NEW TECHNIQUES FOR DIRECT METHODS IN X-RAY CRYSTALLOGRAPHY

BY

ARLENE DONALD

A thesis submitted to the University of Glasgow for the degree of

Doctor of Philosophy in the Faculty of Science

Chemistry Department

August 1995

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

This Thesis is Dedicated to

my parents

Tom and Mary Donald

ACKNOWLEDGEMENTS

I would like to gratefully acknowledge the guidance, support and patience of my supervisor Dr. Chris Gilmore. His advice and direction was pertinent to the research, and for the completion of this thesis.

Thanks must also be given to staff and colleagues of the Chemistry Department, past and present. In particular thanks to Lesley for sharing an office with me and also to Chris and William for their support and friendship. My sincere gratitude goes to Stuart Mackay for all his invaluable advice, time and understanding (and the mints).

I also wish to acknowledge Martin for his continual support and for listening to and enduring all of my complaints.

Finally thanks are due to MAC Science for funding this research, and also for funding attendance at conferences in the U.K and America.

ABSTRACT

The research presented in this thesis involves the development of both new and existing techniques in direct methods. All of the techniques investigated were integrated into a test version of the direct methods program MITHRIL94 and then tested on structures that are known to be difficult to solve. For every technique the results produced were compared to those obtained from the previous version of MITHRIL94.

Chapter 1 is an introduction to direct methods - the underlying principles of the method and a comprehensive summary of the techniques employed, some of which are discussed and expanded in later chapters. This chapter also contains a section on the limitations of direct methods and a flowchart of the modules which are present in the MITHRIL94 version of the program.

The reliability of the phase relationships used in direct methods is an essential step in obtaining the correct structure solution. Chapter 2 discusses the measure of reliability of the triplets (κ) and quartets (κ_{eq}) that exist in MITHRIL94. Since the use of κ_{eq} is relatively slow, this chapter investigates other methods of measuring the reliability of the quartets that are quicker but still produce an accurate measure. The two methods investigated were derived from the variable B, used in the conditional probability equation $P_{1/7}$. These two methods, B_1 and B_2, differed in their treatment of the cross terms. Comparison of these methods with κ_{eq} was achieved by plotting the average phase error against increasing reliability for each of κ_{eq}, B_1 and B_2, with κ also being plotted as a standard. To test the methods fully, negative and positive quartets were treated separately. This chapter provides details of the implementation of these methods into a test version of MITHRIL94 and presents the results for some of the structures tested. The results showed that all three methods provided an accurate measure of reliability, and since the use of B_1 increased the speed of the program by an approximate factor of 10 it was used to measure the reliability of the negative quartets in subsequent investigations.

The third chapter details the principles of optimisation techniques, namely that of simulated annealing and discusses how this technique is applied to crystallography in the form of phase annealing. This chapter includes a detailed account of the

implementation of phase annealing into MITHRIL94. Centric and acentric phases are treated differently during phase annealing and flowcharts for these processes are provided. The phase annealing algorithm was integrated into three of the tangent refinement modules in MITHRIL94 - FASTAN and SWTR, the principles of which are discussed in chapter one, and X-Y which is discussed in this chapter. The phase annealing code was tested on structures from the Sheldrick database of difficult structures. The results for each tangent refinement module employing phase annealing are displayed in separate tables. These show that phase annealing in MITHRIL94 was a definite success for FASTAN and SWTR, especially SWTR where phase annealing produced a dramatic improvement in results compared to the original module.

Phase permutation involves the assignment of values to the phases in the starting set of reflections. Various methods of phase permutation exist including those of magic integer and random phasing, both of which are discussed in chapter 1. Chapter 4 discusses the use of error correcting codes as a phase permutation technique. The first part of this chapter details the background theory of error correcting codes, discussing their relevance to experimental design and the properties required of the error correcting codes for efficient phase permutation. Two error correcting codes were investigated - the Hadamard code and the binary [24,12,8] Golay code. Detailed descriptions of how they were used to permute phases is provided. The error correcting codes were incorporated into a trial version of MITHRIL94 and used with the FASTAN and SWTR tangent refinement modules. The results for each code, employed with each module, are displayed. Again the results are provided from tests on the Sheldrick database of difficult structures and are compared with the results produced using a previous version of MITHRIL94. The results show that the Golay and Hadamard error correcting codes are capable of producing structure solutions and are therefore viable phase permutation techniques.

Since the results gained from error correcting codes and phase annealing were successful the next obvious step was to combine these two techniques. Chapter 5 discusses how these techniques were combined. Again this method was tested with the FASTAN and SWTR modules. The results produced from this investigation were very interesting; with the annealing-code combination in some cases literally working together and in other cases appearing to work against one another.

Chapter 6 discussed the implementation of phase annealing into the graphical user interface version of MITHRIL94, which is contained in the commercial computer program CRYSTAN 6.3. A summary of the facilities in CRYSTAN 6.3 is presented, along with a description of how to initiate phase annealing and the options available within the module.

Appendix A contains listings of subroutines in MITHRIL94 for phase annealing.

Appendix B contains subroutine listings in MITHRIL94 which implement error correcting codes.

CONTENTS

FIGURES

CHAPTER 1. INTRODUCTION TO DIRECT METHODS

CHAPTER 2. ESTIMATING THE RELIABILITY OF THE QUARTETS

TABLES

CHAPTER 1

INTRODUCTION TO DIRECT METHODS

1.0 THE PHASE PROBLEM

When X-rays irradiate a crystal, diffraction occurs producing the characteristic diffraction pattern. The individual diffracted beams are known as reflections and the reflecting planes within the crystal are defined by the Miller indices (h,k,l). The diffraction pattern contains vital information which is used to determine the structure of the crystal.

The following information can be acquired from the pattern:

- By considering the symmetry of the x-ray diffraction pattern, certain physical properties of the crystal and the possible presence of systematic absences, information about the symmetry of the crystal is obtained.
- The geometry of the diffraction pattern is used to determine the shape and size of the unit cell.
- Information regarding the relative atomic positions is obtained by measuring the intensities of the reflections.

The intensity of a reflection is dependent on the positions of the atoms in the unit cell and the scattering power of the atoms and can be measured from the diffraction pattern. They are an important source of information about the structure, providing a direct link to the structure factor amplitudes. The relationship is shown by EQ 1.0.1

$$|F_{\underline{h}}| = \sqrt{I_{\underline{h}}/Lp}$$

(EQ 1.0.1)

where

Lp is the combined polarisation factor and Lorentz factor.

$I_{\underline{h}}$ is the measured intensity of reflection $\underline{h}$

$|F_{\underline{h}}|$ is the relative structure factor amplitude of reflection $\underline{h}$.

Thus if the intensities are known the relative structure factor amplitudes can be determined. For structure solution the structure factor for each reflection must be known. These are complex numbers and are defined as:

$$F_{\underline{h}} = |F_{\underline{h}}| exp\,(i\varphi_{\underline{h}})$$

(EQ 1.0.2)

where

$F_{\underline{h}}$ is the structure factor of reflection $\underline{h}$

$\varphi_{\underline{h}}$ is the phase of reflection $\underline{h}$

The electron density of the structure (therefore information about relative atomic positions) is related to the structure factors by a Fourier transform:

$$\rho\,(\underline{x}) = \frac{1}{V}\sum_{\underline{h}} |F_{\underline{h}}| \cos\,[2\pi \underline{h} \cdot \underline{x} - \varphi_{\underline{h}}]$$

(EQ 1.0.3)

where

$\underline{x}$ is the vector for a point in the unit cell

V is the volume of the unit cell

The summation is over all the observed reflections. The structure factors are the coefficients for the Fourier synthesis and the summation of terms will reconstruct the electron density of the crystal. Apart from the phase, every term in both EQ 1.0.2 and EQ 1.0.3 can be determined experimentally. Unfortunately the phase cannot be established by physical measurements and it must be calculated. Incorrect phases will lead to incorrect electron density and thus an incorrect structure, so the calculation of the phase values is extremely important. This is the Crystallographic Phase Problem.

1.1 Solutions to the Phase Problem

To overcome the phase problem several methods have been devised and are currently used. A brief summary of some of the methods follow.

(i) Patterson Methods (Patterson, 1934) - The Patterson function is a Fourier series which is derived directly from the experimental data. This method is important as it revealed that the structure of a crystal could be derived from the initial data. The function is shown by EQ. 1.1.1

$$P(\underline{u}) = \frac{1}{V}\sum_{\underline{h}} |F_{\underline{h}}|^2 exp\,[-2\pi i \underline{h} \cdot \underline{u}\,]$$

(EQ 1.1.1)

It is the interatomic vectors, not the atomic positions that are found directly by this equation. It is useful in two situations:

- When heavy atoms are present in the structure. The maxima of the Patterson can be used to find the position of the heavy atoms. The rest of the structure can then be found by Fourier techniques. This is known as the heavy atom method.
- When there exists a known arrangement of atoms displaying a characteristic vector pattern. This is known as the molecular replacement search technique. Again the rest of structure is found by refinement.

Drawbacks to the Patterson map are (i) It can really only be successful if a known arrangement of atoms or heavy atoms exist in the structure and (ii) As the structure increases in size the Patterson map decreases in efficiency.

(ii) Isomorphous replacement - this also uses a Patterson map. This technique involves the addition of a heavy atom to a crystal without changing the space group or unit cell in any way. Again once the heavy atom has been located an attempt is made to find the rest of the structure by the use of phase circles. A disadvantage is that it can be difficult to introduce the heavy atom into the structure without changing the crystal or molecular structure.

(iii) Direct methods - so called because a mathematical attempt is made to determine

the phases directly from the structure factor amplitudes. The existence of heavy atoms in the structure or knowledge of part of the structure are not required by direct methods. Being mathematical procedures, direct methods are readily applied as computer programs.

The remaining chapters will explain the development and integration of new techniques into the direct methods program MITHRIL94.

2.0 UNDERLYING THEORY OF DIRECT METHODS

Phases and structure factor amplitudes are not independent of one another and are linked by a degree of knowledge about the electron density. The structure factors are related to the electron density by a Fourier transform so any constraints on the electron density will automatically be constraints on the structure factors. The initial developments in direct methods are better described by considering some of these constraints. One constraint is that obviously the electron density must always be positive at every point within the crystal. This constraint was first applied to structure factor relationships by Harker and Kasper (Harker & Kasper, 1948). For a centrosymmetric space group the phases are restricted to 0 and π, and we can think of the phase $\varphi_{\underline{h}}$ as a sign $s_{\underline{h}}$ of the structure factor, where $\varphi_{\underline{h}} = 0$ corresponds to $s_{\underline{h}} = +1$ and $\varphi_{\underline{h}} = \pi$ corresponds to $s_{\underline{h}} = -1$. Harker and Kasper used the Cauchy -Schwartz inequality to prove that for a centrosymmetric crystal, if $|F_{\underline{h}}|$ and $|F_{2\underline{h}}|$ are both strong then $|F_{2\underline{h}}|$ is probably positive. This inequality is only useful when there is only a small number of atoms in the structure and if the crystal is centrosymmetric. Karle and Hauptman developed determinant inequalities (Karle & Hauptman, 1950) that would give a function a non-negative result everywhere for a Fourier summation. The inequality relationships could result in a reduction of the range of values for a phase only when the intensity of the phase was large (apart from the centrosymmetric case).

It is the development of the probability relationships between the structure factors that is the basis for direct methods.The following are applicable for centrosymmetric structures: Cochran (1952) proposed a constraint on the entire electron density on the unit cell stating that a correct distribution would have some near zero regions with the

electron density large at atomic positions. This eventually led Cochran to the triple product sign relationship (TPSR):

$$s(h)\, s(k)\, s(h+k) \approx 1$$

(EQ 2.0.1)

where $\approx$ means probably equals.

Zachariasen (1952) extended this relationship, where the known signs give an indication of the unknown sign:

$$s(h) \approx s\left[\sum_{k} s(k)\, s(h-k)\right]$$

(EQ 2.0.2)

In the same issue of *Acta Crystallographica* that Cochran and Zachariasen proposed their work the Sayre equation (Sayre, 1952) was introduced:

$$F_{\underline{h}} = \frac{\Theta_{\underline{h}}}{V} \sum_{\underline{k}} F_{\underline{k}} F_{\underline{h}-\underline{k}}$$

(EQ 2.0.3)

where

$$\Theta_{\underline{h}} = \frac{f_{\underline{h}}}{\gamma_{\underline{h}}}$$

(EQ 2.0.4)

$f_{\underline{h}}$ is the scattering factor for each atom

$\gamma_{\underline{h}}$ is the scattering factor of the squared atom and depends only on |h|.

V is the volume of the unit cell.

This equation links the structure factors if the structure consists of equal resolved atoms. If the atoms are to be discrete the structure factors will not be affected by the atomic shape and will all be equal. In order to take advantage of this fact, the structure factors must be expressed as the normalised structure factors $E_{\underline{h}}$.

3.0 NORMALISATION

3.1 The Normalised Structure Factor $E_{\underline{h}}$

The first stage of any direct methods procedure is the normalisation of the structure factors. It is important to note here that the phase of the reflection will not change during normalisation. The magnitude of the normalised structure factors $|E_{\underline{h}}|$ influence every step in direct methods so every factor that could possibly affect the value of $E_{\underline{h}}$ must be closely considered. Measured structure factors will decrease with an increasing diffraction angle since both the atomic vibration and the atom size exert an effect on X-ray diffraction. The size of the vibration is dependent on the atomic mass, the strength of the forces or covalent bonds holding it in place and the temperature. Thermal vibration obviously increases with temperature. As the atom vibrates the electron cloud spreads over a larger area with the consequence that the cloud will have a lower density, which in turn reduces the intensity of the diffraction of x-rays.

These effects must be corrected allowing the structure factors to correspond to point atoms at rest. The normalised structure factors $E_{\underline{h}}$ are defined by EQ. 3.1.1

$$|E_{\underline{h}}|^2 = \frac{K|F_{\underline{h}}|^2}{\varepsilon \sum_{j=1}^{N} f_j^2 e^{-2B\sin^2\theta/\lambda^2}}$$

(EQ 3.1.1)

where

f_j is the scattering factor of the atom and λ is the wavelength of radiation used.

ε is the epsilon factor which takes into account that some reflections may have an average intensity greater than that for general reflections. It is a consequence of point group symmetry.

K is the scale factor which places $F_{\underline{h}}$ on an absolute scale.

B is the isotropic temperature factor defined as:

$$B = 8\pi^2 \overline{U}_j^2$$

(EQ 3.1.2)

where U_j^2 is the mean squared amplitude of atomic vibration.

The observed intensities I_{rel} (corrected for Lorentz and polarisation factors) are related to the intensities I_{abs}, calculated for atoms at rest:

$$I_{rel} = K \cdot I_{abs} e^{-2B\sin^2\theta/\lambda^2}$$

(EQ 3.1.3)

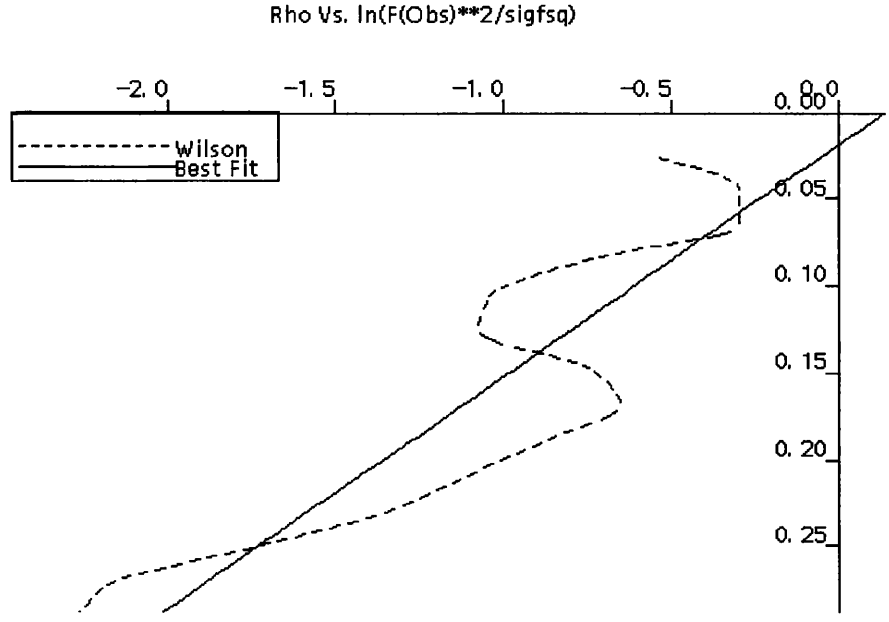
Both B and K can be obtained from a Wilson plot (Wilson, 1942). Rearranging EQ 3.1.3 and multiplying both sides by the natural logarithm gives:

$$ln\left(I_{rel}/\sum f_j^2\right) = lnK - 2B\sin^2\theta/\lambda^2$$

(EQ 3.1.4)

When the left hand side of EQ 3.1.4 is plotted against $(\sin^2\theta)/\lambda^2$ an approximate straight line is obtained. A least squares line is calculated and B can be obtained from the slope and K from the intercept at $\theta = 0$ degrees. An example of a Wilson plot for the centrosymmetric structure diamantane is shown below with K=0.8752 (σ=0.1299) and B=3.7286 (σ=0.7222)

FIGURE 1. The Wilson plot



If the points plotted deviate greatly from a straight line then the K-curve method introduced by Karle and Hauptman (Karle & Hauptman, 1953) is used to determine K and B.

4.0 ORIGIN AND ENANTIOMORPH DEFINITION

4.1 Origin Definition

When a crystal structure has its atomic positions specified with respect to a frame of reference, then that crystal structure is said to be fully defined in real space. In order to achieve this its origin must be defined. In reciprocal space this is done by assigning phases to a small number of $|E_{\underline{h}}|$. If the origin is shifted in real space then the phases of the corresponding reflections in reciprocal space will also be altered as the two are related by a Fourier transform. Thus setting the origin creates a corresponding pattern of relative phases on the diffraction pattern.The structure factor is defined in terms of atomic coordinates:

$$F_{\underline{h}} = \sum_j f_j exp\,(2\pi i \underline{h} \cdot \underline{x}_j)$$

(EQ 4.1.1)

If the origin is moved a distance of Δx the new structure factor can be defined as:

$$F'_h = \sum_j f_j exp\,(2\pi i \underline{h} \cdot (\underline{x}_j - \Delta \underline{x}))$$

(EQ 4.1.2)

Thus if the origin is changed the corresponding shift in phase is:

$$2\pi\varphi'_{\underline{h}} = 2\pi\varphi_{\underline{h}} - 2\pi\,(\underline{h} \cdot \Delta \underline{x})$$

(EQ 4.1.3)

where

$\varphi'_{\underline{h}}$ is the phase at the new origin

$\varphi_{\underline{h}}$ is the phase at the old origin

$\Delta \underline{x} = (\Delta x, \Delta y, \Delta z)$

EQ 4.1.3 is known as the shift theorem. Since the space group does not specify the absolute positions of the symmetry axis but only their direction, a situation arises where there is a choice of origin for each space group. The permissible origin defining reflections and the phases to be assigned are determined by the International Tables of X-ray Crystallography (IUCR, 1989). Reflections used to define the origin should be linearly independent and must not be structure seminvariants or invariants.

5.0 STRUCTURE INVARIANTS AND SEMINVARIANTS

5.1 Structure Invariants

The amplitude of $E_{\underline{h}}$ can be described as structure invariant as it does not differ when there is a shift in the origin. Unfortunately the phase of the structure factor depends on both the atomic positions and the origin, making it impossible to measure, and ultimately resulting in the phase problem. As the amplitudes are invariants then any phase derivation from them must lead to phase invariants, not single phases. Thus there exists certain linear combinations of reflections whose products are structure invariants. It is simple to show that products for $E_{\underline{h}1}E_{\underline{h}2}\ldots E_{\underline{h}n}$ are structure invariant if:

$$\underline{h}_1 + \underline{h}_2 + \ldots + \underline{h}_n = 0$$

(EQ 5.1.1)

For example the product of the following three reflections:

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

(EQ 5.1.2)

reduces to:

$$E'_{\underline{h}}E'_{\underline{k}}E'_{-\underline{h}-\underline{k}} = E_{\underline{h}}E_{\underline{k}}E_{-\underline{h}-\underline{k}}e^{2\pi i(0\cdot\Delta\underline{x})}$$

(EQ 5.1.3)

and finally to:

$$E'_{\underline{h}}E'_{\underline{k}}E'_{-\underline{h}-\underline{k}} = E_{\underline{h}}E_{\underline{k}}E_{-\underline{h}-\underline{k}}$$

(EQ 5.1.4)

The phase sum $\sum_i \varphi(\underline{h}_i)$ that is $\varphi_{\underline{h}} + \varphi_{\underline{k}} + \varphi_{-\underline{h}-\underline{k}}$ must also be a structure invariant. The linear combination of the phases will always have the same value regardless of the choice of origin. It is only the sum of the phases and not the individual phases that are origin independent and relate only to the structure. This makes the structure invariants a very powerful tool in direct methods. The simplest one is just $|E_{\underline{h}}|^2$. The type used in the example, the three phase invariants, are known as triplets or Σ_2 relationships.

5.2 Structure Seminvariants

There exists another set of linear combinations of phases that remain unchanged when the origin of the space group is shifted. The phases will remain the same only if the origin is shifted in a manner allowed by space group symmetry. They are known as structure seminvariants. Thus the following must be true for any structure seminvariant, where $\Delta\underline{x}$ is an origin shift from any permissible origin to the next: $\underline{h} \cdot \Delta\underline{x} = 0$ or n, where n = integer.

Thus for space group $P2_1$ the linear combination of three phases

$$\varphi_{h1k1l1} + \varphi_{h2k2l2} + \phi_{h3k3l3}$$

(EQ 5.2.1)

is seminvariant only if (k1 + k2 + k3) = 0, (h1 + h2 + h3) and (l1 + l2 + l3) are even integers when the origin is shifted between the allowable origins. In general any linear combination of phases that gives a product of (e,0,e), where e is an even integer, will also be a seminvariant. For example:

$\varphi_{eko} + \varphi_{e\bar{k}o}$ and $\varphi_{e\bar{k}o} + \varphi_{eke} + \varphi_{e0o}$

both the above linear combinations of phases are structure seminvariants in the space group $P2_1$. There exist far more seminvariants than invariants and they can be single structure factors, a product of structure factors, a structure factor phase, or a sum of phases. The latter are called Σ_1 relationships and can be used in the phase determination process; in the starting set of reflections to cut down on the number of reflections and in the calculation of some figures of merit. They can also be used in enantiomorph definition.

5.3 Enantiomorph

For non centrosymmetric structures in addition to just defining the structure we must also define the enantiomorph. The enantiomorph is simply the mirror image of the structure. This is done in theory by assigning a phase to a structure invariant or seminvariant which involves large E-magnitudes and whose sum is neither 0 or π. Then all the invariants and seminvariants become uniquely defined. In practise we can assign a phase to just one reflection with a large E-magnitude that belongs to an invariant or seminvariant whose sum is not 0 or π. A change in the enantiomorph will result in all the phases reversing i.e. $\varphi_{\underline{h}}$ will become $-\varphi_{\underline{h}}$ and all the invariants and seminvariants will also reverse. The enantiomorph must be defined at the start of any structure solution or a situation may arise where different reflections are defining different enantiomorphs, creating both of them in the final electron density maps and ultimately confusion.

6.0 THE NEIGHBOURHOOD PRINCIPLE

Once the enantiomorph has been selected, the observed E-magnitudes can then define both the magnitudes and the phases of the structure invariants which are consistent with that enantiomorph. The phases of the structure invariants are not primarily determined by all the E-magnitude sets. Instead they are sensitive only to a small number of reflections which are known as the neighbourhoods of the invariant. The value of the invariant is determined primarily by the reflections within the first neighbourhood, followed by the second etc. The neighbourhoods are nested with the second neighbourhood containing all the reflections from the first plus additional ones.

If the E-magnitude for each reflection within the neighbourhood is known, then an estimate for the invariant can be given using the conditional probability distribution. This estimate is more likely to be correct if the variance of distribution is small. This has the effect that in favourable cases going from the first to the second neighbourhood sometimes increases the chance of a distribution with a small variance and a more reliable estimate for the phase can be obtained.

So how can we estimate the values of the structure seminvariant by conditional probability and what types are used in the process of phase determination?

7.0 THREE PHASE STRUCTURE INVARIANTS

7.1 Triplets

Triplets are three phase structure invariants of the form:

$$\Phi_3 = \varphi_{\underline{h}} + \varphi_{\underline{k}} + \varphi_{\underline{l}}$$

(EQ 7.1.1)

where the sum of the indices:

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

(EQ 7.1.2)

For centrosymmetric structures this phase sum can only have a value of π or zero. The phase sum is usually referred to as the sign of the relationship with a value of zero being positive and π being negative. If the triplet contains two reflections of known phase with large E-magnitudes then the phase of the third reflection can be determined.

7.2 Conditional Probability Distributions

When the number and type of atoms in the unit cell and the magnitudes of the three $E_{\underline{h}}$ involved in the triplet are known, then the probability of the triplet, for the centrosymmetric case, having a value of zero is:

$$P_+(\Phi_3|\kappa) \approx \frac{1}{2} + \frac{1}{2}\tanh\frac{\kappa}{2}$$

(EQ 7.2.1)

(Cochran &Woolfson, 1955)

where

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

(EQ 7.2.2)

and

$$\sigma^n = \sum_{i=1}^{N} Z_i^n$$

(EQ 7.2.3)

and for the non-centrosymmetric case: (Cochran, 1955; Hauptman, 1976a)

$$P(\Phi_3|\kappa) \approx \frac{1}{2\pi I_o(\kappa)} e^{\kappa \cos \Phi_3}$$

(EQ 7.2.4)

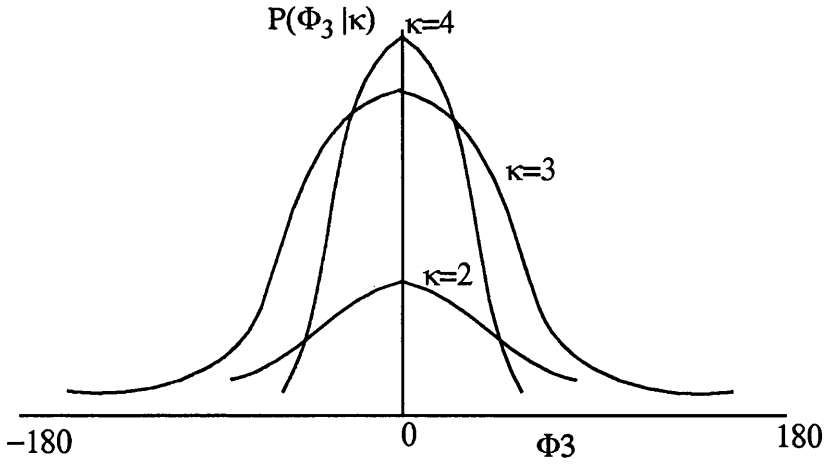
I_0 is a zero order modified Bessel Function. If the structure is comprised of equal atoms then:

$$\frac{\sigma_3}{\sigma_2^{3/2}} = \frac{1}{\sqrt{N}}$$

(EQ 7.2.5)

From EQ 7.2.5 it is obvious that as the structure complexity increases the strength of the probability distribution decreases. If we plot the probability Φ_3 versus the phase angle, then it becomes apparent that for a higher value of κ, the probability that the triplet will equal zero is greater. This can be seen in Figure 2.

FIGURE 2. Cochran distributions for κ =2, 3, 4



The maximum of the function occurs when Φ_3 =0, corresponding to a large κ value. Also the larger the value of κ the smaller the variance of distribution, indicating a

more reliable estimate for the triplet. EQ 7.2.2 links κ to the E-magnitudes of the reflections, thus it is preferable that the triplets be composed of reflections with large E-magnitudes.

The first neighbourhood of a triplet consists of the three magnitudes:

$$|E_{\underline{h}}|, |E_{\underline{k}}|, |E_{\underline{l}}|$$

(EQ 7.2.6)

while the second neighbourhood is formed by considering the following quintet (five phase invariant):

$$\Phi_3 = \varphi_h + \varphi_k + \varphi_l + \varphi_p + \varphi_{-p}$$

(EQ 7.2.7)

and is thus composed of:

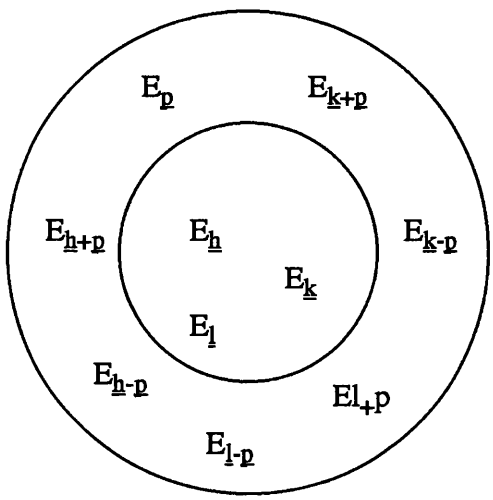
$$|E_{\underline{p}}|, |E_{\underline{h}+\underline{p}}|, |E_{\underline{h}-\underline{p}}|, |E_{\underline{k}+\underline{p}}|, |E_{\underline{k}-\underline{p}}|, |E_{\underline{l}+\underline{p}}| and |E_{\underline{l}-\underline{p}}|$$

(EQ 7.2.8)

where p is a floating vector and $|E_{\underline{p}}|$ must be large.

Figure3. illustrates the neighbourhoods in diagrammatic form.

FIGURE 3. The first and second neighbourhoods for a triplet



8.0 FOUR PHASE STRUCTURE INVARIANTS

8.1 Quartets

Quartets are four phase structure invariants of the form:

$$\Phi_4 = \varphi_{\underline{h}} + \varphi_{\underline{k}} + \varphi_{\underline{l}} + \varphi_{\underline{m}}$$

(EQ 8.1.1)

when

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

(EQ 8.1.2)

The importance of quartets is their use in the solution of structures that crystallise in a symmorphic space group.

8.2 Conditional Probability Distributions

The conditional probability of the quartet using only the four magnitudes is:

$$P(\Phi_4|B) \approx \frac{1}{2\pi I_o(B)} e^{B\cos\Phi_4}$$

(EQ 8.2.1)

where

$$B = 2\frac{\sigma_4}{\sigma_2^2}|E_{\underline{h}}E_{\underline{k}}E_{\underline{l}}E_{\underline{m}}|$$

(EQ 8.2.2)

and

$$\sigma^n = \sum_{i=1}^{N} Z_i^n$$

(EQ 8.2.3)

I_o is a zero order modified Bessel Function. If the structure is comprised of equal atoms then:

$$\frac{\sigma_4}{\sigma_2^2} = \frac{1}{N}$$

(EQ 8.2.4)

A plot of the probability Φ_4 against the phase angle results in a curve where the largest value of B creates the greatest probability that the quartet is equal to zero.

Thus $P(\Phi_4)$ is similar to $P(\Phi_3)$ except that B values of order 1/N tend to be less than κ values of order $1/(\sqrt{N})$. Ultimately then, the estimate of zero for the quartets is less reliable than the estimate for the triplets. To overcome this the reflections in the neighbourhood of the quartets are considered.

The second neighbourhood adds the cross terms:

$$|E_{\underline{h}+\underline{k}}|, |E_{\underline{k}+\underline{l}}| and |E_{\underline{l}+\underline{m}}|$$

(EQ 8.2.5)

The third neighbourhood is formed by introducing an arbitrary vector p and its associated vector q such that:

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

(EQ 8.2.6)

$|E_{\underline{p}}|$ and $|E_{\underline{q}}|$ are best to be large to ensure the probability of this quartet to be high.

The introduction of these vectors results in a second quartet invariant:

$$\Phi_{pq} = \varphi_{\underline{h}} + \varphi_{\underline{k}} + \varphi_{\underline{p}} + \varphi_{\underline{q}}$$

(EQ 8.2.7)

which indirectly defines a third quartet invariant:

$$\Phi_{lm} = \varphi_{\underline{l}} + \varphi_{\underline{m}} + \varphi_{-\underline{p}} + \varphi_{-\underline{q}}$$

(EQ 8.2.8)

Φ_{pq} has a second neighbourhood which consists of

$$|E_h|, |E_k|, |E_q|, |E_p|, |E_{h+k}|, |E_{k+p}| and |E_{h+p}|$$

(EQ 8.2.9)

and Φ_{lm} has a second neighbourhood comprising of

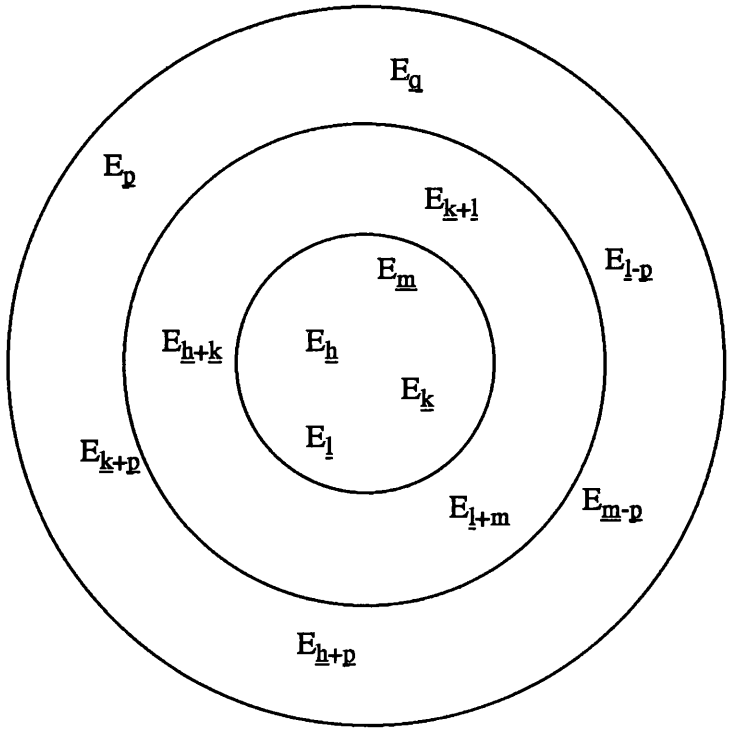
$$|E_l|, |E_m|, |E_p|, |E_q|, |E_{l+m}|, |E_{m-p}| and |E_{-p+q}|$$

(EQ 8.2.10)

Because $\Phi_4 + \Phi_{pq} + \Phi_{lm} = 0$, Φ_4 can be estimated by 13 unique E-magnitudes and cross terms.

The terms are shown in Figure 4. in their appropriate neighbourhoods:

FIGURE 4. The first, second and third neighbourhoods for a quartet



The probability that Φ_4 is equal to zero can also be considered using the principle E-magnitudes and their cross terms (Hauptman,1975a,b 1976b; Giacovazzo,1975,1976b).

$$P\left(\Phi_4 \middle| |E_{\underline{h}}||E_{\underline{k}}||E_{\underline{l}}||E_{\underline{m}}||E_{\underline{h}+\underline{k}}||E_{\underline{k}+\underline{l}}||E_{\underline{l}+\underline{m}}|\right) =$$

$$\frac{1}{L} e^{-2B\cos\Phi_4} I_o\left(\frac{2\sigma_3}{\sigma_2^{3/2}}|E_{\underline{h}+\underline{k}}|X_{12}\right) I_o\left(\frac{2\sigma_3}{\sigma_2^{3/2}}|E_{\underline{k}+\underline{l}}|X_{23}\right) I_o\left(\frac{2\sigma_3}{\sigma_2^{3/2}}|E_{\underline{l}+\underline{m}}|X_{31}\right)$$

(EQ 8.2.11)

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

L is a normalising constant

and

$$B = \frac{2}{N}|E_{\underline{k}}E_{\underline{h}}E_{\underline{l}}E_{\underline{m}}|$$

(EQ 8.2.12)

$$X_{12} = \left(|E_{\underline{h}}|^2|E_{\underline{k}}|^2 + |E_{\underline{l}}|^2|E_{\underline{m}}|^2 + 2|E_{\underline{h}}||E_{\underline{k}}||E_{\underline{l}}||E_{\underline{m}}|\cos\Phi_4\right)^{1/2}$$

(EQ 8.2.13)

$$X_{23} = \left(|E_{\underline{k}}|^2|E_{\underline{l}}|^2 + |E_{\underline{h}}|^2|E_{\underline{m}}|^2 + 2|E_{\underline{h}}||E_{\underline{k}}||E_{\underline{l}}||E_{\underline{m}}|\cos\Phi_4\right)^{1/2}$$

(EQ 8.2.14)

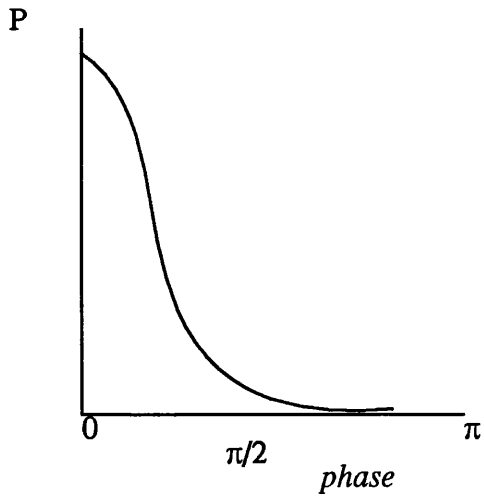
$$X_{31} = \left(|E_{\underline{l}}|^2|E_{\underline{h}}|^2 + |E_{\underline{k}}|^2|E_{\underline{m}}|^2 + 2|E_{\underline{h}}||E_{\underline{k}}||E_{\underline{l}}||E_{\underline{m}}|\cos\Phi_4\right)^{1/2}$$

(EQ 8.2.15)

The most reliable value for Φ_4 occurs when the product $|E_{\underline{h}}||E_{\underline{k}}||E_{\underline{l}}||E_{\underline{m}}|$ is large. Three types of quartet can occur:

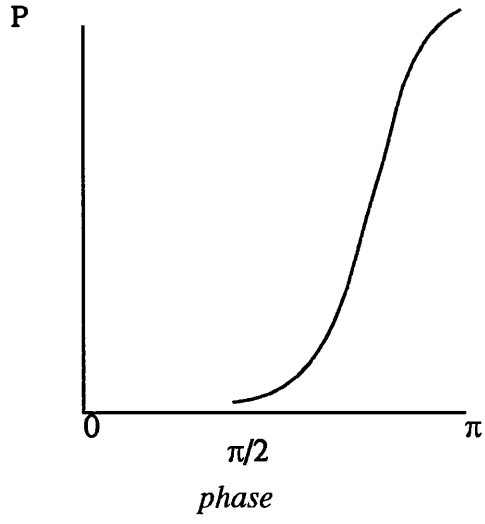
- Positive quartets: When the three cross terms are large $\Phi_4 \approx 0$, i.e. $\cos\Phi_4 \approx +1$ (Schenk, 1973). Although these quartets do contain useful information they are not really used in phase determination procedure as they are highly correlated with triplets. Figure 5. illustrates the probability distribution for positive quartets.

FIGURE 5. The probability distribution for a positive quartet



- Negative quartets: When the E-magnitudes of the cross terms are small then they can be described as negative quartets. The probability is that $\cos\Phi_4 \approx -1$, i.e. $\Phi_4 \approx \pi$ (Hauptman, 1974). They are very important in direct methods as they are used in the phase determination procedure and also in the calculation of the Figure of Merit, NQEST (DeTitta, Edmonds, Langs & Hauptman, 1975). Figure 6. illustrates the probability distribution for negative quartets.

FIGURE 6. The probability distribution for a negative quartet



- Enantiomorph sensitive quartets: If the E-magnitudes of the cross terms have intermediate values, or the E-magnitudes are just a mixture of large and small then they are known as enantiomorph sensitive quartets (Hauptman, 1975a). They are the least reliable as their indications tend to be weak and they are therefore rarely used in phase determination. Φ_4 has a predicted value of $\pm\frac{\pi}{4}$. Figure 7. shows the probability distribution for enantiomorph sensitive quartets.

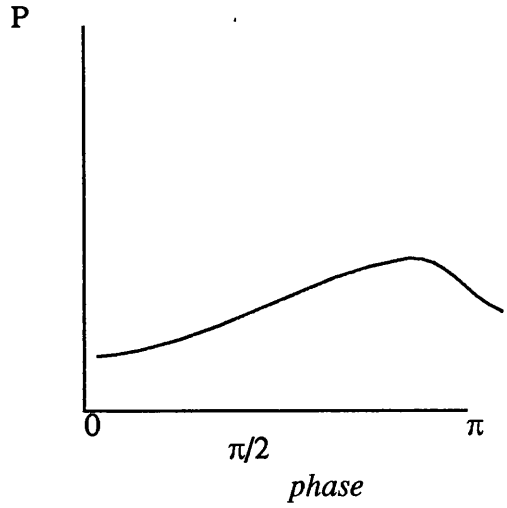


FIGURE 7. The probability distribution of an enantiomorph sensitive quartet

9.0 THE STARTING SET

Once the seminvariants and the invariants have been determined, a starting set of phases can be defined which will be used to generate new phases. This starting set consists of:

- The origin defining reflections.
- The enantiomorph defining reflections (if appropriate).
- The Σ_1 determined phases.
- Reflections of unknown phase which can be used in phase permutation and the symbolic addition procedure. They take on different values for each tangent refinement and are then permuted.

The reflections that are deemed suitable for addition to the starting set are chosen by the convergence procedure (Germain, Main & Woolfson, 1970). Firstly the value for $\alpha(\underline{h})_{est}$ is calculated for each reflection. This requires no prior knowledge of

individual phase angles and is a measurement of reliability of each phase in regards to the remaining phases in the data set. Thus it is just a measure of the connectivity of single reflection to the other reflections through the invariants:

$$\alpha^2(\underline{h}_{est}) = \sum_{k} \kappa(\underline{h}, \underline{k})^2 + \sum_{k} \sum_{l} \kappa(\underline{h}, \underline{k})\, \kappa(\underline{h}, \underline{l}) \frac{I_1\{\kappa(\underline{h}, \underline{k})\}\, I_1\{\kappa(\underline{h}, \underline{l})\}}{I_0\{\kappa(\underline{h}, \underline{k})\}\, I_0\{\kappa(\underline{h}, \underline{l})\}}$$

(EQ 9.0.1)

where $\kappa(\underline{h}, \underline{k})$ is defined earlier in EQ 7.2.2

The procedure is iterative and during each iteration, the least reliable phase is eliminated. This is the reflection which has the lowest value for $\alpha(\underline{h}_{est})$. The reflections remaining are those which are strongly linked together, having maximum connectivity through the invariants. The value of $\alpha(\underline{h}_{est})$ is then recalculated for all the remaining reflections. Reflections which have values of $\alpha(\underline{h}_{est}) = 0$ will be chosen for permutation because their phase cannot be calculated from the phases of the remaining reflections. Also, when the last of the origin and enantiomorph defining reflections are found, they will not be eliminated from the list but kept for the starting set. This method does not select each reflection on its own merits but instead looks at how the reflections are linked together. The convergence procedure usually leads to strong phase development and multiple interactions, avoiding weak links in the convergence map. A weak link occurs where phase information comes from only one invariant. However, care must be taken as any change in the E-magnitudes can lead to a noticable difference in the phasing path. Invariants or seminvariants used in the generation or refinement of new phases are described as the active set, whilst the passive set are those used to calculate the figures of merit and are usually not active. Triplets were initially employed in convergence mapping but any invariant or seminvariant can be used (Freer & Gilmore, 1980).

10.0 PHASING THE STARTING SET

The reflections in the starting set consist of those that are already phased and others which are unphased. The latter will be assigned a range of phase values and are then propagated each time through the convergence map to determine the rest of the phases

in that phase set. Obviously it is important to find methods that reduce the number of phase sets actually calculated without losing the ability of finding the correct one. Usually the starting set of reflections is kept to a minimum and centric phases can only be permuted over the two values of π and zero, whereas acentric phases are not restricted, but for quadrant permutation we can permute over $\pm 3\pi/4$ or $\pm\pi/4$. For n permuted reflections there will be 4^n phase sets each of which then undergo tangent refinement. The options within this multisolution class of methods involve the use of magic integers or random phasing procedures. In Chapter 4 the development and technique of using error correcting codes will be discussed.

10.1 Magic Integers

The technique of magic integers (White & Woolfson, 1975: Declercq, Germain & Woolfson, 1975) can be used in direct methods programs to reduce the number of phase sets generated.

For a sequence of n integers $m_1, m_2, m_3, \ldots\ldots, m_n$, n phases can be represented by the equations:

$$\varphi_i = m_i x$$

(EQ 10.1.1)

where x is a variable in the range $0<x<2\pi$.

The starting set of reflections are phased by assigning incremented values to x, that it x is incremented. Several sets are investigated by making x a suitable number of different values. The enantiomorph can be defined by assigning values to x in the range $0 \leq x \leq \pi$. Thus for each x there is a corresponding set of n phases, with the phase sets being as different from each other as possible. Magic integers are only used on phases that are unrestricted by space group symmetry. The set of integers are an approximation of the phases so it is vital to limit the root mean square (r.m.s.) errors to their smallest. To do this the most efficient set of magic integers is used (Main, 1977).

These are shown in table 1 along with the r.m.s. errors and the number of corresponding permutations in a full factorial design.

TABLE 1. The most efficient sets of magic integers for up to eight phase permutations

n	Sequence								Number of sets (M.I)	Number of sets (F.F.D.)	r.m.s. error
2	2	3							12	16	26
3	3	4	5						20	64	29
4	5	7	8	9					32	256	37
5	8	11	13	14	15				50	1024	42
6	13	18	21	23	24	25			80	4096	47
7	21	29	34	37	39	40	41		128	16384	48
8	34	47	55	60	63	65	66	67	206	65536	49

11.0 PHASE EXTENSION AND REFINEMENT

11.1 The Tangent Formula

The extension and refinement of phases are two different processes which can both be carried out by tangent refinement (Karle & Hauptman, 1956; Karle & Karle 1966). The tangent formula in its weighted form is:

$$\tan\varphi_h \approx \frac{\sum_{\underline{k}} w_{\underline{k}} w_{\underline{h}-\underline{k}} |E_{\underline{k}}||E_{\underline{h}-\underline{k}}| \sin(\phi_{\underline{k}} + \phi_{\underline{h}-\underline{k}})}{\sum_{\underline{k}} w_{\underline{k}} w_{\underline{h}-\underline{k}} |E_{\underline{k}}||E_{\underline{h}-\underline{k}}| \cos(\phi_{\underline{k}} + \phi_{\underline{h}-\underline{k}})}$$

(EQ 11.1.1)

where $w_{\underline{k}}$, $w_{\underline{h}-\underline{k}}$ are weights of reflections $\underline{k}$ and $\underline{h}$ - $\underline{k}$ respectively.

The triplet invariants are used in EQ 11.1.1 to expand the number of phases with the result that there may exist several indications to the value of φ_h. By considering all the appropriate invariant relationships the tangent formula is able to determine probable values of these new phases with the assumption that all the invariants are independent of each other. The various phase values are inserted on the right-hand side of the

equation and an estimate for $\tan\varphi_h$ is produced. The reliability of $\tan\varphi_h$ is measured by the variance $V(\varphi_h)$ which can be expressed as a function of $\alpha(\underline{h})$ (Karle & Karle, 1966).

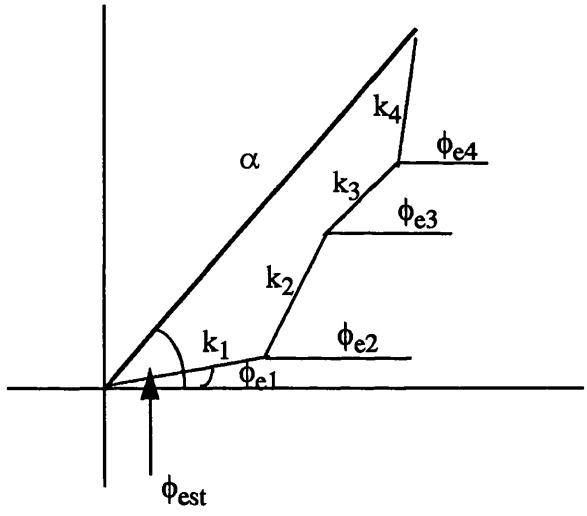
$$\alpha(\underline{h}) = \sum_{\underline{k}} \kappa_{\underline{h}\underline{k}} \cos(\varphi_{\underline{k}} + \varphi_{\underline{h}-\underline{k}})$$

(EQ 11.1.2)

where $\kappa_{\underline{h}\underline{k}}$ is defined in EQ 7.2.2

This estimated value for the phase should be more reliable than the phase prediction using one invariant. If several invariants give a similar indication then the reliability of the phase, α_h, will have a large value. This is illustrated in Figure 8.

FIGURE 8. Combination of phase estimates



where ϕ_{e1} to ϕ_{e4} are the phase estimates with strengths k_1 to k_4 combining to give an overall estimate ϕ_{est} with strength α (Woolfson, 1991).

Phase estimates that are produced early on in the expansion process are based on initial phase estimates and not on invariants involving phases whose values are still to be determined. When all the reflections in the system have phase estimates they can be updated by using tangent refinement. In this case a phase estimate is calculated by inserting all the remaining estimates on the right-hand side of the equation. Tangent refinement is an iterative process, continuing through each cycle until there are negligible shifts in the phase values.

A weighting scheme is employed in the tangent formula as some invariants will have a greater contribution to the value of $\varphi_{\underline{h}}$ than others. The reflections that appear at the bottom of the convergence map i.e. reflections with large E-magnitudes will have maximum connectivity through the triplets and a greater effect on the value compared to those further up the map. It is important that the reflections further up the map are used for the propagation of phase information but as they are poorly determined then they must not exert a large effect in the generation of new phases. Thus reflections at the bottom of the map are weighted at $\omega \approx 1$, while reflections further up the map are assigned lower weights. The weighting scheme is vital in the initial stage of phasing but as refinement proceeds the weights eventually become unity so the scheme exerts little effect in the latter stages. One effective weighting scheme is:

$$W_{\underline{h}} = min\left[\frac{\alpha(\underline{h})}{5}, 1.0\right]$$

(EQ 11.1.3)

With this weighting scheme the weights tend to reach unity too quickly. To prevent this another weighting scheme was introduced by Hull and Irwin (Hull & Irwin, 1978)

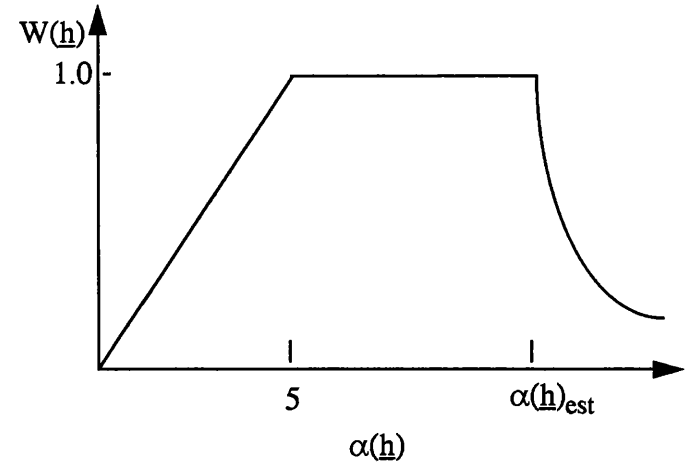
$$W_{\underline{h}} = min\left[\frac{\alpha(\underline{h})}{5}, 1.0, \frac{\alpha(\underline{h})_{est} + 5}{\alpha(\underline{h})}\right]$$

(EQ 11.1.4)

where $\alpha(\underline{h})$ and $\kappa_{\underline{hk}}$ and $\alpha(\underline{h})_{est}$ are defined in EQ 11.1.2, EQ 7.2.2 and EQ 9.0.1 respectively.

If the $\alpha(\underline{h})$ increases to a value which is greater than $\alpha(\underline{h})_{est}$ then the weight is reduced resulting in the value of $\alpha(\underline{h})$ remaining close to its expected value, producing more accurate phases. It is this weighting scheme that is used in the SWTR procedure in MITHRIL94. The graphical form of the Hull & Irwin weighting scheme is shown in Figure 9.

FIGURE 9. Hull & Irwin weighting scheme



It is found that this weighting scheme is useful in solving structures that contain heavy atoms, have pseudo-symmetry, or belong to a symmorphic space group.

11.2 Random Phasing

There exists two main methods of random phasing, developed in two computer based procedures:

11.2.1 YZARC - Linear Equations

An alternative to the use of the phase permutation method for phase refinement is the use of triple-phase relationships treated as linear equations with least squares refinement. This technique is applied in the program YZARC (Baggio, Woolfson, Declercq & Germain, 1978).

The method evolved around the triple-phase relationship where the phases are expressed in cycles with an appropriate weighting scheme. This is shown below by EQ. 11.2.1

$$K\varphi_{\underline{h}} \pm K\varphi_{\underline{k}} \pm K\varphi_{\underline{l}} \approx K(n-b)$$

(EQ 11.2.1)

where K is the weight of the invariant and n is an unknown integer. If the integers are known then the whole system of equations can be written in matrix notation

$$A\varphi = c$$

(EQ 11.2.2)

Routine algebra give a least squares solution

$$\varphi = (A^T A)^{-1} A^T c$$

(EQ 11.2.3)

With an approximate set of phases nearest integers can be found resulting in cyclic refinement of the phases, terminating when the integer values reach stability.

During phase refinement an extremely important quantity is the radius of convergence. This is described as being the distance in phase space from which it is possible to converge to the correct solution. During the study of this refinement technique the radius of convergence was examined. A phase set with random errors added to the phases was refined to see it the r.m.s error could be reduced to a value below 30° . This was the criterion set for a successful convergence (Baggio, Woolfson, Declercq & Germain, 1978). The results obtained from this experiment were surprisingly efficient. When totally random phases were used, a small number of phase sets were still refined to below 30° error. This result introduced the concept of random phasing for the starting set.

11.2.2 RANTAN

In the RANTAN program (Yao Jia Xing, 1981), a large number of phases are given random values and low weights and are then refined using the weighted tangent formula. The weights given to each phase are essential for the procedure to work. Origin defining phases are given weights of 1.0, enantiomorph phases with general values are given weights of 0.85 and random phases are given weights of 0.25. The initial random phase and its corresponding weight will not be changed unless a phase estimate with a higher weight is obtained allowing the phase to be refined.

11.3 Phase Annealing

Simulated Annealing (Kirkpatrick, Gelatt & Vecchi, 1983) is the tool applied to avoid the local minima of a system preventing the determination of the global minimum. For phase annealing the initial phases are random and the consequent "cooling" of the phase sets enables the system to reach its global minimum in phase space. This

method of refinement has already been implemented in SHELX-90 (Sheldrick, 1990). Chapter 3 discusses this alternative refinement method and its implementation into the MITHRIL94 program (Gilmore,1984; Gilmore & Brown 1988).

12.0 FIGURES OF MERIT

After phasing a situation arises where there are numerous phase sets, for example any number between 16 and 4000, all of which must be assessed and ranked so that the most plausible phase set can be determined. Once this set has been chosen an E-map is calculated and examined to see if it reveals the structure. Obviously it is not feasible for every phase set to have an E-map examined as this would be extremely time consuming. Thus the ranking of phase sets in a multisolution environment is vital and is achieved by the calculation of the figures of merit (FOMs).

Ideally, the FOM should be independent of the technique used to generate the phase set. One complication however is that a strong phase relationship is essential in both the derivation of phases and the calculation of the FOM and therefore should be applied in both. The resulting calculation of the FOMs involves a compromise between these two processes. The most widely used FOMs are described in the following sections.

12.1 ABSFOM

ABSFOM (Germain, Main & Woolfson, 1971) measures the extent to which the triple invariant's internal consistency holds for those used in phase estimation.

$$ABSFOM = \frac{\sum_{\underline{h}} \alpha_{\underline{h}} - \sum_{\underline{h}} \langle \alpha_{\underline{h}} \rangle_r}{\sum_{\underline{h}} \langle \alpha_{\underline{h}} \rangle_{est} - \sum_{\underline{h}} \langle \alpha_{\underline{h}} \rangle_r}$$

(EQ 12.1.1)

where

$$\langle \alpha_{\underline{h}} \rangle_r = \langle \sum_{\underline{k}} \kappa^2_{\underline{h}\underline{k}} \rangle^{1/2}$$

(EQ 12.1.2)

$\langle \alpha_{\underline{h}} \rangle_{est}$ and $\alpha_{\underline{h}}$ are defined by EQ 9.0.1 and EQ 11.1.2 respectively.

α_{est} is the estimated value of α

α_r is the expected value of α for random phases.

For random phases ABSFOM = 0, and if the estimated and measured values of α are equal then ABSFOM has a value of 1.0. If the phase set is overconsistent then ABSFOM values are likely to be greater than unity whereas underconsistent sets yield low ABSFOM values. Typical values considered to be reasonable could be anything between 0.8 and 1.4; ABSFOM is not considered the most efficient discriminator between phase sets.

12.2 PSI-ZERO

This was defined by Cochran and Douglas (Cochran & Douglas, 1955; Main, 1977) as equation

$$\Psi_o = \frac{\sum_{\underline{h}} \left| \sum_{\underline{k}} E_{\underline{h}} E_{\underline{h}-\underline{k}} \right|}{\sum_{\underline{h}} \sqrt{\left(\sum_{\underline{k}} \left| E_{\underline{h}} E_{\underline{h}-\underline{k}} \right|^2 \right)}}$$

(EQ 12.2.1)

The value of Ψ_o is independent of the tangent formula and can be used to discriminate between phase sets that have similar ABSFOM values. This is a measure of reliability of the small E magnitudes and therefore phase sets with small values of Ψ_o, usually < 1 , are considered to be correct. If a situation arises where every other FOM calculated for a phase set is good but the ψ_o >> 1 it is possible that a fragment has been found but in the wrong position.

12.3 RESID

RESID is defined by equation

$$Resid = \frac{\sum_{\underline{h}} \left| \alpha_{\underline{h}} - \langle \alpha_{\underline{h}}^2 \rangle^{1/2}_{est} \right|}{\sum_{\underline{h}} \langle \alpha_{\underline{h}}^2 \rangle^{1/2}_{est}}$$

(EQ 12.3.1)

where $\langle \alpha_{\underline{h}} \rangle_{est}$ and $\alpha_{\underline{h}}$ are defined by EQ 9.0.1 and EQ 11.1.2 respectively.

RESID is the residual between the estimated and actual values of α and since it is dependent on α it must also rely on tangent refinement. A correct phase set should have an approximate RESID value of 20%, however greater values could occur when there are a large number of atoms in the unit cell and smaller values could occur when heavy atoms dominate the diffraction.

12.4 NQUEST

This FOM utilises the negative quartets. It is defined as

$$NQUEST = \frac{\sum_{\underline{hklm}} W_{\underline{hklm}} \cos(\varphi_{\underline{h}} + \varphi_{\underline{k}} + \varphi_{\underline{l}} + \varphi_{\underline{m}})}{\sum_{\underline{hklm}} W_{\underline{hklm}}}$$

(EQ 12.4.1)

where for centrosymmetric structures

$$W_{\underline{hklm}} = |1 - 2P^{+}|$$

(EQ 12.4.2)

and for non-centrosymmetric structures

$$W_{\underline{hklm}} = 1/\sigma^2$$

(EQ 12.4.3)

σ^2 is the variance of the probability distribution, and the summation is over all of the negative quartets.

NQUEST is largely independent of the phasing process as it utilises the information from the small E-magnitudes. It is also independent of Ψ_o as the information is used in a different manner. A correct phase set will normally have a value < - 0.2, with the range of NQUEST between -1.0 and +1.0.

12.5 CFOM (Combined Figure of Merit)

When the phasing procedure has finished, the separate FOMs are combined to give CFOM, the combined figure of merit. This is then used to rank the phase sets for the calculation of E-maps. The CFOM shown in EQ 12.5.1 is the one which exists in MITHRIL94 although other direct methods packages may have different FOMs and therefore different CFOMs.

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

$$W_3\frac{(R_\alpha)_{max} - R_\alpha}{(R_\alpha)_{max} - (R_\alpha)_{min}} + W_4\frac{NQEST_{max} - NQUEST}{NQEST_{max} - NQUEST_{min}}$$

(EQ 12.5.1)

where each weight W_1 to W_4 is normally set to unity but can be varied to enhance the information in one FOM. If a FOM has not been calculated for a structure then the corresponding weight will be set at zero. CFOM should be large for a correct phase set and the range is usually between 0 and 4. The maximum value of CFOM is the summation W1+W2+W3+W4.

13.0 ELECTRON DENSITY MAPS

As discussed earlier, the electron density map is related to the structure factors of a reflection by a Fourier transform. The electron density of a structure will be at its greatest around the atomic positions. Therefore by identifying regions of high electron density the elucidation of the relative atomic positions is achieved. The last stage of any direct methods procedure then is the calculation of the E-maps. When phasing is complete the phase sets are ranked according to their figures of merit and for the best phase sets an E-map is calculated and examined for structure information.

13.1 E-maps

An E-map is so called because it is a Fourier synthesis of EQ 1.0.3 but the $|F_{\underline{h}}^{obs}|$ is replaced by $|E_{\underline{h}}|$ (Karle, Hauptman, Karle & Wing, 1958). The reason for this arises from the normalisation procedure which illustrates that reflections with large $|E_{\underline{h}}|$ are often those with high values of $\sin\theta$ and low values of $|F_{\underline{h}}|$. Phases which are the most reliable are found to be those with large E-magnitudes so it is the normalised structure factor which is used to calculate the E-map. This results in the map being very sharp as it is illustrating point atoms at rest and it must be sampled at small grid spacings, usually of the order of $0.3A°$.

The interpretation of an E-map involves three stages:

- A peak search which gives a list of peak coordinates. Peak height should be used to assign atomic types to peaks and obviously requires some knowledge of the chemical nature of the structure.
- Possible bonding between the peaks.
- Identification of chemically reasonable fragments. There are two main methods of identification. (i) Manually by examining the peak list. (ii) Automatic interpretation of the peak list by computer (Main & Hull,1978). This involves considering maximum and minimum bond angles and lengths in known molecular structure.

The user may need a high level of interaction at this stage to make sure that E-maps are interpreted correctly. Computer graphics have proved invaluable in this area, reducing considerably the period of time it takes for full examination and interpretation of the maps.

14.0 THE LIMITATIONS OF DIRECT METHODS

Direct methods, although an extremely powerful tool in structure solution, will sometimes fail to produce any interpretable E-maps. One reason for this can be that the data collected from the diffraction experiment is of very poor quality. However there are some factors which are intrinsic to the procedure.

- The reliability of the phase relationship - this can be undermined if there exist too many atoms in the unit cell. The reliability is also affected by the fact that the probability formula is based on randomly distributed atoms, when in reality the position of the atoms are determined by strict rules governing interatomic angles and lengths.
- The stability of the tangent formula - even when reliable phase relationships are used, they can become unstable during tangent refinement. This results in over refinement of the phases and ultimately E-maps which are useless. The instability can arise because of the assumption that the phase relationships are independent, which is not the case.
- Figures of Merit - in some cases there exists no clear discrimination between phase sets making it a difficult task to find a suitable E-map to examine.
- Large structures - typically they are in the region of excess of 1,500 Daltons. Structures of this size not only reduce the reliability of the phase relationships but the resulting E-maps are cluttered and difficult to interpret.
- The size of the starting set of reflections - the starting set must be efficient enough to produce strong phase development without using excessive computer time. If the reflections involved in the starting set are used in incorrect phase relationships then the process of phasing and refinement can be affected. This is only a problem in multi solution methods. In chapter 4 of this thesis a new method of phasing the starting set of reflections will be discussed.
- Assume Wilson statistics - that is, all the atoms are randomly and uniformly positioned in the unit cell.

15.0 MITHRIL94

In the early 1980s it was decided to bring together all the important theoretical advances made in the 1970s into a single computer program. The result of this NATO funded project was the MITHRIL package (Gilmore, 1984; Gilmore & Brown, 1988). The name MITHRIL is an acronym for **M**ultan with **I**nteractive facilities, **T**riplet checking, **H**igher invariants, **R**andom phasing, **I**ntelligent control of flow and options and **L**inear equations phasing. In 1988 it was decided to upgrade the MITHRIL package to reflect the theory developed during the 1980s, also to introduce some minor bug fixes and to make the package more user friendly.

The program itself is written in standard Fortran 77 in a modular form with each module having its own menu and user options. It is designed to be run in either batch or interactive modes with a great deal of flexibility in the user options for structure solution. The flowchart of the modules through the program is shown in Figure 12. The basic framework is based around a highly modified version of MULTAN80, that incorporates the major features from the following programs:

1. MAGEX (Hull, Viterbo, Woolfson & Shao-Hui, 1981; Shao-Hui & Woolfson, 1982) a magic-integer based program that uses the primary-secondary method for the magic integer representation of the phases. This allows the use of a large number of reflections and relationships to be used from the start in a structure solution.

2. YZARC (Baggio, Woolfson, Declercq & Germain, 1978) is a procedure for the random assignment of phases to the starting set of up to 100 reflections. They are then refined and used for phase extension followed by further phase refinement. It is essentially a forerunner of the RANTAN procedure.

3. RANTAN (Yao Jia-Xing, 1981) is a program that assigns random phases to all reflections and then refines them using the tangent formula with a carefully controlled weighting scheme.

4. LSAM (Germain & Woolfson, 1968) a program to solve structures using a symbolic addition method.

The program has a large number of options and techniques available for structure solution and has proved to be an efficient and powerful tool in the elucidation of many structures. The majority of structures can be solved by MITHRIL using the default options and it is only the most obstinate of data sets that have proved impossible to solve.

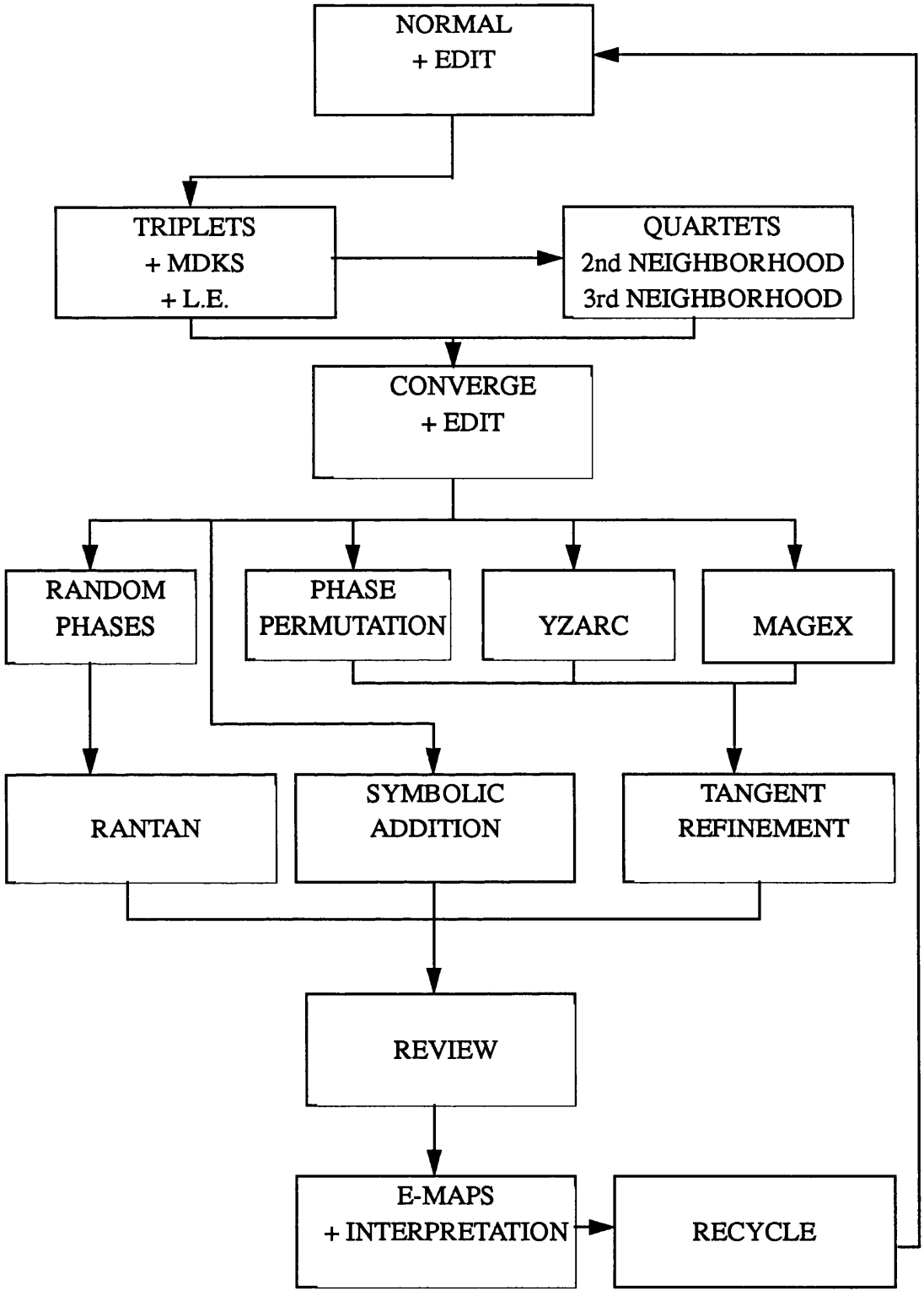


FIGURE 10. Flowchart of the modules of the MITHRIL package

References

Baggio, R., Woolfson, M.M., Declercq, J.P. & Germain, G. (1978). *Acta Cryst.* **A34**, 883-892

Cochran, W. (1952). *Acta Cryst.* **5**, 65-67

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

Cochran, W. & Douglas, A.S. (1955). *Proc. Roy. Soc.* **A227**, 486-500

Cochran, W. & Woolfson, M. M. (1955). *Acta Cryst.* **8**, 1-14

Declerq, J. P., Germain, G. & Woolfson, M. M. (1975). *Acta Cryst.* **A31**, 367-372

DeTitta, G.T., Edmonds, J.W., Langs, D.A. & Hauptman, H. (1975). *Acta Cryst.* **A31**, 472-479

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

Germain, G., & Woolfson, M.M. (1968). *Acta Cryst.* **B24**, 91-96

Germain, G., Main, P. & Woolfson, M.M. (1970). *Acta Cryst.* **B26**, 274-285

Germain, G., Main, P. & Woolfson, M.M. (1971). *Acta Cryst.* **A27**, 368-376

Giacovazzo, C. (1975). *Acta Cryst.* **31**, 252-259

Giacovazzo, C. (1976a). *Acta Cryst.* **32**, 74-82

Giacovazzo, C. (1976b). *Acta Cryst.* **32**, 91-99

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

Gilmore, C.J. & Brown, S.R. (1988). *J. Appl. Cryst.* **21**, 571-572

Harker, D. & Kasper, J.S. (1948). *Acta Cryst.* 1, 70-75

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

Hauptman, H. (1975a). *Acta Cryst.* **A31**, 671-679

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

Hauptman, H. (1976a). *Some Recent Advances in the Probabilistic Theory of the Structure Invariants, Proceedings of the Prague Conference on Computing and Direct Methods in X-Ray Crystallography, July,* 1975, Munksgaard, Copenhagen.

Hauptman, H. (1976b). *Acta Cryst.* **32**, 877-882

Hull, S.E. & Irwin, M.J. (1978). *Acta Cryst.* **A34**, 863-870

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

International Tables for Crystallography, (1989). *Volume A. Space-Group Symmetry,* Kluwer Academic Publishers, Dordrecht/Boston/London

Karle, J. & Hauptman, H. (1953). *Acta Cryst.* **6**, 473-476

Karle, J. & Hauptman, H. (1950). *Acta Cryst.* **3**, 181-187

Karle, J. & Hauptman, H. (1956). *Acta Cryst.* **9**, 635-651

Karle, I.L., Hauptman, H., Karle, J. & Wing, A.B. (1958). *Acta Cryst.* **11**, 257-263

Karle, J. & Karle, I.L. (1966). *Acta Cryst.* **21**, 849-859

Kirkpatrick, S., Gelatt, C.D. & Vecchi, M.P. (1983). *Science* **220**, 671-680

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

Main, P. & Hull, S.E. (1978). *Acta Cryst.* **A34**, 353-361

Sayre, D. (1952). *Acta Cryst.* **5**, 60-65

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

Shao-Hui, Z. & Woolfson, M.M. (1982). *Acta. Cryst.* **A38**, 683-685

Sheldrick, G. M. (1990). *Acta Cryst* **A46**, 467-473

Patterson, A.L. (1934). *Phys. Rev.* **46**, 372-376

White, P.S. & Woolfson, M.M. (1975). *Acta Cryst.* **A31**, 53-56

Wilson, A.J.C. (1942). *Nature* **150**, 151-152

Woolfson, M. M. (1991). *Direct methods of solving crystal structures.* Plenum Press, New York, 87-102

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

Zachariasen, W. H. (1952). *Acta Cryst.* **5**, 68-73

CHAPTER 2

ESTIMATING THE RELIABILITY OF QUARTETS

1.0 THEORY

1.1 Introduction

As described in Chapter 1, the reliability of the triplets is measured by κ, a variable used in the conditional probability expressions $P_{1/3}$ and $P^{+}{}_{3}$. This variable determines which triplets will be used throughout the convergence mapping and tangent refinement. The conditional probability distribution $P_{1/3}$, utilising κ is the Cochran distribution (Cochran, 1955):

$$P\left(\Phi_3\left(\underline{h}, \underline{k}\right)\right) = \frac{1}{2\pi I_o(\kappa)} e^{\kappa \cos \Phi_3\left(\underline{h}, \underline{k}\right)}$$

(EQ 1.1.1)

where κ is:

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

(EQ 1.1.2)

and

$$\sigma^n = \sum_{i=1}^{N} Z_i^n$$

(EQ 1.1.3)

To mix the quartets with these relationships, the quartets must also have an appropriate measure of reliability on the same scale as κ. The method used in MITHRIL94 (Freer & Gilmore, 1980) is a numerical integration of the full quartet formula. A disadvantage of this method is that it is relatively slow with $P_{1/7}$ being evaluated at every 45 degrees, to calculate the mode and variance of the distribution. Obviously it would be more efficient to use a method that was quicker and still produced an accurate measure of reliability for the quartets. The remainder of this chapter will compare the direct use of the quantity B, corresponding to κ, which is present in the conditional probability expressions $P_{1/7}$ and $P^{+}{}_{7}$ (Hauptman, 1974) with the numerical integration method used in MITHRIL94 (Gilmore, 1984; Gilmore & Brown, 1988).

1.1.1 κ_{eq} as an estimate of reliability

The non-centrosymmetric case

The procedure that exists in MITHRIL94 for the non - centrosymmetric case is as follows:

- Firstly the relevant joint conditional probability distribution $P(\Phi)$ is calculated in 45° intervals from 0° to 180° with $P(\Phi)$ representing either $P_{1/7}$ or $P_{1/13}$.
- The maximum value of the $P(\Phi)$, the mode $|\Phi_m|$, is then found.
- The distribution is normalised via numerical integration using Simpson's rule, such that.

$$\int_0^{2\pi} P(\Phi)\, d\Phi = 1$$

(EQ 1.1.1.1)

Simpson's rule is discussed in the next section.

- The associated variance V, where $V = \sigma^2$ is also derived via numerical integration of the normalised distribution, such that

$$V = \int_0^{2\pi} (\Phi - |\Phi_m|)^2 P(\Phi)\, d\Phi$$

(EQ 1.1.1.2)

- Each quartet is then assigned an equivalent κ value κ_{eq} related to V (in degrees^2) by a best fit equation which is derived empirically

$$\kappa_{eq} = -\left(\frac{9.0}{\sigma^2}\right) + \left(\frac{510}{\sigma}\right) - 24.0 + (0.35 \times \sigma) - \left(\left(1.8 \times 10^{-3}\right) \times \sigma^2\right)$$

$$+ 1.5\exp\left(-\frac{15}{10000.0} \times \sigma\right) + 1.5\exp\left(-\frac{15}{100000} \times \sigma^2\right)$$

(EQ 1.1.1.3)

By default only quartets with a κ_{eq} >0.6 are kept, although this is a user option.

Centrosymmetric case

A procedure similar to the one described above is used in the centrosymmetric case however the probability P^+ is converted to κ^{eq} directly via the relationship shown below

$$\kappa^{eq} = \left|\log_e\left(P^+ / 1 - P^+\right)\right|$$

(EQ 1.1.1.4)

In this section however, only the quartets and triplets of non-centrosymmetric structures will be tested since 1.1.1.4 is unambiguous and quick to compute.

1.1.2 Numerical Integration - Simpson's rule

Simpson's rule belongs to the group of classical formulas for integrating a function whose value is known at equally spaced steps. A function f(x) has known values at a sequence of abscissas

$$x_0 x_1 \ldots, x_N, x_{N+1}$$

(EQ 1.1.2.1)

which are spaced apart by a constant step h,

$$x_i = x_0 + ih$$

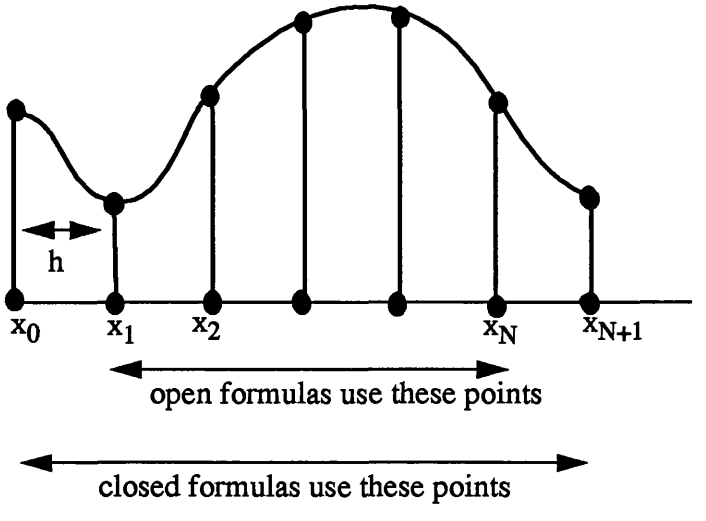
(EQ 1.1.2.2)

with $i = 0, 1, \ldots, N+1$.

(EQ 1.1.2.3)

The integration of the function f(x) occurs between a lower limit a and an upper limit b. If the integration formula uses the value of the function at the endpoints f(a) or f(b) this is called a *closed formula.* An *open formula* is one where the integral is estimated using only the x_i's between a and b. This is seen is Figure1 (Press, Flannery Teukolsky, Vetterling, 1986)

FIGURE 1. Integration of functions - open and closed formulas



Simpson's rule is an example of a closed Newton-Cotes formula:

$$\int_{x_1}^{x_3} f(x)\,dx = h\left[\frac{1}{3}f_1 + \frac{4}{3}f_2 + \frac{1}{3}f_3\right] + O\left(h^5 f^{(4)}\right)$$

(EQ 1.1.2.4)

In this case $f^{(4)}$ means the fourth derivative of the function f evaluated at an unknown place in the interval and $O\left(h^5 f^{(4)}\right)$ is the residual error.

1.1.3 The use of B as an estimate of reliability

The conditional probability equation for $P_{1/7}$ (Hauptman, 1975a,b, 1976b; Giacovazzo, 1975, 1976b) is:

$$P_{1/7} \approx \frac{1}{L} e^{-2B\cos\Phi_4} I_o\left(\frac{2\sigma_3}{\sigma_2^{3/2}} R_{12}X_{12}\right) I_o\left(\frac{2\sigma_3}{\sigma_2^{3/2}} R_{23}X_{23}\right) I_o\left(\frac{2\sigma_3}{\sigma_2^{3/2}} R_{31}X_{31}\right)$$

(EQ 1.1.3.1)

where B is:

$$B = \frac{1}{\sigma^{3/2}}\left(3\sigma_2{}^3 - \sigma_2\sigma_4\right) R_1R_2R_3R_4$$

(EQ 1.1.3.2)

and $R_1 = |E_{\underline{h}}|$, $R_2 = |E_{\underline{k}}|$, $R_3 = |E_{\underline{l}}|$, $R_4 = |E_{\underline{m}}|$

(EQ 1.1.3.3)

From EQ 1.1.3.2 and EQ 1.1.2 we can see that B and κ are defined in terms of |E| along with the number of atoms in the unit cell and are thus dependent on the individual structures. As discussed in Chapter 1, to produce a more reliable estimate for the quartets, the 3 cross terms are used in the conditional probability distribution $P_{1/7}$. Thus if we are to use B directly as an estimate, then the cross terms should also be included in the expression. Two equations for B were tested, each one dealing with the cross terms in a different manner. These equations are

:

$$B_1 = \frac{1}{\sigma^{3/2}}\left(3\sigma_2^{\,3} - \sigma_2\sigma_4\right)R_1R_2R_3R_4 \times \left(1.0 - R_{12}^{\,2} - R_{23}^{\,2} - R_{31}^{\,2}\right)$$

(EQ 1.1.3.4)

(Hauptman, 1974)

$$B_2 = \frac{1}{\sigma^{3/2}}\left(3\sigma_2^{\,3} - \sigma_2\sigma_4\right)R_1R_2R_3R_4 \times \left(R_{12}^{\,2} + R_{23}^{\,2} + R_{31}^{\,2} - 2.0\right)$$

(EQ 1.1.3.5)

(Hauptman, 1976b)

where $R_{12} = |E_{\underline{h}+\underline{k}}|$, $R_{23} = |E_{\underline{k}+\underline{l}}|$, $R_{31} = |E_{\underline{l}+\underline{h}}|$

(EQ 1.1.3.6)

To compare these two methods the average phase error is plotted against increasing reliability and the resulting trend in the graphs noted. The trend should be one of decreasing phase error with increasing reliability. These graphs will then be compared to decide whether the direct use of B as a measurement of reliability is viable or not. As a standard, the κ value will also be plotted against its corresponding phase error, to test its power as an estimate of reliability for the triplets for the same structure.

2.0 EXPERIMENTAL

The quartets using B_1 or B_2 were integrated into the QUARTET subroutine. This subroutine was kept intact and the new equations added as options so the resulting subroutine contained both the original and the new expressions. The reliability of each quartet was then calculated using either the κ_{eq}, B_1 or B_2 methods described earlier. The known phases of each reflection were input with the Miller indices and the structure factors during normalisation and used to calculate the true value of the quartet. To test all 3 equations fully, negative and positive quartets are treated separately. Apart from entering the command to read the known phases and the command for the generation of the positive quartets, the relevant modules of MITHRIL94 were run completely by default. The reliability of the quartet and its corresponding true value were stored in a file and used to calculate the average phase error. The whole procedure was repeated, until all methods had been tested. The general process can be seen in the flow diagram below.

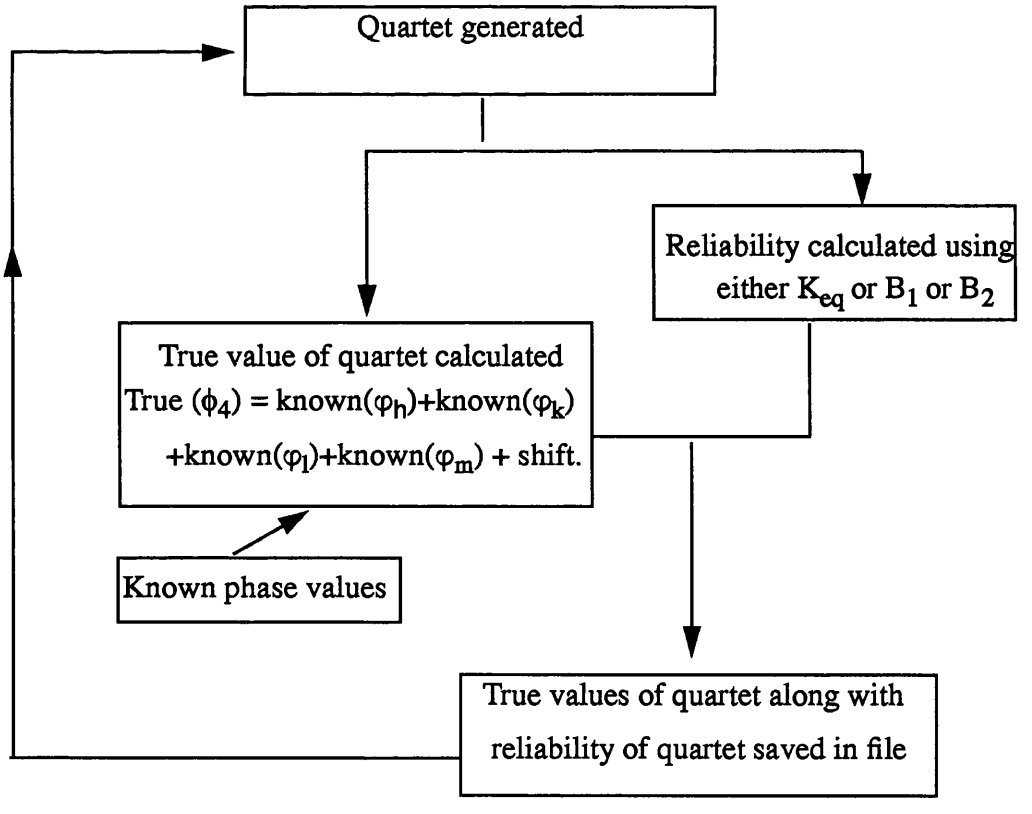


FIGURE 2. Flowchart for calculating the true value and the reliability of a quartet

The shift value used to calculate the true value of the quartet is a consequence of translational symmetry. The true value and the reliability of the quartet were stored in a computer file which was then accessed by a small FORTRAN program to calculate the average phase error of quartets at certain ranges of reliability. This process is shown in the below.

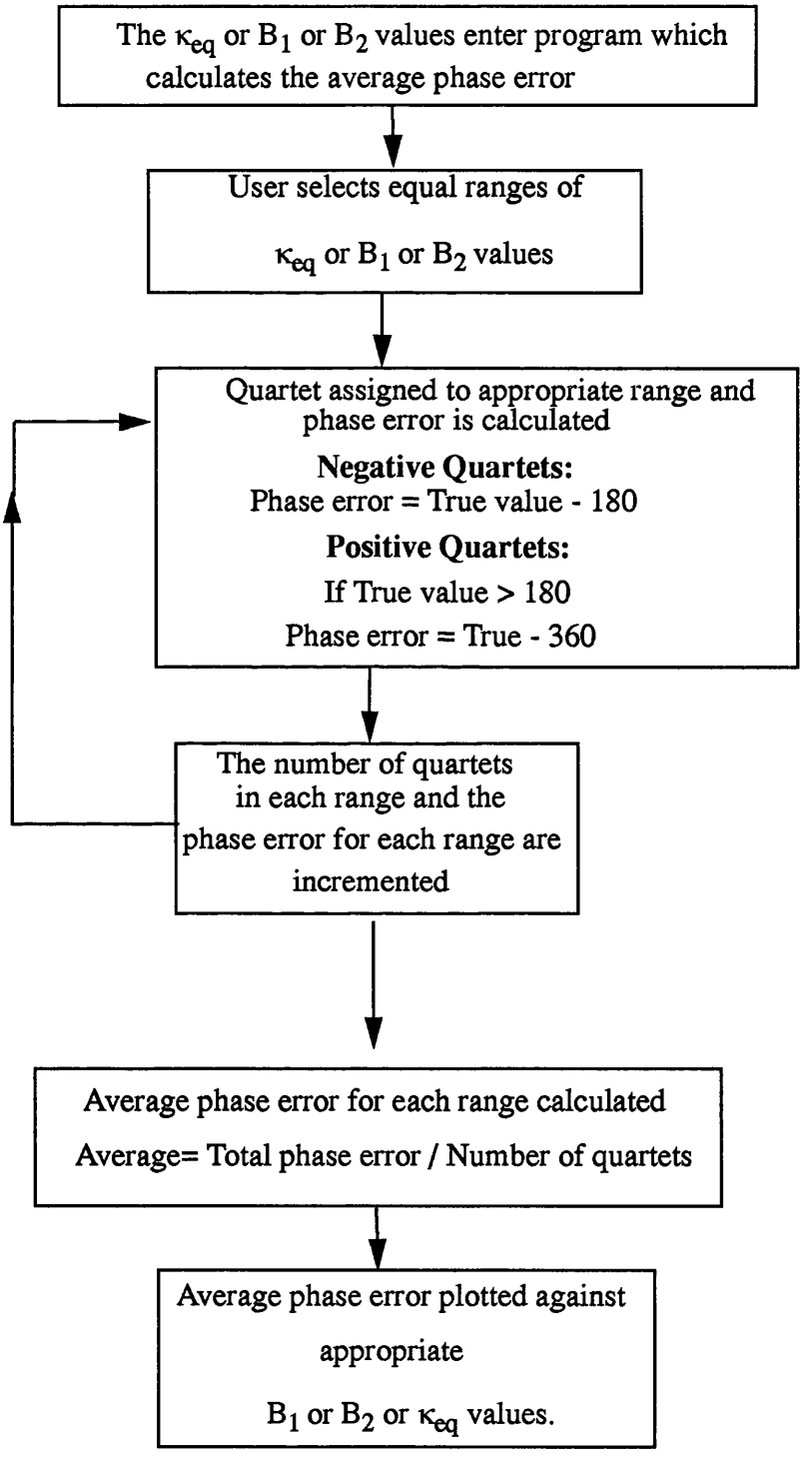


FIGURE 3. Flowchart for calculating the average phase error of a quartet

A similar process was carried out for the triplets:

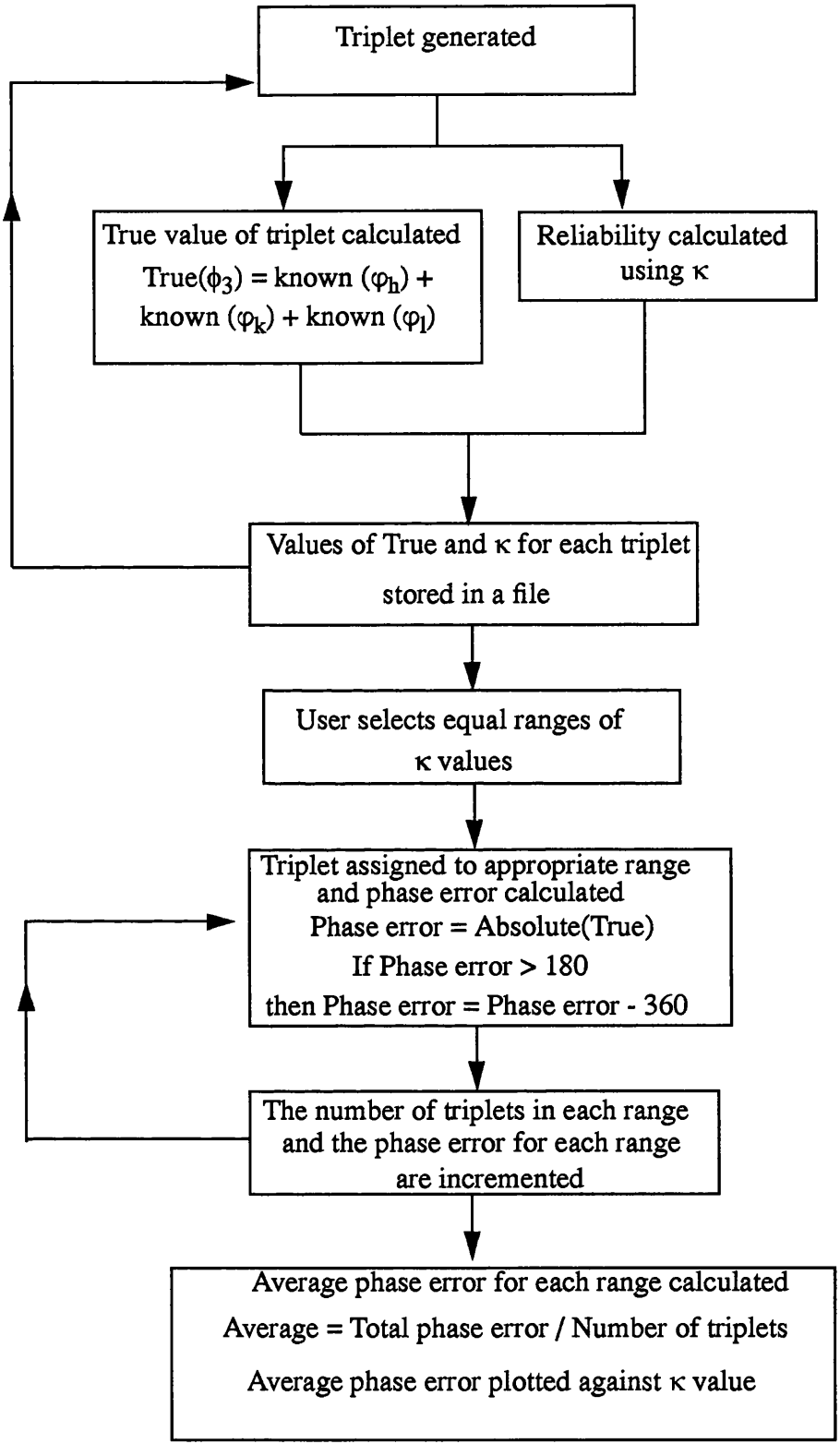


FIGURE 4. Flowchart for calculating the average phase error of a triplet

Graphs were constructed using the program XGRAPH which is a public domain graphics program written by David Harrison, the University of California. The graphs plotted the average phase error against κ_{eq}, B_1, B_2 or κ.

2.1 Smoothing the graphs

The data for the graphs were smoothed as a graphical technique and the program used was called SMOOFT from Numerical Recipes (Press, Flannery, Teukolsky, Vetterling, 1986). This program smooths an array of ordinates (y's) which in this case are the phase errors, in order of increasing abscissas (x's) which in this case are the κ_{eq}, B_1 or B_2. Firstly the program removes any linear trend, then a fast Fourier transform is used to filter the data. The linear trend is re-inserted at the end. The amount of smoothing, that is the number of points the data must be smoothed over, is user controlled. This number must never be greater than half the number of original points as this results in the data losing features. For all the structures this was investigated thoroughly until smooth graphs were obtained. These test graphs were then compared to see which method gives the most reliable indication for the quartets.

2.2 Structures investigated

The structures used in the investigation were all chosen from the Sheldrick database of difficult structures, so called because direct methods have difficulty solving them or because they are in unusual high symmetry space groups. Reasons why some of the structures create a problem for direct methods are:

- Their space group symmetry.
- They contain heavy atoms.
- Poor quality X-ray data.
- Large number of atoms in asymmetric unit.

However for many of the structures it is impossible to quote just one definite reason which results in their failure to solve. From previous investigations it was discovered that quartets played an active role in the solution of some of the structures and four such structures were tested here. In total, six structures were investigated.

These were:

TABLE 1. Difficult structures tested

Structure	Space Group	Formula
Azet	$Pca2_1$	$C_{21}H_{16}ClNO$
Bed 2	I4	$C_{26}H_{26}N_4O_4$
Loganin	$P2_12_12_1$	$C_{17}H_{26}O_{10}$
Gold2	Cc	$C_{28}H_{16}$
Munich1	C2	$C_{20}H_{16}$
Apapa	$P4_12_12$	$C_{30}H_{37}N_{15}O_{16}P_2.6H_2O$

- Azet - The generation of quartets helped in the solution of this structure which is known to be unstable under tangent refinement.
- Bed - Quartets are used actively in the convergence mapping.
- Loganin - This structure does not cause MITHRIL94 any real problems.
- Goldman2 - Quartets again play an active role in solving the structure.
- Munich1 - The reliability of the quartets is important as very few quartets produced.
- Apapa - Again, few quartet relationships were generated but they play an active role in solving the structure.

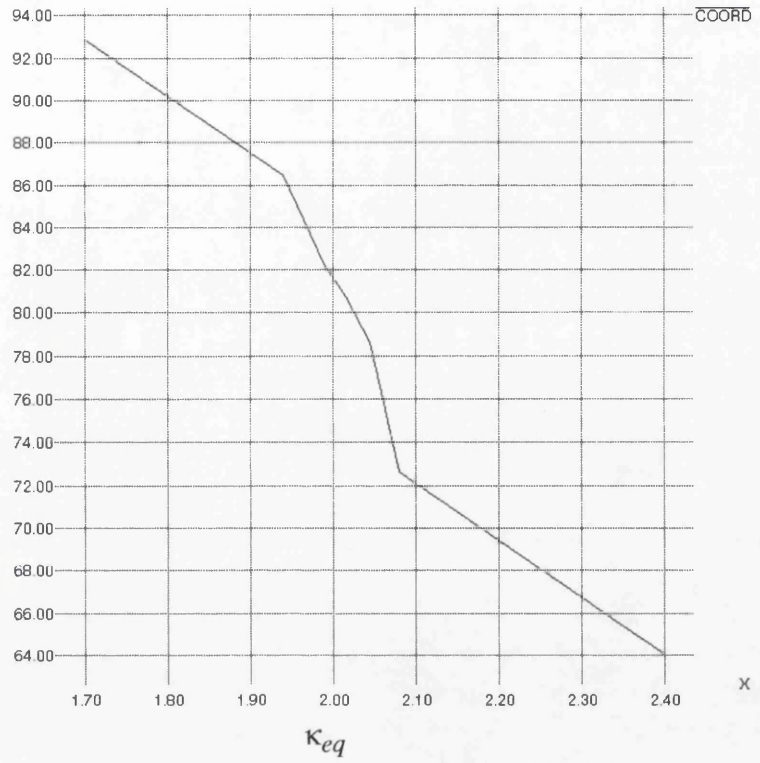
Obviously just using B_1 or B_2 has the advantage that the process will be on the whole faster, but it must still provide an accurate measure of reliability. The graphs for only some of the structures are shown as most of the structures produced graphs showing the same trend. The results for all of the structures are shown in a table at the end of the chapter.

3.0 RESULTS

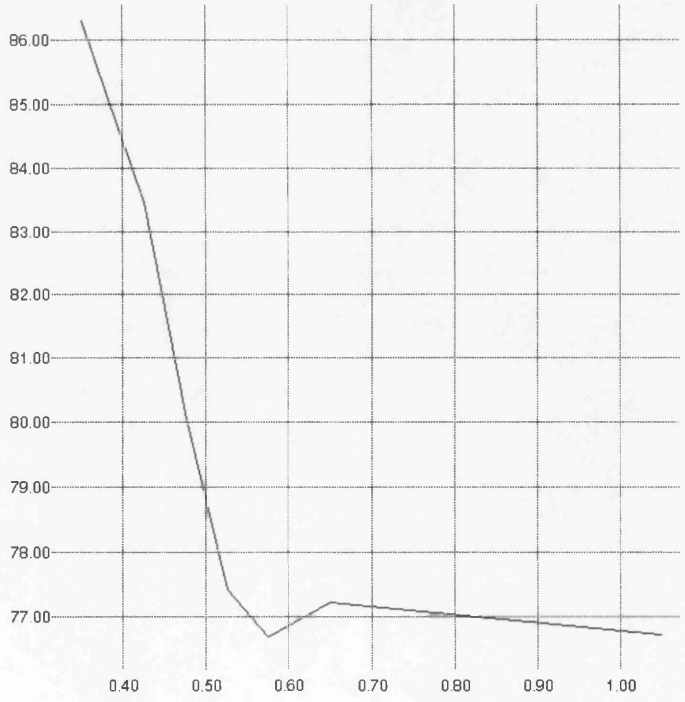
3.1 Negative Quartets

FIGURE 5. AZET

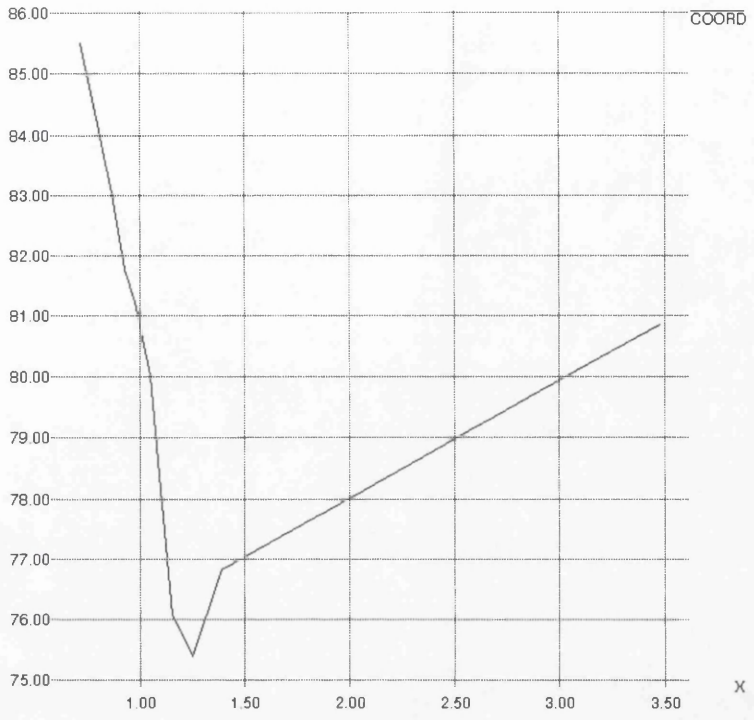
(a.) Azet κ_{eq} *phase error*



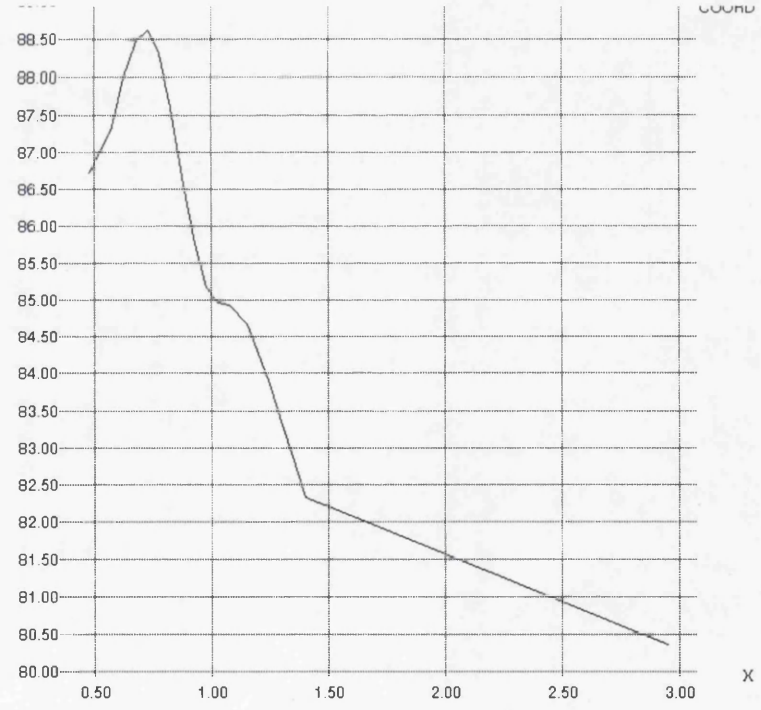
(b.) Azet B_1



(c.) Azet B_2



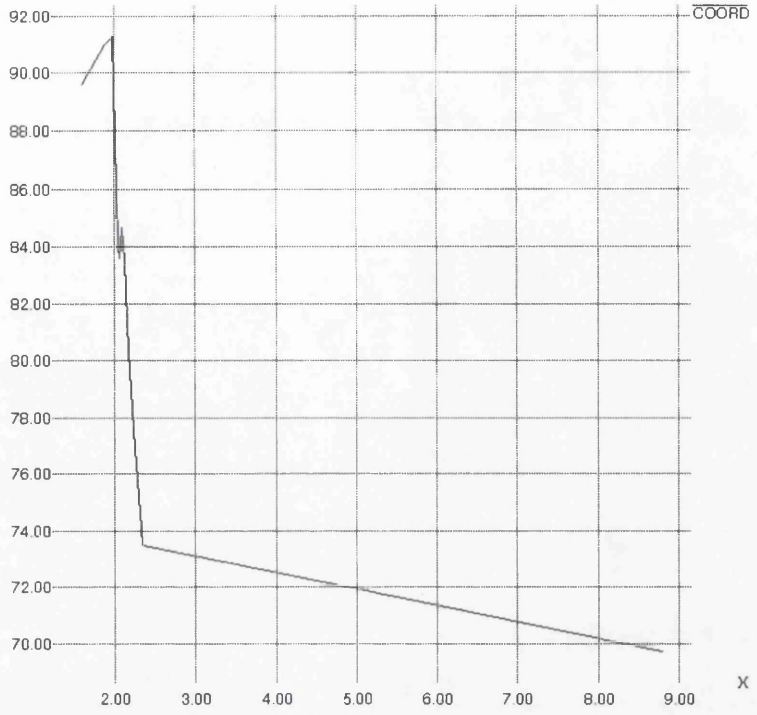
(d.) Azet Triplets



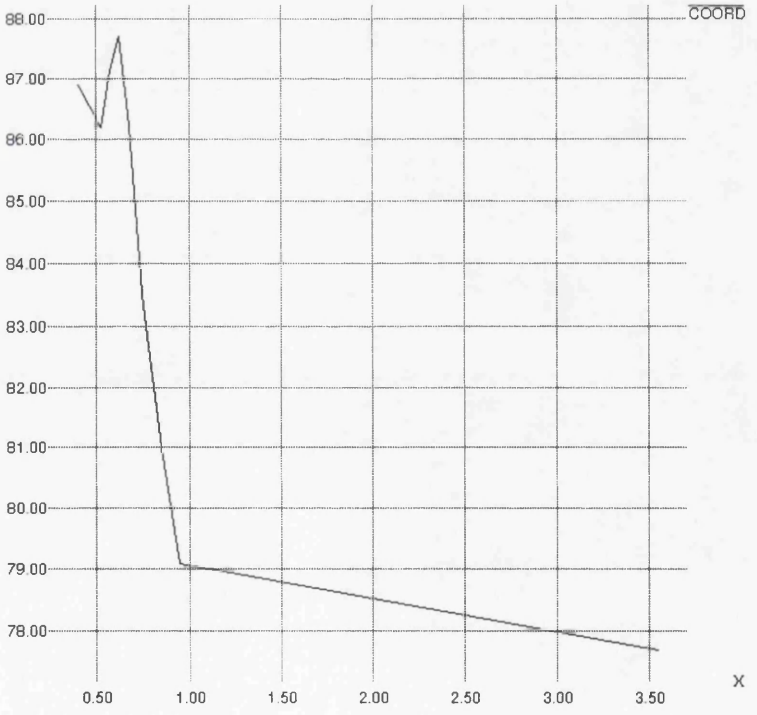
The triplets for this structure behave as expected, exhibiting decreasing phase error as κ increases. A decrease in phase error is more pronounced for κ_{eq}, followed by B_1, indicating that B_1 and not B_2 is a viable alternative method for scaling the reliability of this structure.

FIGURE 6. BED

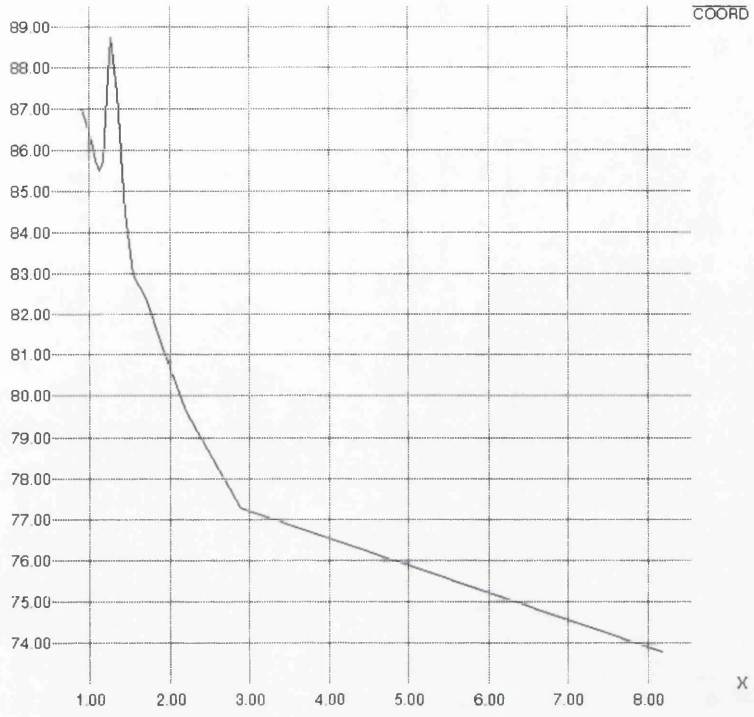
(a.) Bed κ_{eq}



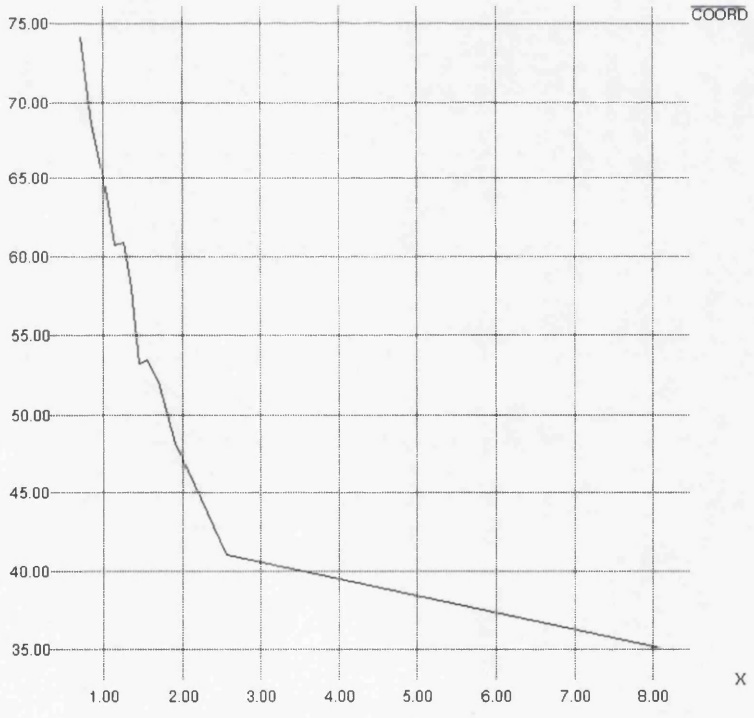
(b.) Bed B_1



(c.) Bed B_2



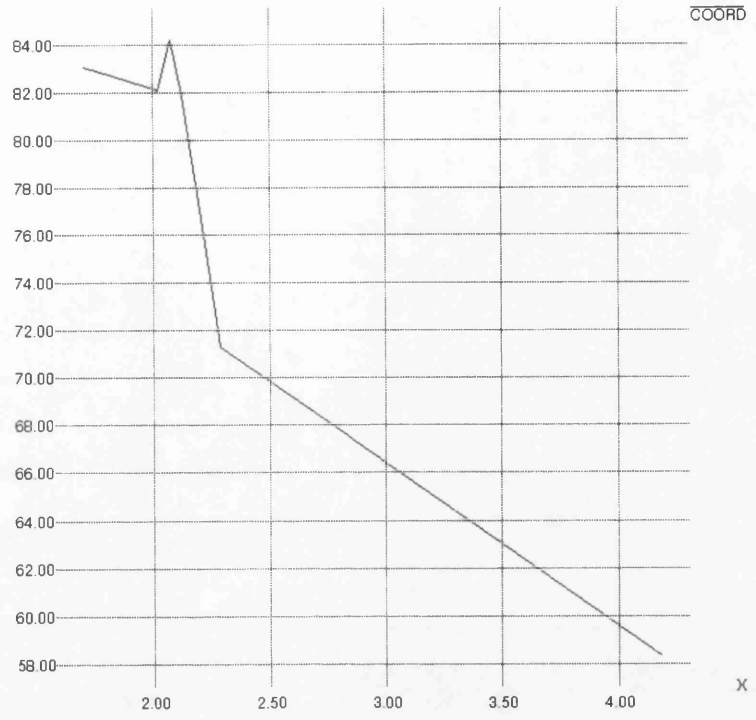
(d.) Bed Triplets



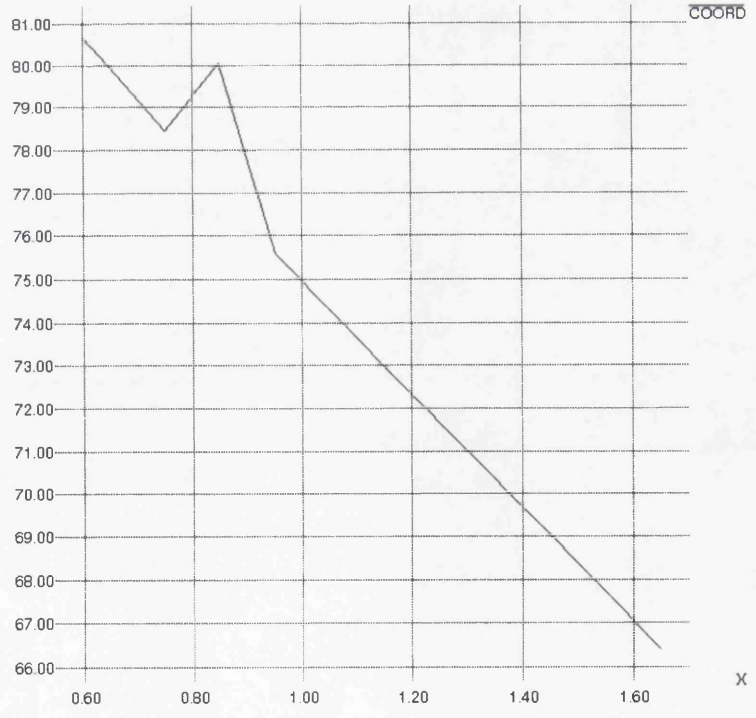
The triplets again follow the correct trend showing decreasing phase error. A decrease in phase error also occurs for κ_{eq}, B_1 and B_2 indicating that all three methods would prove viable methods of estimation for this structure.

FIGURE 7. MUNICH1

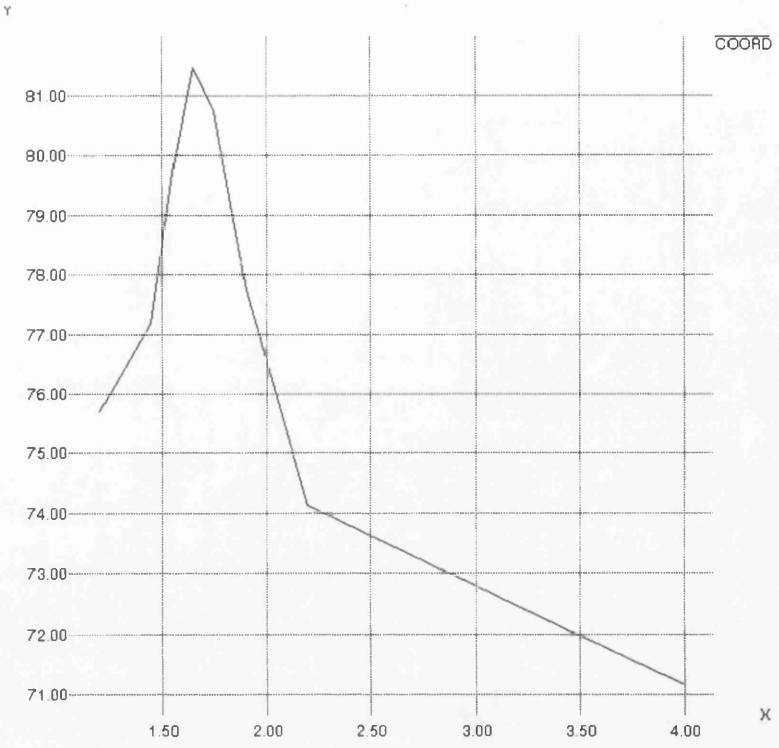
(a.) Munich1
κ_{eq}



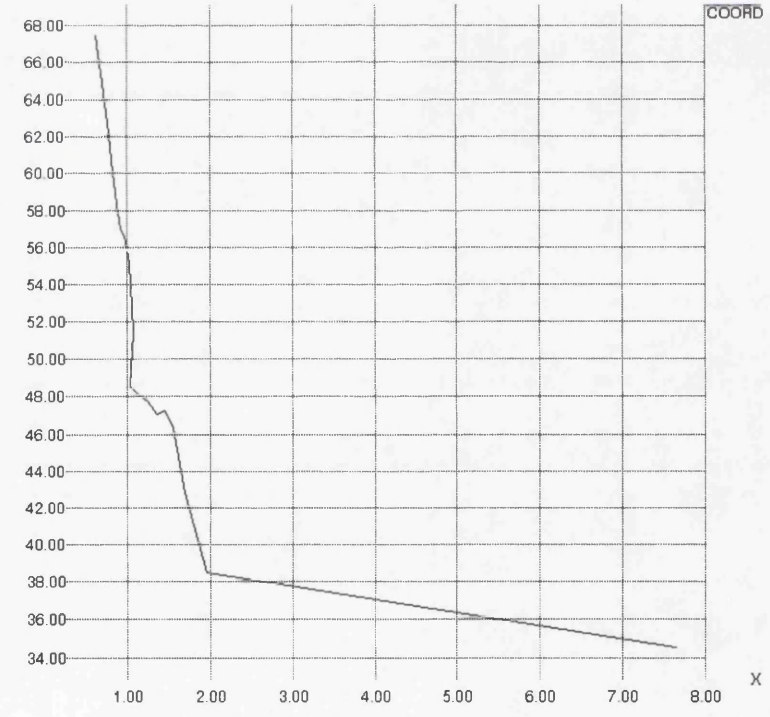
(b.) Munich1
B_1



(c.) Munich1 B_2



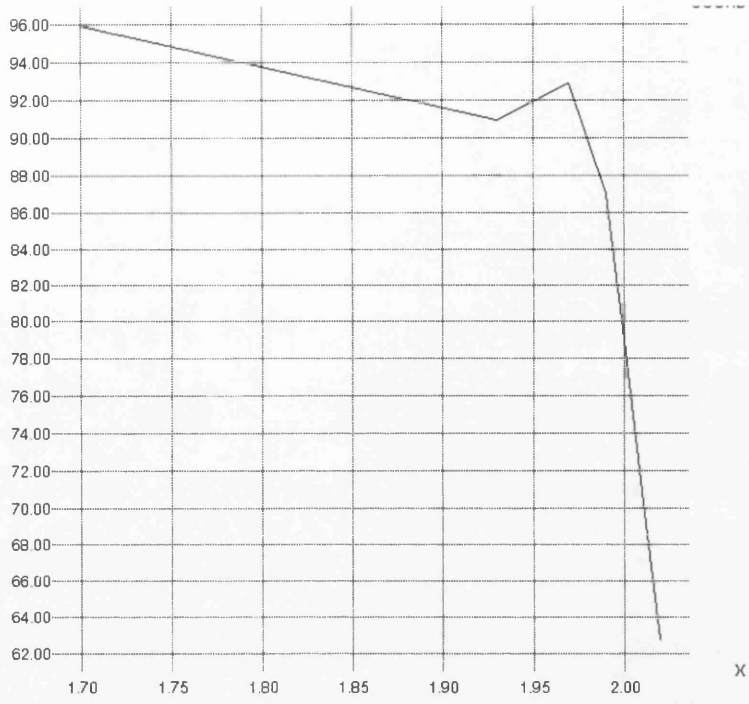
(d.) Munich1 Triplets



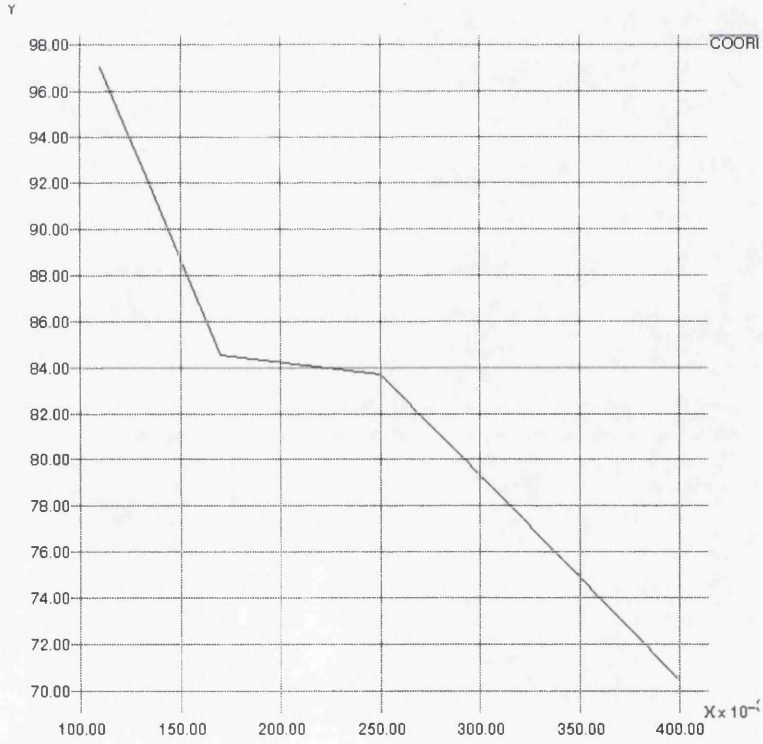
All three methods have an overall decrease in phase error. This again is more pronounced with κ_{eq} and B_1, although the final phase error produced is smaller for κ_{eq}.

FIGURE 8. APAPA

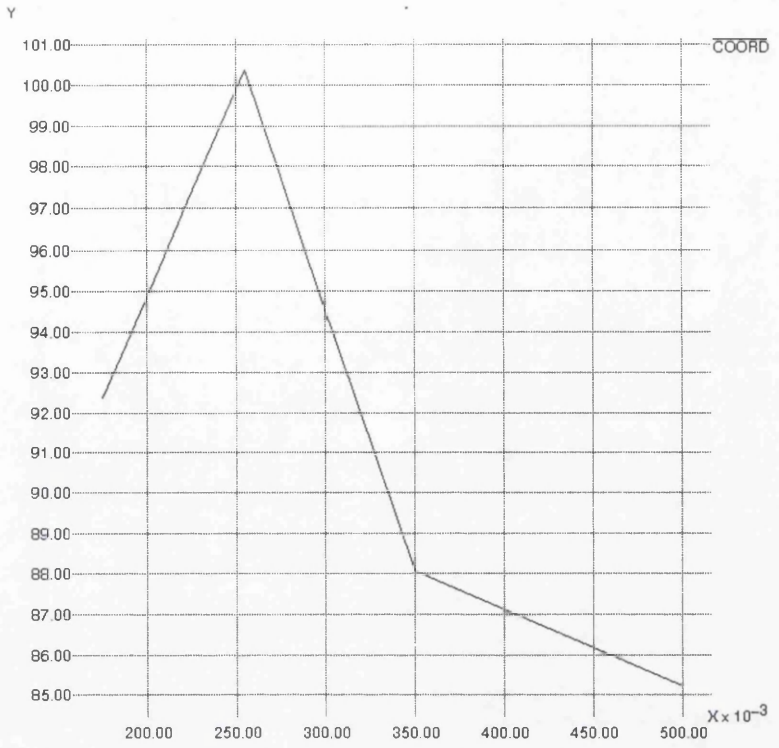
(a.) Apapa κ_{eq}



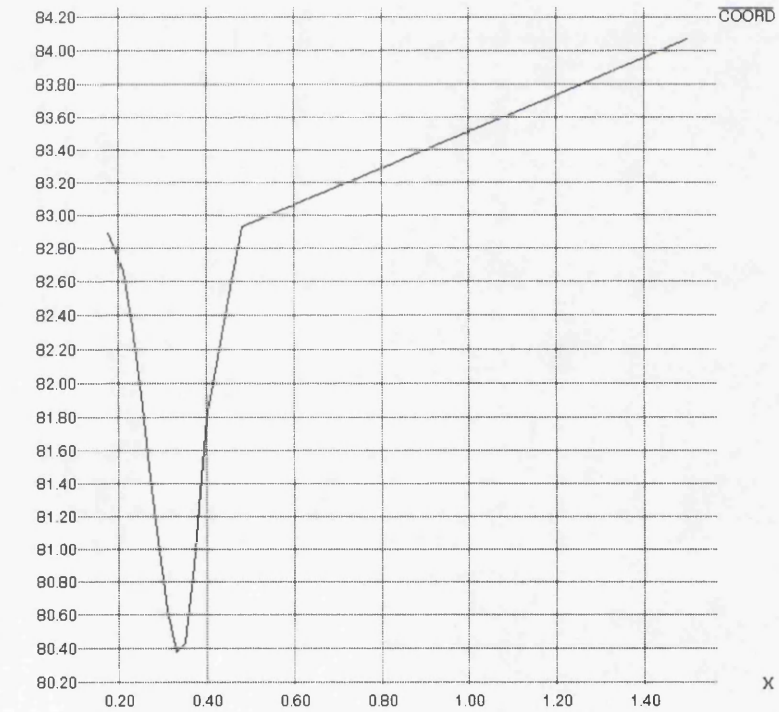
(b.) Apapa B_1



(c.) Apapa B_2



(d.) Apapa Triplets

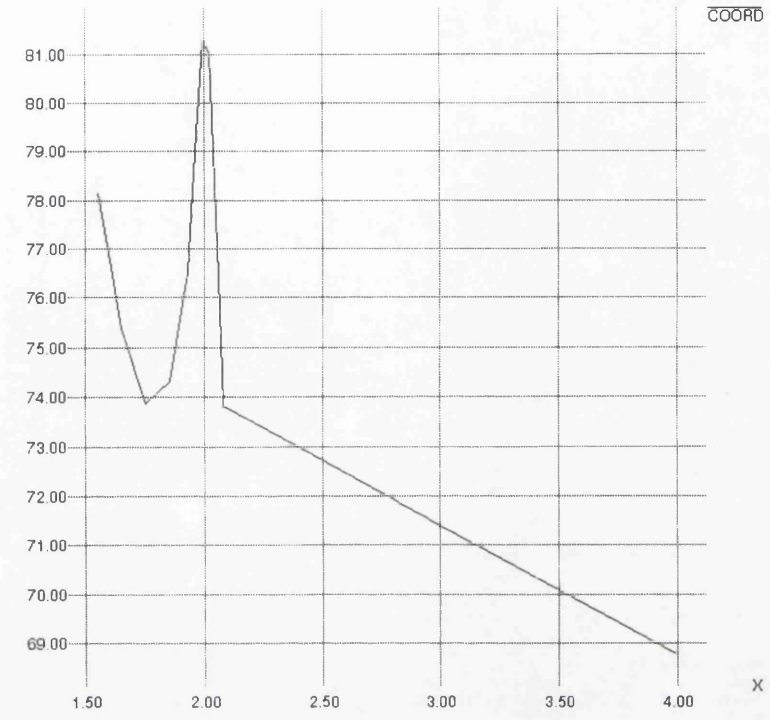


Again, the best graph produced for the quartets is κ_{eq} followed closely by B_1. The graph for the triplets shows an increase in the phase error. This results in the convergence map for this structure having few contributors.

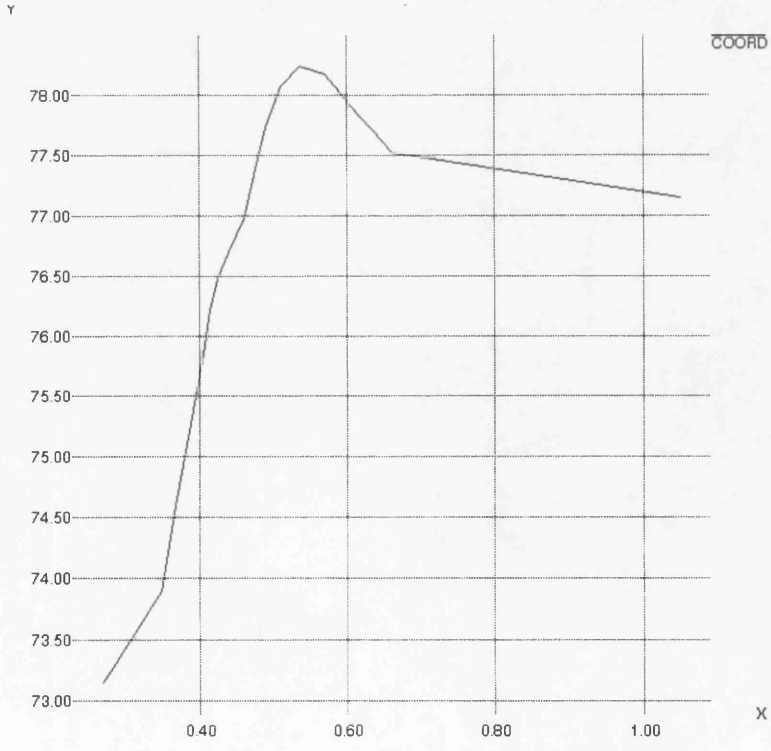
3.2 Positive Quartets

FIGURE 9. AZET

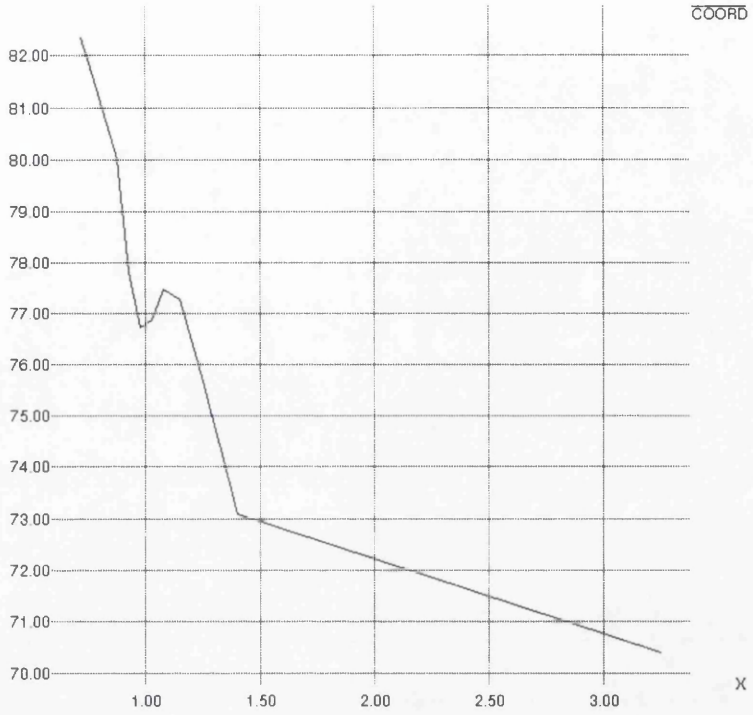
(a.) Azet κ_{eq}



(b.) Azet B_1



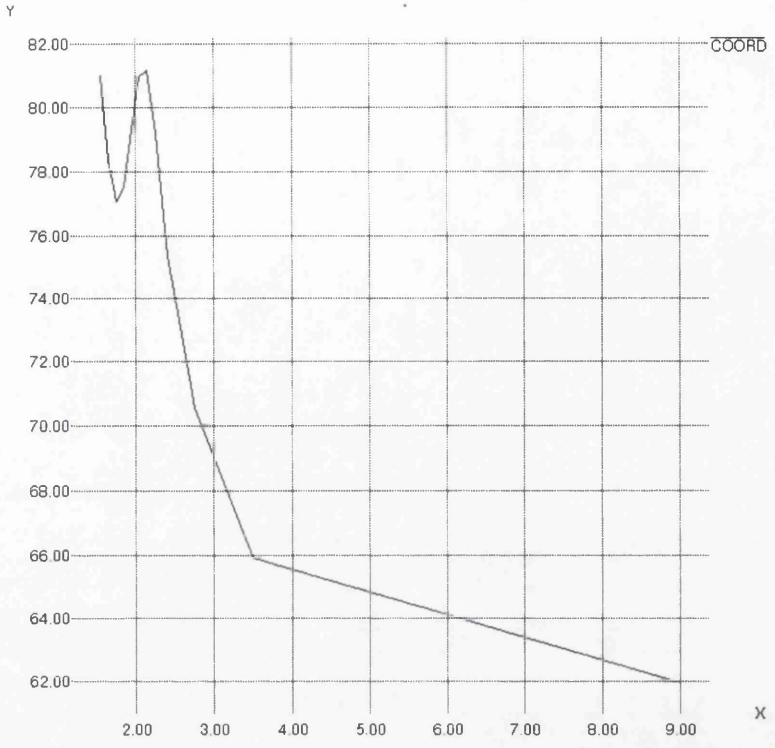
(c.) Azet B_2



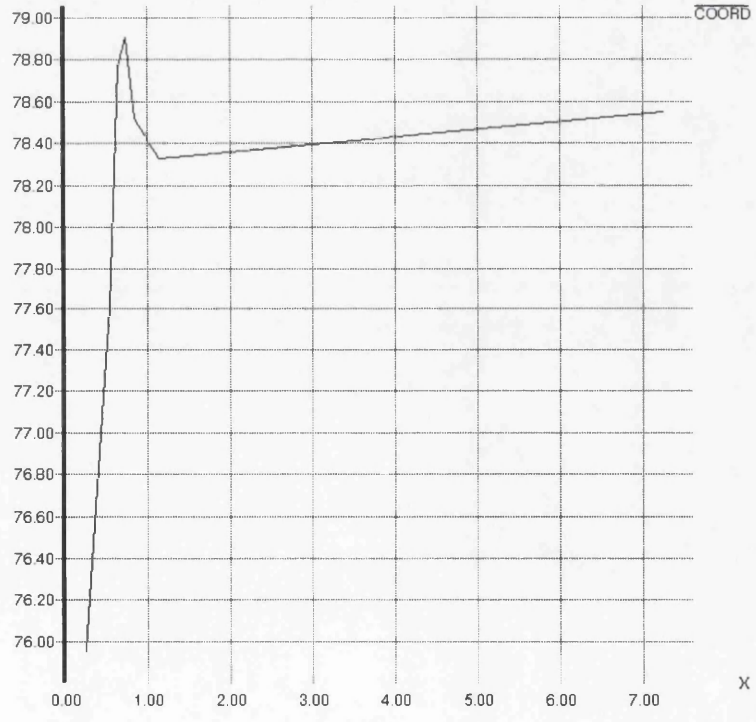
Azet B_1 has a slight increase in phase error whereas B_2 shows a decrease. The graph for κ_{eq} shows a sharp increase then decrease, so for the generation of positive quartets B_2 appears to produce the best variable to use.

FIGURE 10. BED

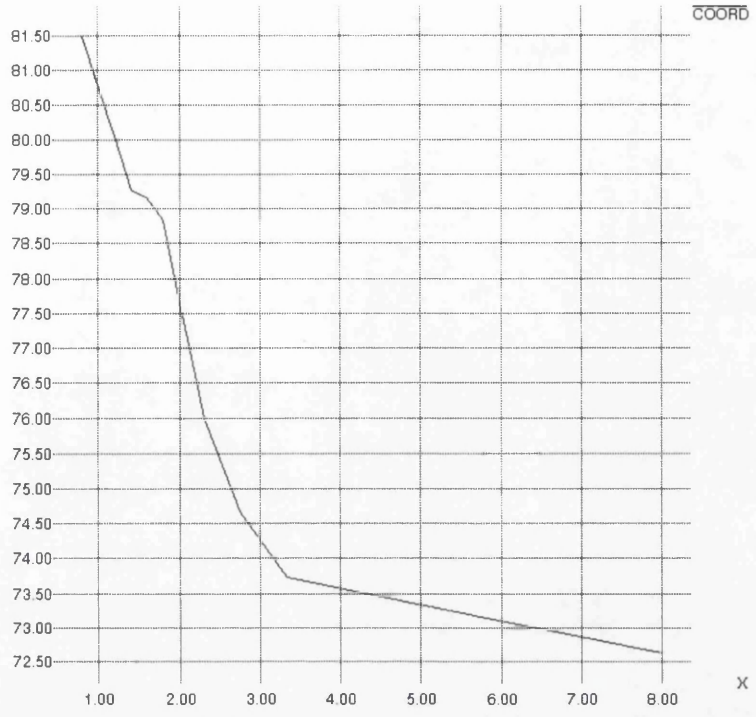
(a.) Bed κ_{eq}



(b.) Bed B_1



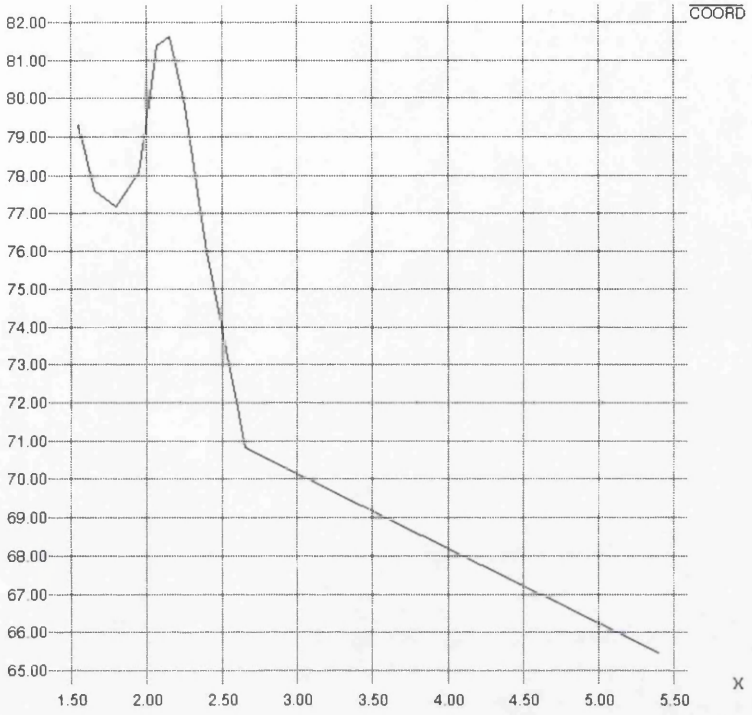
(c.) Bed B_2



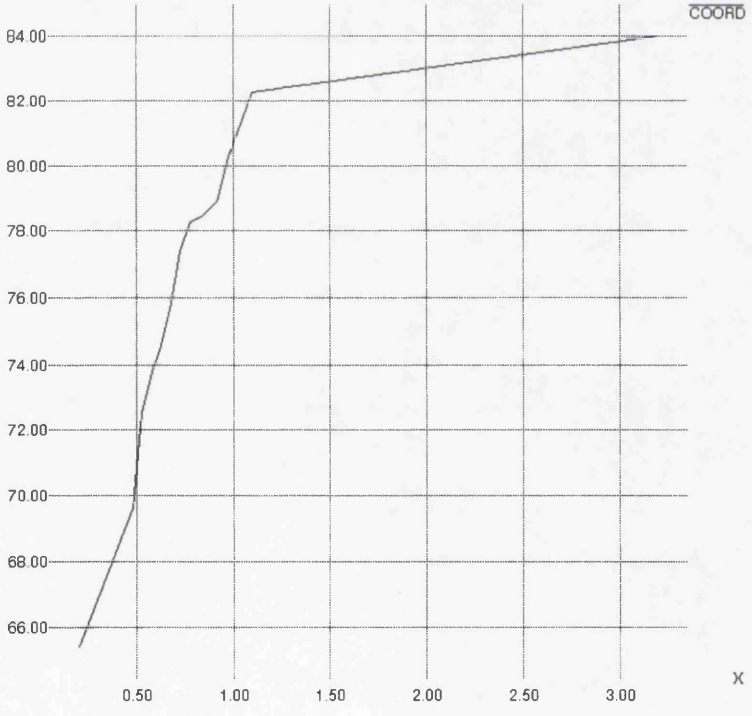
For this structure B_2 again produces the best results, along with κ_{eq}, with a consistent decrease in phase error. However κ_{eq} produces the lowest value of phase error. The graph produced for B_1 goes against the trend with an increase in phase error.

FIGURE 11. Munich1

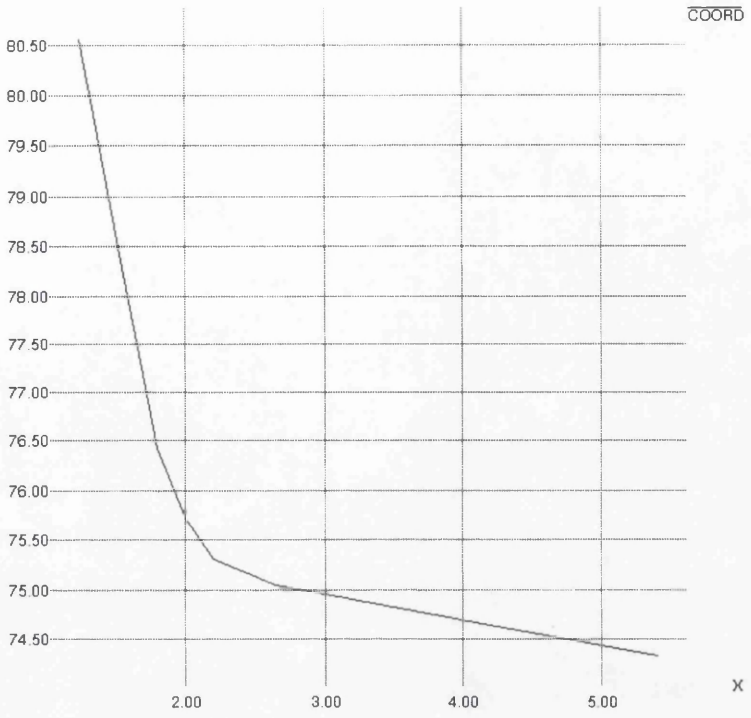
(a.) Munich1
κ_{eq}



(b.) Munich1
B_1



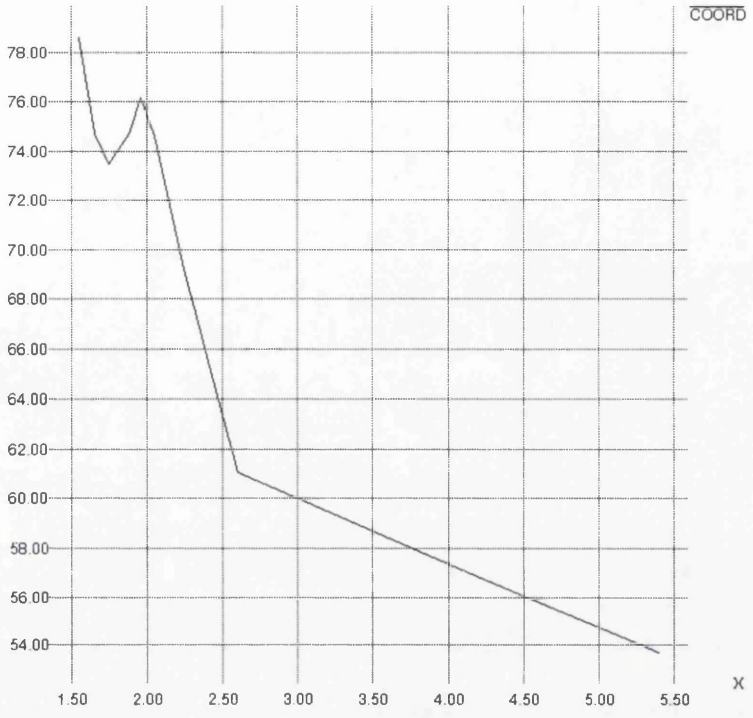
(c.) Munich1
B_2



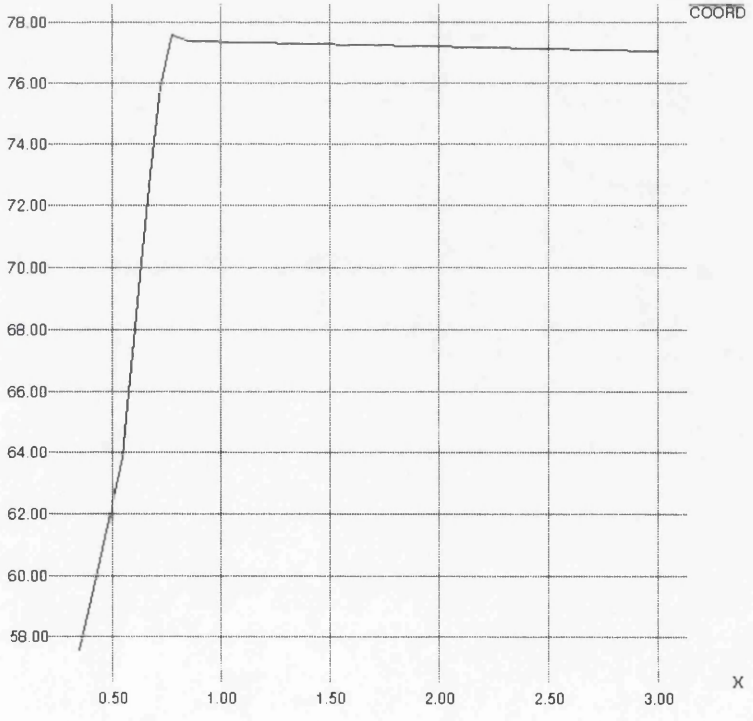
The graphs again indicate that B_2 and κ_{eq} are viable techniques for scaling positive quartets.

FIGURE 12. Gold2

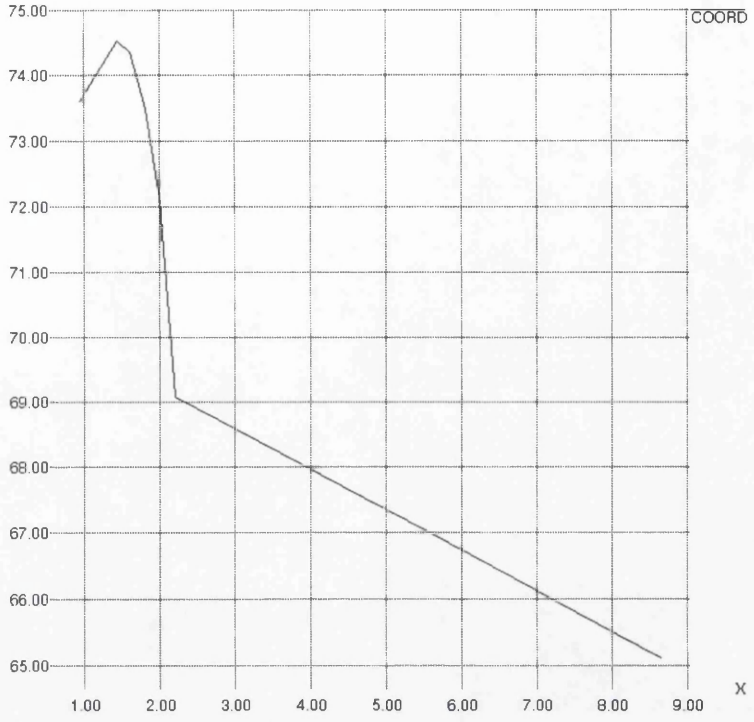
(a.) Gold2
κ_{eq}



(b.) Gold2
B_1



(c.) Gold2
B_2



The graphs again indicate that B_2 and κ_{eq} are viable techniques for scaling positive quartets whereas the phase error increases with B_1.

4.0 DISCUSSION

Table2 below shows a summary of results for all the structures, where + indicates a good result and - a poor result:

TABLE 2. Phase error with increasing reliability

	Positive Quartets			Negative Quartets			Triplets
Structure	κ_{eq}	B_1	B_2	κ_{eq}	B_1	B_2	κ
Munich1	+	-	+	+	+	+	+
Loganin	+	+	+	+	+	+	+
Gold2	+	-	+	+	+	-	-
Bed 2	+	-	+	+	+	+	+
Azet	+	-	+	+	+	-	+
Apapa	+	-	-	+	+	+	-

By looking first at the results for the negative quartets it is obvious that, on the whole all three methods provided an accurate measurement of reliability. For most structures κ_{eq} produced the best graphs followed by B_1, so B_1 could be used as a quicker easier alternative for estimation of the negative quartets. B_2 was a success for 4 of the 6 structures so in some cases it could also be used as a alternative.

From the results produced by the positive quartets, it is obvious that κ_{eq} and B_2 provide the most accurate measures of reliability. B_1 is only able to produce an accurate measurement for loganin, a structure that has never caused MITHRIL94 any problems. Thus an alternative method for estimation for the positive quartets would be the use of B_2. From Table 2 we can also conclude that κ_{eq}, the method which is presently in MITHRIL94, produces a very accurate estimation of reliability. The use of B_1 and B_2 increases the speed of the program by approximately a factor of 10 thus for the investigations discussed in the Chapters which follow, it is B_1 that is used to measure the reliability of the negative quartets.

References

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

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

Giacovazzo, C. (1975). *Acta Cryst.* **A31,** 252-259

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

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

Gilmore, C.J. & Brown, S.R. (1988). *J. Appl. Cryst.* **21,** 571-572

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

Hauptman, H. (1975a). *Acta Cryst.* **A31,** 671-679

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

Hauptman, H. (1976b). *Acta Cryst.* **32,** 877-882

Press, W. H. & Flannery, B. P. & Teukolsky, S. A. & Vetterling, W. T. (1986). *NUMERICAL RECIPES, The Art of Scientific Computing* 103-105, Cambridge University Press

CHAPTER 3

PHASE ANNEALING AS A METHOD OF REFINEMENT IN MITHRIL94

1.0 OPTIMISATION TECHNIQUES

1.1 Theory

Optimisation techniques are used to find the value of those variables which maximise or minimise a function. This also permits the value of the function at these extrema to be calculated. Finding the maximum and minimum of a function is basically the same problem, as the value of x which maximises the function f(x) minimises -f(x). Functions to be optimised depend on one or more independent variables and the optimisation is described as being constrained or unconstrained depending on whether constraints on the variables exist. The constraints may be either linear or non-linear.

The global minimum is the lowest possible value of the function. This is the desired end point of an optimisation. However there also exist values of the function, described as local minima, where the value is the lowest in a finite neighbourhood. These can create problems as they make it more difficult to find the actual global minima.

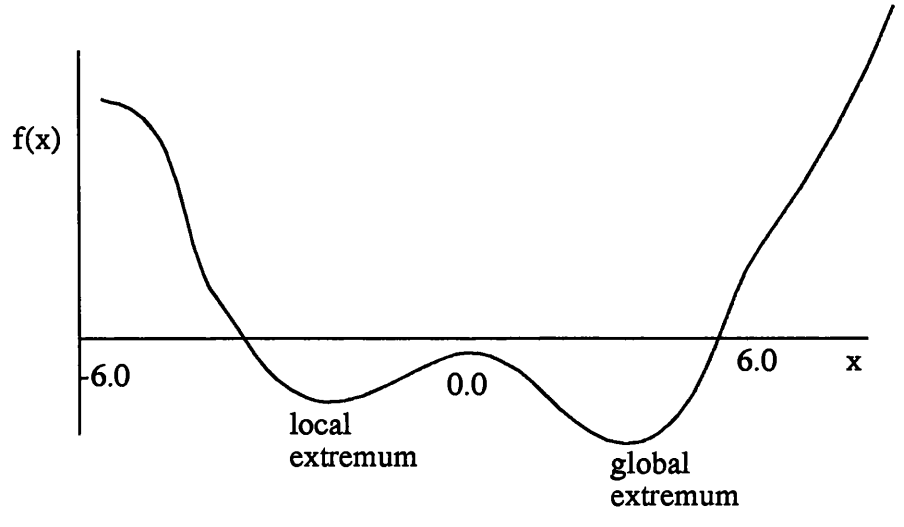


FIGURE 1. Extrema of a function

There are two main strategies for finding the global minimum (Press, Fannery, Tuekolsky and Vetterling, 1986):

- Begin from a large range of initial values for the variables and collate all information about local minima found. The global minimum is the smallest of these values.

Or

- Once a local minimum has been found, change its value slightly, optimise the function again and examine to see if the minimum found is the original one or a new better (lower) one.

There are various methods of optimisation for finding the minimum of a function, but one of the simplest and oldest is the Newton - Raphson method which involves the calculation of the first and second derivative. EQ 1.1.1 shows the technique

$$x_{new} = x_{initial} - \frac{f(x)}{f'(x)}$$

(EQ 1.1.1)

Firstly the variable x is given an initial value denoted $x_{initial}$ and the first and second derivatives are calculated. Using the above equation, a value of x_{new} is found which is now used as the new estimate for x and the whole process begins again. The Newton - Raphson method is an iterative technique, i.e. a technique that starts with a known or guessed initial value and is repeated until a new configuration is found. This new configuration becomes the new starting position and the process is repeated until the system shows no improvement. This technique is only viable when first and second derivatives exist for the function f(x). Another problem for this technique which is universal to all optimisation techniques is that if the function falls into a local minimum it is very difficult for it to escape and find the global minimum (Press, Fannery, Tuekolsky and Vetterling, 1986).

2.0 CRYSTALLOGRAPHIC OPTIMISATION PROBLEMS

Various optimisation problems exist throughout the stages of crystal structure solution:

2.1 Parameter Refinement

When there exists several independent observations of some quantity x:

$$x_i, i = 1, 2, 3, \ldots, n$$

(EQ 2.1.2)

and where the x_i do not agree, then it is convenient to pick the 'best fit' for x_i. Several different techniques could be used to calculate the 'best fit':

- If the probability of an error of a given size fell sharply to zero at some particular size and $x_1 < x_2 < \ldots < x_n$, then the choice would be $\frac{1}{2}(x_1 + x_n)$. This minimises the absolute error $|x-x_1|$. However experimental errors rarely fall in this category.
- If a few of the x_i have large errors but the majority of the x_i are highly accurate and $x_1 < x_2 < \ldots < x_n$, then the best fit would be $x_{(n+1)/2}$ if n is odd and $\frac{1}{2}[x_{n/2} + x_{(n/2+1)}]$ if n is even.

The technique used to refine parameters in crystallography is the method of least squares. The simple form of least squares is represented by:

$$\sum_{i=1}^{n} (x - x_i)^2$$

(EQ 2.1.3)

This is suitable when the x_i are all expected to be equally accurate. In crystallography, the least squares equation is (Rollet, 1984):

$$\sum_{\underline{h}} w_{\underline{h}} \left[\left| F_{\underline{h}}^{obs} \right| - \left| F_{\underline{h}}^{cal} \right| \right]^2$$

(EQ 2.1.4)

where $F_{\underline{h}}^{calc}$ is a function of both atomic coordinates and thermal parameters and

$w_{\underline{h}}$ is a weight related to the variance of observation.

The relevant observations are non linear and the equations are linearised using the Taylor series. The unknown quantities are called the parameters. The solution of the linearised equations via matrix algebra produces a new set of parameters, which should be an improvement on the original. Iteration continues until the 'best fit' is produced. This technique has the disadvantage that it can be trapped in local minima.

2.2 Unit cell determination

The orientation matrix lies at the centre of the whole data collection process. The correct diffractometer angles can be calculated for any reflection, once this matrix has been determined. The matrix is seen below;

$$A = \begin{bmatrix} a'_x & b'_x & c'_x \\ a'_y & b'_y & c'_y \\ a'_z & b'_z & c'_z \end{bmatrix}$$

FIGURE 2. The orientation matrix

where a', b', c' are the crystal fixed set of reciprocal lattice axes and the z axis is coincident with the diffractometer ϕ axis.

x and y are defined as this second axis set.

These nine elements contain information about the unit cell and orientation of the crystals. The process of data collection involves finding reflections, assigning integer indices to them and then determining the unit cell and orientation. The initial matrix and cell are mostly determined from low-angle reflections and are not precise. The unit cell parameters must be optimised by refinement and checked. The most common method is linear least squares refinement of the nine elements from reflection indices and reciprocal space coordinates calculated from the observed diffractometer diffraction angles (Tichy, 1970).

2.3 The tangent formula in direct methods

For most direct methods, random phases are input into the tangent formula and optimised using this formula:

$$\tan\varphi_h \approx \frac{\sum_{\underline{k}} w_{\underline{k}} w_{\underline{h}-\underline{k}} |E_{\underline{k}}||E_{\underline{h}-\underline{k}}| \sin(\phi_{\underline{k}} + \phi_{\underline{h}-\underline{k}})}{\sum_{\underline{k}} w_{\underline{k}} w_{\underline{h}-\underline{k}} |E_{\underline{k}}||E_{\underline{h}-\underline{k}}| \cos(\phi_{\underline{k}} + \phi_{\underline{h}-\underline{k}})}$$

(EQ 2.3.1)

where $w_{\underline{k}}$, $w_{\underline{h}-\underline{k}}$ are weights.

With random phases the starting point can be a long way from the correct phases and the equations are non linear, so again there is a danger that the optimisation is stuck in a local minima. Simulated annealing is an optimisation technique that is able to escape local minima to find the global minimum. The technique of simulated annealing in phase refinement was introduced by Sheldrick (Sheldrick, 1990) as a method of improving the radius of convergence of the tangent formula. The latter parameter is described as the maximum distance between the sampling points and the remaining points of a full factorial design. Before discussing this technique (called phase annealing) and its implementation into MITHRIL94, the similarity between optimisation problems and statistical mechanics must be discussed to describe the relevance of simulated annealing to a non-thermodynamic system.

3.0 Combinatorial Optimisation - The Travelling Salesman

Combinatorial optimisation applies to a situation where the space over which the function is defined is not N-dimensional space of N continuously variable parameters, but a large discrete configuration space where the numbers of elements in the space is so large that it is not possible to try every combination fully (Christofides, Mingozzi, Toth and Sandi, 1979). Phase space can be thought of to be both N-dimensional space of N continuously variable parameters (acentric reflections) and also a discrete large configuration space (centric reflections). One of the most famous combinatorial optimisation problems is that of the *travelling salesman.* The salesman has a list of N cities and the distances between them. He must then plan a route where he is able to visit each city just once, using the shortest mileage possible. Again, one of the two strategies explained earlier can be put into action, that is start from an initial point (solution) and change factors to get a better solution only stopping when no more improvements can be made. For example, the initial random route of a certain length L is changed by swapping the sequence the cities are visited in. This is shown diagrammatically in Figure 3 (James, 1989)

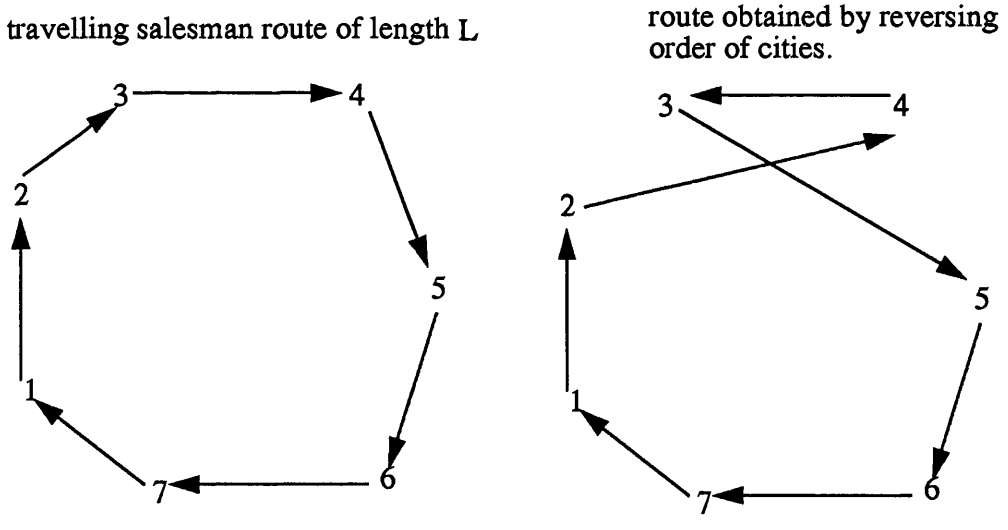


FIGURE 3. The travelling salesman

If this new route is shorter than the original then it taken to be the new best route and the process is continued, if not more cities from the original route are swapped to try and generate a shorter route. As usual though, by using this strategy, the danger of local minima still exists. This problem has been largely solved by simulated annealing.

4.0 Statistical Mechanics

Statistical mechanics describe a collection of methods for analysing the properties of atoms in their solid or liquid states (Kirkpatrick, Gelatt, Vecchi, 1983). The basis of simulated annealing is formed from an analogy with thermodynamics. At high temperatures, the molecules of a liquid are able to move freely with respect to each other. As the liquid is cooled, the stored heat energy is lost and the atoms start to align themselves to form either a crystalline solid or glass. The crystal is the minimum energy state for the atoms and the glass is a local energy minimum. The question that thus arises is how nature is able to avoid local minima and find the global minimum, i.e. to form crystals. The answer lies in the rate of cooling. If the temperature drops rapidly then the atoms do not have sufficient time to redistribute themselves correctly and the substance formed is a glass with no crystalline order and the system has not found its global minimum. This is called quenching the system. However, if the temperature is dropped slowly then the atoms do have time to reach an equilibrium and are able to redistribute themselves correctly. The particles then have a very high probability of solidifying in crystal form, the global minimum. This is called annealing. If the system does find itself in a local minimum, it can only reach the global minimum if its energy is increased slightly to escape the local minimum. Thus annealing does not follow a perfect continual decreasing of energy.

The main similarity between combinatorial optimisation and statistical mechanics is that in both cases the global minimum is sought. The method described earlier which attempted to solve the travelling salesman problem resembles the process of quenching. Obviously then if we know why nature is able to find its global minimum this information can be applied to optimisation problems. This results in the Metropolis algorithm (Metropolis, Rosenbluth, Rosenbluth, Teller and Teller, 1953).

5.0 Simulated Annealing - The Metropolis Algorithm

In nature, particles are able to arrange themselves by chance, into a higher energy state to escape local minima. This is achieved using the Boltzmann probability distribution.

$$Prob(E) \propto exp(-E/(kT))$$

(EQ 5.1)

where

E is an energy state, T is the temperature and k is the Boltzmann constant.

When a system has reached thermal equilibrium at a temperature T, then its energy is said to be probabilistically distributed among all the different energy states E. At a low temperature there is still the slight chance that the system can be in a high energy state. This means that there must also be the chance for the system to escape this higher energy and find its global energy state. The system is therefore able to travel up and downhill in energy. However, as the temperature of the system falls even lower the chance that the particles will arrange themselves in any state apart from the global minimum decreases, although the possibility still exists. In an attempt to improve and therefore solve optimisation problems, Metropolis published a paper which incorporated nature's annealing process. This reference contains an algorithm for the simulation of atoms at thermal equilibrium. Obviously for optimisation problems the concept of thermodynamic temperature is not applicable so an effective temperature of optimisation is introduced. It is this that is decreased during optimisation. The procedure is again iterative, however uphill movements, which are controlled, are incorporated. A brief outline of this process is as follows:

- A parameter is changed and the resulting change in energy ΔE is calculated.
- If $\Delta E \leq 0$, the change is accepted and this new configuration becomes the starting point of the next step.
- If $\Delta E > 0$, then the probability of the new state relative to the old is calculated from the Boltzmann probability equation above. In this algorithm k is a constant but usually not the Boltzmann constant.
- A random number between 0 and 1 is generated.
- If Prob(E) is greater than the random number then the new configuration is retained.

- If Prob(E) is less than the random number the original configuration is used in the next step.
- The effective temperature of optimisation T, *the annealing schedule* is slowly reduced throughout the steps to ensure that the last configuration is truly the global minima.

As discussed earlier, this process, when applied to the travelling salesman problem, largely solves it. *The Travelling salesman* problem belongs to a class known as the NP-complete (nondeterministic polynomial time complete) problems, whose computation time for an exact solution increases exponentially with N, the number of parameters. Theoretically, simulated annealing can also be applied to every NP-complete problem (James, 1989).

6.0 Phase Annealing

The tangent formula can be used to optimise random phases. To improve this optimisation simulated annealing can be applied. This is called phase annealing (Sheldrick, 1990). Again, because this optimisation cannot be described in terms of continuous motion, a Boltzmann distribution must be applied to allow the system to reach thermodynamic equilibrium. This is controlled by a temperature T, which throughout the optimisation is slowly lowered. The energy of the system is described as a combination of the potential and kinetic energy. The potential energy is the function that is to be minimised and the kinetic energy enables the system to escape from wide shallow minima. Allowing the temperature to drop slowly, further increases the possibility of finding the global minima. The following two sections describe the principles of the phase annealing process present in SHELX and then a description of how this process was integrated into MITHRIL94. Centric and acentric phases are treated differently during phase annealing.

6.0.1 Centric Phases

Centric phases are restricted with values of 0, π or $\pm\frac{\pi}{2}$ or as in trigonal space groups of $\pm\frac{\pi}{3}$. For phase annealing a derivation of the Cochran and Woolfson formula must be applied (Cochran & Woolfson, 1955)

$$P_+ = \frac{1}{2} + \frac{1}{2}\tanh\left(\frac{\alpha}{2}\right)$$

(EQ 6.0.1.1)

where

$$\alpha = 2|E_h|E_k E_{h-k}/N^{1/2}$$

(EQ 6.0.1.2)

and P_+ is the probability that E_h takes the sign of α. When the sign of α is +ve, P_+ measures the probability of the lower energy state and P_- where $P_-=1-P_+$, measures the probability of the higher energy state. This gives the ratio of

$$\frac{P_-}{P_+} = e^{-\alpha}$$

(EQ 6.0.1.3)

The next stage is then to simply multiply α by a variable β which is inversely proportional to the temperature resulting in the Boltzmann ratio.

$$\frac{P_-}{P_+} = \frac{e^{-\alpha}}{\beta}$$

(EQ 6.0.1.4)

In SHELX and MITHRIL94 this ratio has been slightly changed to

$$\frac{P_-}{2P_+} = \frac{e^{-\alpha}}{\beta}$$

(EQ 6.0.1.5)

It is at this stage that the Metropolis algorithm is applied to invert the phases. The tangent formula produces a phase for which the α is calculated. This is used in the Boltzmann ratio. If

$$\frac{P_-}{2P_+} > R$$

(EQ 6.0.1.6)

where R is a random number in the region 0 to 1, then the higher energy state P_- is accepted as the new phase. If this equation does not hold then the lower energy state and thus the original phase is kept. Throughout this process the temperature factor is reduced slowly, by increasing β, to anneal the system. In this case α is associated with the potential energy of the system and β represents the kinetic energy of the system. Centric phases in a non-centrosymmetric structure are treated in exactly the same way.

6.0.2 Acentric Phases

In this case a parameter $\Delta\varphi$ is added to the phase produced by the tangent formula (Sheldrick, 1990)

$$\cos(\Delta\varphi) = [4\beta\alpha + ln(R)] / [4\beta\alpha - ln(R)]$$

(EQ 6.0.2.1)

Where R is a random number in the range 0 to 1 and the sign of $\cos(\Delta\varphi)$ is random. This formula is purely empirical, and the value of 4 (NDAMP) was chosen to give similar mean phase shifts for general and restricted phases at the beginning of phase determination.

MITHRIL94 contains three tangent refinement subroutines which were used in phase annealing

- FASTAN which uses the standard tangent refinement formula, described in chapter 1.
- SWTR tangent formula which uses the Hull and Irwin weighting scheme, described in Chapter 1.
- X-Y which uses the x-y tangent formula.

6.1 X-Y tangent formula

The possibility of X-Y as a random approach to the phase problem (Debaerdemaeker and Woolfson, 1989) was discovered by the same authors (1983) whilst researching the maximisation of various functions for the refinement of random phases. The process involves a parameter shift method which is also an optimisation method (Bhuiya & Stanley, 1963) and involves the x-y function seen below

$$\Psi = \sum_{\underline{h}} [X(\underline{h}) - Y(\underline{h})]$$

(EQ 6.1.1)

where

$$X(\underline{h}) = \sum_{\underline{h}} |E(\underline{h})\,E(\underline{h}')\,E(\underline{h}-\underline{h}')| \cos[\varphi(\underline{h}) - \varphi(\underline{h}') - \varphi(\underline{h}-\underline{h}')]$$

(EQ 6.1.2)

and

$$Y(\underline{h}) = \sum_{\underline{h}} |E(\underline{h})\,E(\underline{h}')\,E(\underline{h}-\underline{h}')| \sin[\varphi(\underline{h}) - \varphi(\underline{h}') - \varphi(\underline{h}-\underline{h}')]$$

(EQ 6.1.3)

Although no logical explanation is available for its success in refinement, this method is found to be efficient in its task of phase extension and refinement. The tangent formula followed from the maximisation of $\sum X(\underline{h})$ (Debaerdemaeker, Tate & Woolfson, 1985) and in this equation a correct phase set would also be expected to produce a large and positive $\sum X(\underline{h})$. Also for a correct phase set Y(h) would be small in magnitude and making its value equal to zero (its expectation value) mirrors the process by which Karle and Hauptman (1956) actually derived the tangent formula.

7.0 EXPERIMENTAL

The simulated annealing algorithm was integrated into the FASTAN, SWTR and X-Y tangent refinement subroutines. In MITHRIL94 each subroutine was kept intact and

the annealing algorithm added so that the resulting subroutine contains the original tangent refinement formula and the new simulated annealing code as an option.

The simulated annealing algorithm was used to modify the phases calculated by the refinement subroutine. The relevant tangent refinement code was then used to finish the refinement. The exact number of cycles of phase annealing and tangent refinement needed to produce the optimum results was extensively researched. The results of this were to produce algorithms that defined different numbers of optimum cycles for phase annealing and tangent refinement. These are specific to each subroutine. Also the shift for acentric reflections was calculated using a variable called NDAMP seen in EQ 6.0.2.1. To investigate what value for NDAMP produced optimum results, many different values were tested. The different parameters used in each subroutine and the integration of the annealing process into MITHRIL94 will be discussed below.

7.1 The β value

The first stage concerned the value of β. This is the parameter that is inversely proportional to the temperature and is used in the Boltzmann ratio. The correct value of β depends on the structure being tested and is calculated using the following equation (Sheldrick, 1990).

$$\beta = -ln(B) / \langle \alpha^2 \rangle^{\frac{1}{2}} \qquad \text{(EQ 7.1.1)}$$

where $\langle \alpha^2 \rangle^{\frac{1}{2}}$ is over all reflections used and B is a constant

Again each subroutine was rigorously tested with various values for the initial B and the optimum value was established. In each cycle, β was increased to allow annealing. The initial value of B used in each subroutine, along with the appropriate number of cycles of phase annealing and the number of cycles of tangent refinement per phase set

and the optimum value of NDAMP is shown in Table 1.

TABLE 1. Optimum values for parameters used in phase annealing

Subroutine	Value of B	Cycles of annealing	Cycles of refinement	Value of NDAMP
FASTAN	0.8	40	3	4
SWTR	0.9	80	5	4
X-Y	0.3	30	4	8

7.2 Phase Annealing in MITHRIL94

Phase annealing code was integrated into all three subroutines with the parameters shown in table 1. and the phases were annealed in a similar manner to the process in SHELX.

7.2.1 Centric Phases

The value of β was incremented each cycle of annealing, to mimic the lowering of the temperature of the system. The flow diagram shown on the next page displays the events that occur for each cycle of annealing per phase set.

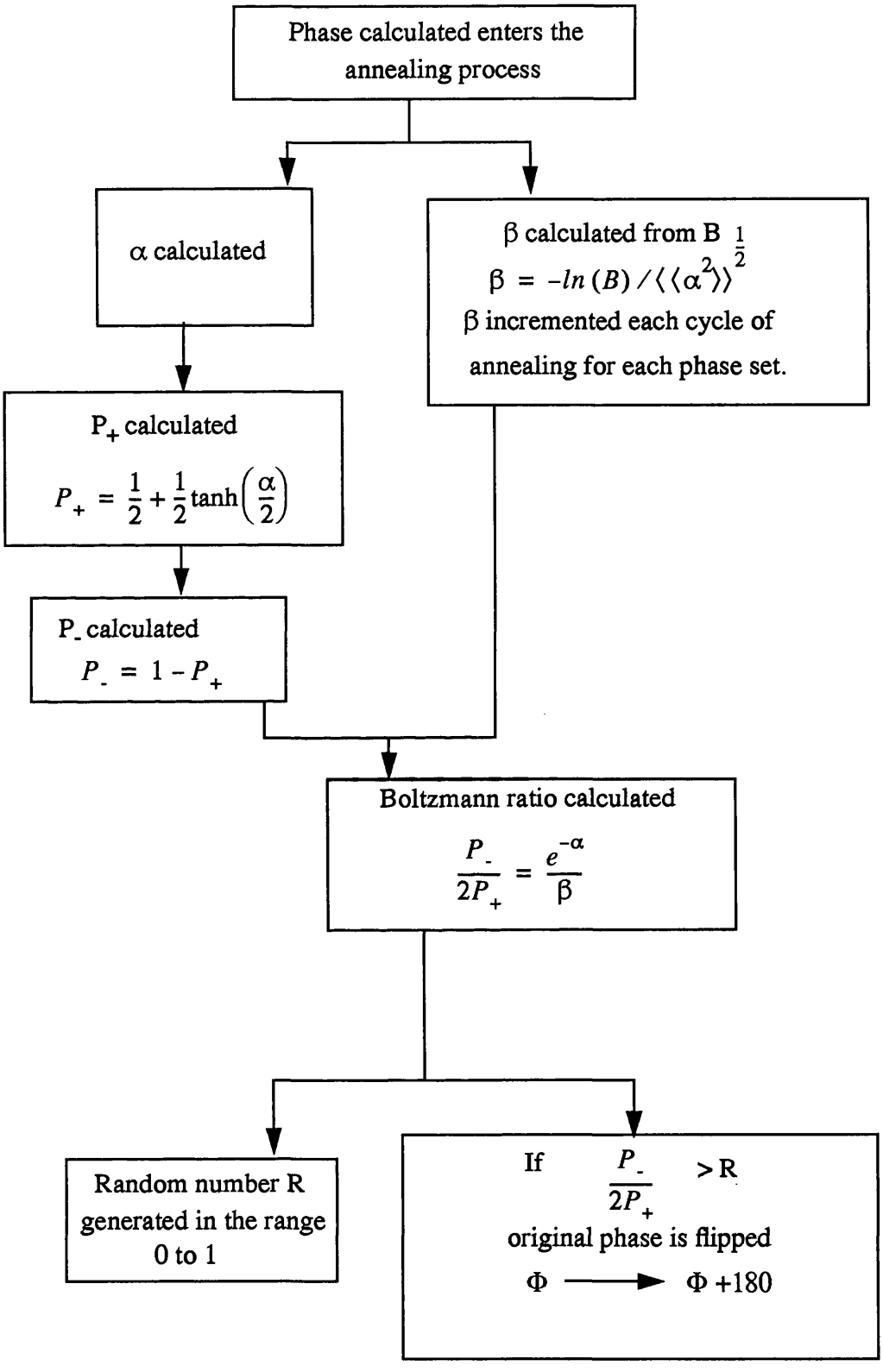


FIGURE 4. Flowchart for annealing of centric phases per phase set

As the temperature factor was lowered, the Boltzmann ratio decreased, resulting in fewer phases being flipped. This is equivalent to the system finding its global minimum. A figure of merit was then calculated for each phase set. If the temperature is slowly lowered then the well defined phases (high α) will tend to become established more quickly whilst the remaining phases are free to explore phase space. This means that phases linked by strong phase relationships will be fixed faster and if a good, but not necessarily correct solution is found, it will remain more stable than a phase set which has low mean α (Sheldrick, 1990). The number of phases flipped in the first cycle was usually around 120 and decreased to approximately zero by the time the annealing finished.

7.2.2 Acentric Phases

For each phase set the following procedure occurred during each cycle of annealing:

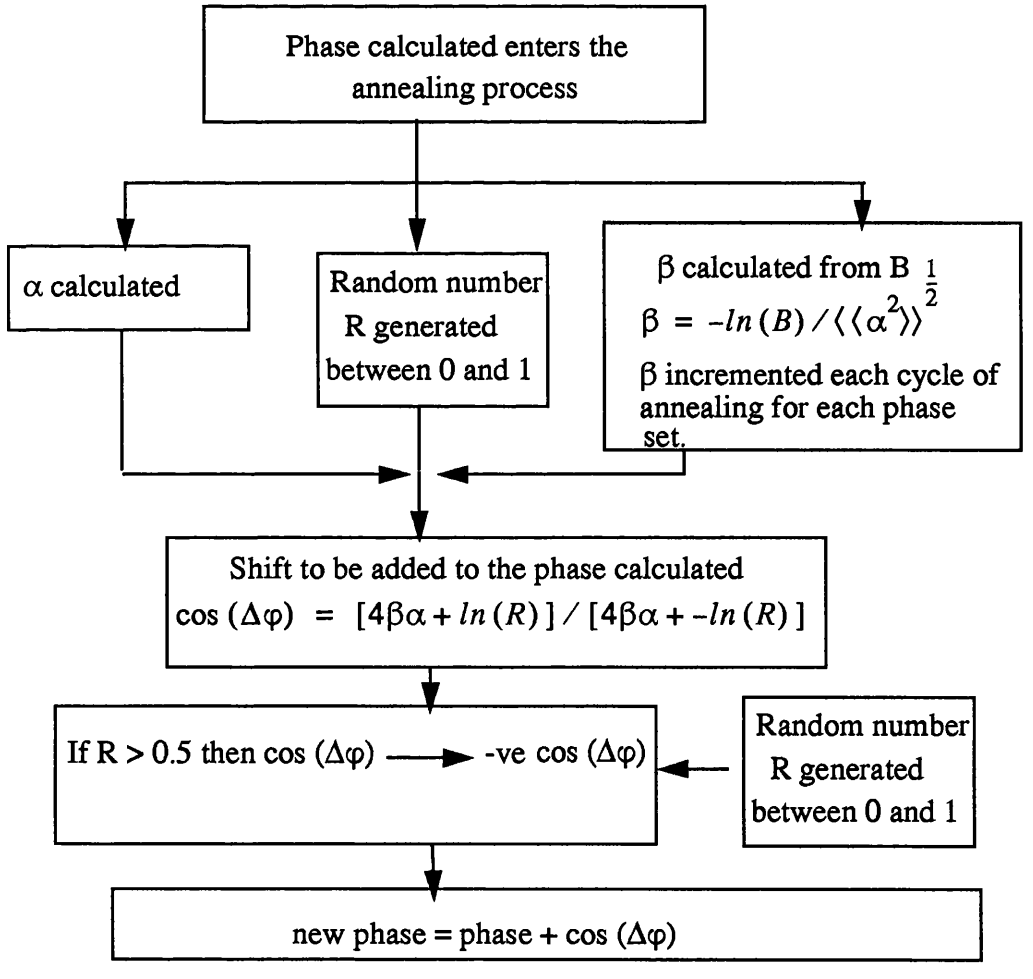


FIGURE 5. Flowchart for phase annealing of acentric phases per phase set

Again the β increases each cycle and therefore the shift added to each phase set should decrease each cycle. The shift usually starts in the region of 180 and decreases to around 10 degrees. If phase annealing is performed with a β of zero (this is an infinite temperature) the phases will remain random therefore as the β increases the temperature decreases as this corresponds to phase annealing.

8.0 RESULTS

The phase annealing code was tested on structures from the Sheldrick database of difficult structures which were known to be difficult to solve and these results were compared to those obtained from testing the original code on the same structures. The results produced from each subroutine with phase annealing were compared to the results obtained from the original code. These results are shown in 3 different tables:

- Table 2- FASTAN with annealing and without annealing.
- Table 3- SWTR tangent refinement with annealing and without annealing.
- Table 4- X-Y with annealing and without annealing.

This sort of iterative process should end with the global minimum for phase space; however, the structure solution may not be at this minimum, as the function we are minimising may not define accurately enough the phase space. Because of this it is best to approach the problem with a multisolution approach, with many random starting positions chosen. Therefore 1000 phase sets were produced for each structure, irrespective of whether phase annealing was in operation or not. Thus the numbers shown in the tables correspond to the number of correct phase sets produced from a total of 1000 phase sets.

TABLE 2. FASTAN with and without annealing

Structure	Space Group	FASTAN with anneal	FASTAN without anneal
Diamantane	$P4_2/n$	574	649
Quinol	$R\bar{3}$	150	72
Selenid	$P2_1$	3	2
Azet	$Pca2_1$	1	0
Tursch 10	$P6_322$	63	33
Bed	I4	126	14
Loganin	$P2_12_12_1$	79	49
Diol	$I\bar{4}2d$	0	0
Apapa	$P4_12_12$	0	0
Tpala	$P2_1$	10	12
Tval	P1	0	1
Newqb	$P\bar{1}$	2	2
Goldman2	Cc	98	44
Munich1	C2	8	0
MBH2	P1	2	3
PGE2	P1	7	2
SUOA	$P2_12_12_1$	0	0

TABLE 3. SWTR with and without annealing

Structure	Space Group	SWTR with anneal	SWTR without anneal
Diamantane	$P4_2/n$	325	274
Quinol	$R\bar{3}$	91	53
Selenid	$P2_1$	9	7
Azet	$Pca2_1$	10	6
Tursch 10	$P6_322$	85	15
Bed	I4	101	23
Loganin	$P2_12_12_1$	84	8
Diol	$I\bar{4}2d$	0	0
Apapa	$P4_12_12$	2	0
Tpala	$P2_1$	14	12
Tval	P1	33	74
Newqb	$P\bar{1}$	0	0
Goldman2	Cc	30	2
Munich1	C2	6	0
MBH2	P1	53	19
PGE2	P1	1	1
SUOA	$P2_12_12_1$	2	0

TABLE 4. X-Y with and without annealing

Structure	Space Group	X-Y with anneal	X-Y without anneal
Diamantane	$P4_2/n$	301	149
Quinol	$R\bar{3}$	17	15
Selenid	$P2_1$	10	11
Azet	$Pca2_1$	1	0
Tursch 10	$P6_322$	5	2
Bed	I4	47	9
Loganin	$P2_12_12_1$	11	3
Diol	$I\bar{4}2d$	1	0
Apapa	$P4_12_12$	0	0
Tpala	$P2_1$	12	7
Tval	P1	0	0
Newqb	$P\bar{1}$	0	0
Goldman2	Cc	0	1
Munich1	C2	0	0
MBH2	P1	0	0
PGE2	P1	1	1
SUOA	$P2_12_12_1$	0	0

9.0 DISCUSSION

9.1 Table 1 - FASTAN with or without phase annealing

From the 17 structures tested, the results of 9 have improved with phase annealing, 3 have worsened with phase annealing and 4 have stayed the same.

Of the 4 structures that remained the same, 3 did not produce any correct phase sets with or without annealing. These were:

- Diol
- Apapa
- SUOA

The 4th one, Newqb produced 2 correct phase sets for both phase annealing and normal tangent refinement. This structure has always caused MITHRIL94 problems and does not normally solve using defaults. Using random numbers for FASTAN therefore increases the chances of finding correct phase sets for this structure.

The results for 4 structures showed no improvement using phase annealing. This was not too disappointing as the results for 3 of the structures were still very close:

- Diamantane still managed to produce 574 correct phase sets from 1000 (compared to 649 from 1000 without annealing).
- MBH2 produced 2 phase sets compared to 3 correct phase sets with normal FASTAN.
- Similarly Tpala only produced 2 correct phase sets less than normal FASTAN.
- Tval was the only structure that could be considered disappointing as it did not solve at all with phase annealing, whereas normal FASTAN found 1 correct phase set.

The results for 9 of the structures improved with phase annealing. Most notable were:

- Quinol - this centrosymmetric structure produced double the number of correct phase sets with phase annealing.
- Azet - this structure is unstable under regular tangent refinement and produced only1 correct phase set with annealing, however without annealing Azet was not solved.
- Bed - the results for Bed were quite outstanding with the annealing improving the results by a factor of 9.

- Munich1 - 8 correct phase sets were produced with phase annealing. This structure was not solved without annealing.

FASTAN with phase annealing has made a definite improvement for most of the test structures, managing to solve 2 structures that normal FASTAN using random numbers could not.

9.2 Table2 - SWTR with or without phase annealing

The results produced by this subroutine were excellent, with 13 of the 17 structures tested increasing the number of correct phase sets when phase annealing was used. Most of the improvements were quite outstanding with some structures increasing correct phase sets by a factor of 10:

- Tur10 - 85 correct phase sets with phase annealing compared to 15 when phase annealing is not used.
- Bed - again the results for this structure improved dramatically when phase annealing was employed, in this case 101 correct phase sets with annealing compared to only 23 without.
- Loganin - this structure although not usually a problem to MITHRIL94 produced only 8 correct phase sets for normal FASTAN, compared to 84 with phase annealing.
- Goldmann2 - when no annealing was used only 2 correct phase sets were produced compared to 30 correct when annealing was employed.

The best results however for this subroutine were those for Apapa and SUOA. Both structures produced no correct phase sets with FASTAN however with SWTR they both solved, producing 2 correct phase sets each. SWTR without annealing was unable to solve them. The result for Apapa was particularly pleasing as MITHRIL94 has never been able to solve this structure before using only default parameters.

The structures which were not solved by SWTR irrespective of whether phase annealing was used were:

- DIOL - this was not solved by FASTAN either.
- Newqb - this was solved by FASTAN.
- Tval - the results for this structure again worsened when annealing was used, but phase annealing was still able to produce 33 correct phase sets (without annealing - 74 correct phase sets).

Overall, phase annealing with SWTR produced a dramatic improvement in results compared to normal SWTR.

9.3 Table3 - X -Y with or without phase annealing

The results for this subroutine may look disappointing at first as out of 17 structures only the results for 8 improved. However of the other structures, 5 could not be solved at all by X-Y, with annealing making no improvement on this. Only the results for 2 of the structures worsened:

- Goldman2 - no correct solutions were found using annealing compared to 1 correct solution without.
- Selenid - the results only decreased by 1 correct phase set when annealing was used.

The best results were for:

- Diamantane - the number of correct solutions produced by annealing was 301 compared to only 149 without.
- Azet - X-Y was unable to solve this structure unless phase annealing was used.
- Bed - the results for Bed were again highly successful for phase annealing with 49 correct solutions being produced, compared to only 9 without.

The best result for X-Y however was for a structure that the previous subroutines failed completely to solve:

- Diol - Phase annealing produced 1 correct solution whereas normal X-Y could not solve this structure.

The results for X-Y are not as clear cut as the results for the other 2 subroutines. On the whole annealing did make some improvements, namely for diol. However the fact that it was unable to improve the results for those structures which the original program was unable to solve was quite disappointing. However X-Y is the least used of the refinement subroutines.

Phase annealing has been incorporated into a commercial version of MITHRIL94 which is discussed in chapter 6.

References

Bhuiya, A. K. & Stanley, E. (1963). *Acta Cryst.* **16**, 981-984

Christofides, N., Mingozzi, A., Toth, P., and Sandi, C. (1979). *Combinatorial Optimisation* (London and New York: Wiley-Interscience)

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

Debaerdemaeker, T. & Woolfson, M. M. (1983). *Acta Cryst.* **A39**, 193-196

Debaerdemaeker, T., Tate, C. & Woolfson, M. M.(1985). *Acta Cryst.* **A41**, 286 -290

Debaerdemaeker, T. & Woolfson, M. M. (1989). *Acta Cryst.* **A45**, 349 -353

James, M. (1989) *COMPUTER SHOPPER, Annealing we will go.* 93-95

Karle, J. & Hauptman, H. (1956). *Acta Cryst.* **9**, 45 -55

Kirkpatrick, S., Gelatt, C.D. & Vecchi, M.P. (1983). *Science* **220**, 671-680

Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H. & Teller, E. (1953). *J. Chem. Phys.* **21**, 1087-1092

Press, W. H. & Flannery, B. P. & Teukolsky, S. A. & Vetterling, W. T. (1986) *NUMERICAL RECIPES, The Art of Scientific Computing* 274-326, Cambridge University Press

Rollet, J. S. (1984). *Methods and applications in crystallographic computing,* Edited by S.R. Hall and T. Ashida, pp161-174, Clarendon Press, Oxford

Sheldrick, G. M. (1990). *Acta Cryst* **A46**, 467-473

Tichy, K. (1970). *Acta Cryst.* **A26**, 295-296

CHAPTER 4

ERROR CORRECTING CODES AS A PHASE PERMUTATION TECHNIQUE IN MITHRIL94

1.0 THEORY

1.1 Introduction

The convergence procedure introduced by Germain, Main & Woolfson (1970) determines the starting set of reflections to be used in phase permutation, symbolic addition or random phasing. This set consists of the origin and enantiomorph defining reflections (if appropriate), the Σ_1 phased reflections and reflections of unknown phase. These reflections are selected for permutation to ensure multiple interactions and strong phase development via triplets. Phase permutation involves the assignment of values to the phases in the starting set. As discussed earlier there exists various phase permutation methods including those of magic integers and random phasing.

Bricogne (1993) has shown how certain error correcting codes can be used to sample phase space efficiently and hence be used as a phase permutation technique. To test this alternative permutation method, the use of error correcting codes has been investigated and added into the MITHRIL94 program (Gilmore,1984; Gilmore & Brown 1988). The remainder of this chapter will discuss the background theory and experimental results of this new method.

2.0 THE RELEVANCE OF EXPERIMENTAL DESIGNS TO ERROR CORRECTING CODES

Experimental designs are used to provide information simultaneously about the effects of a variety of possible factors. They are efficient, accurate and complete. The use of experimental designs is already seen in protein crystallography, for example in the screening of heavy atom derivatives and in finding the optimum conditions for crystallisation (Carter, 1992,1994). For phase permutation we require a method that is able to sample phase space as efficiently and uniformly as possible. The relevance of error correcting codes to experimental design (Hill, 1993) is shown by the following example:

Seven brands of fertiliser are to be tested on seven types of crops, noting the effect pairs of fertiliser have on each crop. Obviously one method would be to test every combination of fertilisers on every crop but this is time consuming and the results may not always be easy to interpret. Instead a more economical method would be to design an experiment where

- each crop receives the same number of fertilisers
- each pair of fertilisers are compared on the same number of crops.

Mathematically all that is involved is a set S of *varieties* (the fertilisers) and subsets of S (consisting of fertilisers tested on each crop). The subsets are called *blocks*. In this experiment

- each block has the same number of entries
- each pair of varieties is contained in the same number of blocks

Take S={1,2,3,4,5,6,7} as the 7 varieties of fertiliser. 7 subsets of S (B1-B7) can occur so that each pair of fertilisers only appears once.

FIGURE 1. Subsets B1 to B7 of set S

B1	B2	B3	B4	B5	B6	B7
{1,2,4}	{2,3,5}	{3,4,6}	{4,5,7}	{5,6,1}	{6,7,2}	{7,1,3}

This experimental design is a *balanced block design* as it consists of a set S of v elements, called *points* or *varieties* and a collection of b subsets of S, called *blocks,* such that, for some fixed k,r and λ

- each block contains exactly k points
- each point lies in exactly r blocks
- each pair of points occurs together in exactly λ blocks.

These designs are referred to as (b,v,r,k,λ)-designs (Hill, 1993).

The subsets form the blocks of a (7,7,3,3,1) balanced block design as there are 7 blocks (fields), 7 varieties(fertiliser), each block contains 3 points and each point lies in 3 blocks. The comparison between the pairs of fertilisers is balanced with each pair of points (fertiliser) occurring together in exactly 1 block.

There is a simple geometric representation of this design known as a seven-point plane or Fano plane.

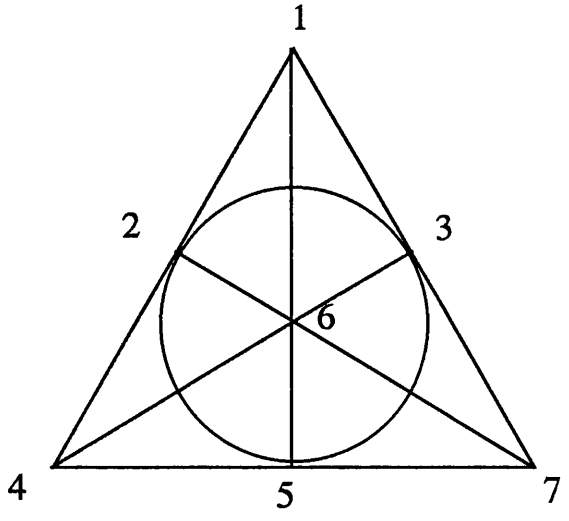
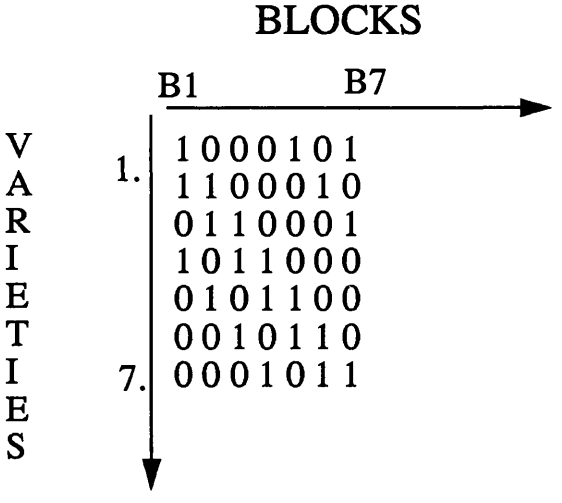


FIGURE 2. The seven - point plane

The fertilisers 1,2....,7 are represented by points and the blocks are represented by lines (6 straight lines and a circle)

Alternatively this (7,7,3,3,1)-design can be displayed in matrix form.This is known as the incidence matrix of the design as shown in Figure 2.

FIGURE 3. The Incidence Matrix of the design.



Each column of the matrix represents one block and each row provides information about a particular variety, for example the first row shows that fertiliser 1 is tested on crop 1(B1),5(B5) and 7(B7) and column one shows that crop 1 is tested with the block varieties {1,2,4}

The advantages of using balanced block designs are:

- the work load of the experiment is reduced.
- time spent on experiment reduced.
- potentially the balanced block design gives better results than non-systematic methods.

The advantages of using the incidence matrix to describe the design are:

- the structure of the design is visibly clearer with no irrelevant information
- it is easy to scan the columns of the matrix to retrieve information

Error correcting codes are linked to experimental design by the incidence matrix of the design. Each row of the incidence matrix is composed of a sequence of 0's and 1's. These are binary error correcting codewords.

3.0 ERROR CORRECTING CODES

Error correcting codes are used to correct errors when messages are transmitted through a noisy communication channel.

The channel may be:

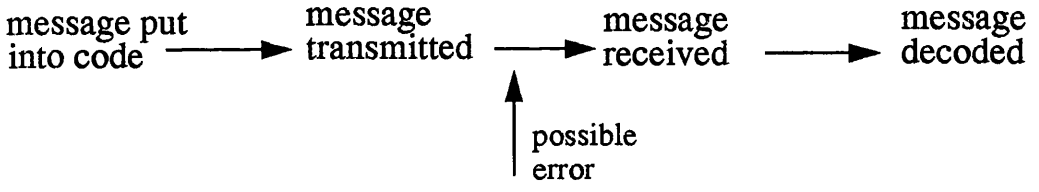
- telephone line
- high frequency radio link
- satellite communication link

The noise may be

- human error
- thermal noise
- imperfections in equipment

If errors do occur in transmission then the received message could be different and have a different meaning from the original message causing disruption and confusion.

FIGURE 4. Transmission of messages



Error correcting codes are used to encode data by adding redundancy to a message so that if errors do occur they can be identified and corrected and the original message can be recovered.

A q-ary code is a given set of sequences of symbols where each symbol is chosen from a set F_q of q distinct elements.

The set F_q is called the *alphabet* and in the binary case it is denoted F_2. If q is a prime power then we take the alphabet F_q to be the finite field of order q. This is defined as a field which has a finite number of elements, this number being called the *order* of the

field. $(F_q)^n$ is the set of all ordered n-tuples $a=a_1a_2.....$ and the elements of $(F_q)^n$ are called *vectors* or *words*. For example, the (7,7,3,3,1)-design can be denoted $(F_2)^7$. The error correcting codes discussed later in this chapter are binary codes thus this investigation occurs in the finite field F_2.

The main purpose of coding theory is to send messages quickly with as much accuracy as possible. The concept of "parity" is simple use of this. An n-block of binary digits has *even parity* if the sum modulo 2 of the digits is zero (in other words, there are an even number of 1's).Otherwise the n-block has *odd parity*.The parity check is just the addition of the digits and using the parity check enables us to detect any single error that may have occurred in transmission.

Example: For 1100 and 1101 add redundancy so that the resulting vectors have even parity.

$$1\,1\,0\,0 \longrightarrow 1\,1\,0\,0\,0 \quad \text{and} \quad 1\,1\,0\,1 \longrightarrow 1\,1\,0\,1\,1$$

Any single error in the transmission of these vectors makes the parity of the codewords odd thus the parity check fails so a single error must have occurred in the codeword. These are single error detecting codewords.

For example if a single error was to occur in one of the codewords so that the vector received was 11010:

$$1\,1\,0\,0\,0 \longrightarrow 1\,1\,0\,1\,0 \quad \text{and} \quad 1\,1\,0\,1\,1 \longrightarrow 1\,1\,0\,1\,0$$

In this case the error can be detected but not corrected as the error could have occurred in either of the codewords. This would cause the receiver difficulty if retransmission of the vectors is impossible. For correction to occur the received vector would have to be "closer" to one of the codewords than to any other.

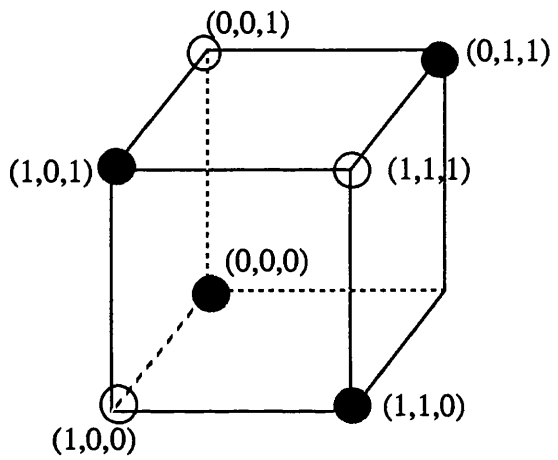
The concept of one codeword being "closer" to another is made precise by introducing a legitimate distance function on $(F_q)^n$ called the Hamming distance.

The Hamming Distance between two vectors x and y of $(F_q)^n$ is the number of places in which they differ denoted d(x,y). This was introduced by Hamming in 1950 using a

unit cube in Euclidean n-dimensional space E^n. The vertices of the cube are the 2^n n-tuples of 0s and 1s. Thus a subset of the vertices will be a binary code of length n.

Figure 5. illustrates a single error detecting code.

FIGURE 5. A single error detecting code

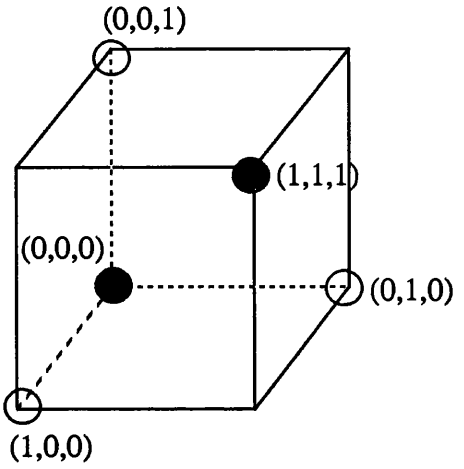


The darkened vertices are the codewords 000, 011, 101 and 110 and they represent a code with four codewords of length three. The light vertices show the three codewords formed when a single error occurs in the transmission of codeword 101. In this case again the single error can be detected but not corrected, i.e. the received codeword 001 could have been formed from single errors in the transmission of codewords 101, 000 or 011 and is therefore impossible to correct.The Hamming distance between the codewords 101 and 011 denoted d(101,011) = 2, as does d(101,000). This is also the number of edges in shortest route between the two vertices of the cube (Thompson, 1983).

The *minimum distance* of a code is the smallest of all the distances between two different codewords of the code and is denoted d(C). In this case d(C) = 2. Single errors can be detected if d(C) = 2 but for single error correction to occur d(C) must at least ≥ 3.

This is illustrated by Figure 6.

FIGURE 6. A single error correcting code



In this case the darkened vertices are the codewords 000 and 111 and they represent a code with two codewords of length three. The minimum distance d(C) between the two codewords is three. The light vertices show the three codewords received when a single error occurs in the transmission of codeword 000. The error can be detected by parity check and it can be corrected by considering the Hamming distance d(000,001)=1 and d(111,001)=2. The received vector can be corrected to its nearest codeword by considering the smallest of the Hamming distances.In this case the value is 1 and the vector can be corrected to the original 000. This example only applies for the occurrence of single errors. If two errors occurred then they would be detected but not corrected as the receiver would not be able to tell from which codeword the received vector originated and how many errors had actually occurred. In general a code with minimum distance d will ***detect*** up to d/2 errors and ***correct*** up (d-1)/2 errors.

The codes investigated for their use in phase permutation can be described as (n,M,d)-codes where n is length of code, preferably small for fast transmission, M is the number of codewords in the code which is preferably large for transmission of as many messages as possible and d, the minimum distance is preferably large. Obviously a good code will have a large minimum distance to correct many errors.

The above requirements for the formation of a successful code unfortunately conflict with each other and usually one of the parameters n, M or d is optimised for given values of the other two. Frequently M is optimised for given values of n and d.

Error correcting codes are linked to experimental design by the incidence matrix of the design and specific properties of the codes are required for phase permutation, to sample phase space efficiently so giving good results and saving computer time.

4.0 PROPERTIES REQUIRED OF ERROR CORRECTING CODES

The error correcting codes investigated for their use in phase permutation were chosen to provide efficient covering of phase space and to have a balanced structure for 1-phase, 2-phase and 3-phase interactions. Two questions must be addressed before discussing these codes and the manner the codes were used in phase permutation:

- what is efficient covering of phase space?
- how do the codes produce efficient covering?

4.1 Efficient Covering of Phase Space

As discussed in chapter one, multisolution methods involve assigning approximate numerical values to the unknown phases in the starting set. Centric phases can take the values 0 and π or $\pm\frac{\pi}{2}$ or $\pm\frac{\pi}{3}$. For acentric phases the value is not restricted but for quadrant permutation we can permute over the values $\pm 3\pi/4$ or $\pm\pi/4$. Thus the total number of combinations n, to be permuted by the tangent formula is:

$$n = 4^{Nu} \times 2^{Nr}$$

(EQ 1.4.1.1)

where Nu is the number of acentric reflections and Nr is the number or reflections with restricted phases.

Looking at this equation, it is clear that the number of combinations increases dramatically as Nu and Nr increase and it is obviously not feasible to examine all the combinations in a full factorial phase permutation design when Nu is large. Usually there is a limit put on the number of reflections allowed in the starting set. Using

magic integers this limit can be reduced. Table one shows the comparison between the number of phase sets produced by quadrant permutation of phases and magic integer permutations.

TABLE 1. Number of phase sets produced by quadrant permutation compared to number produced by magic integer permutation

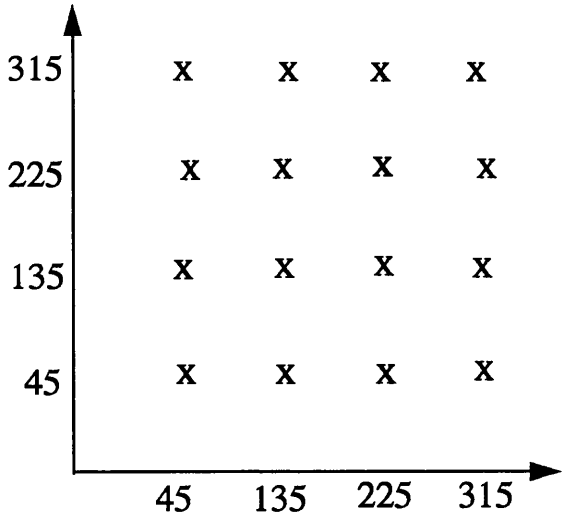
QUADRANT		MAGIC INTEGERS	
n	no of permutations	n	no of permutations
1	4	1	4
2	16	2	12
3	64	3	20
4	256	4	32
5	1024	5	50
6	4096	6	80
7	16384	7	128

As seen from this table magic integers reduce the number of permutations dramatically (Giacovazzo, Monaco, Viterbo, Scordari, Gilli, Zanotti, Catti, 1992). The question still to be addressed though is how efficiently magic integers are able to sample the phase space.

The following examples show how we could permute two acentric reflections and the covering of phase space that each produces.

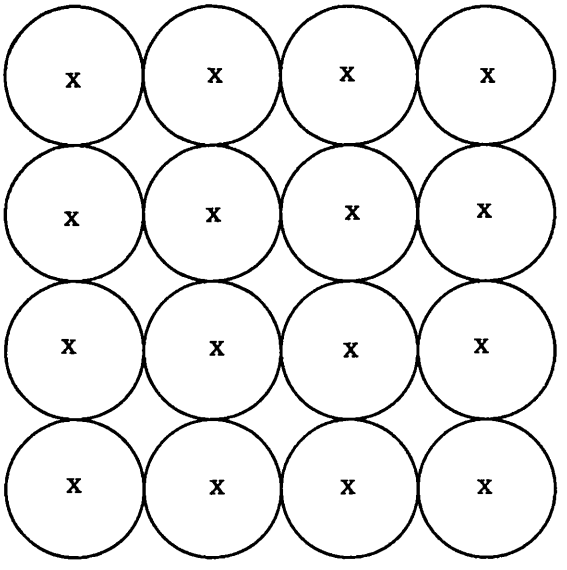
Figure 7. illustrates permuting every possible value for the reflections so that the entire phase space is sampled:

FIGURE 7. Sampling the phase space (A)



The covering generated by this sampling is seen below:

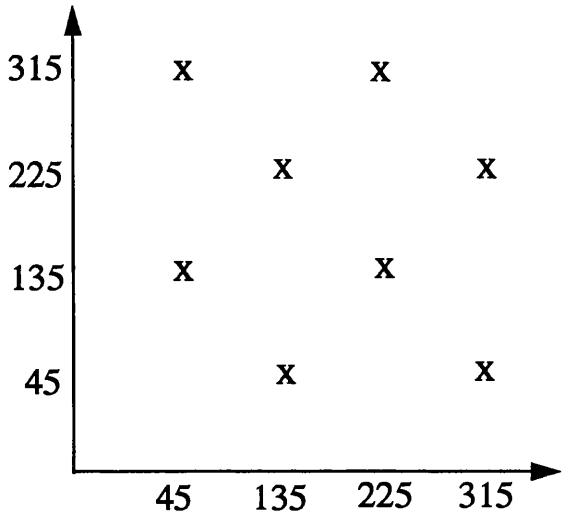
FIGURE 8. Covering of phase space (A)



The covering shown in this example is similar to the covering expected from a full quadrant permutation factorial design (full factorial design)

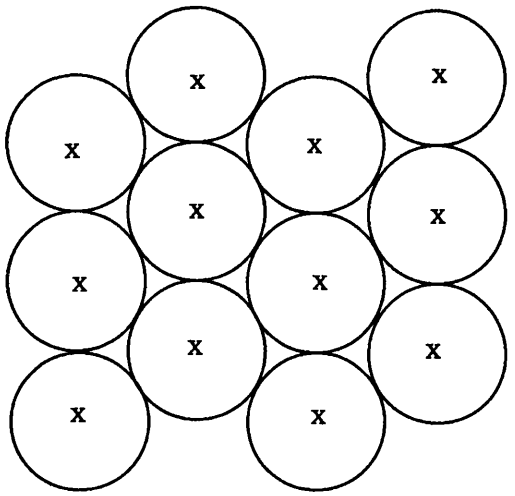
If however we do not permute every value for the phases but sample the phase space in the manner shown in Figure 9, the covering in Figure 10 is generated.

FIGURE 9. Sampling the phase space (B)



the following covering is generated:

FIGURE 10. Covering of phase space (B)

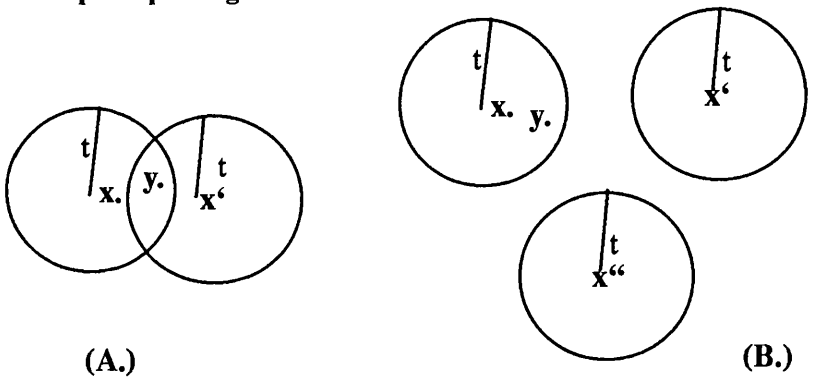


This is a denser and hence more efficient covering and is similar to that expected from a magic integer permutation.

4.2 The covering of phase space and error correcting codes

For the codes to produce efficient covering they must fulfil a sphere packing bound. This means that for a t-error correcting code, the M spheres of radius t centred on codewords fill the whole of $(F_q)^n$ without overlapping. This is the definition of a *perfect code* (Hill, 1993). If the spheres centred on the codewords are not disjoint then the codewords are not effective at error correcting. This is shown diagrammatically in Figure 11 below where y is a vector received when t or fewer errors occur in codeword x where a sphere of radius t and centre x is denoted S(x,t).

FIGURE 11. Sphere packing bound



Very simply in example A, the received vector y occurs in the overlap, that is in both S(x,t) and S(x',t) and it cannot be corrected to its closest codeword, whereas in example B the received vector may be different from the centre of the sphere S(x,t) but it cannot escape from it and is thus corrected to the nearest codeword x. So for error correcting properties we require the code to be perfect. For phase permutation we require the code to be perfect for much the same reasons. If the spheres centred on the codewords are not disjoint then the covering created would not be efficient as duplicate phase information could be contained in the overlap. For phase permutation we require a code that creates efficient covering of phase space. It is the code which generates the covering, not the opposite way round. Designs using error correcting codes for phase permutation take advantage of the periodicity of the phase angles and display this efficiency.

5.0 CODES USED IN PHASE PERMUTATION

The use of two different error correcting codes was investigated

- The Hadamard code
- The Golay code

5.1 The Hadamard Configurations

Hadamard configurations are (v,k,λ)–designs in which v=4m-1, k=2m-1 and λ=m-1 for an integer m≥2 (Street, Street, 1987). The seven-point plane illustrated earlier is a design for m=2. The designs are useful because knowledge of one design can lead to the formation of many others. Hadamard configurations are closely related to Hadamard matrices (Hadamard codes). The Hadamard matrix is an nxn matrix comprising the integers ±1. If we change all the -1 to 0 the resulting matrix is similar to the incidence matrix of the design.

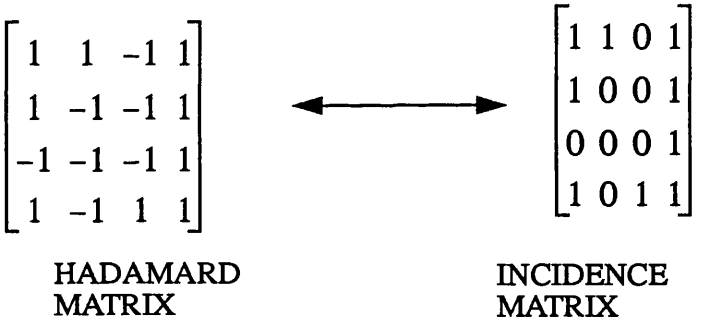


FIGURE 12. Hadamard and Incidence matrices

Large Hadamard matrices are generated by the repetition of smaller matrices. By using the 4x4 matrix above, the 8x8 matrix can be formed by simply repeating the matrix in the following manner where the original matrix is A.

$$\begin{bmatrix} [A] & [A] \\ [A] & -[A] \end{bmatrix} = B$$

The resulting matrix B can be used to form the 16x16 matrix in the same way.

5.2 The Hadamard Code and Phase Permutation

An nxn Hadamard matrix is generated with each row comprising ± 1. The starting set of reflections are assigned phases from the information contained in each row of the Hadamard code. This is done in the following manner:

Centric reflections are generally given a value of 0 or π or $\pm\frac{\pi}{2}$ or $\pm\frac{\pi}{3}$ so they have one degree of freedom and a single sign is used for phase permutation:

- +1 = 0 degrees or +90 degrees
- -1 = 180 degrees or -90 degrees

Acentric reflections have a choice of four phase values, one from each quadrant. They have two degrees of freedom and two signs are required to define the relevant quadrant:

- -1, -1 =225 degrees
- -1, +1 = 315 degrees
- +1, +1 = 45 degrees
- +1, -1 = 135 degrees

Each row of the Hadamard code defines one phase set which is input into the phase extension and refinement subroutines in MITHRIL94 (Gilmore,1984;Gilmore & Brown 1988) and figures of merit for that phase set are calculated and produced. The next phase set is just the complement of the above row, thus 2n phase sets are generated. One property of Hadamard matrices is that by definition multiplication of rows and columns by -1 and also swapping of rows and columns does not change the matrix.

5.3 The Hadamard Code and the Covering of Phase Space

The Hadamard code gives a sparse, but economical and efficient covering of phase space. The Hadamard code produces a covering similar to the covering produced by the magic integers in that all phase values are sampled but certain combinations are not fully permuted. The code is also efficient, as a Hadamard code of 16 phase sets will have one phase set with, at the most, only one wrong phase. The following sign combinations for phase permutation were introduced by Woolfson (Woolsfson, 1954). This is shown in table 2.

TABLE 2. Sign Combinations

Term	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	+	+	+	+	+	+	+	+	-	-	-	-	-	-	-	-
2	-	-	-	-	+	+	+	+	-	-	-	-	+	+	+	+
3	+	+	-	-	+	+	-	-	+	+	-	-	+	+	-	-
4	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+
5	+	-	-	+	+	-	-	+	-	+	+	-	-	+	+	-
6	+	+	-	-	-	-	+	+	-	-	+	+	+	+	-	-
7	-	+	-	+	+	-	+	-	+	-	+	-	-	+	-	+

The combinations above are extremely useful as, irrespective of what the correct sign combination actually is, each of the 128 possible 7-sequences differs from one of this set in at most one sign. Good (Good, 1954) extended this to a set S of 2048 possible 15 bit sequences, where each of the 32,768 possible 15-sequences differs in at most one sign from one of the set of 2048. This theory is described as Woolfson substantialisation as one member of S must be substantially correct. Substantialisation can be described as the maximum distance between the correct solution is the complete factorial design and the nearest sampling point (Carter, 1994). In this case the number of permutations have been reduced by a factor of 16, with the acceptance of one wrong sign.

The Hadamard codes generated and tested in this investigation were:

- The n=32 matrix, producing 64 phase sets instead of the 2^{32} produced from a full factorial design.

- The n=64 matrix, producing 128 phase sets instead of the 2^{64} produced from a full factorial design,
- The n=128 matrix, producing 256 phase sets instead of the 2^{128} produced from a full factorial design.

The best results produced were from n=64, and these results will be discussed later in this Chapter.

5.4 The Binary [24,12,8] Golay Code

This code was devised in 1949 by Marcel J.E.Golay (Hill, 1993) and it generates 4096 codewords of length 24. The actual Golay code is the [23,12,7] code and the [24,12,8], the *(extended Golay code)* is obtained from this by adding a parity check. The [23,12,7] can be regained by puncturing, that is removing the last digit from each codeword. It is the extended code that formed the basis of this investigation.

5.5 The Golay Code and Phase Permutation

Again phase information was extracted from the code and each row of the code, that is each codeword was translated into one phase set. The translation was similar to that of the Hadamard, with centric restricted reflections using one bit and acentric reflections using two bits of codeword to define the relevant quadrant.

5.6 The Golay Code and the Covering of Phase Space

The efficiency of covering of the [24,12,8] Golay code is quite staggering. The starting set is comprised of a maximum of 24 reflections. A full factorial design would therefore require 2^{24} (16,777,216) permutations. The use of the Golay code reduces this to 4096, but it is guaranteed that one of these phase sets will have at the most only 4 wrong sign indications. The [23,12,7] Golay code will have one codeword with only 3 wrong sign indications. The covering produced by the Golay code is not too surprising as this code is used in the construction of the Leech lattice, and it is of interest to discuss this here.

5.7 The Golay Code and the Leech Lattice

As discussed before, the codes investigated for their use in phase permutation can be described as (n,M,d)-codes where n is the code length, preferably small for fast transmission, M is the number of codewords in the code which is preferably large for transmission of as many messages as possible and d, the minimum distance is preferably large (Conway & Sloane, 1988). Keeping the codewords short enough for fast transmission whilst maintaining a large minimum distance is equivalent geometrically to packing spheres around an origin as densely as possible (Sloane, 1984). This is described as the sphere packing problem and involves finding the densest way of arranging spheres in space by packing as many spheres as possible into a large volume, assuming them to be rigid, non overlapping and all the same size.

In 1965 at Glasgow University, John Leech constructed a lattice packing spheres in 24 dimensional space. This lattice is based on the Golay code and it is described as the densest known packing of spheres in 24 dimensions (Sloane, 1984). It is known as the Leech lattice. In mathematics 24 dimensional space just consists of points with 24 coordinates instead of 3. Each sphere in the lattice is able to "kiss" 196,560 others. This means that the spheres are arranged in such a way that a central sphere is touched by 196,560 surrounding ones. The centres of the spheres in the Leech lattice all have the form 2C+4X or I+2C+4Y where I is the point (1,1,.....,1) and X and Y range over all the points that are integers. The Leech lattice is used in many diverse areas of mathematics for example in topology, group theory and polynomial work. The covering of the Golay code must be extremely efficient if it is able to help construct such a dense packing of spheres and the formation of the lattice emphasises the remarkable power that the codes possess.

6.0 EXPERIMENTAL AND RESULTS

The use of the Hadamard and Golay codes for phase permutation was tested on the Sheldrick database of difficult structures. The reasons why these structures are difficult to solve are discussed in a previous chapter. However, as discussed before, for some of the structures it is impossible to say exactly why they are difficult to solve using direct methods. It is for this reason that the database was used to investigate the use of error correcting codes.

The investigation was carried out in the following manner:

1. The error correcting codes were incorporated into a trial version of MITHRIL94 and the starting set of reflections phased using either the Hadamard or the Golay code by the method described previously.

2. In both cases the convergence subroutine was modified to allow for the starting sets to be larger. It is also believed, within certain limits, that it is more beneficial to permute a larger starting set of reflections than to sample more finely with fewer reflections as the additional reflections are said to provide a larger "dynamic range" (Carter, 1994).

(i) The Golay code is composed of 24 digits with each digit having one degree of freedom. Centric reflections have one degree of freedom and are defined by one digit from the code, thus the starting set for centrosymmetric structures contained 24 reflections. Acentric reflections have two degrees of freedom and are equivalent to two signs of the code. Thus the starting set for noncentrosymmetric structures contained less than 24 reflections, the exact number depending on how many centric and acentric reflections were present.

Two other extended forms of the Golay code having degrees of freedom of 96 and 192 were also investigated in the following manner:

- A random number was generated and multiplied by 4096 with the resulting integer formed equalling a row of the Golay code.
- When this row had been used to assign phases to the starting set of reflections another row was randomly picked.

- This continued until a maximum of 96 or 192 reflections had been chosen, depending on which code was being investigated. Again centric reflections required one piece of code and acentric reflections required two bits of the code. The results produced from the two extended forms of the Golay code were promising but only the results from the original [24,12,8] code will be discussed, as overall they achieved the best results.

(ii) The Hadamard code has 64 degrees of freedom, thus structures with a centrosymmetric space group had a starting set of reflections containing 64 reflections. Structures with a non centrosymmetric space group had less than 64 reflections depending on the mixture of acentric and centric reflections.

3. The phases were then expanded and refined using one of two weighted tangent refinement subroutines, the MULTAN weighting scheme or the Hull-Irwin statistically weighted tangent refinement subroutine (Hull & Irwin, 1978). As discussed in Chapter 1, this can be useful in cases of psuedosymmetry, symmorphic space groups, over-consistent phase sets and if the structure contains heavy atoms.

4. Figures of merit were produced for each phase set and the number of phase sets that were "correct" and solved the structure were noted. For the Golay code figures of merit for 4096 phase sets were produced. The Hadamard code produced figures of merit for 128 phase sets.

5. The structures were then tested on MITHRIL94 without using the error correcting codes to phase the starting set, but instead by using magic integers.

6. To compare the magic integer results with the results obtained for the Golay code the appropriate number of reflections in the starting set were chosen to produce 4096 or as near to 4096 phase sets as possible. For comparison with the Hadamard code the starting set of reflections had to produce 128 or as near to 128 phase sets as possible.

7. The results can be seen in four tables:

- Table3 - MITHRIL94 with or without the Hadamard code
- Table4- MITHRIL94 with or without the Golay code
- Table5 - MITHRIL94 with or without the Hadamard code(SWTR)
- Table6 - MITHRIL94 with or without the Golay code (SWTR)

TABLE 3. The Hadamard code

STRUCTURE	SPACE GROUP	MITHRIL94 WITH H(64) FROM 128 FIGS	MITHRIL94 WITHOUT H(64)
DIAM	$P4_2/n$	8	18
QUINOL	$R\bar{3}$	3	5
SELENID	$P2_1$	11	25
AZET	$Pca2_1$	3	1
TUR 10	$P6_322$	0	1
BED 2	I4	6	5
LOGANIN	$P2_12_12_1$	1	7
DIOL	$I\bar{4}2d$	0	3
APAPA	$P4_12_12$	0	0
TPALA	$P2_1$	6	3
TVAL	P1	2	0
NEWQB	$P\bar{1}$	1	0
GOLD 2	Cc	4	12
MUNICH 1	C2	1	0
MBH2	P1	4	18
PGE2	P1	0	5
SUOA	$P2_12_12_1$	0	25

TABLE 4. The Golay code

STRUCTURE	SPACE GROUP	MITHRIL94 WITH THE GOLAY CODE FROM 4096 FIGS	MITHRIL94 WITHOUT THE GOLAY CODE
DIAM	$P4_2/n$	309	430
QUINOL	$R\bar{3}$	314	341
SELENID	$P2_1$	110	80
AZET	$Pca2_1$	5	6
TUR 10	$P6_322$	44	48
BED 2	I4	207	278
LOGANIN	$P2_12_12_1$	97	57
DIOL	$I\bar{4}2d$	1	10
APAPA	$P4_12_12$	0	0
TPALA	$P2_1$	2	1
TVAL	P1	4	5
NEWQB	$P\bar{1}$	5	1
GOLD 2	Cc	505	278
MUNICH 1	C2	3	20
MBH2	P1	554	20
PGE2	P1	21	16
SUOA	$P2_12_12_1$	6	13

TABLE 5. The Hadamard code with SWTR

STRUCTURE	SPACE GROUP	MITHRIL94 WITH SWTR AND H(64) FROM 128 FIGS	MITHRIL94 WITH SWTR AND WITHOUT H(64)
SELENID	$P2_1$	12	17
AZET	$Pca2_1$	4	1
TUR10	$P6_322$	0	1
TPALA	$P2_1$	7	3
TVAL	P1	1	0
MUNICH 1	C2	1	1
MBH2	P1	3	0
PGE2	P1	12	3
SUOA	$P2_12_12_1$	0	0

TABLE 6. The Golay code with SWTR

STRUCTURE	SPACE GROUP	MITHRIL94 WITH SWTR AND THE GOLAY CODE FROM 4096 FIGS	MITHRIL94 WITH SWTR AND WITHOUT THE GOLAY CODE
SELENID	$P2_1$	1	0
AZET	$Pca2_1$	6	0
TUR10	$P6_322$	61	52
TPALA	$P2_1$	1	2
TVAL	P1	13	9
MUNICH1	C2	1	2
MBH2	P1	8	3
PGE2	P1	10	6
SUOA	$P2_12_12_1$	4	6

7.0 DISCUSSION

For every structure MITHRIL94 was run with default parameters, that is no attempt was made to try and solve the structure by utilising any of the non-default options available in MITHRIL94. This was done to test the power of the codes. It is impossible to explain why the codes worked well for some structures and not for others just as it is impossible to say why one code is successful on one structure whilst the other code is not. The results will be discussed therefore in view of whether the error correcting codes can be used as a viable phase permutation technique.

7.1 Table3 - MITHRIL94 with and without the Hadamard Code

The results in this table do not look outstanding at first glance, but the very fact that structures are solved by the codes is the proof that this code does work. Also as discussed before the Hadamard (64) code produces an extremely sparse economical covering and only 128 solutions. From the seventeen structures tested, six of the structures gave good results, namely:

- Azet - MITHRIL94 with the Hadamard code produced 3 correct solutions from 128 figures of merit, without the Hadamard code only 1 was correct.
- Tval - This structure has an incomplete data set. MITHRIL94 without using the Hadamard code failed to solve this structure, with the code 2 complete structures were found from the 128 figures of merit.
- Newqb - MITHRIL94 with the Hadamard code produced one correct solution whereas without the codes the structure was not solved. This is the structure with very low symmetry and a lack of phase relationships and has always been a problem for MITHRIL94 to solve. The results produced from using the codes were therefore pleasing.
- Munich1 - MITHRIL94 with the Hadamard code produced one correct solution whereas again without the codes the structure was not solved.
- Tpala - MITHRIL94 with the Hadamard code produced six correct solutions whereas MITHRIL94 without produced only one.

A few of the results were disappointing namely for those structures that the codes could not solve, yet without using the codes did solve. However for most of the remaining structures, MITHRIL94 using error correcting codes could solve the structure although MITHRIL94 without the codes seemed to perform better.

7.2 Table5 - MITHRIL94 with and without the Hadamard Code (SWTR)

Nine structures were tested using the SWTR subroutine which incorporates the Hull and Irwin weighting scheme. From the structures tested, five of them gave better results using the error correcting codes. The best of these was:

- PGE2 - this structure was not solved by the Hadamard code using the standard Multan weighting scheme. However using the Hadamard code with SWTR produces 12 solutions compared with the 3 solutions produced without using the code.

7.3 Table4- MITHRIL94 with and without the Golay Code

On the whole the Golay code appeared to produce better results than the Hadamard, this is not surprising as the covering produced by the Golay code is much more efficient and much less sparse than the Hadamard. The most notable successes of the Golay code were:

- Loganin - 97 structures produced by the code, compared to only 57 without using the code.
- Newqb - This was an extremely good result for Newqb with 5 solutions produced using the Golay code compared to only 1 being produced without the code.
- Goldman2 - 505 solutions produced by the Golay code compared to only 278 being produced without the code.
- MBH2 - This structure produced the best results for the Golay code with the solution being found 554 times compared to only 20 times when the code was not used.

Although the results produced by MITHRIL94 without the Golay code were sometimes better, the Golay code using the Multan weighting scheme was always able to find at least one solution for each structure.

7.4 Table6 - MITHRIL94 with and without the Golay Code (SWTR)

Nine structures were tested using the SWTR subroutine which incorporates the Hull and Irwin weighting scheme. From the structures tested, six of them gave better results using the error correcting codes. The best of these was:

- Tur10 - This structure is included in the database as it has an unusual space group. The result for MITHRIL with SWTR and using the Golay code was the best result gained for this structure.

8.0 Summary

The previous results clearly show that the Golay and Hadamard error correcting codes are capable of producing structure solutions. The codes are therefore viable phase permutation techniques. The use of the codes for phase permutation also provides a significant improvement in the results for some structures where conventional direct methods either fails or is poor. Thus the results from this experiment prove that coding theory is a useful tool in multisolution direct methods and shows potential for further development.

References

Bricogne, G. (1993). *Acta Cryst.* **D49,** 37-60

Carter, C. W., Jr. (1992). in *Crystallisation of Proteins and Nucleic Acids: A Practical Approach,* edited by A. Ducruix & R. Giege Oxford: IRL Press 47-71

Carter, C. W., Jr. (1994). *Entropy Maximization, Permutation and Likelihood Scoring Methods for Improving Macromolecular Electron Density Maps:* Daresbury School

Conway, J. H. & Sloane, N. J. A. (1988). *Sphere Packings, Lattices and Groups.* New York: Springer-Verlag, 75-79

Germain, G., Main, P. & Woolfson, M.M. (1970). *Acta Cryst.* **B26,** 274-285

Giacovazzo, C., Monaco, H.L., Viterbo, D., Scordari, F., Gilli, G., Zanotti, G., Catti, M. (1992). *Fundamentals of Crystallography.* **5,** Oxford University Press, 319-402

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

Gilmore, C. J. & Brown, S. R. (1988). *J. Appl. Cryst.* **21,** 571-572

Good, I. J. (1954). *Acta Cryst.* **7,** 603-604

Hill, R. (1993). *A First Course in Coding Theory.* **2,** 11-29

Hull, S.E. & Irwin, M.J. (1978). *Acta. Cryst.* **A34,** 863-870

Street, A., Street, D. J. (1987). *Combinatorics of Experimental Design.* Clarendon Press - Oxford, 47-52

Sloane, N. J. A. (1984). *Scientific American.* **250,** 116-125

Thompson, T. M. (1983). *From error correcting codes through sphere packings to simple groups,* [Washington]: Mathematical Association of America

Woolfson, M. M. (1954). *Acta Cryst.* **7,** 65-67

CHAPTER 5

ERROR CORRECTING CODES AND PHASE ANNEALING COMBINED

1.0 THEORY

1.1 Introduction

As discussed in chapter 4, error correcting codes are able to sample phase space efficiently and hence they can be used as a phase permutation technique (Bricogne,1993). Furthermore, the results from chapter 3 illustrate that phase annealing is also successful, flipping the phases until the "global" correct phase set is found (Sheldrick, 1990). In view of these results, the next obvious step must be to combine these methods, with the error correcting codes phasing the starting set followed by annealing to refine the phases. The remainder of this chapter will discuss how these techniques were combined and the results they produced.

2.0 EXPERIMENTAL

Based on the analysis of the results gained from the coding theory investigation it was decided that the code to be applied with simulated annealing would be the Golay code, as overall it seemed to produce the best results. From the results produced from the phase annealing research, the use of the Golay code-annealing combination was applied only in the FASTAN and SWTR subroutines. By this stage, both phase annealing and coding theory techniques existed as options in MITHRIL94 (Gilmore, 1984; Gilmore & Brown, 1988) so mixing them was relatively simple, involving only a small amount of programming. The flow diagram on the next page illustrates the basic procedure:

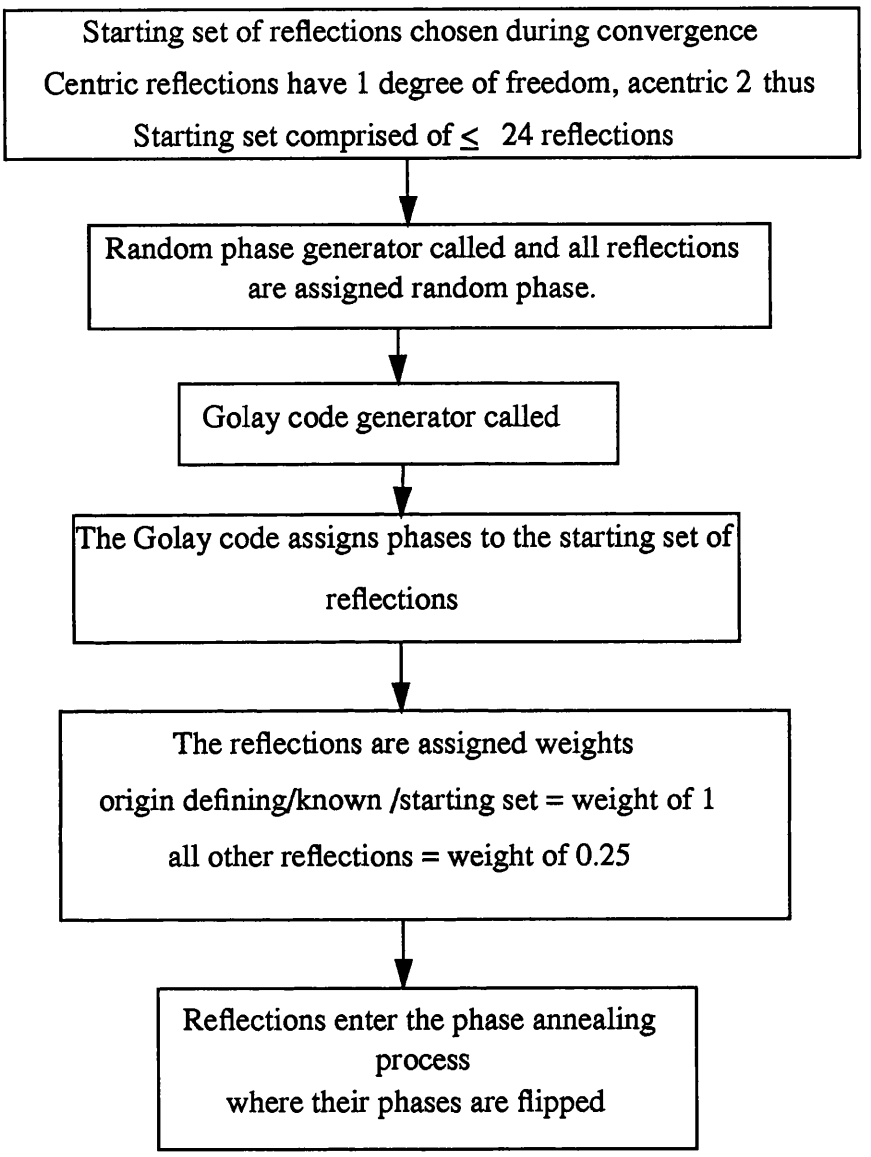


FIGURE 1 . Flowchart for phase annealing and coding theory combined

The origin defining, known and starting set of reflections were all assigned a weight of 1 so that they were not initially flipped by the phase annealing process and would retain their original assigned phases. The rest of the reflections were assigned weights of 0.25 and their original random phases were refined as required. Figures of merit were produced for each phase set and these were analysed to see if they represented a

"correct" phase set. The number of phase sets produced was 4096. The structures used to investigate the mixing of these techniques were again taken from the Sheldrick database of difficult structures. The results can be seen in two Tables:

- Table 1 - FASTAN with phase annealing and the Golay code
- Table 2 - SWTR with phase annealing and the Golay code

Each table compares the results from the Golay code only, annealing only and the results produced from mixing them both.

3.0 RESULTS

TABLE 1. FASTAN with annealing and the Golay code

Structure	The Golay Code (from 4096)	Annealing (from 1000)	Annealing and Golay code (from 4096)
Diam	309	574	751
Quinol	314	150	138
Selenid	110	3	98
Azet	5	1	3
Tursch 10	44	63	6
Bed2	207	126	250
Loganin	97	79	130
Diol	1	0	2
Apapa	0	0	0
Tpala	2	10	7
Tval	4	0	6
Newqb	5	2	10
Gold2	505	98	161
Munich1	3	8	15
PGE2	21	7	6
Suoa	6	0	3

TABLE 2. SWTR with annealing and Golay code

Structure	The Golay Code (from 4096)	Annealing (from 1000)	Annealing and Golay code (from 4096)
Selenid	1	9	155
Azet	6	10	70
Tursch 10	61	85	0
Bed2	42	101	280
Loganin	31	84	163
Apapa	0	2	0
Tpala	1	14	13
Tval	3	33	46
Newqb	1	0	1
Gold2	17	30	132
Munich1	1	6	19
PGE2	4	10	12
Suoa	4	2	14

4.0 DISCUSSION

The tables shown do not contain results for MITHRIL94 run on default, without annealing or the use of the codes, since what is being investigated here is just the cumulative effect of applying both techniques simultaneously. The results will be discussed in view of whether the joint use of the methods is better or worse than the results from each individual method.

4.1 FASTAN with annealing and the Golay code

From the 16 structures tested, 7 increased their correct solutions. A few are discussed below:

- Diam - this structure performed very well under both annealing and the Golay code and when these techniques were mixed it produced 751 correct phase sets from 4096, more than doubling the success rate produced from the use of the Golay code alone.
- Diol - the Golay code produced 1 correct phase set and annealing was unable to solve the structure at all. However the combination of these techniques produced 2 correct phase sets, indicating that for this structure the mixing of these methods gives better results.
- Tval - this structure produced results which were similar to diol; 4 phase sets were correct with the Golay code, whereas with annealing no correct phase sets were produced. However, when both techniques were applied, 6 correct phase sets were found.
- Newqb - the result for newqb was the best for this structure in this entire study, with 10 correct phase sets being produced.

For the rest of the structures:

The decrease in correct solutions when both techniques were applied was not very significant, with the overall correct phase sets being greater than one technique and less than the other and all the structures producing correct phase sets. However there are 2 structures whose results are worth discussion:

- Tursch 10 - annealing produced 63 correct phase sets and the Golay code produced 44 correct phase sets. However the annealing-codes combination drastically decreased the correct phase sets, producing to only 6.
- SUOA - the Golay code produced 6 correct phase sets, annealing produced zero. Only 3 phase sets were produced when the techniques were combined.

4.2 SWTR with annealing and the Golay code

From the 13 structures tested, nine increased the number of correct phase sets produced quite dramatically. A few of these are discussed below:

- Selenid - only 1 solution was produced from 4096 for the Golay code and 9 solutions for annealing. However when the techniques were applied together 155 correct phase sets were created.
- Azet - 6 correct phase sets were generated for the Golay code and 10 for annealing, however 70 correct solutions were created with the combination of these techniques.
- Loganin - 163 solutions produced when the techniques were combined, which is double the number of correct solutions for annealing and over 5 times more than the results for the Golay code.

In only one case was there a decrease in the number of correct solutions with the combination of both techniques.

- Tval - the number of correct solutions was 1 less than the annealing result and 12 more than in the Golay code case.

Two structures produced results that are worth discussion:

- Apapa - this structure has only been solved by using SWTR with phase annealing and the combination of techniques in this investigation was unable to produce any correct phase sets.
- Tursch 10 - again the results produced for this structure were very strange. 61 and 85 correct phase sets were produced for the Golay code and annealing respectively, however the combination of these techniques failed to generate any correct phase sets. This is probably a consequence of the mode of sampling phase space.

4.3 Conclusions

Overall the combined application of these two techniques has produced very interesting results. When the annealing-codes combination was a success it seemed that the techniques literally worked together to produced a definite increase in correct phase sets. This increase was far greater than the number of correct phase sets produced from either of the individual methods. In a few cases however, the opposite could be said, as the number of correct phase sets decreased with the annealing-codes combination. In view of these results the combined use of codes and annealing is a viable phase permutation technique and another useful tool in multisolution direct methods.

References

Bricogne, G. (1993). *Acta Cryst.* **D49,** 37-60

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

Gilmore, C. J. & Brown, S. R. (1988). *J. Appl. Cryst.* **21,** 571-572

Sheldrick, G. M. (1990). *Acta Cryst* **A46,** 467-473

CHAPTER 6

IMPLEMENTING ANNEALING IN CRYSTAN 6.3

1.0 INTRODUCTION

As discussed in Chapter 3, phase annealing was implemented into three of the tangent refinement modules in MITHRIL94

- FASTAN which uses the standard tangent refinement formula, described in Chapter 1.
- SWTR tangent formula which uses the Hull and Irwin weighting scheme (Hull & Irwin, 1978), described in Chapter1.
- X-Y (Debaerdemaeker and Woolfson, 1989) which uses the X-Y tangent formula, described in Chapter 3.

This version of MITHRIL94, containing the phase annealing facility, has been adapted for CRYSTAN 6.3 which has a graphical user interface (GUI). The commands entered by the user for phase annealing are exactly the same as the earlier line based commands except phase annealing is controlled by buttons and selectors making the program more user friendly. This version of the program also has the added advantage of producing graphical output. The GUI version of MITHRIL94 is contained in a commercial computer program, CRYSTAN 6.3 (MAC Science, 1995).

1.1 CRYSTAN 6.3

CRYSTAN is a powerful, state-of-the art computer program for solving, refining and publishing crystal structures automatically from X-ray diffraction data. It works in a windows environment with SUN and Silicon Graphics computers and offers the following facilities:

- SG-Merge: A computer program to automatically assign space groups, merge intensity data and correct for crystal decomposition.
- Test i to test for centrosymmetric non-centrosymmetric structures.
- An AUTO option to solve and refine crystal structures including H atom addition.
- A program, MODEL, to manipulate structures during the process of solution and refinement.
- A real-time, graphical least squares program, RTG-LSQ, that interactively refines crystal structure.
- A structure factor program.
- A multi-optional Fourier program, FOURIER.
- Five different absorption correction methods in the ABSORB module.

- MOPAC for calculating theoretical charges on refined structures.
- A PUBLISH program for generating tables, CIF files and a report for publishing the final structure.
- Five different colour graphics options:

  Interactive ORTEP.

  PLUTO

  PLOTQ for drawing contoured Fourier maps.

  XXMOL for three-dimensional colour graphics.

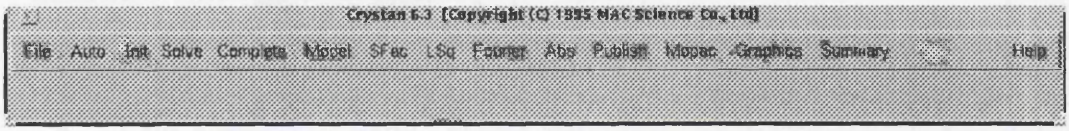
  PReDS for 3d viewing of density maps & models.

CRYSTAN, as well as offering the facilities listed above, also contains five programs to solve crystal structures automatically. They are SIR, MITHRIL94, SHELXS-86, DIRDIF and Monte-Carlo MULTAN. All of these programs have been adapted to have a graphical user interface (GUI). The current version of MITHRIL94 available in CRYSTAN 6.3 contains exactly the same features as before, the only difference being the interface.

1.2 Interface to Crystan

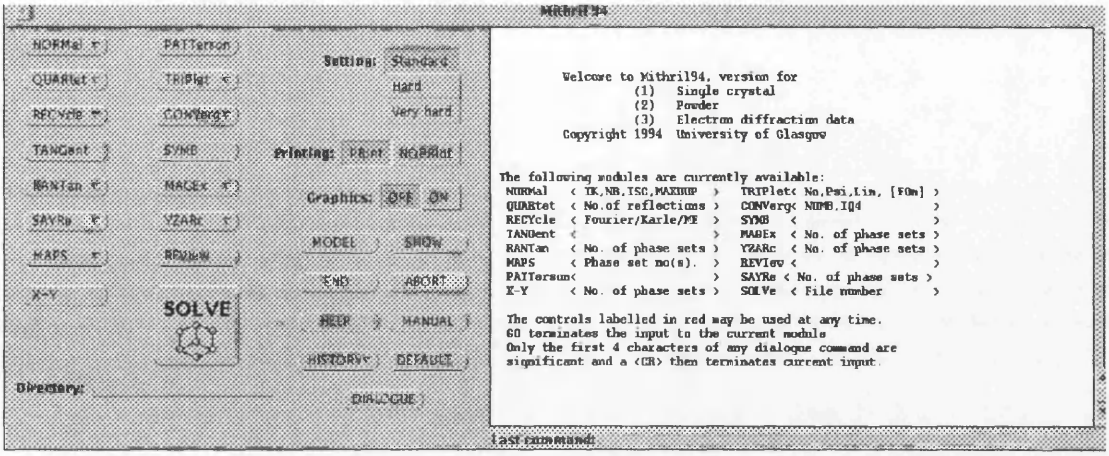
To initiate MITHRIL94, select **SOLVE** from the main Crystan-GM menu, then **MITHRIL-94** and **EXEC**. The CRYSTAN window is seen below:

FIGURE 1. The CRYSTAN window:



MITHRIL94 is loaded and reads the reflection and model information from the files created by Crystan. Whilst loading, the cursor changes to a small clock to indicate that MITHRIL94 is busy. When it reverts to an arrow, MITHRIL94 is ready to interact with the user. Buttons, selectors and menus operate in the normal OPEN LOOK manner. All MITHRIL94 commands are accessed through these menus, buttons and selectors. The MITHRIL94 interface is shown by Figure 2. on the next page.

FIGURE 2. The MITHRIL94 interface



2.0 Using MITHRIL94

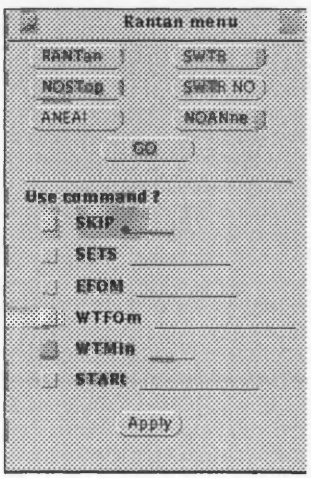
To use simulated annealing in the GUI version of MITHRIL94 the annealing options are present in the RANTAN and X-Y modules.

By entering **RANT** the RANTAN module is initialised.

2.1 RANTAN (Yao Jia-Xing, 1981)

In one respect the RANTAN module is similar to TANGENT in that it refines phase angles. The difference is that RANTAN uses random phases for all the unknown phases and refines them using the tangent formula, rather than using the phase permutation techniques of TANGENT. RANTAN is never called automatically.

FIGURE 3. The RANTAN dialog box



The following commands, apart from the first, may be entered in any order:

RANTan NO. OF PHASE SETS TO GENERATE.

Calls and initialises the RANTAN module. The number of phase sets to generate is entered. The default is:

No. of phase sets = 100 + 50 * Idif + 100 * Ivdif

where:

Idif = 0 / 1 for Standard / Hard structure.
Ivdif = 0 / 1 for Standard / Very hard structure.

SWTR [NO]

This command causes the Hull - Irwin weighting scheme to be used rather than the traditional MULTAN80 weights. This is a default when Karle recycling is used, otherwise the standard procedure is used. If for some reason the Hull - Irwin scheme is to be used but the user wishes to revert to the standard weighting method, then the command:

SWTR NO where "NO" is a keyword is used.

SKIP N

This command is usually used for restarts. It causes the first N phase sets to be skipped before starting tangent refinement.

WTMIn WEIGHTS OF UNKNOWN REFLECTIONS.

The random phases have weights of 0.25 assigned to them before refinement begins, but if a different value is wanted, then this command can be used. Some experimentation with these weights can be useful in difficult cases.

EFOM [ALl]/[NOne] or EFOM NO., CUT; EFOM NO., CUT; EFOM NO., CUT

This invokes the early figures of merit. There are three EFOMs numbered as follows:

(1) ψ_0 + Resid (R_{Karle})
(2) NQEST.
(3) ψ_0 alone (applied later in refinement than (1) above).

These are the numbers used on the EFOM command. The cut values listed on the command are the maximum values that the early figure of merit may have when tested; any solution with values greater than these cut-offs are rejected. The defaults are: (1) 1.8 (2) 0.0 (3) 1.6

Usually the EFOMs are not used. The command:

EFOM NONE

also has this effect, and can be used to switch off EFOMs already invoked. If you wish to use all three early figures of merit then the command:

EFOM ALL CUT(1) CUT(2) CUT(3)

can be used, where CUT(1), CUT(2) etc. refer to the cut-off parameters discussed above. If all three cut-offs are absent then the default values are used for all these parameters. If only one appears, it is presumed to apply to the first EFOM; two parameters are assumed to apply to the first two etc. A zero parameter gives defaults. E.g.

EFOM ALL 1.2 0.1 2.0

applies cut-offs of 1.2, 0.1 and 2.0 respectively; whereas:

EFOM ALL 0 0 2.0

will give defaults (1.8 and 0.0) for the first two EFOM's and a cut-off of 2.0 for the third. If only certain EFOM's are wanted, then do not use the "ALL" keyword. Instead enter the EFOM number followed by the required cut-off. In these circumstances, only the specified early figures of merit are invoked. E.g.

EFOM 2 0.1 3 2.0 or EFOM 3 2.0 2 0.1

invokes the second EFOM with a cut-off of 0.1 and the third with a cut-off of 2.0; the first EFOM is not used. Note that all the EFOM requirements must appear on a single EFOM command.

It is quite difficult to devise suitable defaults applicable to all situations, and some parameter tuning may be necessary. EFOMs are not available with the Hull-Irwin weighting scheme.

WTFOM W1, W2, W3, W4

This command defines the relative weights of the five figures of merit used by the RANTAN module when calculating a combined figure of merit (CFOM). The defaults are as follows:

Figure of Merit	Weight	Default	Default in symmorphic cases
ABS FOM	W1	1.0	0.6
ψ-ZERO	W2	1.0	1.2
RESID	W3	1.0	0.6
NQEST	W4	1.0	1.3

TABLE 1. Weight defaults for FOM's

If there are no appropriate relationships for a particular figure of merit then a weight of zero is assigned. The relative weights are normalised such that the maximum CFOM value is equal to the total number of figures of merit contributing to it. The LOGLIK figure of merit, if calculated, is not used in the combined figure of merit.

NOSTop

If the RANTAN module finds a solution with figures of merit that satisfy the following conditions:

- Resid (R_{Karle}) less than 20.0
- ψ_0 less than 1.25
- NQEST less than -0.15
- The figures of merit above are within 5% of the best so far.

(assuming that these figures of merit are available), then the module assumes that this is the correct solution and exits. Users in the interactive modes will be questioned first if they wish to accept this solution, but batch users will not. The command NOSTOP switches off these tests. So do the HARD and VERY_HARD options.

SETS N1, N2, N3, N4..... etc.

With this option only the phase sets with numbers N1, N2 etc. are investigated via the RANTAN module. This is useful for re-runs. Unlike the SKIP command, the other phase sets do not need to be on file 11.

STARt IX, IY

Two odd integers used to seed the random number generator. The default values are 1 and 1 (cf. YZARC).

As usual, the commands TITLE, END, MENU, LEVEL, NOPRINT, PRINT, DEFAULT, HARD, VERY_HARD, MODEL, SHOW and X are available as appropriate.

ANEA

Requests simulated annealing. If the traditional MULTAN80 weights are being used then the default parameters are 40 cycles of annealing followed by 3 cycles of refinement. If the Hull-Irwin weighting scheme is used the default parameters are 80 cycles of annealing followed by 5 cycles of refinement.

NOAN

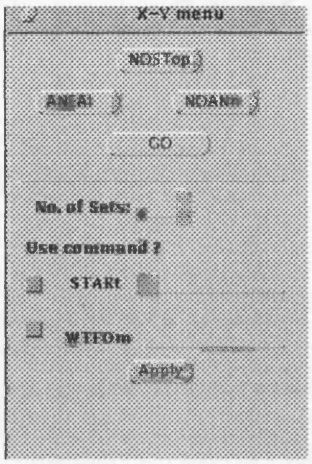
Switches off simulated annealing.

By entering **RANT** followed by **SWTR** followed by **ANEA** initialises the use of phase annealing with the Hull-Irwin (Hull & Irwin, 1978)weighting scheme or by entering **RANT** followed by **ANEA** initialises the use of phase annealing with the use of traditional MULTAN80 weights (Main, Fiske, Germain, Hull, Declercq, Lessinger & Woolfson, 1980).

2.2 X-Y(Debaerdemaeker and Woolfson, 1989)

This is an alternative to tangent refinement and is accessed by entering **X-Y** in the MITHRIL94 interface.

FIGURE 4. The X-Y dialog box



The options are:

NOSTop

If the X-Y module finds a solution with figures of merit that satisfy the conditions:

- Resid (R_{Karle}) less than 20.0
- ψ_0 less than 1.25
- NQEST less than -0.15.
- The figures of merit above are within 5% of the best so far

Then the module assumes that this is the correct solution and exits. Users in the interactive mode will be questioned first if they wish to accept this solution, but batch users will not. The command NOSTOP switches off these tests. So do the HARD and VERY_HARD options. NOSTOP is often worthwhile as the correct solution can often be missed under the early stop algorithm.

ANEA

Requests simulated annealing. The default parameters are 30 cycles of annealing followed by 4 cycles of refinement.

NOAN

Switches off simulated annealing.

SETS

Defines the number of phase sets to be produced.

STARt IX, IY

Two odd integers used to seed the random number generator. The default values are 1 and 1 (cf. YZARC).

WTFOm W1, W2, W3, W4

This command defines the relative weights of the four figures of merit used by the TANGENT module when calculating a combined figure of merit (CFOM). The defaults are as follows:

TABLE 2. Weight defaults for FOM's

Figure of Merit	Weight	Default	Default in symmorphic cases
ABS FOM	W1	1.0	0.6
ψ-ZERO	W2	1.0	1.2
RESID	W3	1.0	0.6
NQEST	W4	1.0	1.3

If there are no appropriate relationships for a particular figure of merit then a weight of zero is assigned. The relative weights are normalised such that the maximum CFOM value is equal to the total number of figures of merit contributing to it. The LOGLIK figure of merit, if calculated, is not used in the combined figure of merit.

By entering **X-Y** followed by **ANEA** allows phase annealing to be used with the X-Y module.

References

Debaerdemaeker and Woolfson, *Acta Cryst.* (1989). **A45**, 349-353

Hull, S.E. & Irwin, M.J. (1978). *Acta Cryst.* **A34**, 863-870

MAC Science (Material Analysis and Characterisation) Co, Ltd. Tokyo, Japan

Main, P. et al. (1980). *Multan-80 Manual*, University of York

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

APPENDIX A

LISTINGS OF SUBROUTINES IN MITHRIL94:

PHASE ANNEALING

INTRODUCTION TO APPENDIX A

This appendix contains the source code for the subroutines

- FASTAN
- SWTR

which contain the phase annealing code from MITHRIL94 and are described fully in Chapter 3 of this thesis. The source code for X-Y is not included in this appendix as the phase annealing code is very similar to the two subroutines listed.

```
C-------------------------------------------------------------------
      SUBROUTINE FASTAN(IREJ,SIGD)                                    FASTAN
      LOGICAL TEST
      DIMENSION CTABLE(360)
      COMMON /ANNEAL /ANEA
C**** GENERATE STARTING SETS OF PHASES FROM CONVERGENCE RESULTS
      COMMON /IO /NSPEC,NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(200),A(200),ICH1(10),ICH2(10),TITLE(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,IVERB
      COMMON /BLK1 /IND2(80000),IND3(80000),IND4(80000),IND5(80000),
     1 EEE(80000),
     1 SUMNUM(800),SUMDEN(800),ALPHA(800),WT(800),IPHAZ(800),
     2 IORDE(1300),MKG(800),PALF(800),IZ(800),MKANG(800),EALF(800),
     3 LIM(1301),MAXHKL(3),SUMN(800),SUMD(800),E(800)
      COMMON /BLK2 /IS(2,3,24),TS(3,24),P(6),CX(9),NSYM,ICENT,
     1 NEX,LATT,LAT,PTS,KSYS,IAPX,NGP,NTOT,NASU,NRD,AMX(5),
     2 AMN(5),LINE(17),LINEX(17),WTFOM(5),IMK,ITAN,IPUB,ISKIP,NLIM,WLIM
     3 ,PAD2(1302)
      COMMON /BLK3 /STABLE(450),NQ1(500),NQ2(500),AQ(500),NQTOT,NQ4,NQ5,
     1 FOM(400),I4,I5,IQTOT,CFOM,RN4,RN5,ISTP,CVR(10),VVR(10),NA(10),
     3 NW(10),NO(10),NK,SIGMA,PAD3(170)
      COMMON /BLK4 /TALF(800),RALF(800),ISPZRO(500),PAD4(448)
      COMMON /BLK5 /NUMB,NUMSET,NRAL,NANT(4),
     1 ALFRAN,NSX,IEF3,IEF4,IEF5,NDET,NAT,IFOM(3),IEFOM,IZRO,IXRAN,
     2 IYRAN,IRAND,WMIN,IWMIN,NSREQ,CUTT(3),ICC,IMP,PAD5(5)
       LOGICAL ANEA
       EQUIVALENCE (STABLE(91),CTABLE(1)),(LK10,LK(10)),(LK5,LK(5)),
     1 (LK17,LK(17)),(LK16,LK(16)),(LK6,LK(6)),(LK29,LK(29)),
     1 (LK1,LK(1))
C**** I1 /I0 (BESSEL FUNCTION)
      VEC(U)=U*(U+0.4807) /((U+0.8636)*U+1.3943)
C**** CONTROL CHARACTER FOR EFOMS
      ICC=IH(13)
       IF(IRAND.EQ.1.AND.IEFOM.EQ.0) ICC=IH(45)
      RN4=0.0
      RN5=0.0
      CFOM=0.0
C     CODE TO INITIALISE THE BOLTZ VALUE
       IF(ANEA)THEN
      BOLTZ=0.8

C     CODE TO SET THE MAX NUMBER OF CYCLES
      MAXCY=40
C     CODE TO INITIALISE THE TOTAL SHIFT AND NUMBER TIMES SHIFT CALCUL.
      TSHIF=0.0
      NSHIF=0.0
      ENDIFIF
      MARK=0
      NNN = MIN0(100, NDET)
       IF(IRAND.EQ.1) NNN=NDET
C     CODE TO CALCULATE THE VALUE OF SHELDRICKS BETA
       IF(ANEA)THEN
        BETAS=-ALOG(BOLTZ) /SQRT((ALFRAN**2) /NNN)
        ENDIFIF

      CUT = 25.0
      SALF = 0.0
      NCYCLE=0
      NCYCLE2=0
       IF ( ICENT .EQ. -1) IABSCENT=0
```

```
C**** CLEAR ARRAYS AT START OF DETERMINATION
      DO 990 I=1,NUMB

      EALF(I)=0.0
      TALF(I)=0.0
      PALF(I)=0.0
      SUMN(I)=0.0
      SUMD(I)=0.0
      SUMNUM(I)=0.0
  990 SUMDEN(I)=0.0
C**** START OF NEXT CYCLE
 1000 SUMALF=SALF
      TSHIF=0.0
      NSHIF=0.0
      MSHIF=0.0
      FLIPHAZ=0.0
      SALF=0.0
C     CODE TO INCREMENT THE ANNEALING CYCLES
      NCYCLE=NCYCLE+1
      NCYCLE2=NCYCLE2+1

       IF (NCYCLE2 .LE .25..AND.ANEA )    BETAS=BETAS+0.002
       IF(NCYCLE2.GT.25.AND.NCYCLE2.LE.35.AND.ANEA)BETAS=BETAS+0.03
       IF (NCYCLE2 .GT.35.AND. NCYCLE2 .LE. 40.AND.ANEA)BETAS=BETAS+0.05


      IABSCENT=ICENT


C****  TEST  =  TRUE  IF EXPECTED ALPHA IS TO BE CALCULATED
C**** I.E. 2 CYCLES BEFORE FIRST EFOM AND FINAL 2 CYCLES
C**** IN THE CASE OF RANTAN TEST IS TRUE 2 CYCLES BEFORE FIRST
C**** EFOM OR FINAL 2 CYCLES
        IF(IRAND.EQ.1) THEN
        TEST=((NCYCLE.EQ.5.OR.NCYCLE.EQ.6).AND.IEFOM.GT.0).OR.MARK.GT.0
        ELSE
        TEST=(CUT.LT.3.0.AND.CUT.GT.1.5.AND.IEFOM.GT.0).OR.MARK.GT.0
        ENDIFIF
        IF(ANEA)THEN
        TEST= .TRUE.
        ENDIFIF
       DO 1500 I=1,NNN
       LL=IORDE(I)
       LI=LIM(LL)+1
       LS=LIM(LL+1)
       IF(LI.GT.LS) GO TO 1200
       DO 1100 JJ=LI,LS
       IMA=0
       IEF=0
       IL=IND3(JJ)
       ILA=IABS(IL)
       W=ABS(WT(ILA))
       IF(W.LT.WLIM) GOTO 1100
       IRK=IND2(JJ)
       IRA=IABS(IRK)
       WW=ABS(WT(IRA))
       IF(WW.LT.WLIM) GOTO 1100
       W=W*WW
C**** IS THERE A POSSIBILITY OF A QUARTET ?
       IF(IEF3.NE.0) THEN
C**** YES
C**** QUARTET POSSIBLY PRESENT
         IM=IND4(JJ)
         IMA=IABS(IM)
```

```
          IF(IMA.EQ.0) GOTO 110
          WW=ABS(WT(IMA))
          IF(WW.LT.WLIM) GOTO 1100
          W=W*WW
          IPI=IND5(JJ)
          ENDIFIF
  110  EE=EEE(JJ)*W
       IF(IMA.GT.0) GOTO 9110
          IF(TEST)THEN
          VECX=EE*VEC(EEE(JJ))
          EALF(LL)=EALF(LL)+VECX
          TALF(LL)=TALF(LL)+EE*EE-VECX*VECX
          ENDIFIF
 9110  IP=IND5(JJ)+2160
          IF(IABSCENT.EQ.0) THEN
C****     NON-CENTROSYMMETRIC
          IP=IP+ISIGN(IPHAZ(ILA),IL)
C            write(*,*)IPHAZ(ILA),NCYCLE,IP
          IF(IRA.GT.0) IP=IP+ISIGN(IPHAZ(IRA),IRK)
          IF (IEF3.NE.0 .AND. IMA .GT. 0) IP=IP+ISIGN(IPHAZ(IMA),IM)
          IARG=MOD(IP,360)+1
C           write(*,*) IP,NCYCLE
          SUMNUM(LL)=SUMNUM(LL)+EE*STABLE(IARG)
          SUMDEN(LL)=SUMDEN(LL)+EE*CTABLE(IARG)
             IF(IMA.GT.0) THEN
             SUMN(LL)=SUMN(LL)+EE*STABLE(IARG)
             SUMD(LL)=SUMD(LL)+EE*CTABLE(IARG)
             ENDIFIF
          ELSE
C****     CENTROSYMMETRIC
   150    IP=IPHAZ(ILA)+IP
          IF(IRA.GT.0) IP=IP+IPHAZ(IRA)
          IF(IMA.GT.0) IP=IP+IPHAZ(IMA)
          IF(MOD(IP,360).NE.0) EE=-EE
          SUMNUM(LL)=SUMNUM(LL)+EE
          IF(IMA.GT.0) SUMN(LL)=SUMN(LL)+EE
          ENDIFIF
  1100  CONTINUE
  1200  ID=15*IABS(MKANG(LL))-14
        IF(ID.EQ.1.OR.IABSCENT.EQ.1) GO TO 1300
        T2=SUMNUM(LL)*STABLE(ID)+SUMDEN(LL)*CTABLE(ID)
        SUMNUM(LL)=T2*STABLE(ID)
        SUMDEN(LL)=T2*CTABLE(ID)
  1300  ALFA=(SUMNUM(LL)-SUMN(LL))**2+(SUMDEN(LL)-SUMD(LL))**2
        IF(ALFA.EQ.0.0) GOTO 1490
        IF (MKANG(LL).LE.0) GOTO 1320
        SALF=SALF+ALFA
        IF(ALFA.LT.CUT.AND.IRAND.EQ.0) GOTO 1490
        WATE=1.0
        IF(ALFA.LT.25.0) WATE=0.2*SQRT(ALFA)
        IF(WT(LL) .LT. 0.0 .AND. WATE .LT. (-WT(LL))) GO TO 1320
        WT(LL)=AMAX1(WATE,0.15)
  1111    IF(IABSCENT.NE.1) THEN
          PHAZ=RTOD*ATAN2(SUMNUM(LL),SUMDEN(LL))
          ELSE
          PHAZ=90.0-SIGN(90.0,SUMNUM(LL))
          ENDIFIF
        IF(PHAZ.LE.0.0) PHAZ=PHAZ+360.0

        IPHAZ(LL)=PHAZ+0.5
```

```
C       CODE TO MISS OUT THE FOLLOWING TANGENT REFINEMENT
          IF(NCYCLE2.GE. 1 .AND. NCYCLE2 .LE. MAXCY.AND.ANEA)GOTO  3333
          IF(NCYCLE2.GT.MAXCY.AND.NCYCLE2.LE.MAXCY+3.AND.ANEA )GOTO  1320
C       END OF CODE
          IF(ANEA)THEN
          GOTO 3333
          ELSE
          GOTO 1320
          ENDIFIF

C       TO DETERMINE WHETHER USE CENTRO /NONCENTRO CODE
 3333   IF(ABS(MKANG(LL)).EQ.1)GOTO 4444
C       SIMULATED ANNEALING CODE FOR CENTROSYMMETRIC CASE
C       CODE TO CALCULATE RANDOM NUMBERS BETWEEN O AND 1
            RNUMB=RAND(IXRAN,IYRAN)
C       CODE TO WORK OUT SHELDRICKS PROBABILITY
             PR1=ABS(SUMNUM(LL)*SIGD*0.5) /2
              PROBPL=0.5*TANH(PR1)+0.5
          IF (PROBPL .LT. 0.5) PROBPL=1-PROBPL

C       CODE TO CALCLATE (1-PROB) AND THUS RATIO
          PROBMI=(1-PROBPL)

C       CODE TO CALCULATE BOLTZMANN RATIO
          RATIO=0.5*((PROBMI /PROBPL)) /(EXP(BETAS))
C       CODE TO DETERMINE WHETHER PHASES SHOULD BE FLIPPED
          IF((IPHAZ(LL).EQ.180) .AND. (RATIO .GT. RNUMB))THEN
C       COUNT NUMBER OF FLIPPED PHASES
          FLIPHAZ=FLIPHAZ+1

          IPHAZ(LL)=0
          SUMNUM(LL)= -SUMNUM(LL)
            ELSE
          IF((IPHAZ(LL) .EQ. 0) .AND.(RATIO.GT.RNUMB))THEN
          FLIPHAZ=FLIPHAZ+1
          IPHAZ(LL)=180
          SUMNUM(LL)= -SUMNUM(LL)
            ELSE
          IF((IPHAZ(LL).EQ.360) .AND.(RATIO .GT.RNUMB))THEN
          FLIPHAZ=FLIPHAZ+1

          IPHAZ(LL)=180
          SUMNUM(LL)= -SUMNUM(LL)
            ELSE
          IF(IPHAZ(LL).EQ.360) THEN
          IPHAZ(LL)=0
          ENDIFIF
          ENDIFIF
          ENDIFIF
          ENDIFIF
C      TO MISS OUT THE NONCENTROSYMMETRIC CODE
         GOTO 1320
C       SIMULATED ANNEALING CODE FOR NONCENTROSYMMETRIC STRUCTURES

C       CODE TO CALCULATE RANDOM NUMBER BETWEEN O AND 1
 4444     RANON=RAND(IXRAN,IYRAN)
C       CODE TO DETERMINE SHIFT
          UPPER=(4*BETAS*ALFA)+(ALOG(RANON))
          DENOM =(4*BETAS*ALFA)-(ALOG(RANON))
          SHIFT=UPPER /DENOM
C       CODE TO CONVERT SHIFT INTO DEGREES
          SHIFT=RTOD*ACOS(SHIFT)
C       CODE TO DETERMINE THE SIGN OF THE SHIFT
          RSIGN=RAND(IXRAN,IYRAN)
```

```
          IF(RSIGN.LT.0.5)SHIFT=-SHIFT
C         CODE TO CONVERT SHIFT INTO INTEGERS
          SHIFT=AINT(SHIFT)

C         CODE TO ADD SHIFT ONTO RANDOM PHASE
          IPHAZ(LL)=IPHAZ(LL)+SHIFT
C         CODE TO CALCULATE TOTAL SHIFT
          TSHIF=TSHIF+ABS(SHIFT)

C         CODE TO CALCULATE NUMBER OF SHIFTS
          NSHIF=NSHIF+1

 1320   ALPHA(LL)=ALFA

          IF ( TEST ) PALF(LL)=EALF(LL)*EALF(LL)+TALF(LL)

          IF(LI.GT.LS) GO TO 1490
          WLL=ABS(WT(LL))
          DO 1450 JJ=LI,LS
          IPI=0
          IMA=0
          IFR=0
          IFM=0
          WR=1.0
          WM=1.0
          III=1
          IL=IND3(JJ)
          ILA=IABS(IL)
          WL=ABS(WT(ILA))
          IFL=ISIGN(IPHAZ(ILA),IL)
          IF(WL.GE.WLIM) III=2
          IRK=IND2(JJ)
          IRA=IABS(IRK)
          WR=ABS(WT(IRA))
          IF(WR.GE.WLIM) III=III+1
          IF(III.LT.2) GOTO 1450
          IFR=ISIGN(IPHAZ(IRA),IRK)
          IF(IEF3.EQ.0) GOTO 215
          IF(IND4(JJ).EQ.0) GOTO 215
C**** POSSIBLE QUARTET
            IM=IND4(JJ)
            IF(IM.NE.0) THEN
               IMA=IABS(IM)
               WM=ABS(WT(IMA))
               IF(WM.GE.WLIM) III=III+1
               IF(III.LT.3) GOTO 1450
               IFM=ISIGN(IPHAZ(IMA),IM)
            ENDIFIF
  215   IP=IND5(JJ) + IPI
C**** PART 1 : PHASE OF IL
          IF(WR.LT.WLIM) GOTO 260
          IF(WM.LT.WLIM) GOTO 300
          EE=EEE(JJ)*WR*WLL*WM
          IF(IMA.GT.0) GOTO 9215
            IF(TEST)THEN
            VECX=EE*VEC(EEE(JJ))
            EALF(ILA)=EALF(ILA)+VECX
            TALF(ILA)=TALF(ILA)+EE*EE-VECX*VECX
            ENDIFIF
 9215   IF(IABSCENT.EQ.1) GOTO 270
C**** NON-CENTROSYMMETRIC
          IARG=MOD(2160-(IFR-IPHAZ(LL)+IFM+IP)*ISIGN(1,IL),360)+1
          SUMNUM(ILA)=SUMNUM(ILA)+EE*STABLE(IARG)
          SUMDEN(ILA)=SUMDEN(ILA)+EE*CTABLE(IARG)
```

```
          IF(IMA.GT.0) THEN
          SUMN(ILA)=SUMN(ILA)+EE*STABLE(IARG)
          SUMD(ILA)=SUMD(ILA)+EE*CTABLE(IARG)
          ENDIFIF
         GOTO 260
C**** CENTROSYMMETRIC
  270 IARG=IPHAZ(IRA)-IPHAZ(LL)+2160
       IF(IMA.GT.0) IARG=IARG+IPHAZ(IMA)
       IF(MOD(IARG+IP,360).NE.0) EE=-EE
       SUMNUM(ILA)=SUMNUM(ILA)+EE
       IF(IMA.GT.0) SUMN(ILA)=SUMN(ILA)+EE
C**** PART2 PHASE OF IR
  260 IF(WL.LT.WLIM) GOTO 1450
       IF(WM.LT.WLIM) GOTO 300
       EE=EEE(JJ)*WLL*WL*WM
       IF(IMA.GT.0) GOTO 9260
         IF(TEST)THEN
         VECX=EE*VEC(EEE(JJ))
         EALF(IRA)=EALF(IRA)+VECX
         TALF(IRA)=TALF(IRA)+EE*EE-VECX*VECX
         ENDIFIF
 9260      IF(IABSCENT.EQ.0) THEN
C****      NON-CENTROSYMMETRIC
           IARG=MOD(2160-(IFL-IPHAZ(LL)+IFM+IP)*ISIGN(1,IRK),360)+1
           SUMNUM(IRA)=SUMNUM(IRA)+EE*STABLE(IARG)
           SUMDEN(IRA)=SUMDEN(IRA)+EE*CTABLE(IARG)
           IF(IMA.GT.0) THEN
             SUMN(IRA)=SUMN(IRA)+EE*STABLE(IARG)
             SUMD(IRA)=SUMD(IRA)+EE*CTABLE(IARG)
             ENDIFIF
           ELSE
C****      CENTROSYMMETRIC
  250        IARG=IPHAZ(ILA)+IPHAZ(LL)+IP+2160
             IF(IMA.GT.0) IARG=IARG+IPHAZ(IMA)
             IF(MOD(IARG,360).NE.0) EE=-EE
             SUMNUM(IRA)=SUMNUM(IRA)+EE
             IF(IMA.GT.0) SUMN(IRA)=SUMN(IRA)+EE
           ENDIFIF
C**** PART3 :PHASE OF IM
  300 IF(WR.LT.WLIM) GOTO 1450
      IF(IMA.EQ.0) GOTO 1450
  330 EE=EEE(JJ)*WLL*WL*WR
        IF(IABSCENT.EQ.0) THEN
C****   NON-CENTROSYMMETRIC
        IARG=MOD(2160-(IFL-IPHAZ(LL)+IFR+IP)*ISIGN(1,IM),360)+1
        SUMNUM(IMA)=SUMNUM(IMA)+EE*STABLE(IARG)
        SUMDEN(IMA)=SUMDEN(IMA)+EE*CTABLE(IARG)
        SUMN(IMA)=SUMN(IMA)+EE*STABLE(IARG)
        SUMD(IMA)=SUMD(IMA)+EE*CTABLE(IARG)
        ELSE
C****   CENTROSYMMETRIC
  350   IARG=IPHAZ(ILA)+IPHAZ(LL)+IPHAZ(IRA)+IP+2160
        IF(MOD(IARG,360).NE.0) EE=-EE
        SUMNUM(IMA)=SUMNUM(IMA)+EE
        SUMN(IMA)=SUMN(IMA)+EE
        ENDIFIF
 1450 CONTINUE
 1490 SUMNUM(LL)=0.0
      SUMDEN(LL)=0.0
      SUMN(LL)=0.0
      SUMD(LL)=0.0
      EALF(LL)=0.0
      TALF(LL)=0.0
 1500 CONTINUE
```

```
C        CALCULATE THE AVERAGE SHIFT
         MSHIF=TSHIF /NSHIF

C        CODE TO STOP THE REFINEMENT AND GET FOMS PRODUCED
         IF(NCYCLE2 .LE. MAXCY+2.AND.ANEA) GOTO 1000
         IF(NCYCLE2 .GT.MAXCY+2.AND. ANEA)GOTO 980
C**** FASTAN CONTROL STATEMENTS
  981    IF(IRAND.NE.1) THEN
         CUT=AMAX1(0.65*CUT,0.05)
         IF(CUT.GT.1.5) GO TO 1000
         IF (NNN.EQ.NDET) GOTO 1530
           ELSE
C**** RANTAN CONTROL PARAMETERS SET
         IF(NCYCLE-6) 1000,340,1530
         ENDIFIF
   340 IF(IEFOM.EQ.0) GOTO 1520
         IF(IFOM(1).EQ.0) GOTO 1540
C**** CALCULATE FIRST EARLY FIGURE OF MERIT
        CALL EFOM(NNN,IREJ,VALUE)
        IF (IREJ.EQ.1) GOTO 5000
C**** 2ND FIGURE OF MERIT WHICH IS BASED ON NQEST
  1540 IF(IFOM(2).EQ.0) GOTO 1520
        CALL NQEST
        IF(IQTOT.EQ.0) GOTO 1520
          IF(CFOM.GT.CUTT(2)) THEN
          IREJ=1
          GOTO 5400
          ENDIFIF
  1520 IF(IRAND.EQ.1) GOTO 1530
        NNN=NDET
        GO TO 1000
  1530 IF((IFOM(3).EQ.0.OR.CUT.GT.0.34.OR.CUT.LT.0.33).AND.IRAND.EQ.0)
       1 GOTO 1550
        IF(.NOT.(IFOM(3).EQ.1.AND.(NCYCLE.EQ.6.OR.NCYCLE.EQ.9)).AND.IRAND
       1 .EQ.1) GOTO 1550
C**** CALCULATE THIRD  EARLY FIGURE OF MERIT
        CALL EFOM(NDET,IREJ,VALUE)
        IF (IREJ.EQ.1) GOTO 5200
  1550 IF((SALF-SUMALF) /SALF.GT.0.02.AND.MARK.EQ.0) GO TO 1000
        IF (MARK.NE.0) GOTO 1700
        DO 1600 LL=1,NUMB
  1600 MKANG(LL)=IABS(MKANG(LL))
  1700 MARK=MARK+1
C**** 2 CYCLES OF REFINING ALL PHASES AT END
        IF(MARK.LE.2) GO TO 1000
C**** CALCULATE FINAL F.O.M.S AND OUTPUT RESULTS
    980  RESID = 0.0
        SUMEO = 0.0
        SUMALF = 0.0
        NUNDET = NDET-NUMB
        ALFEST=0.0
        DO 4920 LL=1,NUMB
          IF (WT(LL) .EQ. 0.0) THEN
          NUNDET = NUNDET + 1
          GOTO 4920
          ENDIFIF
        ALPHA(LL) = SQRT(ALPHA(LL))
C         write(*,*)ALPHA(LL)
        PALF(LL) = SQRT(PALF(LL))
        ALFEST = ALFEST + PALF(LL)
        SUMALF = SUMALF + ALPHA(LL)
C         write(*,*)SUMALF,ALPHA(LL)
C        IF (IPHAZ(LL) .LE. 0) IPHAZ(LL) = IPHAZ(LL) + 360
        SUMEO = SUMEO + PALF(LL)
```

```
 4920 CONTINUE
C**** CALCULATE ABSOLUTE FIGURE OF MERIT
C       write(*,*)SUMALF,ALFRAN,ALFEST
        ABSFOM = (SUMALF - ALFRAN) / (ALFEST - ALFRAN)
C**** CALCULATE FINAL PSI ZERO FIGURE OF MERIT
        CALL EFOM(NDET,I,PSIZRO)
        PSIZRO = PSIZRO / AMIN1(1.3,ABSFOM)
C**** CALCULATE A SCALED RESIDUAL
        SC = 1.0
        IF (IHVY.GT.0) SC = AMIN1(1.3,SQRT(AMAX1(ABSFOM,1.0)))
        DO 4930 LL=1,NUMB
          IF (WT(LL).NE.0.0) THEN
          RESID=RESID+ABS(SC*PALF(LL)-ALPHA(LL))
          ENDIFIF
 4930 CONTINUE
          IF (SUMEO.LE.0.1) THEN
          RESID = 100.0
          ELSE
          RESID = 100.0 * RESID / SUMEO
          ENDIFIF
        CALL NQEST
C++++
C++++ CALCULATE  THE  NEW FIGURE OF MERIT AS A FINAL F.O.M.
C++++
        CALL MYFOM(NUMSET,RFT,RMS,GLIK,STABLE,ITLE)
C++++
C++++ WRITE OUT THE NEW FIGURE FIGURE OF MERIT IN PLACE OF NQEST AND
C++++ PUT LOGLIK AT THE END
C++++
 5555  WRITE(NOUT,4950)NUMSET,ABSFOM,PSIZRO,RESID,RN4,GLIK,NUNDET,NCYCLE
 4950 FORMAT(1H+,I4,F8.4,F7.3,2F8.2,F9.2,I4,'(',I2,')')

        IF(LEVEL.GT.0) WRITE(NSPEC,7002) NUMSET,ABSFOM,PSIZRO,RESID,
       1 RN4,GLIK,NUNDET,NCYCLE
          CALL FLUSH(NSPEC)
 7002 FORMAT(1H ,I4,F7.2,F8.3,F8.2,F7.2,F8.2,I11,I12)
C**** PACK WEIGHTS AND PHASES IN TO ONE WORD
        DO 4935 LL=1,NUMB
        IF(WT(LL).LT.WMIN) IPHAZ(LL)=360
 4935 IPHAZ(LL)=IPHAZ(LL)*LK10+INT(WT(LL)*100.0)
        WRITE (NTAPED) NUMSET,ABSFOM,PSIZRO,RESID,RN4,GLIK,(IPHAZ(I),
       1 I=1,NUMB)
C**** TEST FOR A VERY PROMISING SOLUTION
        IF(ISTP.EQ.1) RETURN
        IF (PSIZRO.LT.AMN(2)) AMN(2) = PSIZRO + 0.05
        IF (RESID.LT.AMN(3)) AMN(3) = RESID + 0.5
        IF (PSIZRO.GT.1.3.OR.PSIZRO.GT.AMN(2)) RETURN
        IF (RESID.GT.20.0.OR.RESID.GT.AMN(3)) RETURN
        IF (CFOM.GT.-.15.AND.NQTOT.GT.0) RETURN
C**** LOOKS LIKE A GOOD MINIMUM IN PHASE SPACE
        IREJ = -1
        RETURN
C**** OUTPUT RELEVANT REJECTION MESSAGE
 5000 WRITE (NOUT,5100) ICC,NUMSET,VALUE
 5100 FORMAT (A1 ,I4,10X,' Rejected because 1st. EFOM=',F6.3)
        IF(LEVEL.GT.0) WRITE(NSPEC,7050)       NUMSET,VALUE
          CALL FLUSH(NSPEC)
 7050 FORMAT(1H ,I4,10X,' Rejected because 1st. EFOM=',F6.3)
        RETURN
 5200 WRITE (NOUT,5300) ICC,NUMSET,VALUE
 5300 FORMAT (A1 ,I4,10X,' Rejected because 3rd. EFOM=',F6.3)
        IF(LEVEL.GT.0) WRITE(NSPEC,7051)       NUMSET,VALUE
          CALL FLUSH(NSPEC)
 7051 FORMAT(1H ,I4,10X,' Rejected because 3rd. EFOM=',F6.3)
```

```
      RETURN
 5400 WRITE(NOUT,5500) ICC,NUMSET,CFOM
 5500 FORMAT(A1 ,I4,10X,' Rejected because 2nd. EFOM=',F6.3)
      IF(LEVEL.GT.0) WRITE(NSPEC,5600)       NUMSET,CFOM
        CALL FLUSH(NSPEC)
 5600 FORMAT(1H ,I4,10X,' Rejected because 2nd. EFOM=',F6.3)
      RETURN
      END

C-------------
```

SWTR

```
      SUBROUTINE SWTR(IREJ,SIGD)
C**** STATISTICALLY WEIGHTED TANGENT FORMULA
      DIMENSION CTABLE(360)
      COMMON /ANNEAL /ANEA
      COMMON /IO /NSPEC,NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
     1 NTAPEF,NTAPEG,IH(200),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,IVERB
      COMMON /BLK1 /IND2(80000),IND3(80000),IND4(80000),IND5(80000),
     1 EEE(80000),
     1 SUMNUM(800),SUMDEN(800),ALPHA(800),WT(800),IPHAZ(800),
     2 IORDE(1300),MKG(800),PALF(800),IZ(800),MKANG(800),EALF(800),
     3 LIM(1301),MAXHKL(3),SUMN(800),SUMD(800),PAD1(346)
      COMMON /BLK2 /IS(2,3,24),TS(3,24),P(6),CX(9),NSYM,ICENT,
     1 NEX,LATT,LAT,PTS,KSYS,IAPX,NGP,NTOT,NASU,NRD,AMX(5),
     2 AMN(5),LINE(17),LINEX(17),WTFOM(5),IMK,ITAN,IPUB,ISKIP,NLIM,WLIM
     3 ,PAD2(1302)
      COMMON /BLK3 /STABLE(450),NQ1(500),NQ2(500),AQ(500),NQTOT,NQ4,NQ5,
     1 FOM(400),I4,I5,IQTOT,CFOM,RN4,RN5,ISTP,CVR(10),VVR(10),NA(10),
     2 NW(10),NO(10),NK,SIGMA,PAD3(170)
      COMMON /BLK4 /TALF(800),RALF(800),ISPZRO(500),PAD4(448)
      COMMON /BLK5 /NUMB,NUMSET,NRAL,NANT(4),
     1 ALFRAN,NSX,IEF3,IEF4,IEF5,NDET,NAT,IFOM(3),IEFOM,IZRO,IXRAN,
     2 IYRAN,IRAND,WMIN,IWMIN,NSREQ,CUTT(3),ICC,IMP,PAD5(5)
      LOGICAL ANEA
      EQUIVALENCE (STABLE(91),CTABLE(1))
      EQUIVALENCE (LK16,LK(16)),(LK10,LK(10)),(LK5,LK(5)),(LK2,LK(2)),
     1 (LK17,LK(17)),(LK6,LK(6))
C**** 11 /10 (BESSEL FUNCTION)
      VEC(U)=U*(U+0.4807) /((U+0.8636)*U+1.3943)
      RN4=0.0
      RN5=0.0
      CFOM=0.0
      SCALE=SIGMA /0.09
      NCYCLE=0
      MARK = 0
      NNN = MIN0(100, NDET)
      IF(IRAND.EQ.1) NNN=NDET
      IF (KARLE.EQ.1) NNN=NDET
      CUT=12.5
      IF(KARLE.EQ.1) CUT=2.0

C     CODE TO INITIALISE THE BOLTZ VALUE
      IF(ANEA)THEN
        BOLTZ=0.9

C     CODE TO CALCULATE THE VALUE OF SHELDRICKS BETA
        BETAS=-ALOG(BOLTZ) /SQRT((ALFRAN**2) /NNN)
C     CODE TO SET THE MAX NUMBER OF CYCLES
        MAXCY=80
C     CODE TO INITIALISE THE TOTAL SHIFT AND NUMBER TIMES SHIFT CAL
        TSHIF=0.0
        NSHIF=0.0
C     END OF ADDITON
        NDAMP=4
C     CODE TO INITIALISE THE NUMBER OF FLIPPED PHASES
        FLIPHAZ=0.0
       ENDIFIF

       NCYC=-2*KARLE
C**** DO NOT REFINE STARTING PHASES IF HEAVY ATOM(S) PRESENT
      IF(IHVY.GT.0) NCYC=0
      SALF = 0.0
```

```
      NNIN=0


C**** INITIALISE ARRAYS AT START OF EACH CYCLE
 1000 DO 1010 I=1,NUMB
      FLIPHAZ=0.0
      SUMNUM(I)=0.0
      SUMDEN(I)=0.0
      SUMD(I)=0.0
      SUMN(I)=0.0
      EALF(I)=0.0
      TALF(I)=0.0
      RALF(I)=0.0
      IF(NCYC.LT.0) MKANG(I)=IABS(MKANG(I))
      IF(ABS(WT(I)).LT.WLIM) IPHAZ(I)=-1
 1010 CONTINUE
      SUMALF=SALF
      SALF=0.0
      NCYCLE=NCYCLE+1

C       CODE TO CALCULATE THE BETA FOR NON /CENTRO STRUCTURES
        IF (ICENT .EQ.1 .AND. ANEA)THEN
          BETAS=BETAS+0.03
        ELSE
          IF(NCYCLE .LE.5 .AND. ANEA) BETAS=BETAS+0.001
          IF(NCYCLE.GT.5.AND.NCYCLE.LE.15 .AND. ANEA)BETAS=BETAS+0.03
          IF(NCYCLE.GT.15 .AND. NCYCLE .LE. 40 .AND. ANEA)BETAS =BETAS+0.4
          IF(NCYCLE.GT.40.AND.NCYCLE.LE.60 .AND. ANEA)BETAS=BETAS+1.0
          IF(NCYCLE .GT.60.AND.NCYCLE.LE.80 .AND. ANEA) BETAS=BETAS+1.5
        ENDIFIF
      NEXT=NNIN
      NNIN=0
C**** CALCULATE TOP & BOTTOM OF TANGENT FORMULA FOR EACH REFLEXION
      DO 1250 I=1,NNN
      LL=IORDE(I)
      LI=LIM(LL)+1
      LS=LIM(LL+1)
      IF(LI.GT.LS) GO TO 1250
      DO 1100 JJ=LI,LS
      IMA=0
      IFR=0
      IFM=0
      WTIMA=1.0
      KN=7
      IF(IPHAZ(LL).GE.0) KN=KN-1
      IL=IND3(JJ)
      ILA=IABS(IL)
      IF(IPHAZ(ILA).GE.0) KN=KN-2
      IF(KN.EQ.7) GO TO 1100
      IRK=IND2(JJ)
      IRA=IABS(IRK)
C***  AT LEAST A TRIPLET
 1047 IF(IPHAZ(IRA).GE.0) KN=KN-3
      IF(KN.GT.4) GO TO 1100
      WTIRA=ABS(WT(IRA))
      IF(IEF3.EQ.0) GOTO 1041
      IF(IND4(JJ).EQ.0) GOTO 1041
C***  QUARTET
      IM=IND4(JJ)
      IMA=IABS(IM)
      KN=KN+4
      IF(IPHAZ(IMA).GE.0) KN=KN-4
      IF(KN.GT.6) GOTO 1100
      IFM=ISIGN(1,IPHAZ(IMA))
```

```
      WTIMA=ABS(WT(IMA))
 1041 IP=IND5(JJ)
      VECE=VEC(EEE(JJ))
      IF(IEF3.EQ.0.OR.IRA.EQ.0) GOTO 1043
      IF(IMA.EQ.0.OR.KN.NE.1) GOTO 1043
 1043 GOTO(1040,1040,1060,1050,1070),KN
C**** PHASE OF LL TO BE DETERMINED
 1040 EE=EEE(JJ)*ABS(WT(ILA))*WTIRA**WTIMA
         IF(IMA.LE.0) THEN
         VECX=EE*VECE
         EALF(LL)=EALF(LL)+VECX
         TALF(LL)=TALF(LL)+EE*EE-VECX*VECX
         RALF(LL)=RALF(LL)+EE*EE
         ENDIFIF
 1042 IFL=ISIGN(IPHAZ(ILA),IL)
      IFR=ISIGN(IPHAZ(IRA),IRK)
      IARG=MOD(IFL+IFR+IFM+IP+2520,360)+1
      SUMNUM(LL)=SUMNUM(LL)+EE*STABLE(IARG)
      SUMDEN(LL)=SUMDEN(LL)+EE*CTABLE(IARG)
         IF(IMA.GT.0)THEN
         SUMN(LL)=SUMN(LL)+EE*STABLE(IARG)
         SUMD(LL)=SUMD(LL)+EE*CTABLE(IARG)
         ENDIFIF
      IF(KN.NE.1) GO TO 1100
C**** PHASE OF IL TO BE DETERMINED
 1060 EE=EEE(JJ)*ABS(WT(IRA)*WT(LL))*WTIMA
         IF(IMA.LE.0) THEN
         VECX=EE*VECE
         EALF(ILA)=EALF(ILA)+VECX
         TALF(ILA)=TALF(ILA)+EE*EE-VECX*VECX
         RALF(ILA)=RALF(ILA)+EE*EE
         ENDIFIF
 1061 IF(KN.NE.1.AND.IRA.GT.0) IFR=ISIGN(IPHAZ(IRA),IRK)
      IARG=MOD(2520-(IFR+IFM-IPHAZ(LL)+IP)*ISIGN(1,IL),360)+1
      SUMNUM(ILA)=SUMNUM(ILA)+EE*STABLE(IARG)
      SUMDEN(ILA)=SUMDEN(ILA)+EE*CTABLE(IARG)
         IF(IMA.GT.0) THEN
         SUMN(ILA)=SUMN(ILA)+EE*STABLE(IARG)
         SUMD(ILA)=SUMD(ILA)+EE*CTABLE(IARG)
         ENDIFIF
      IF(KN.NE.1) GOTO 1100
      IF(IRA.EQ.0) GOTO 1100
C**** PHASE OF IRK TO BE DETERMINED
 1050 EE=EEE(JJ)*ABS(WT(ILA)*WT(LL))*WTIMA
         IF(IMA.LE.0) THEN
         VECX=EE*VECE
         EALF(IRA)=EALF(IRA)+VECX
         TALF(IRA)=TALF(IRA)+EE*EE-VECX*VECX
         RALF(IRA)=RALF(IRA)+EE*EE
         ENDIFIF
 1051 IF(KN.NE.1) IFL=ISIGN(IPHAZ(ILA),IL)
      IARG=MOD(2520-(IFL+IFM-IPHAZ(LL)+IP)*ISIGN(1,IRK),360)+1
      SUMNUM(IRA)=SUMNUM(IRA)+EE*STABLE(IARG)
      SUMDEN(IRA)=SUMDEN(IRA)+EE*CTABLE(IARG)
         IF(IMA.GT.0) THEN
         SUMN(IRA)=SUMN(IRA)+EE*STABLE(IARG)
         SUMD(IRA)=SUMD(IRA)+EE*CTABLE(IARG)
         ENDIFIF
      IF(KN.NE.1) GOTO 1100
      IF(IMA.EQ.0) GOTO 1100
C**** QUARTET
 1070 EE=EEE(JJ)*ABS(WT(LL)*WT(ILA))*WTIRA
      IARG=MOD(2520-(IFL+IFR-IPHAZ(LL)+IP)*ISIGN(1,IM),360)+1
      SUMNUM(IMA)=SUMNUM(IMA)+EE*STABLE(IARG)
```

```
      SUMDEN(IMA)=SUMDEN(IMA)+EE*CTABLE(IARG)
      SUMN(IMA)=SUMN(IMA)+EE*STABLE(IARG)
      SUMD(IMA)=SUMD(IMA)+EE*CTABLE(IARG)
      IF(KN.NE.1) GOTO 1100
 1100 CONTINUE
 1250 CONTINUE
C**** UPDATE PHASES AT END OF CYCLE
      DO 1500 I=1,NNN
      LL=IORDE(I)
      IF(NCYC.EQ.(-2).AND.IPHAZ(LL).LT.0) GO TO 1500
      IF (MKANG(LL).LE.0) GOTO 1490
      ID=15*MKANG(LL)-14
        IF(ID.NE.1) THEN
        T2=SUMNUM(LL)*STABLE(ID)+SUMDEN(LL)*CTABLE(ID)
        SUMNUM(LL)=T2*STABLE(ID)
        SUMDEN(LL)=T2*CTABLE(ID)
        ENDIFIF
 1300 ALFA=(SUMNUM(LL)-SUMN(LL))**2+(SUMDEN(LL)-SUMD(LL))**2
      IF(ALFA.EQ.0.0) GOTO 1500
      WATE=1.0
      IF(ALFA.LT.25.0) WATE=0.2*SQRT(ALFA)
      PALF(LL)=TALF(LL)+EALF(LL)*EALF(LL)
      WX1= ALFA / PALF(LL)
      IF (WX1.GT.1.03) WX =1.03-(WX1-1.03)*SCALE
      IF(WX1.GT.1.03)WX=WX /((1.2178*WX-1.0698)*WX+0.858)
      IF(WX.LT.WATE.AND.WX1.GT.1.03) WATE=WX
      IF(WT(LL).LT.0.0.AND.ALFA.LT.ALPHA(LL)) GO TO 1500
      SALF = SALF + WATE
      WT(LL)=AMAX1(WATE,0.15)
      PHAZ=RTOD*ATAN2(SUMNUM(LL),SUMDEN(LL))
      IF(PHAZ.LT.0.5) PHAZ=PHAZ+360.0
      IPHAZ(LL)=PHAZ+0.5
C     CODE TO MISS OUT THE FOLLOWING TANGENT REFINEMENT
        IF(NCYCLE.GE.1.AND.NCYCLE.LE.MAXCY .AND. ANEA)GOTO  3333
        IF(NCYCLE.GT.MAXCY.AND.NCYCLE.LE.MAXCY+5.AND.ANEA)GOTO  1320

        IF(ANEA)THEN
         GOTO 3333
        ELSE
         GOTO 1320
        ENDIFIF

C     TO DETERMINE WHETHER USE CENTRO /NONCENTRO CODE
 3333       IF(ABS(MKANG(LL)).EQ.1)GOTO 4444
C     SIMULATED ANNEALING CODE FOR CENTROSYMMETRIC CASE

C     CODE TO CALCULATE SHELDRICKS PROBABILITY
            PR1=ABS(SUMNUM(LL)*SIGD) /2
            PROBPL=0.5*TANH(PR1)+0.5
        IF (PROBPL .LT. 0.5) PROBPL=1-PROBPL
C     CODE TO CALCLATE (1-PROB) AND THUS RATIO
        PROBMI=(1-PROBPL)
C     CODE TO CALCULATE BOLTZMANN RATIO
        RATIO=0.5*((PROBMI /PROBPL) /EXP(BETAS))
C     CODE TO CALCULATE RANDOM NUMBERS BETWEEN O AND 1
        RNUMB=RAND(IXRAN,IYRAN)
C     CODE TO DETERMINE WHETHER PHASES SHOULD BE FLIPPED
        IF((IPHAZ(LL).EQ.180) .AND. (RATIO .GT. RNUMB))THEN
C      TO KEEP A COUNT OF THE FLIPPED PHASES
        FLIPHAZ=FLIPHAZ+1

       IPHAZ(LL)=0
       SUMDEN(LL)= -SUMDEN(LL)
        ELSE
```

```
      IF((IPHAZ(LL) .EQ. 0) .AND.(RATIO.GT.RNUMB))THEN
      FLIPHAZ=FLIPHAZ+1
      IPHAZ(LL)=180
      SUMDEN(LL)= -SUMDEN(LL)
        ELSE
      IF((IPHAZ(LL).EQ.360) .AND.(RATIO .GT.RNUMB))THEN
      FLIPHAZ=FLIPHAZ+1
      IPHAZ(LL)=180
      SUMDEN(LL)= -SUMDEN(LL)
        ELSE
     IF(IPHAZ(LL).EQ.360) THEN
      IPHAZ(LL)=0
        ENDIFIF
        ENDIFIF
        ENDIFIF
        ENDIFIF
        GOTO 1320
C     CODE TO CALCULATE RANDOM NUMBER BETWEEN 0 AND 1
 4444   RANON=RAND(IXRAN,IYRAN)

C     SIMULATED ANNEALING CODE FOR NONCENTROSYMMETRIC STRUCTURES
C     CODE TO DETERMINE ALFA
       ALFA=SQRT((SUMNUM(LL)-SUMN(LL))**2+(SUMDEN(LL)-SUMD(LL))**2)
C     CODE TO DETERMINE SHIFT
       UPPER=(NDAMP*BETAS*ALFA)+(ALOG(RANON))
       DENOM =(NDAMP*BETAS*ALFA)-(ALOG(RANON))
       SHIFT=UPPER /DENOM
C     CODE TO CONVERT SHIFT INTO DEGREES
       SHIFT=RTOD*ACOS(SHIFT)
C     CODE TO DETERMINE THE SIGN OF THE SHIFT
       RSIGN=RAND(IXRAN,IYRAN)
       IF(RSIGN.LT.0.5)SHIFT=-SHIFT
C     CODE TO CONVERT SHIFT INTO INTEGERS
       SHIFT=AINT(SHIFT)
       IPHAZ(LL)=IPHAZ(LL)+SHIFT

C     CODE TO CALCULATE TOTAL SHIFT
         TSHIF=TSHIF+ABS(SHIFT)
C     CODE TO CALCULATE NUMBER OF SHIFTS
         NSHIF=NSHIF+1
 1320  ALPHA(LL)=ALFA
       IF(KARLE.EQ.1.AND.WT(LL).GT.0.9)  MKANG(LL)=-IABS(MKANG(LL))
 1490 NNIN=NNIN+1
 1500 CONTINUE
C     CALCULATE THE AVERAGE SHIFT
         MSHIF=TSHIF /NSHIF
C     CODE TO STOP THE REFINEMENT AND GET FOMS PRODUCED
      IF(NCYCLE .LT. MAXCY+5 .AND. ANEA) GOTO 1000
      IF(NCYCLE .EQ. MAXCY+5 .AND. ANEA)GOTO 1620

  981  IF(KARLE.EQ.0.OR.IRAND.EQ.1)GOTO 1540
C**** KARLE RECYCLING CONTROL STATEMENTS
      NCYC=NCYC+1
      IF(NCYC.LE.0) GO TO 1000
      IF(MARK.GT.0) GO TO 1520
      IF(NNIN.GT.NEXT.AND.NCYC.LT.4.AND.NNIN.LT.NUMB-5) GO TO 1000
      DO 1515 LL=1,NUMB
 1515 MKANG(LL)=IABS(MKANG(LL))
 1520 MARK=MARK+1
      IF(MARK.LE.2) GO TO 1000
      IF(IRAND.EQ.1) GOTO 1620
      WRITE(NOUT,1530) NNIN
 1530 FORMAT( / /26X,' Number of phases generated for Fourier =',I6)
      RESID=0.0
```

```
      ABSFOM=1.0
      PSIZRO=1.0
      RN4=-1.0
      RN5=-1.0
      NUMSET=1
      ISTP=1
      GO TO 1890
C**** SWTR CONTROL STATEMENTS
 1540 CUT=AMAX1(0.5*CUT,0.05)

      IF(CUT.GT.0.1) GOTO 1000
      IF(NNN.EQ.NDET) GO TO 1550
      NNN=MIN0(NNN+NNN /2,NDET)
      GOTO 1000
 1550 IF((SALF-SUMALF) /SALF.GT.0.02.AND.MARK.EQ.0) GO TO 1000
      IF (MARK.EQ.3) GOTO 1620
      DO 1600 LL=1,NUMB
 1600 MKANG(LL)=IABS(MKANG(LL))
      MARK = MARK+1
      GOTO 1000
C**** CALCULATE AND OUTPUT FINAL FIGURES OF MERIT
 1620 ALFEST=0.0
      ALFRAN=0.0
      SALF=0.0
      SUMEO=0.0
      RESID=0.0
      NUNDET=NDET-NUMB
      DO 1700 I=1,NUMB
      IF (ABS(WT(I)).LT.0.05.OR.(IRAND.EQ.1.AND.INT(100.0*WT(I)).EQ.
     1 (-WMIN)))NUNDET=NUNDET+1
      IF (ABS(WT(I)).LT.0.05) GOTO 1700
      ALPHA(I)=SQRT(ALPHA(I))
      PALF(I)=SQRT(PALF(I))
      ALFEST=ALFEST+PALF(I)
      ALFRAN=ALFRAN+SQRT(RALF(I))
      SALF=SALF+ALPHA(I)
      SUMEO=SUMEO+PALF(I)
 1700 CONTINUE
C**** CALCULATE ABSOLUTE FIGURE OF MERIT
      ABSFOM = (SALF-ALFRAN) /(ALFEST-ALFRAN)
C**** CALCULATE FINAL PSIZERO FIGURE OF MERIT
      CALL EFOM(NDET,I,PSIZRO)
      PSIZRO = PSIZRO  / AMIN1(1.3,ABSFOM)
C**** CALCULATE A SCALED RESIDUAL
      SC = 1.0
      IF (IHVY.GT.0) SC = AMIN1(1.3,SQRT(AMAX1(ABSFOM,1.0)))
      DO 1720 LL=1,NUMB
      IF (WT(LL).EQ.0.0) GOTO 1720
      RESID=RESID+ABS(SC*PALF(LL)-ALPHA(LL))
 1720 CONTINUE
      IF (SUMEO.LE.0.1) RESID = 100.0
      IF (SUMEO.GT.0.1) RESID = 100.0 * RESID  / SUMEO
      CALL NQEST
C++++
C++++ CALCULATE  THE  NEW FIGURE OF MERIT AS A FINAL F.O.M.
C++++
      CALL MYFOM(NUMSET,RFT,RMS,GLIK,STABLE,ITLE)

C++++
C++++ WRITE OUT THE NEW FIGURE FIGURE OF MERIT IN PLACE OF NQEST AND
C++++ PUT LOGLIK AT THE END
C++++

 5555      WRITE(NOUT,1880)NUMSET,ABSFOM,PSIZRO,RESID,RN4,GLIK,NUNDET,NCYCLE
```

```
      SUBROUTINE PHASE(MARK,ISETX,XP,DX)                              PHASE
C**** GENERATE STARTING SETS OF PHASES FROM CONVERGENCE RESULTS

        INTEGER    GOL GEN, GOL COD
        DIMENSION IZ2(800),ALPHA2(800)

C       COMMON BLOCK FOR THE GENERATION AND USE OF GOLAY CODE
        COMMON   / GOLAY C /        GOL GEN ( 24, 0:11 ) ,
     .                              GOL COD ( 24, 4096 ), CODE

C       COMMON BLOCK FOR THE GENERATION AND USE OF HADAMARD CODE
        COMMON   / HADAM R /          H 2 ( 2, 2) , H 4 ( 4, 4) , H 8 ( 8, 8),
     .                              H12 (12,12) , H16 (16,16) , H32 (32,32),
     .                              H64 (64,64),H128 (128,128)
       INTEGER     H2, H4, H8, H12, H16, H32, H64,H128
           COMMON /IO /NSPEC,NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
      1 NTAPEF,NTAPEG,IH(200),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,IVERB
       COMMON /BLK1 /IND2(80000),IND3(80000),IND4(80000),IND5(80000),
      1 EEE(80000),
      1 SUMNUM(800),SUMDEN(800),ALPHA(800),WT(800),IPHAZ(800),IORDE(1300)
      2,MKG(800),PALF(800),IZ(800),MKANG(800),EALF(800),LIM(1301),
      3 MAXHKL(3),SUMN(800),SUMD(800),E(800)
       COMMON /BLK2 /IS(2,3,24),TS(3,24),P(6),CX(9),NSYM,ICENT,
      1 NEX,LATT,LAT,PTS,KSYS,IAPX,NGP,NTOT,NASU,NRD,AMX(5),
      2 AMN(5),LINE(17),LINEX(17),WTFOM(5),IMK,ITAN,IPUB,ISKIP,NLIM,WLIM
      3 ,PAD2(1302)
       COMMON /BLK3 /STABLE(450),NQ1(500),NQ2(500),AQ(500),NQTOT,NQ4,NQ5,
      1 FOM(400),I4,I5,IQTOT,CFOM,RN4,RN5,ISTP,CVR(10),VVR(10),NA(10),
      3 NW(10),NO(10),NK,SIGMA,PAD3(170)
       COMMON /BLK5 /NUMB,NUMSET,NRAL,NANT(4),
      1 ALFRAN,NSX,IEF3,IEF4,IEF5,NDET,NAT,IFOM(3),IEFOM,IZRO,IXRAN,
      2 IYRAN,IRAND,WMIN,IWMIN,NSREQ,CUTT(3),ICC,IMP,PAD5(5)
        LOGICAL  CODE
       NAN = IABS(NANT(1))
 1000 MARK = 0


       IF (CODE)THEN
        CALL GOLAY
       ENDIFIF
       IF(IRAND.EQ.0) GOTO 1300
C**** GENERATE NEXT SET OF RANDOM PHASES FOR RANTAN
       IXS=IXRAN
       IYS=IYRAN
       IF(NUMSET.GT.NSREQ) RETURN
       NNN=NDET
       MS=0
       MARK=1
       IF (MS.EQ.1) GO TO 2040
 2010 DO 1035 I=1,NUMB
       IF (MKG(I)) 2020,2020,1030
 2020 IF (IABS(MKANG(I)).EQ.1) GO TO 1025
       MPH=15*IABS(MKANG(I)-1)
       IRA=360*RAND(IXRAN,IYRAN)+0.5
       IF (IRA.GT.180) MPH=MPH+180
       IZ(I)=MPH*1000+IWMIN
       IF (I.EQ.NAN) IZ(I)=MPH*1000+99
       GO TO 1035
 1025 MPH=360*RAND(IXRAN,IYRAN)+0.5
       IZ(I)=MPH*1000+IWMIN
       IF (I.EQ.NAN) IZ(I)=MPH*1000+85
```

```
      GO TO 1035
1030 IZ(I)=MKG(I)
1035 CONTINUE
      WRITE(NOUT,1321) IXS,IYS
1321 FORMAT(57X,' Random numbers IX,IY are: ',2I12)
      MS=1
      GO TO 1070
2040 J=2+ICENT
      DO 1045 I=J,4
      IF (NANT(I)) 1045,2060,1050
1045 CONTINUE
      GO TO 2060
1050 DO 1055 K=J,I
      NANT(K)=-NANT(K)
      NN=IABS(NANT(K))
      IZ(NN)=-(IZ(NN)-200)+360200
1055 CONTINUE
      GO TO 1070
2060 DO 1065 I=J,4
      NANT(I)=IABS(NANT(I))
1065 CONTINUE
      MS=0
      GO TO 2010
1070 DO 2300 LL=1,NUMB
      I=IORDE(LL)
      IF (LL.GT.NNN) GO TO 1081
      IWT=IZ(I)-1000*(IZ(I) /1000)
      ALPHA(I)=0.0025*IWT*IWT
      IF (IWT-100) 1080,2200,2200
1080 WT(I)=-0.01*FLOAT(IWT)
      GO TO 2180
1081 WT(I)=0.0
      GO TO 2300
2200 WT(I)=1.0
      MKANG(I)=-IABS(MKANG(I))
2180 IPHAZ(I)=MOD(IABS(IZ(I) /1000)+360,360)
      IF (IPHAZ(I) .EQ. 0) IPHAZ(I) = 360
2300 CONTINUE
      RETURN


      IF(CODE)GOTO 1300

C**** GENERATE NEXT SET OF PHASES FOR REGULAR PHASE PERMUTATION
 1300 IF (NRAL .EQ. 0) GO TO 1010
      XP = XP - DX
      IF(CODE)GOTO 1006
      IF (XP+720.1) 1004,1004,1006
 1004 XP = XP + 360.0
      GO TO 1010
 1006 MARK = 1

 1010  NDF=0
       NNDF=0
       IF (CODE) THEN
        NCODE=10000
       ELSE
        NCODE=NUMB
       ENDIFIF
        ICODE=1

       DO 1200 I=1,NUMB
C       CODE TO CALCULATE RANDOM NUMBERS BETWEEN O AND 1
```

```
        ALPHA(I) = 0.0
        IF (IZ(I)) 1100,1020,1040
**** UNKNOWN PHASE
 1020 WT(I) = 0.0
C     TO MAKE SURE ONLY THE STARTING SET OF PHASES USES GOLAY CODE
        IF(CODE)GOTO 7777
          GO TO 1180
 7777   IF(CODE)GOTO 1200

C**** ORIGIN FIXING OR KNOWN PHASE
 1040 IWT=IZ(I)-1000*(IZ(I) /1000)
        ALPHA(I)=0.0025*FLOAT(IWT*IWT)
        IF(IWT-100) 1060,1170,1170

 1060 WT(I)=-0.01*FLOAT(IWT)
        IF(IZ(I) .GT. 0 .AND. CODE)GOTO 1180
C       TO MAKE SURE ONLY THE STARTING SET OF PHASES USES GOLAY CODE
         IF (CODE) GOTO 8888
        GO TO 1180
 8888 IF (CODE) GOTO 1200

C**** HAS STARTING SET PHASE BEEN INCREMENTED
 1100 IF (MARK) 1170,1120,1170
        IF (MARK) 1170,1120,1170
 1120 IF (IABS(MKANG(I)) .EQ. 1) GOTO 1170
        IZ(I) = IZ(I) - 180000
        IF(IZ(I)+720000) 1140,1140,1160
 1140 IZ(I)=IZ(I)+360000
        GO TO 1170
 1160 MARK = 1
 1170 WT(I) = 1.0
C         write(*,*) IZ(I),I
        MKANG(I) = -IABS(MKANG(I))
        IF (IZ(I) .GT. 0.AND. CODE) GO TO 1180

C       USING THE GOLAY CODE TO ASSIGN PHASES
C       TO CONTROL THE ROW OF GOLAY CODE
           IF(CODE) NOR=NUMSET
C       CENTROSYMMETRIC
           IF(ABS(MKANG(I)) .NE. 1 .AND. CODE)THEN
C        TO CONTROL THE COLUMN OF GOLAY CODE
            NDF=NDF+1
            IF (NDF.GT.24)GOTO 4444
            IF( GOL COD(NDF,NOR).EQ.0) IPHAZ(I)=360
            IF(GOL COD(NDF,NOR).EQ.1)IPHAZ(I)=180
          ENDIFIF

C       NON CENTROSYMMETRIC
           IF(ABS(MKANG(I)) .EQ. 1 .AND. CODE)THEN
             NDF=NDF+2
            IF (NDF.GT.24)GOTO 4444
            IF(GOL COD(NDF-1,NOR).EQ.0.AND.GOL COD(NDF,NOR).EQ.0)
     1          IPHAZ(I)=45
            IF(GOL COD(NDF-1,NOR).EQ.0.AND.GOL COD(NDF,NOR).EQ.1)
     1          IPHAZ(I)=135
            IF(GOL COD(NDF-1,NOR).EQ.1.AND.GOL COD(NDF,NOR).EQ.0)
     1          IPHAZ(I)=315
            IF(GOL COD(NDF-1,NOR).EQ.1.AND.GOL COD(NDF,NOR).EQ.1)
     1          IPHAZ(I)=225

          ENDIFIF

           IF(CODE)GOTO 1200
```

```
      IF (IABS(MKANG(I)) .NE. 1) GO TO 1180
      IF (IZ(I) .GT. 0) GO TO 1180
      IPHAZ(I) = AMOD((FLOAT(IZ(I))*XP), 360.0) + 0.5
     GO TO 1190
1180 IPHAZ(I) = MOD(IABS(IZ(I) /1000), 360)
1190 IF (IPHAZ(I) .EQ. 0) IPHAZ(I) = 360


1200 CONTINUE

4444   IF (ICENT.NE.0) GOTO 1220
       IF(MARK.EQ.0) GO TO 1220
       IF (CODE)GOTO 1220

     IF (NANT(1)) 1210,1220,1215
1210 IF (COS(DTOR * FLOAT(IPHAZ(NAN)))) 1000,1000,1220
1215 IF (SIN(DTOR * FLOAT(IPHAZ(NAN)))) 1000,1000,1220
1220 IF (MARK) 1230,1230,1270
1230 J = 2 + ICENT
     DO 1240 I=J,4
     IF (NANT(I)) 1240,1360,1250
1240 CONTINUE
      IF (CODE) GOTO 1250
     GO TO 1360
1250 DO 1260 K=J,I
     NANT(K) = -NANT(K)
     NN = IABS(NANT(K))
     IZ(NN) = -(IZ(NN) - 200) + 360200
      IF (CODE) GOTO 1260
     IPHAZ(NN) = MOD(IABS(IZ(NN) /1000), 360)
1260 CONTINUE

     MARK = 1

1270 IF (NUMSET .LE. ISKIP .OR. NUMSET .LT. ISETX) RETURN
C**** OUTPUT STARTING POINT
     J = 0

     DO 1340 I=1,NUMB
     IF (IZ(I) .EQ. 0) GOTO 1340
     J = J + 1
     LINEX(J) = IPHAZ(I)
     IF (J .LT. 17) GO TO 1340
     WRITE(NOUT,1320) LINEX
1320 FORMAT(57X,17I4)
     J = 0
1340 CONTINUE
800   continue
      IF(J .GT. 0) WRITE(NOUT,1320) (LINEX(K),K=1,J)
1360 RETURN
     END
```

```
      SUBROUTINE PHASE(MARK,ISETX,XP,DX)                              PHASE
C**** GENERATE STARTING SETS OF PHASES FROM CONVERGENCE RESULTS

        INTEGER    GOL GEN, GOL COD
C       COMMON BLOCK FOR THE GENERATION AND USE OF GOLAY CODE
       COMMON   / GOLAY C /       GOL GEN ( 24, 0:11 ) ,
     .                            GOL COD ( 24, 128 )
C       COMMON BLOCK FOR THE GENERATION AND USE OF GOLAY CODE
       COMMON   / HADAM R /       H 2 ( 2, 2) , H 4 ( 4, 4) , H 8 ( 8, 8),
     .                            H12 (12,12) , H16 (16,16) , H32 (32,32),
     .                            H64 (64,64),CODE
       INTEGER    H2, H4, H8, H12, H16, H32, H64
       COMMON /IO /NSPEC,NIN,NOUT,NTAPEA,NTAPEB,NTAPEC,NTAPED,NTAPEE,
      1 NTAPEF,NTAPEG,IH(200),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,IVERB
       COMMON /BLK1 /IND2(80000),IND3(80000),IND4(80000),IND5(80000),
      1 EEE(80000),
      1 SUMNUM(800),SUMDEN(800),ALPHA(800),WT(800),IPHAZ(800),IORDE(1300)
      2,MKG(800),PALF(800),IZ(800),MKANG(800),EALF(800),LIM(1301),
      3 MAXHKL(3),SUMN(800),SUMD(800),E(800)
       COMMON /BLK2 /IS(2,3,24),TS(3,24),P(6),CX(9),NSYM,ICENT,
      1 NEX,LATT,LAT,PTS,KSYS,IAPX,NGP,NTOT,NASU,NRD,AMX(5),
      2 AMN(5),LINE(17),LINEX(17),WTFOM(5),IMK,ITAN,IPUB,ISKIP,NLIM,WLIM
      3 ,PAD2(1302)
       COMMON /BLK3 /STABLE(450),NQ1(500),NQ2(500),AQ(500),NQTOT,NQ4,NQ5,
      1 FOM(400),I4,I5,IQTOT,CFOM,RN4,RN5,ISTP,CVR(10),VVR(10),NA(10),
      3 NW(10),NO(10),NK,SIGMA,PAD3(170)
       COMMON /BLK5 /NUMB,NUMSET,NRAL,NANT(4),
      1 ALFRAN,NSX,IEF3,IEF4,IEF5,NDET,NAT,IFOM(3),IEFOM,IZRO,IXRAN,
      2 IYRAN,IRAND,WMIN,IWMIN,NSREQ,CUTT(3),ICC,IMP,PAD5(5)
        LOGICAL  CODE
       NAN = IABS(NANT(1))
 1000  MARK = 0
       IF (CODE)THEN
        CALL HADAM
       ENDIFIF
       IF(IRAND.EQ.0) GOTO 1300
C**** GENERATE NEXT SET OF RANDOM PHASES FOR RANTAN
       IXS=IXRAN
       IYS=IYRAN
       IF(NUMSET.GT.NSREQ) RETURN
       NNN=NDET
       MS=0
       MARK=1
       IF (MS.EQ.1) GO TO 2040
 2010  DO 1035 I=1,NUMB
       IF (MKG(I)) 2020,2020,1030
 2020  IF (IABS(MKANG(I)).EQ.1) GO TO 1025
       MPH=15*IABS(MKANG(I)-1)
       IRA=360*RAND(IXRAN,IYRAN)+0.5
       IF (IRA.GT.180) MPH=MPH+180
       IZ(I)=MPH*1000+IWMIN
       IF (I.EQ.NAN) IZ(I)=MPH*1000+99
       GO TO 1035
 1025  MPH=360*RAND(IXRAN,IYRAN)+0.5
       IZ(I)=MPH*1000+IWMIN
       IF (I.EQ.NAN) IZ(I)=MPH*1000+85
       GO TO 1035
 1030  IZ(I)=MKG(I)
 1035  CONTINUE
       WRITE(NOUT,1321) IXS,IYS
 1321  FORMAT(57X,' Random numbers IX,IY are: ',2I12)
```

```
      MS=1
      GO TO 1070
2040  J=2+ICENT
      DO 1045 I=J,4
      IF (NANT(I)) 1045,2060,1050
1045  CONTINUE
      GO TO 2060
1050  DO 1055 K=J,I
      NANT(K)=-NANT(K)
      NN=IABS(NANT(K))
      IZ(NN)=-(IZ(NN)-200)+360200
1055  CONTINUE
      GO TO 1070
2060  DO 1065 I=J,4
      NANT(I)=IABS(NANT(I))
1065  CONTINUE
      MS=0
      GO TO 2010
1070  DO 2300 LL=1,NUMB
      I=IORDE(LL)
      IF (LL.GT.NNN) GO TO 1081
      IWT=IZ(I)-1000*(IZ(I) /1000)
      ALPHA(I)=0.0025*IWT*IWT
      IF (IWT-100) 1080,2200,2200
1080  WT(I)=-0.01*FLOAT(IWT)
      GO TO 2180
1081  WT(I)=0.0
      GO TO 2300
2200  WT(I)=1.0
      MKANG(I)=-IABS(MKANG(I))
2180  IPHAZ(I)=MOD(IABS(IZ(I) /1000)+360,360)
      IF (IPHAZ(I) .EQ. 0) IPHAZ(I) = 360
2300  CONTINUE
      RETURN

C**** GENERATE NEXT SET OF PHASES FOR REGULAR PHASE PERMUTATION

1300  IF (NRAL .EQ. 0) GO TO 1010
      XP = XP - DX
      IF(CODE)GOTO 1006
      IF (XP+720.1) 1004,1004,1006

1004  XP = XP + 360.0
      GO TO 1010
1006  MARK = 1

1010   NDF=0
       NNDF=0
       IF (CODE) THEN
        NCODE=10000
       ELSE
        NCODE=NUMB
       ENDIFIF
       ICODE=1

       DO 1200 I=1,NUMB
       ALPHA(I) = 0.0
       IF (IZ(I)) 1100,1020,1040
**** UNKNOWN PHASE
 1020 WT(I) = 0.0
C       TO MAKE SURE ONLY THE STARTING SET OF PHASES USES GOLAY CODE
        IF(CODE)GOTO 7777
         GO TO 1180
 7777   IF(CODE)GOTO 1200
```

```
C**** ORIGIN FIXING OR KNOWN PHASE
 1040 IWT=IZ(I)-1000*(IZ(I) /1000)
      ALPHA(I)=0.0025*FLOAT(IWT*IWT)
      IF(IWT-100) 1060,1170,1170

 1060 WT(I)=-0.01*FLOAT(IWT)
      IF(IZ(I) .GT. 0 .AND. CODE)GOTO 1180
C     TO MAKE SURE ONLY THE STARTING SET OF PHASES USES GOLAY CODE
      IF (CODE) GOTO 8888
      GO TO 1180
 8888 IF (CODE) GOTO 1200

C**** HAS STARTING SET PHASE BEEN INCREMENTED
 1100 IF (MARK) 1170,1120,1170
      IF (MARK) 1170,1120,1170
 1120 IF (IABS(MKANG(I)) .EQ. 1) GOTO 1170
      IZ(I) = IZ(I) - 180000
      IF(IZ(I)+720000) 1140,1140,1160
 1140 IZ(I)=IZ(I)+360000
      GO TO 1170
 1160 MARK = 1
 1170 WT(I) = 1.0
C       write(*,*) IZ(I),I
      MKANG(I) = -IABS(MKANG(I))
      IF (IZ(I) .GT. 0.AND. CODE) GO TO 1180

C     USING THE HADAMARD CODE TO ASSIGN PHASES
C     CENTROSYMMETRIC
        IF(ABS(MKANG(I)).NE.1.AND.CODE.AND.NUMSET.LE.64)THEN
C     TO CONTROL THE COLUMN OF HAD CODE
          NDF=NDF+1

C     TO CONTROL THE ROW OF THE HADAMARD CODE
          NOR=NUMSET
          IF( H64(NDF,NOR).EQ.-1) IPHAZ(I)=360
          IF(H64(NDF,NOR).EQ.1)IPHAZ(I)=180
        ENDIFIF

C     ASSIGN PHASES TO THE COMPLEMENT OF THE HADAMRD CODE
      IF(ABS(MKANG(I)).NE.1.AND.CODE.AND.NUMSET.GT.64.AND.
     1 NUMSET .LE. 128)THEN
C     TO CONTROL THE COLUMN OF THE HAD CODE
          NDF=NDF+1
C    TO CONTROL THE ROW OF THE HADAMARD CODE
          NOR=NUMSET-64
          IF(NDF .GT.128)GOTO 4444
          IF( H64(NDF,NOR).EQ.-1) IPHAZ(I)=180
          IF(H64(NDF,NOR).EQ.1)IPHAZ(I)=360
        ENDIFIF

C       CODE FOR ACENTRIC
        IF(ABS(MKANG(I)).EQ.1.AND.CODE.AND.NUMSET.LE.64)THEN
            NDF=NDF+2
            NOR=NUMSET
          IF(H64(NDF-1,NOR).EQ.-1.AND.H64(NDF,NOR).EQ.-1)
     1      IPHAZ(I)=45
          IF(H64(NDF-1,NOR).EQ.-1.AND.H64(NDF,NOR).EQ.1)
     1      IPHAZ(I)=135
          IF(H64(NDF-1,NOR).EQ.1.AND.H64(NDF,NOR).EQ.-1)
     1      IPHAZ(I)=315
          IF(H64(NDF-1,NOR).EQ.1.AND.H64(NDF,NOR).EQ.1)
     1      IPHAZ(I)=225

        ENDIFIF
```

```
C       ASSIGN PHASES TO THE COMPLEMENT FOR ACENTIC

          IF(ABS(MKANG(I)).EQ.1.AND.CODE.AND.NUMSET.GT.64
     1      .AND.NUMSET .LE.128)THEN
              NDF=NDF+2
              NOR=NUMSET-64
              IF(NDF .GT.128)GOTO 4444
            IF(H64(NDF-1,NOR).EQ.-1.AND.H64(NDF,NOR).EQ.-1)
     1        IPHAZ(I)=225
           IF(H64(NDF-1,NOR).EQ.-1.AND.H64(NDF,NOR).EQ.1)
     1          IPHAZ(I)=315
           IF(H64(NDF-1,NOR).EQ.1.AND.H64(NDF,NOR).EQ.-1)
     1          IPHAZ(I)=135
           IF(H64(NDF-1,NOR).EQ.1.AND.H64(NDF,NOR).EQ.1)
     1          IPHAZ(I)=45

         ENDIFIF
        IF(CODE)GOTO 1200

         IF (IABS(MKANG(I)) .NE. 1) GO TO 1180
         IF (IZ(I) .GT. 0) GO TO 1180
         IPHAZ(I) = AMOD((FLOAT(IZ(I))*XP), 360.0) + 0.5
        GO TO 1190
 1180 IPHAZ(I) = MOD(IABS(IZ(I) /1000), 360)
 1190 IF (IPHAZ(I) .EQ. 0) IPHAZ(I) = 360


 1200 CONTINUE

 4444    IF (ICENT.NE.0) GOTO 1220
C        write(*,*)IPHAZ
         IF(MARK.EQ.0) GO TO 1220
         IF (CODE)GOTO 1220

       IF (NANT(1)) 1210,1220,1215
 1210 IF (COS(DTOR * FLOAT(IPHAZ(NAN)))) 1000,1000,1220
 1215 IF (SIN(DTOR * FLOAT(IPHAZ(NAN)))) 1000,1000,1220
 1220 IF (MARK) 1230,1230,1270
 1230 J = 2 + ICENT
       DO 1240 I=J,4
       IF (NANT(I)) 1240,1360,1250
 1240 CONTINUE
        IF (CODE) GOTO 1250
       GO TO 1360
 1250 DO 1260 K=J,I
       NANT(K) = -NANT(K)
       NN = IABS(NANT(K))
       IZ(NN) = -(IZ(NN) - 200) + .360200
        IF (CODE) GOTO 1260
       IPHAZ(NN) = MOD(IABS(IZ(NN) /1000), 360)
 1260 CONTINUE
       MARK = 1
 1270 IF (NUMSET .LE. ISKIP .OR. NUMSET .LT. ISETX) RETURN
         IF (NUMSET .LE. ISKIP .OR. NUMSET .LT. ISETX) RETURN
C**** OUTPUT STARTING POINT
       J = 0

       DO 1340 I=1,NUMB
       IF (IZ(I) .EQ. 0) GOTO 1340
       J = J + 1
       LINEX(J) = IPHAZ(I)
       IF (J .LT. 17) GO TO 1340
       WRITE(NOUT,1320) LINEX
 1320 FORMAT(57X,17I4)
```

```
      J = 0
1340  CONTINUE
      IF(J .GT. 0) WRITE(NOUT,1320) (LINEX(K),K=1,J)
1360  RETURN
      END
```