NX.GE.30) THEN
        XAXIS=1.0
        YAXIS=0.0
        ZAXIS=0.0
      ELSE
        GOTO 25
      ENDIF
      CALL WTEXT(40,480,2,1,0,0,0,'Now choose axis for rotation')
C===> GET ROTATIONAL MATRIX
   24 CALL RMTRX(XAXIS,YAXIS,ZAXIS,ANGL,ROTMAT)
   26 DO 30 J=1,NAT
        XOLD=XB(1,J)-XCENT
        YOLD=XB(2,J)-YCENT
        ZOLD=XB(3,J)-ZCENT
C===> ADJUST ATOMIC COORDINATES
        XB(1,J)=XOLD*ROTMAT(1,1)+YOLD*ROTMAT(1,2)+ZOLD*ROTMAT(1,3)+XCENT
        XB(2,J)=XOLD*ROTMAT(2,1)+YOLD*ROTMAT(2,2)+ZOLD*ROTMAT(2,3)+YCENT
        XB(3,J)=XOLD*ROTMAT(3,1)+YOLD*ROTMAT(3,2)+ZOLD*ROTMAT(3,3)+ZCENT
        XXB(1,J)=(XB(1,J)-XMIN)*SCALE
        XXB(2,J)=(XB(2,J)-YMIN)*SCALE
C===> SCREEN COORDINATES
        IX(1,J)=IFIX(XXB(1,J))+100
        IX(2,J)=IFIX(XXB(2,J))+50
   30 CONTINUE
C===> DISPLAY FRAGMENT
      CALL SHWFRG(NAT,XB)
      CALL RMENU(1)
C===> NOW LOOK FOR FURTHER INSTRUCTION
      CALL RDLTPN(MX,MY,LPEN)
      IF(.NOT.LPEN) GOTO 26
C===> ADJUST MX & MY TO BE IN THE RANGE 0 TO 512
      KX=(MX-10)*2
      KY=(MY-1)*2
```

```
      IF(KY.GT.40)GOTO 26
      IF(KY.LE.20)THEN
C===> QUICKEN
        IF(KX.LE.70.AND.KX.GE.50)THEN
          AA=ABS(ANGL)
          IF(AA.LT.0.07)ANGL=1.05*ANGL
C===> SLOW DOWN
        ELSE IF (KX.LE.170.AND.KX.GE.150)THEN
          AA=ABS(ANGL)
          IF(AA.GT.0.01)ANGL=0.95*ANGL
C===> +VE ROTATION
        ELSE IF(KX.LE.270.AND.KX.GE.250)THEN
          ANGL=ABS(ANGL)
C===> -VE ROTATION
        ELSE IF(KX.LE.370.AND.KX.GE.350)THEN
          ANGL=-ABS(ANGL)
        ENDIF
        GOTO 24
C===> AXIS FOR ROTATION
      ELSE IF (KY.GE.20.AND.KY.LE.40) THEN
C===> STOP
        IF(KX.LE.350.AND.KX.GE.330) THEN
          CALL RMENU(0)
          RETURN
C===> Z
        ELSE IF(KX.LE.250.AND.KX.GE.230) THEN
          XAXIS=0.0
          YAXIS=0.0
          ZAXIS=1.0
C===> Y
        ELSE IF(KX.LE.150.AND.KX.GE.130) THEN
          XAXIS=0.0
          YAXIS=1.0
          ZAXIS=0.0
C===> X
        ELSE IF(KX.LE.50.AND.KX.GE.30) THEN
          XAXIS=1.0
          YAXIS=0.0
          ZAXIS=0.0
        ELSE
          GOTO 26
        ENDIF
C===> GET NEW MATRIX
        GOTO 24
      ENDIF
      GOTO 26
      RETURN
      END
C
      SUBROUTINE RMENU(I)
$INT2
C===> MENU FOR ROTATE OPTION
      CALL SETDAT(0)
      CALL WTEXT(40,30,2,2,0,I,0,'* X')
      CALL WTEXT(140,30,2,2,0,I,0,'* Y')
      CALL WTEXT(240,30,2,2,0,I,0,'* Z')
      CALL WTEXT(340,30,2,2,0,I,0,'* STOP')
      CALL WTEXT(60,10,2,2,0,I,0,'* Q')
      CALL WTEXT(160,10,2,2,0,I,0,'* S')
      CALL WTEXT(260,10,2,2,0,I,0,'* +VE')
      CALL WTEXT(360,10,2,2,0,I,0,'* -VE')
```

```
      RETURN
      END
      SUBROUTINE RMTRX(X,Y,Z,THETA,ROTMAT)
$INT2
      DIMENSION ROTMAT(3,3)
C===> CALCULATES ROTATIONAL MATRIX
C===> RETURN UNIT MATRIX IF THETA =0 OR X=Y=Z=0
      IF (ABS(THETA).LT.1.E-6) GOTO 10
      COST=COS(THETA)
      C=(1.-COST)
      S=SIN(THETA)
      SX=S*X
      SY=S*Y
      SZ=S*Z
C===> ROTATIONAL MATRIX - APROPRIATE TO THE AXIS OF ROTATION
      ROTMAT(1,1) =C*X+COST
      ROTMAT(2,2) =C*Y+COST
      ROTMAT(3,3) =C*Z+COST
      ROTMAT(1,2) =-SZ
      ROTMAT(2,1) =SZ
      ROTMAT(2,3) =-SX
      ROTMAT(3,2) =SX
      ROTMAT(1,3) =SY
      ROTMAT(3,1) =-SY
      RETURN
C===> MAKE UP UNIT MATRIX
10    DO 30 I=1,3
        DO 20 J=1,3
20        ROTMAT(I,J)=0.
30      ROTMAT(I,I)=1.
      RETURN
      END
C
      SUBROUTINE STRFIL(X,XB,NAT,CELL,JCON,NSYMOP)
$INT2
C===> STORES A FRAGMENT CONNECTIVITY ON A FILE THAT CAN BE USED IN THE
C===> STAND ALONE VERSION OF MITHGRAPH
      INTEGER*4 NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,NTAPEF,IK,
     1 NTAPEG,IH,ICH1,ICH2,ITLE,IR,ICALL,NRC,NCH,NREAD,IEND,NREF,MAXH,
     2 ISPFL,LK,IDEF,ICHK,ISYMP,LEVEL,IFLOW,KUSE,INXT,IRDY,NULL,IDIF,
     3 IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH,ICLCK,ICNT,NAT,ICON,JBOND,NOFRAG
     4 ,JCON,NSYMOP
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFL,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),ICLCK,ICNT
      COMMON/GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
     1 ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
      DIMENSION XB(4,200),JCON(200),X(4,200),CELL(6)
      CHARACTER*12 FILNAM
      LOGICAL*4 FILE
C===> GET THE BIGGEST ATOM NUMBER
      BIG=0.0
      DO 10 J=1,NAT
        IF(XB(4,J).GT.BIG) THEN
          BIG=XB(4,J)
        ENDIF
   10 CONTINUE
      NUMB=IFIX(BIG)
C===> NOW GET THE FILE NAME
```

```
   12 WRITE(*,15)
   15 FORMAT(/' Enter the name of the MITHGRAPH2 file - '\)
      READ(*,20)FILNAM
   20 FORMAT(A)
C===> CHECK IF FILE ALREADY EXISTS
      INQUIRE(FILE=FILNAM,EXIST=FILE)
      IF(FILE)THEN
        WRITE(*,25)
   25   FORMAT(/' There is a file of that name already on the disk'/
     1                 ' Do you wish to use it ? - (y/n)'\)
        CALL YN(NN)
        IF(NN.EQ.1)GOTO 12
        OPEN(FILE=FILNAM,UNIT=15,FORM=' FORMATTED')
      ELSE
        OPEN(FILE=FILNAM,UNIT=15,FORM=' FORMATTED',STATUS=' NEW')
      ENDIF
C===> NOW WRITE THE FILE
      WRITE(15,30)ITLE
   30 FORMAT(68A1)
      WRITE(15,31)(CELL(J),J=1,6)
   31 FORMAT(6F10.4)
      WRITE(15,35)NAT,NUMB,NSYMOP,SCALE,XMIN,YMIN,JKIND,JPRINT
   35 FORMAT(3I4,3F12.6)
      DO 40 J=1,NAT
       WRITE(15,45)(XB(L,J),L=1,4),(X(K,J),K=1,4),(IX(I,J),I=1,2)
   45  FORMAT(7F9.5,F9.3,2I4)
   40 CONTINUE
      DO 60 J=1,NUMB
       WRITE(15,65)JCON(J),NTYPE(J),(LCON(J,K),K=1,4),(KL(J,L),L=1,4)
   65  FORMAT(I10,9I6)
   60 CONTINUE
      RETURN
      END
C
      SUBROUTINE FILES(X,XB,NAT,CELL,JCON,NSYMOP)
$INT2
C===> DETERMINES WHICH TYPE OF FILE IS TO BE WRITTEN
      DIMENSION CELL(6),X(4,200),XB(4,200),JCON(200)
      INTEGER*4 NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,NTAPEF,IK,
     1 NTAPEG,IH,ICH1,ICH2,ITLE,IR,ICALL,NRC,NCH,NREAD,IEND,NREF,MAXH,
     2 ISPFL,LK,IDEF,ICHK,ISYMP,LEVEL,IFLOW,KUSE,INXT,IRDY,NULL,IDIF,
     3 IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH,ICLCK,ICNT,NAT,ICON,JBOND,NOFRAG
     4 ,JCON,NSYMOP
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IEND,NREF,PI,DTOR,MAXH,ISPFL,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),ICLCK,ICNT
      COMMON/GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
     1 ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
      CALL GMENU1(0)
C===> DISPLAY MENU OF FILE TYPES
      CALL FILMEN(1)
C===> READ LIGHT-PEN
   20 CALL RDPEN(NX,NY)
      IF(NY.GT.20.OR.NY.LT.0)GOTO 20
      IF(NX.GT.0.AND.NX.LE.20)THEN
C===> WRITE MITHGRAPH2 FILE
        CALL STRFIL(X,XB,NAT,CELL,JCON,NSYMOP)
      ELSE IF(NX.GE.190.AND.NX.LE.210)THEN
C===> CHEMMOD FILE
```

```
        CALL CHEMOD(NAT,JCON,X,CELL)
      ELSE IF(NX.GE.350.AND.NX.LE.370)THEN
C===> NEITHER
        GOTO 25
      ELSE
        GOTO 20
      ENDIF
   25 CALL FILMEN(0)
      CALL GMENU1(1)
      RETURN
      END
C
      SUBROUTINE FILMEN(I)
$INT2
C===> DISPLAY FILE OPTIONS
      CALL SETDAT(48)
      CALL WTEXT(40,500,2,1,0,I,0,'Enter type of file required')
C===> NOW WRITE IN BLUE
      CALL SETDAT(2)
      CALL WTEXT(10,10,2,2,0,I,0,'* MITHGRAPH2')
      CALL WTEXT(200,10,2,2,0,I,0,'* CHEMMOD')
      CALL WTEXT(360,10,2,2,0,I,0,'* NEITHER')
C===> SET PEN TO DRAW WHITE
      CALL SETDAT(0)
      RETURN
      END
C
      SUBROUTINE SORTZ(XB,NAT,NORDER)
$INT2
C===> SORTS THE POINTS WITH RESPECT TO THEIR Z-COORDINATE
C===> SPACE-FILLING ROUTINE THEN STARTS WITH ATOM WITH LOWEST Z-COORDINATE
      DIMENSION XB(4,200),NORDER(200),RXB(200)
      INTEGER*4 NAT
C===> INITIALISE ARRAYS
      DO 5 J=1,NAT
        RXB(J)=XB(3,J)
        NORDER(J)=J
    5 CONTINUE
      M=2
   10 NT=NAT/M
      IF(NT.EQ.0)NT=1
      IFIN=NAT-NT
   20 MARK=0
      DO 30 I=1,IFIN
        J=I+NT
C===> CHECK WHETHER TWO POINTS ARE IN THE CORRECT ORDER
        IF(RXB(I).LT.RXB(J)) GOTO 30
C===> SWAP OVER IF THEY ARE NOT
        X=RXB(J)
        RXB(J)=RXB(I)
        RXB(I)=X
        L=NORDER(J)
        NORDER(J)=NORDER(I)
        NORDER(I)=L
        MARK=1
   30 CONTINUE
      IF(MARK.EQ.1) GOTO 20
C===> ALL DONE ?
      IF(NT.EQ.1) GOTO 40
      M=2*M
      GOTO 10
```

```
   40 RETURN
      END
C
      SUBROUTINE UPDATE(XB, X, NAT, NSYMOP, JCON, NELSE)
$INT2
C===> PUTS NEW GROUP OF ATOMS ONTO THE MODEL FILE IN PREPARATION FOR
C===> FURTHER FOURIER RECYCLING
      CHARACTER*2 NAM(200),LIST(12)
      INTEGER*4 NAT,NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,NTAPEF,
     1 NTAPEG,IH,ICH1,ICH2,ITLE,IR,ICALL,NRC,NCH,NREAD,IDUM,NREF,MAXH,
     2 ISPFLG,LK,IDEF,ICHK,ISYMP,LEVEL,IFLOW,KUSE,INXT,IRDY,NULL,IDIF,
     3 IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH,ICLCK,ICNT,IS,NSYM,ICENT,LAT,
     4 IAPX,MLAT,NEQV,MCON,INT,KFRAG,JUNK,LFRAG,MFRAG,NOFRAG,NN,MFRN,IFMT,
     5 KMATCH,ITYPE,NOINT,MNPPF,NSYMOP,JCON,NELSE,NTAPEH
      COMMON/IO/NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(155),A(200),ICH1(10),ICH2(10),ITLE(68),IR(68),
     2 ICALL,NRC,NCH,NREAD,IDUM,NREF,PI,DTOR,MAXH,ISPFLG,ZERO,LK(32),
     3 IDEF,ICHK,ISYMP,LEVEL,IFLOW(25),KUSE(40),INXT,IRDY,RTOD,NULL,
     4 IDIF,IVDIF,KARLE,IHVY,IGPFL,NDIFF,ISPCH(30),ICLCK,ICNT
      COMMON /BLK2/ IS(2,3,24),TS(3,24),NSYM,ICENT,LAT,IAPX,
     1 CELL(6),T(3,3),DXMAX(3),TL(3,4),FS(3,3,24),MLAT,NEQV,MCON,INT,
     2 FOM,KFRAG(20),JUNK,LFRAG(20),MFRAG,NOFRAG,NN,MFRN,IFMT(20),
     3 KMATCH(20),ITYPE,NOINT,MNPPF
      COMMON/GRAPH/IX(2,200),LCON(200,4),NTYPE(200),NFRG,NUMPNT,KL(200,4)
     1 ,NPAGE,NFRAG,SCALE,IIX(2,200),NAMPNT,XMIN,YMIN,JKIND,JPRINT
      DIMENSION X(4,200),XB(4,200),JCON(200)
      DATA LIST/'C ','N ','O ','S ','P ','H ','F ','Cl','Br','I ','C ','  '/
      NTAPEH=16
C===> IF FRAGMENT HAS ARISEN FROM PHASE FILE CALCULATION THEN NEED TO GET THE
C===> ASSOCIATED MODEL FILE OF THE STRUCTURE
      IF(NELSE.EQ.1) THEN
        WRITE(*,15)
   15   FORMAT(/'Option not available in this mode')
        RETURN
      ENDIF
      REWIND NTAPEH
C===> GET ATOM NAMES
      DO 100 N=1,NAT
        IF(LCON(N,1).EQ.0)GOTO 100
        NMB=IFIX(XB(4,N))
        IF(NTYPE(NMB).GT.0.AND.NTYPE(NMB).LE.4)NAM(N)=LIST(1)
        IF(NTYPE(NMB).GT.4.AND.NTYPE(NMB).LE.9)NAM(N)=LIST(2)
        IF(NTYPE(NMB).GT.9.AND.NTYPE(NMB).LE.11)NAM(N)=LIST(3)
        IF(NTYPE(NMB).GT.11.AND.NTYPE(NMB).LE.13)NAM(N)=LIST(4)
        IF(NTYPE(NMB).EQ.14)NAM(N)=LIST(5)
        IF(NTYPE(NMB).EQ.15)NAM(N)=LIST(6)
        IF(NTYPE(NMB).EQ.16)NAM(N)=LIST(7)
        IF(NTYPE(NMB).EQ.17)NAM(N)=LIST(8)
        IF(NTYPE(NMB).EQ.18)NAM(N)=LIST(9)
        IF(NTYPE(NMB).EQ.19)NAM(N)=LIST(10)
        IF(NTYPE(NMB).EQ.20)NAM(N)=LIST(11)
  100 CONTINUE
C===> COUNT THE NUMBER OF LINES TO BE LEFT UNALTERED ON THE MODEL FILE
      NTIMES=3+NSYMOP
      DO 200 J=1,NTIMES
        READ(NTAPEH,9000)ICALL,IR
 9000   FORMAT(A4,68A1)
  200 CONTINUE
C===> NOW READY TO PUT NEW FRAGMENT'S ATOMIC COORDINATES ONTO MODEL FILE
      WRITE(NTAPEH,9001)
 9001 FORMAT('GROU')
```

```
C===>  PUT GROUP OF ATOMS ONTO THE MODEL FILE
       DO 300 L=1,NAT
         IF(LCON(L,1).EQ.0.OR.JCON(L).EQ.0)GOTO 300
         WRITE(NTAPEH,9002)NAM(L),(X(K,L),K=1,3)
 9002    FORMAT('ATOM ',A2,2X,3F9.5)
  300  CONTINUE
C====  PUT BLANKS ON TO THE END OF THE MODEL FILE TO AVOID OVERLAPPING WITH
C====  ANY PREVIOUS GROUPS
       DO 400 J=1,10
         WRITE(NTAPEH,9003)LIST(12)
 9003    FORMAT(A2)
  400  CONTINUE
       ENDFILE NTAPEH
       REWIND NTAPEH
       WRITE(*,9004)
 9004  FORMAT(/'     ---- New model file written ----'/)
       RETURN
       END
C
       SUBROUTINE PAUSE
C===>  ROUTINE TO CAUSE COMPUTER TO PAUSE FOR 3 SECONDS
       DIMENSION A(100),B(100),C(100)
       DO 5 K=1,60
         DO 10 J=1,100
           B(J)=FLOAT(J)*0.01
           C(J)=FLOAT(J)*0.02+12
           A(J)=B(J)/C(J)
   10    CONTINUE
    5  CONTINUE
       RETURN
       END
```

APPENDIX C

THE GRAPHDRVR LIBRARY OF GRAPHICS ROUTINES

APPENDIX C - The GRAPHDRVR library of graphics routines

Authors: M. Lindley & J. K. Tyler, University of Glasgow, 1986

1. READY - Check to see if processor is ready to receive new command

2. DRAWW - Set pen to draw white

3. DRAWB - Set pen to draw black

4. CLEARW - Clear screen to white

5. CLEARB - Clear screen to black

6. INITIL - Initialises processor and clears both memory planes to black

7. INITL1 - Initialise page 1 to black

8. INITL2 - Initialise page 2 to black

9. COLOUR(I) - Set colour register to desired colour

10. WCOLOR(I) - Set pen to draw white and load colour register with desired colour

11. BCOLOR(I) - Set pen to draw black and load color register with desired colour

12. SETPAG(I) - Sets page register where user sends:
    I = 0 for draw to page 1, 1 for draw to page 2,
    2 for draw to hidden page 2 & 3 for draw to hidden page 1

13. VTYPE(I) - Defines vector type
    I = 0 for solid, 1 is for dotted, 2 is for dashed and 3 is for dash-dotted lines

14. SETDAT(I) - Set data register, used to allow writing on top of previous write with out additive effect

15. CHKINT(I) - Checks for interrupt - if CTRL-C key pressed then value of true is returned in I

16. LOADXY(IX,IY) - Sets screen cursor to position given by IX,IY

17. POINT(IX,IY) - Sets screen cursor to position given by IX,IY and print point

18. RELVEC(IDX,IDY,ITYPE,IBW,ICOL) - Draws relative vectors where :
    IDX and IDY are signed offsets from current X,Y position -
    ITYPE is vector type
    IBW is pen - 0 is for black vectors and 1 is for white vectors
    ICOL is colour of vector

19. RVECUP(IDY) - Draws relative vector in +ve y-axis

20. RVECDN(IDY) - Draws relative vector in -ve y-axis

21. CLRPLN(I) - Clears memory plane to black where user sends:
I = 0 for plane 1, 1 for plane 2, 3 for hidden plane 2 and
3 for hidden plane 1.

22. WTEXT(IX,IY,ISCX,ISCY,IORIEN,IBW,ICOL,ICHAR)
Writes text on monitor where :
IX and IY are coordinates for start of text
ISCX and ISCY are X and Y scale factors (0-15)
IORIEN - orientation of text (0 = horizontal, 1 = slant
horizontal, 2 = vertical and 3 = slant vertical)
IBW - black or white text
ICOL - colour of text
ICHAR - base address of character string

23. WTEXT(IX,IY,ISCX,ISCY,IORIEN,IBW,ICOL,ICHAR)
Writes integer string on monitor where :
IX and IY are coordinates for start of text
ISCX and ISCY are X and Y scale factors (0-15)
IORIEN - orientation of text (0 = horizontal, 1 = slant
horizontal, 2 = vertical and 3 = slant vertical)
IBW - black or white text
ICOL - colour of text
ICHAR - base address of integer string

24. RDLTPN(IX,IY,LIT) - Read light pen where:
IX and IY return position of cursor after good read
LIT returns true if a good read occurs
IX and IY are 8 bit values and need to be adjusted to
be in range 0 - 512

25. CLRBLK(NPAG,IY) - Clear block of screen using 4x4 character
beginning on LHS of screen
NPAG - indicates plane - see CLRPLN for explanation
IY - Cursor position on y-axis

26. SIRCLE(IX,IY,IR,IBW) - Draws spacefilled circle where IX and IY
are centre of circle,IR is radius of circle and IBW indicates
black or white

27. CIRCLE(IX,IY,IR,IBW) - Draws circle of radius IR
IR can not be bigger than 255
IX and IY are centre of circle
IBW indicates black or white

28. HARDCP - Reads memory plane and writes display to printer using
parallel port. Set to use EPSON FX printer

29. IMC - Initialise the mouse channel

30. IGETBY() - Reads a byte from the optical mouse channel.
Note that no parameters are enclosed in the parentheses.

31. ITIME() - Gets clock in 64th of a second

32. CLSCN - Clears the console

33. SCDUMP - Reads memory plane and writes display to printer using
parallel port. Set to use NEC Pinwriter P3 printer

34. RMWON - Switches read modified write on & stops the screen cursor overwriting text.

35. RMWOFF - Switches read modified write off

APPENDIX D

PROGRAM LISTING OF CONVERT

APPENDIX D - Program listing of CONVERT

```
C===>
C===>     CCCC     000     M   M   V   V   EEEEE   RRRR     TTTTT
C===>    C        O   O    MM  M   V   V   E       R   R      T
C===>    C        O   O    M M M   V   V   EEEE    RRRR       T
C===>    C        O   O    M  MM    V V    E       R  R       T
C===>     CCC      000     M   M     V     EEEEE   R   R      T
C===> ========================================================
C===>
C===> A program to convert the binary phase file or maps file
C===> produced by MITHRIL into an ASCII file containing the same
C===> information. The ASCII file can then be copied onto the
C===> STRIDE computer.
C===>
C===> Author:  S.R.Brown,  University of Glasgow, June 1987
C===>
C===>
C===> FILES:
C===>          NTAPED = FILE FROM MITHRIL
C===>          NOUT   = NEW ASCII FILE TO TRANFER TO THE STRIDE
C===>
      INTEGER FLAG(2250)
      DIMENSION ITLE(68),TS(3,24),IS(2,3,24),CX(6),MAXHKL(3),
     1 IHKL(800),E(800),NH(10),NO(10),CVR(10),VVR(10),CFOM(2250),
     2 IPHAZ(800),LINE( 12),CLINE(6),CELL(6),X(4,200),LMAP(200)
      DIMENSION IMAP(50000)
      LOGICAL*4 FILE
      CHARACTER*30 PFILE,NFILE,ANSWER*1
      DATA FLAG/2250*0/
      DATA NTAPED,NOUT,IPFILE,IMFILE/11,12,0,0/
C===>
C===> ----------- WARNING !   UNIX SPECIFIC -----------------
C===>
C===> SET FP TO NO-PIPE MODE (TO YIELD MORE INFORMATION WHEN PROGRAM
C===> CRASH OCCURS)
c      INCLUDE "/usr/include/fpedefs.f"
c      call fpsetxflags (NOPIPEMODE)
C===> Clear the screen
      CALL SYSTEM("clear")
C===>
C===> ------------ END OF WARNING -------------------------
C===>
      WRITE(*,9000)
 9000 FORMAT(7X,' ********************************************'/
     1       7X,' * MASSCOMP-STRIDE  FILE CONVERSION PROGRAM  *'/
     #       7X,' * ------------------------------------   *'/
     2       7X,' ********************************************'/)
    2 WRITE(*,3)
    3 FORMAT(//' Which type of file do you wish to convert ?'//
     1 ' M - Maps file'/' P - Phase file'/' Q - quit'//
     2 ' Type m, p or q: ',$)
      READ(*,5)ANSWER
    5 FORMAT(A)
C===>
C===> WHICH OPTION?
```

```
C===> PHASE FILE CONVERSION
C===>
      IF(ANSWER.EQ.'P'.OR.ANSWER.EQ.'p')THEN
        IPFILE=1
    9   WRITE(*,10)
   10   FORMAT(//' Enter the name of the MITHRIL phase file - ',$)
        READ(*,20)PFILE
   20   FORMAT(A)
C===>
C===> OPEN FILE IF OK TO DO SO
C===>
        CALL CHECK(NTAPED,PFILE,NYY)
        IF(NYY.EQ.1)GOTO 9
        GOTO 15
C===>
C===> MAPS FILE CONVERSION REQUESTED
C===>
      ELSE IF(ANSWER.EQ.'M'.OR.ANSWER.EQ.'m')THEN
        IMFILE=1
   12   WRITE(*,11)
   11   FORMAT(//' Enter the name of the MITHRIL maps file - ',$)
        READ(*,21)PFILE
   21   FORMAT(A)
C===>
C===> CHECK THAT IT IS OK TO OPEN FILE
C===>
        CALL CHECK(NTAPED,PFILE,NYY)
        IF(NYY.EQ.1)GOTO 12
        GOTO 15
      ELSE IF(ANSWER.EQ.'Q'.OR.ANSWER.EQ.'q')THEN
        STOP
      ENDIF
C===>
C===> UNRECOGNISABLE INPUT
C===>
      WRITE(*,8)
    8 FORMAT(//' ***** Invalid input - please try again *****')
      GOTO 2
C===>
C===> Now get the new file
C===>
   15 WRITE(*,40)
   40 FORMAT(/' Enter the name of the STRIDE file - ',$)
      READ(*,50)NFILE
   50 FORMAT(A)
      OPEN(FILE=NFILE,UNIT=NOUT,STATUS='NEW')
C===>
C===> PHASE FILE CONVERSION
C===>
      IF(IPFILE.EQ.1) THEN
        REWIND NTAPED
        READ(NTAPED) MARK,ITLE,ICENT,LAT,NSYM,TS,IS,MAXHKL,
     1  (CX(J),J=1,6),NAT,NUMB,(IHKL(I),E(I),I=1,NUMB) ,NN,NO,CVR,
     2  VVR,NK,IAPX,ISYMP,IVDIF,IDIF
C===>
C===> CONVERT TO ASCII
C===>
        WRITE(NOUT,9001) ITLE,ICENT,LAT,NSYM
```

```
 9001    FORMAT(68A1/3I5)
         WRITE(*,8001) ITLE
 8001    FORMAT(1H ,68A1/'0Symmetry operations are:'/)
         DO 100 J=1,NSYM
           WRITE(NOUT,9002) (TS(I,J),(IS(K,I,J),K=1,2),I=1,3)
 9002      FORMAT( 3(F10.5,2I5))
           WRITE(*,8002) (TS(I,J),(IS(K,I,J),K=1,2),I=1,3)
 8002      FORMAT( 1H ,3(F10.5,2I5))
  100    CONTINUE
         WRITE(NOUT,9003) MAXHKL,CX,NAT,NUMB
 9003    FORMAT(3I5/6F10.5/2I5)
         WRITE(*,8003) NUMB
 8003    FORMAT(1H0,'There are',I5,' E-magnitudes'/)
         II=0
         DO 110 J=1,NUMB
           II=II+1
           LINE(II)=IHKL(J)
           CLINE(II)=E(J)
           IF(II.LT.3) GOTO 110
           WRITE(NOUT,9004) (LINE(L),CLINE(L),L=1,3)
 9004      FORMAT( I12,F10.5,I12,F10.5,I12,F10.5)
           DO 101 L=1,3
             CLINE(L)=0.0
  101      LINE(L)=0
           II=0
  110    CONTINUE
         IF(II.GT.0) WRITE(NOUT,9004) (LINE(L),CLINE(L),L=1, 3)
         WRITE(NOUT,9005) NN,NO,CVR,VVR,NK,IAPX,ISYMP,IVDIF,IDIF
 9005    FORMAT( 10I6/10I6/10F7.3/10F7.3/5I5)
C===>
C===> FIRST RECORD IS NOW READ FROM NTAPED
C===> FIND OUT HOW MANY PHASE SETS ARE ON THE FILE
C===>
  150    READ(NTAPED) I
         IF(I.GE.0) GOTO 150
C===>
C===> END OF PHASE SETS
C===>
         READ(NTAPED) NSET,(CFOM(J),J=1,NSET)
         WRITE(*,8007) NSET
         WRITE(NOUT,9005) NSET
 8007    FORMAT(1H ,'There are',I5,' phase sets on the original file'
     1/)
C===>
C===> NOW THE PHASE SETS TO BE CONVERTED:
C===>
 200     WRITE(*,8030)
 8030    FORMAT('0Which phase set do you wish to transfer ?'/
     1           '0(A zero indicates no further transfers) '/)
         READ(*,*) IT
         IF(IT.LE.0) GOTO 250
         REWIND NTAPED
         READ(NTAPED)
         DO 119 I=1,NSET
           READ(NTAPED)NNO,ABS,PSI,RESID,RN4,RN5,
     1                          (IPHAZ(IJ),IJ=1,NUMB)
           IF(NNO.NE.IT) GOTO 119
           WRITE(NOUT,9006)NNO,ABS,PSI,RESID,RN4,RN5,CFOM(NNO)
 9006      FORMAT(I5,6F10.5)
```

```
          FLAG(NNO)=1
          WRITE(NOUT,9007)(IPHAZ(IJ),IJ=1,NUMB)
 9007     FORMAT(12I6)
          WRITE(*,8004) NNO
 8004     FORMAT('0***** Phase set number',I5,' transferred *****')
          GOTO 200
  119   CONTINUE
C===>
C===> COULD NOT FIND PHASE SET
C===>
        WRITE(*,8050) IT
 8050   FORMAT('0Could not find the required phase set (no.',I5,')')
        GOTO 200
C===>
C===> NOW TRANSFER THE CFOM VALUES
C===>
  250   NNO=-1
        DO 300 J=1,NSET
          IF(FLAG(J  ).EQ.0) CFOM(J  )=-99.9
  300   CONTINUE
        ABS=-1.0
        WRITE(NOUT,9006) NNO,ABS,ABS,ABS,ABS,ABS,ABS
        WRITE(NOUT,9051) (CFOM(J),J=1,NSET)
 9051   FORMAT(10F7.2)
C===>
C===> MAPS FILE CONVERSION
C===>
      ELSE IF (IMFILE.EQ.1)THEN
        REWIND NTAPED
        NUM=1
        READ(NTAPED)CELL,NEQV,ICENT,LAT,TS,IS,IDIF,IVDIF,MLAT
        WRITE(NOUT,2000)CELL,NEQV,ICENT,LAT,IDIF,IVDIF,MLAT
 2000   FORMAT(6F9.4,6I5)
        DO 260 J=1,24
          DO 270 L=1,3
            WRITE(NOUT,2001)IS(1,L,J),IS(2,L,J),TS(L,J)
 2001       FORMAT(2I6,F9.5)
  270     CONTINUE
  260   CONTINUE
        READ(NTAPED)NX,NZ,NY
        WRITE(NOUT,2005)NX,NZ,NY
 2005   FORMAT(3I6)
C===>
C===> NEED TO GET RMIN AND RMAX - POSITION NTAPED ACCORDINGLY
C===>
        DO 295 K=1,NY
          READ(NTAPED)
  295   CONTINUE
        READ(NTAPED)RMIN,RMAX
C===>
C===> CALCULATE A SCALE FACTOR
C===>
        ABSMIN=ABS(RMIN)
        IF(ABSMIN.GT.RMAX) THEN
          USCF=RMIN
        ELSE
          USCF=RMAX
        ENDIF
```

```
        SCF=127.0/USCF
        REWIND NTAPED
        READ(NTAPED)
        READ(NTAPED)
C===>
C===> READ THE MAP POINTS OFF NTAPED
C===>
        MIN=1
        MAX=(NX+2)*NZ
        DO 305 K=1,NY
          READ(NTAPED)(IMAP(LL),LL=MIN,MAX)
C===>
C===> IGNORE THE TWO "REPEATED" X-SECTIONS
C===>
          NUMB=0
          DO 400 J=1,NZ
            DO 330 KK=1,NX
              NUMB=NUMB+1
C===>
C===> SCALE ALL POINTS BETWEEN -127 AND +127
C===>
              LMAP(KK)=NINT(FLOAT(IMAP(NUMB))*SCF)
  330       CONTINUE
C===>
C===> SKIP THE TWO "REPEATS"
C===>
            NUMB=NUMB+2
C===>
C===> WRITE MAP POINTS TO OUTPUT FILE
C===>
            WRITE(NOUT,2002)(LMAP(NN),NN=1,NX)
 2002       FORMAT(16I4)
  400     CONTINUE
  305     CONTINUE
          NUM=NX*NY*NZ
  310     WRITE(*,7001)NUM
 7001     FORMAT(//' There are',I8,' map points'/)
          READ(NTAPED)RMIN,RMAX
          WRITE(NOUT,2003)RMIN,RMAX
 2003     FORMAT(2F9.2)
          READ(NTAPED)NAT,NSYM,((X(I,J),I=1,4),J=1,NAT)
          WRITE(NOUT,7002)NAT,NSYM
 7002     FORMAT(2I4)
          DO 315 JJ=1,NAT
            WRITE(NOUT,7003)(X(L,JJ),L=1,4)
 7003       FORMAT(4F10.4)
  315     CONTINUE
        ENDIF
C===>
C===> ALL DONE
C===>
        ENDFILE NOUT
        REWIND NOUT
        WRITE(*,8010)
 8010 FORMAT(1H0,'********* All done *********'/)
        STOP
        END
C
```

```
      SUBROUTINE CHECK( NTAPED, PFILE, NY)
C===>
C===> Check that the phase file exists
C===>
      LOGICAL*4 FILE
      CHARACTER*30 PFILE
      INQUIRE( FILE=PFILE, EXIST=FILE)
      IF( FILE) THEN
        OPEN( FILE=PFILE, UNIT=NTAPED, STATUS=' OLD' , FORM=' UNFORMATTED' )
        NY=0
      ELSE
C===>
C===> FILE DOES NOT EXIST
C===>
        NY=1
      ENDIF
      RETURN
      END
```