

# Computer Program for the Least-Squares Determination of the Phases of the Crystal Structure Factors

J. HARRISON HANCOCK, JANET E. FISHER,  
AND HERBERT A. HAUPTMAN

*The Mathematics Staff  
Optical Sciences Division*

December 16, 1970



**NAVAL RESEARCH LABORATORY**  
**Washington, D.C.**

## CONTENTS

Abstract	ii
Problem Status	ii
Authorization	ii
1. INTRODUCTION	1
2. THE BASIC SET OF PHASES	2
3. THE STRUCTURE INVARIANTS	3
4. APPROXIMATE EVALUATION OF INITIAL PHASES BY MEANS OF LEAST SQUARES	4
5. THE COMPUTER PROGRAMS	5
6. CAPACITY AND ARRAY DIMENSIONS	9
7. PROGRAM <u>BUILDUP</u> INPUT AND OUTPUT	11
Data Input to Program <u>Buildup</u>	11
Output from Program <u>Buildup</u>	12
8. PROGRAM <u>LS PHASES</u> INPUT AND OUTPUT	13
Data Input to Program <u>LS Phases</u>	13
Output from Program <u>LS Phases</u>	14
REFERENCES	15
Appendix A—Listing of Program <u>Buildup</u>	17
Appendix B—Hypothetical Example of a <u>Buildup</u> Run	25
Appendix C—Listing of Program <u>LS Phases</u>	29
Appendix D—Hypothetical Example of an <u>LS Phases</u> Run	39

## ABSTRACT

An effective technique, employing the Principle of Least Squares, which leads from the values of the crystal structure invariants to the values of individual phases, has recently been obtained. In the present report the details of the computer program which implements this technique are described. An auxiliary program to identify the required structure invariants has also been written and is contained herein.

## PROBLEM STATUS

This is the final report on this problem. The problem was terminated as of June 30, 1970.

## AUTHORIZATION

NRL Problem N01-19  
Project RR 002-07-41-5065

Manuscript submitted July 27, 1970.

COMPUTER PROGRAM FOR THE LEAST-SQUARES DETERMINATION  
OF THE PHASES OF THE CRYSTAL STRUCTURE FACTORS

1. INTRODUCTION

It is assumed that the unit cell of a crystal consists of  $N$  identical atoms. Denote by  $\vec{r}_j$  the position vector of the atom labeled  $j$  and by  $\phi$  the phase of the normalized structure factor  $E$ . Then the equation

$$E_{\vec{h}} = |E_{\vec{h}}| \exp(i\phi_{\vec{h}}) = \frac{1}{N^{1/2}} \sum_{j=1}^N \exp(2\pi i \vec{h} \cdot \vec{r}_j) \quad (1.1)$$

leads directly to

$$\left\langle E_{\vec{h}}^* \exp(-2\pi i \vec{h} \cdot \vec{r}) \right\rangle_{\vec{h}} = \frac{1}{N^{1/2}} \left\langle \sum_{j=1}^N \exp[2\pi i \vec{h} \cdot (\vec{r}_j - \vec{r})] \right\rangle_{\vec{h}} = \left. \begin{aligned} &= \frac{1}{N^{1/2}} \text{ if } \vec{r} = \vec{r}_j \\ &= 0 \text{ if } \vec{r} \neq \vec{r}_j \end{aligned} \right\} \quad (1.2)$$

It is clear from (1.2) that the crystal structure is determined by the complex normalized structure factors  $E_{\vec{h}}$ . Since the position vectors  $\vec{r}_j$  depend on the choice of origin, as well as on the crystal structure, (1.1) does not imply that the normalized structure factors  $E_{\vec{h}}$  are determined by the crystal structure alone. As it turns out, however, the magnitudes  $|E_{\vec{h}}|$  of the normalized structure factors are in fact uniquely determined by the crystal structure and are independent of the choice of origin. The values of the phases  $\phi_{\vec{h}}$ , on the other hand, depend on the choice of origin as well as on the crystal structure [1, 2].

A finite number of magnitudes  $|E_{\vec{h}}|$  are obtainable from experiment, while the phases  $\phi_{\vec{h}}$  cannot be measured. Nevertheless, (1.2) is still useful in the determination of crystal structures since the prior knowledge that the unit cell is composed of  $N$  identical point atoms severely restricts the possible values of the phases  $\phi_{\vec{h}}$ . In fact, once the origin and enantiomorph have been specified, the magnitudes  $|E_{\vec{h}}|$  are sufficient to determine uniquely those phases  $\phi_{\vec{h}}$  with the property that the left side of (1.2), as a function of  $\vec{r}$ , is then zero everywhere except at  $N$  points where it has the value  $1/N^{1/2}$ . Owing to the finite number of data available from experiment, (1.2) is actually nonvanishing in  $N$  discrete regions, not points, and is only approximately zero elsewhere. In practice, therefore, it is the maxima of (1.2) which yield the atomic positions.

---

Present addresses: Janet E. Fisher, Central NOMIS Office, Naval Ordnance Station, Indian Head, Md. 20640; and Herbert A. Hauptman, Medical Foundation of Buffalo, 73 High Street, Buffalo, N. Y. 14203.

The problem of actually determining the values of the phases  $\phi_{\vec{h}}$  when only the magnitudes  $|E_{\vec{h}}|$  are given is the so-called phase problem. The solution of this problem is facilitated by the introduction of certain linear combinations of the phases, the structure invariants

$$\phi_{\vec{h}_1}^* + \phi_{\vec{h}_2}^* + \phi_{\vec{h}_3}^*, \quad (1.3)$$

where

$$\vec{h}_1 + \vec{h}_2 + \vec{h}_3 = 0. \quad (1.4)$$

The structure invariants are important because they serve as the connecting link between the magnitudes  $|E|$  and the phases  $\phi$ . On the one hand, the magnitudes of the structure invariants, or, equivalently, the values of

$$\cos(\phi_{\vec{h}_1}^* + \phi_{\vec{h}_2}^* + \phi_{\vec{h}_3}^*), \quad (1.5)$$

where (1.4) holds, are uniquely determined by the magnitudes  $|E_{\vec{h}}|$ . On the other hand, once the origin and enantiomorph have been specified, the values of the structure invariants (1.5) determine uniquely the values of the individual phases  $\phi_{\vec{h}}$ . The problem then is to calculate the values of the structure invariants (1.5) from the observed magnitudes of the structure factors and then, assuming the values of the former to have been found, to determine the values of the individual phases. Methods for solving the former problem have been described elsewhere [3, 4], and a computer program for calculating the values of a special class of structure invariants (1.5) is now available (NRL Report 7157). The present report is devoted to a description of a computer program for calculating the values of the individual phases, given the values of the structure invariants (1.5), which is based on the Principle of Least Squares.

## 2. THE BASIC SET OF PHASES

For definiteness it will be assumed that the space group is  $P2_1$ . The same method, with obvious modifications, is valid for the other noncentrosymmetric (or centrosymmetric) space groups.

First, as many phases  $\phi_{2h_0 \ 2\ell} = 0$  or  $\pi$  as possible are determined by means of  $\Sigma_1$  [5]:

$$E_{2h_0 \ 2\ell} \approx N^{1/2} \left\langle (-1)^k (|E_{hk\ell}|^2 - 1) \right\rangle_k \quad (2.1)$$

Next, the values of a linearly independent pair of phases

$$\phi_{\vec{k}_1}^* = \phi_{h_1 \ 0 \ \ell_1}, \quad \phi_{\vec{k}_2}^* = \phi_{h_2 \ 0 \ \ell_2}$$

are arbitrarily specified (i. e., either 0 or  $\pi$ ), and the value of a third phase

$$\phi_{\vec{k}_3}^* = \phi_{h_3 \ 1 \ \ell_3}$$

is also arbitrarily specified (between 0 and  $2\pi$ , e. g., 0), thus uniquely fixing the origin [1]. It should be noted that the theory of invariants and seminvariants developed in the latter reference permits alternative schemes for origin specification when these seem desirable, e. g., when no  $|E_{h_0\ell}|$  is large. The vectors  $\vec{k}_1, \vec{k}_2, \vec{k}_3$  are chosen so that  $|E_{\vec{k}_1}|, |E_{\vec{k}_2}|, |E_{\vec{k}_3}|$  are large, and in such a way that many of the  $|E_{\vec{k}_i + \vec{k}_j}|$ ,  $i, j = 1, 2, 3$ , are also large, i. e., so that the  $|E_{\vec{k}_i}|$  "interact" strongly with each other. It is also desirable that the  $|E_{\vec{k}_i}|$  interact strongly with those  $|E_{\vec{h}}|$  for which the values of the  $\phi_{\vec{h}}$  have been previously determined by  $\Sigma_1$ .

Finally, owing to the space group symmetry, the value of the phase  $\phi_{hk\ell}$  determines the values of the three additional phases  $\phi_{h\bar{k}\ell}, \phi_{h k \bar{\ell}}, \phi_{h \bar{k} \bar{\ell}}$  if  $k \neq 0$  while, if  $k = 0$ , only one additional phase,  $\phi_{h \ 0 \ \bar{\ell}}$ , is known. (Similarly, the phase  $\phi_{0 \ k \ 0}$  determines only the one additional phase  $\phi_{0 \ \bar{k} \ 0}$ .)

The basic set of phases consists of the origin-determining phases, several others determined by means of  $\Sigma_1$ , and the related phases whose values are determined by the space-group symmetries. Thus the values of all phases in the basic set are known.

### 3. THE STRUCTURE INVARIANTS

The whole phase determination rests on the basic set of phases. Ordinarily one attempts to find the values of only those phases  $\phi_{\vec{h}}$  whose corresponding magnitudes  $|E_{\vec{h}}|$  are large, say greater than unity. Such phases will be said to be permissible. The identity of the structure invariants whose values are needed in order to obtain preliminary values of a limited set of permissible phases is then determined iteratively as follows.

Corresponding to each pair  $\phi_{\vec{k}}, \phi_{-\vec{h}-\vec{k}}$  of the basic set having the property that the phase  $\phi_{\vec{h}}$  is permissible, one constructs the associated structure invariant

$$\phi_{\vec{h}} + \phi_{\vec{k}} + \phi_{-\vec{h}-\vec{k}} \tag{3.1}$$

and the product

$$\frac{2}{N^{1/2}} |E_{\vec{h}} E_{\vec{k}} E_{\vec{h} + \vec{k}}| = A_{\vec{k}} \tag{3.2}$$

For each such vector  $\vec{h}$ , the sum

$$\sum_{\vec{k}} A_{\vec{k}}$$

(which may consist of only a single term), taken over all allowed vectors  $\vec{k}$ , is computed. These sums (one for each  $\vec{h}$ ) are arranged in decreasing order and the largest one selected, thus defining a unique vector  $\vec{h}$  and several (perhaps only one) structure invariants

$$\cos(\phi_{\vec{h}} + \phi_{\vec{k}} + \phi_{-\vec{h}-\vec{k}}) \tag{3.3}$$

the values of which, for the purpose of the present report, are assumed to be known [3]. Since the values of the  $\phi_{\vec{k}}, \phi_{-\vec{h}-\vec{k}}$ , and the several  $\cos(\phi_{\vec{h}} + \phi_{\vec{k}} + \phi_{-\vec{h}-\vec{k}})$  are thus presumed to be known, the value of  $\phi_{\vec{h}}$  is actually overdetermined (in general); its evaluation by least squares will be described later in Section 4. The new phase  $\phi_{\vec{h}}$  and its three (or possibly only one) symmetry-related phases are added to the basic set of known phases and the process repeated. During the second cycle, however, the three largest sums

$$\sum_{\vec{k}} A_{\vec{k}}$$

are selected, so that three new phases are determined, rather than only one as in the first cycle. During the third, fourth, fifth, . . . , cycles, the numbers of new phases determined are five, seven, nine, . . . . Thus the order in which the values of the phases are to be obtained is determined, and the identity of the structure invariants  $\cos(\phi_{\vec{h}} + \phi_{\vec{k}} + \phi_{-\vec{h}-\vec{k}})$  whose values are required, is also found. Clearly, the rate at which new phases are acquired may be varied if deemed desirable.

The first structure invariant  $\cos(\phi_{\vec{h}} + \phi_{\vec{k}} + \phi_{-\vec{h}-\vec{k}})$  whose value is different from  $\pm 1$  leads to two possible values for  $\phi_{\vec{h}} + \phi_{\vec{k}} + \phi_{-\vec{h}-\vec{k}}$  differing only in sign. The sign of this

invariant  $\phi_{\vec{h}} + \phi_{\vec{k}} + \phi_{-\vec{h}-\vec{k}}$  may be arbitrarily specified, thus fixing the enantiomorph and leading to a unique value for  $\phi_{\vec{h}}$  [1]. Equivalently, from the present point of view, there exist two solutions  $\phi_{\vec{h}}$  of the least squares problem (Section 4) and these yield the same minimum sum of squares. Either of these two values for  $\phi_{\vec{h}}$  is chosen arbitrarily and the enantiomorph is thus fixed. The values of all the remaining phases are then uniquely determined.

#### 4. APPROXIMATE EVALUATION OF INITIAL PHASES BY MEANS OF LEAST SQUARES

Suppose now that the structure invariants (3.3) called for by the iterative process described in Section 3 have been computed, as has been assumed for the present purpose. Thus, for each fixed vector  $\vec{h}$  whose corresponding phase  $\phi_{\vec{h}}$  is to be determined, the values of several structure invariants

$$\cos(\phi_{\vec{h}} + \phi_{\vec{k}} + \phi_{-\vec{h}-\vec{k}}) = c_{\vec{k}} \quad (4.1)$$

are presumed to be known. The values of the several phases  $\phi_{\vec{k}}$ ,  $\phi_{-\vec{h}-\vec{k}}$  are also assumed to be known in accordance with Section 3. It is natural to suppose further that the larger the number  $n$  of contributors to the average from which  $c_{\vec{k}}$  is computed [3] and the larger the value of  $|E_{\vec{h}} E_{\vec{k}} E_{\vec{h}+\vec{k}}|$ , the more accurate the computed value of  $c_{\vec{k}}$  will be. Hence a weight  $w_{\vec{k}}$  is defined by means of

$$w_{\vec{k}} = |E_{\vec{h}} E_{\vec{k}} E_{\vec{h}+\vec{k}}| \sqrt{n} \quad (4.2)$$

The phase  $\phi_{\vec{h}}$  is then determined by minimizing, in accordance with the Principle of Least Squares,

$$\Phi = \frac{\sum_{\vec{k}} w_{\vec{k}}^2 (\cos(\phi_{\vec{h}} + \phi_{\vec{k}} + \phi_{-\vec{h}-\vec{k}}) - c_{\vec{k}})^2}{\sum_{\vec{k}} w_{\vec{k}}^2} \quad (4.3)$$

which, after a straightforward but lengthy calculation, finally reduces to

$$\Phi = \frac{1}{2} C_2 \cos 2\phi_{\vec{h}} - \frac{1}{2} S_2 \sin 2\phi_{\vec{h}} - 2 C_1 \cos \phi_{\vec{h}} + 2 S_1 \sin \phi_{\vec{h}} + c \quad (4.4)$$

where

$$C_2 = \frac{\sum_{\vec{k}} w_{\vec{k}}^2 \cos 2(\phi_{\vec{k}} + \phi_{-\vec{h}-\vec{k}})}{\sum_{\vec{k}} w_{\vec{k}}^2} = \left\langle \cos 2(\phi_{\vec{k}} + \phi_{-\vec{h}-\vec{k}}) \right\rangle_{\vec{k}}, \quad (4.5)$$

$$S_2 = \frac{\sum_{\vec{k}} w_{\vec{k}}^2 \sin 2(\phi_{\vec{k}} + \phi_{-\vec{h}-\vec{k}})}{\sum_{\vec{k}} w_{\vec{k}}^2} = \left\langle \sin 2(\phi_{\vec{k}} + \phi_{-\vec{h}-\vec{k}}) \right\rangle_{\vec{k}}, \quad (4.6)$$

$$C_1 = \frac{\sum_{\vec{k}} w_{\vec{k}}^2 c_{\vec{k}} \cos(\phi_{\vec{k}} + \phi_{-\vec{h}-\vec{k}})}{\sum_{\vec{k}} w_{\vec{k}}^2} = \left\langle c_{\vec{k}} \cos(\phi_{\vec{k}} + \phi_{-\vec{h}-\vec{k}}) \right\rangle_{\vec{k}}, \quad (4.7)$$

$$S_1 = \frac{\sum_{\vec{k}} w_{\vec{k}}^2 c_{\vec{k}} \sin(\phi_{\vec{k}} + \phi_{-\vec{h}-\vec{k}})}{\sum_{\vec{k}} w_{\vec{k}}^2} = \left\langle c_{\vec{k}} \sin(\phi_{\vec{k}} + \phi_{-\vec{h}-\vec{k}}) \right\rangle_{\vec{k}}, \quad (4.8)$$

and

$$c = \frac{\sum_k w_k \left( \frac{1}{2} + c_k^2 \right)}{\sum_k w_k} = \left\langle \frac{1}{2} + c_k^2 \right\rangle_k, \quad (4.9)$$

so that the quantities  $C_2$ ,  $S_2$ ,  $C_1$ ,  $S_1$ , and  $c$  are all known. Then  $\phi_{\vec{h}}$  is uniquely determined by minimizing  $\Phi$  as given by (4.4). The chief aim in the present report is to describe the computer program which solves this least-squares problem. It may happen occasionally, particularly in one of the early cycles of the iteration, that two distinct values of  $\phi_{\vec{h}}$  lead to essentially the same minimum value for  $\Phi$ . In this case one may be led to two or more solutions, and the incorrect ones will have to be rejected by inspection of the corresponding Fourier Series (1.2). Finally, the first structure invariant  $\cos(\phi_{\vec{h}} + \phi_{\vec{k}} + \phi_{-\vec{h}-\vec{k}})$  whose value differs significantly from  $\pm 1$  yields two possible values for  $\phi_{\vec{h}} + \phi_{\vec{k}} + \phi_{-\vec{h}-\vec{k}}$ , differing only in sign, and therefore two possible values for  $\phi_{\vec{h}}$ . Choosing either of these two values for  $\phi_{\vec{h}}$  is equivalent to choosing one of the two possible signs for  $\phi_{\vec{h}} + \phi_{\vec{k}} + \phi_{-\vec{h}-\vec{k}}$ , and therefore to selecting one of the two possible enantiomorphs. Once this is done, the remaining phases are then uniquely determined. It has been found in practice that it is feasible to determine the values of several dozen phases by this process. The more efficient tangent formula [Eq. 5.62, Ref. 6]

$$\tan \phi_{\vec{k}} = \frac{\sum_{\vec{k}} |E_{\vec{k}} E_{\vec{h}-\vec{k}}| \sin(\phi_{\vec{k}} + \phi_{\vec{h}-\vec{k}})}{\sum_{\vec{k}} |E_{\vec{k}} E_{\vec{h}-\vec{k}}| \cos(\phi_{\vec{k}} + \phi_{\vec{h}-\vec{k}})} \quad (4.10)$$

may then be used in order to evaluate the remaining permissible phases.

## 5. THE COMPUTER PROGRAMS

The two Fortran computer programs, written for the CDC-3800 computer, described in this report are: (a) Buildup, which establishes both the order of phase determinations and the identity of the needed structure invariants (3.3), and (b) LS Phases, which assumes that the buildup has been ascertained and the cosine invariants have been identified and computed, and then proceeds to use these cosines for actual phase determinations. These two programs are intended to implement the procedures described in Sections 3 and 4, respectively.

Buildup was initially written to explore the possibility of "building up" from a given set of initial phases to the full set of permissible phases. This is attempted in a cycle-by-cycle process in which the vectors for which phases are assumed to be known at the start of each cycle are allowed to interact with each other in the manner described in Section 3. For convenience, vectors  $(h, k, \ell)$  are stored within the computer memory in canonical form only; for the present space group  $P2_1$  this form is defined as that in which  $h$  and  $k$  are nonnegative, and  $\ell$  is also nonnegative if  $h$  is 0. It is to be recognized throughout that each such stored vector actually represents four (or two) vectors related through symmetry transformations.

In the interactive process, each vector  $\vec{k}$ , taken in canonical form, is paired in turn with all symmetry-related forms of itself and of each other vector currently in the basic set—the second vector of the pair being the vector  $-\vec{h}-\vec{k}$  of Section 3. The interactions may be carried out completely by pairing each vector in the computer-stored list of base vectors with the several forms of each vector which does not precede it in the stored list. For each pairing, the third vector  $\vec{h}$ , formed by taking the negative sum of the first two, is put into canonical form. If this third vector is to be found in the E deck,

i. e., if it represents a permissible phase, the trio of matching vectors is termed a contributor to the phase  $\phi_{\vec{h}}$ .

Contributors are tabulated within the computer by third vector  $\vec{h}$  taken in canonical form; information tabulated includes the sum of A values and the number of contributors. When the interaction of all vectors input to the cycle is complete, the tabulated information corresponding to all vectors not already in the basic set is examined. After rejecting those which do not have a prespecified minimum number of contributors and minimum sum of A values, the remaining vectors are examined, and the quota of pickups for the present cycle is sought from among those with the largest  $\Sigma A_k$ . Of course, if either the quota or the two minimum requirements are relatively large, it may not be possible to meet the full quota for every cycle, especially at the outset. Even if not, the prespecified quotas for the succeeding cycles are not increased.

For the sake of efficiency, the procedure actually followed for the interactions after the first cycle is to use information brought forward concerning interactions through, and including, the previous cycle, and to limit new interactions to those between pickups in the immediately previous cycle and all current members of the basic set.

Buildup now offers several options adapting it specifically to use as a preliminary to phase determination by cosine invariants:

(a) Contributors may be rejected if the corresponding cosine invariants cannot be reliably computed. For the present space group  $P2_1$  this means vector triples of the form

$$h \ k \ \ell \qquad h \ -k \ \ell \qquad -2h \ 0 \ -2\ell$$

with  $k \neq 0$ .

(b) A printout may be output to list the matching vector triples, the three  $|E|$  values, and the A value.

(c) The information indicated under (b) may be punched on cards for use as input to a program to compute the cosine invariants.

(d) The vectors in the original basic set and those added, in the order of pickup, may be punched, together with their  $|E|$  values, for use as input to LS Phases—or for other purposes such as application of the tangent formula.

(e) An E deck may be punched, consisting of the same information, in the same order, as that output by the immediately previous option, but in the E deck format. This deck is ordinarily a greatly abridged version of the original E deck, assumed to represent all permissible phases.

In practice it may be desirable to use Buildup for a preliminary run in order to get a feel for the number of invariants needed; this is feasible since Buildup runs quickly. The input would be set to select the option of disallowing certain contributors; of the optional outputs, probably only the E deck output would be selected. As many cycles may be run as conceivably needed. After examining the printout, one decides how many phases to compute (how many cycles to include) and reruns Buildup set for this number. For the rerun the newly punched E deck would be used instead of the original one, after omitting any unneeded  $h, k, \ell, |E|$  items. Use of an E deck limited to those vectors which are in the final basic set ensures that only those invariants usable in the phase-determining sequel are output by Buildup. The omission of the unneeded  $h, k, \ell, |E|$  items from the E deck is easily accomplished since the items are punched in the order of pickup, and any unneeded ones would therefore be at the end of the deck. In the new Buildup run, all optional outputs would ordinarily be requested, except a new E deck.

The computer input and output for Buildup will be described later in Section 7. Appendix A gives a listing of the Fortran program for Buildup, and Appendix B gives information on a very brief sample run. The example chosen is hypothetical, without physical significance, and is intended to be illustrative and tutorial only.

The second program described in this report is LS Phases, which uses the Principle of Least Squares actually to determine phases, in the order prescribed by Buildup. The basic equations are (4.4) through (4.9), with  $w_{\vec{k}}$  as defined by (4.2). Equation (4.4) was slightly modified, to the following form, adjudged somewhat more convenient for computer use:

$$\Phi = \sin\phi_{\vec{h}} (-C_2 \sin\phi_{\vec{h}} - S_2 \cos\phi_{\vec{h}} + 2S_1) - 2C_1 \cos\phi_{\vec{h}} + (c + 0.5 C_2). \quad (5.1)$$

As a preliminary to minimizing  $\Phi$  it is necessary to calculate the coefficients  $C_1$ ,  $S_1$ ,  $C_2$ ,  $S_2$ , and  $c$ . To this end the vectors in the basic set as brought forward into the cycle are allowed to interact with each other in a manner similar to that for Program Buildup, and the sums required for the calculation of the coefficients are accumulated as the interaction proceeds. This serves as prologue to all phase calculations for the cycle; note that the calculations include not only the phases of the new pickups but also the recalculation ("refinement") of phases for which approximate values are assumed known at the start of the cycle.

Despite the strong similarity, the present interactive process and that for Buildup have several points of difference. In the present program the entire set of vectors in the basic set at the start of each cycle must be allowed to interact with itself rather than utilizing the results of those interactions which have occurred in previous cycles; this is because the latest phase information as brought forward from the immediately previous cycle is used, as the interaction proceeds, to accumulate the sums for the coefficients in (5.1). Also, in LS Phases runs, prior knowledge, furnished by a running of Buildup, of just which vectors will be in the basic set at the end of the cycle means that, as the interaction proceeds, the program can retain information on these vectors only, rather than on all vectors in the E deck.

Consistent with the fact that only the canonical forms of vectors are retained in the computer memory, only those phases corresponding to canonical-form vectors are stored. This indicates the following process: When a vector triple  $\vec{h}$ ,  $\vec{k}$ ,  $-\vec{h} - \vec{k}$  is identified as a contributor to the phase  $\phi_{\vec{h}}$ , a symmetry transformation is applied to  $\vec{h}$  to put it into canonical form (either of two such transformations may be equally effective if  $\vec{h}$  is not three dimensional), and the same transformation is applied to  $\vec{k}$  and  $-\vec{h} - \vec{k}$ . Thus the stored phases  $\phi_{\vec{k}}$  and  $\phi_{-\vec{h}-\vec{k}}$  must be transformed accordingly before contributing to the accumulating sums for the coefficients.

Once the interactions for the cycle are complete and all sums required for the coefficients are accumulated in the computer storage, the actual solution to the problem of minimizing  $\Phi$  is handled by a subroutine, Rev 2 (including Entry Srch). At the outset, a table of sines and cosines is constructed for  $\phi_{\vec{h}}$  between 0 and  $2\pi$  inclusive (actually 0 and 6.32), at intervals of 0.01. For each phase determination, the coefficients appearing in (5.1) are furnished to Rev 2, which brackets the one (or two) minimizing  $\phi_{\vec{h}}$  between successive entries in the sine-cosine table and finally uses parabolic interpolation. Corresponding to each minimizing  $\phi_{\vec{h}}$ , the subroutine returns the pair  $\phi_{\vec{h}}$ ,  $\Phi$ , where  $\phi_{\vec{h}}$  is taken to be in the range  $-\pi$  to  $\pi$ . There should never be more than two solutions to the minimizing problem. If three are found, the message "3PHI" is printed, followed by the three  $\phi_{\vec{h}}$ ,  $\Phi$  pairs, and a normal return to the main program is taken.

If there is a double solution, the program selects one; in the usual case of unequal  $\phi$ , the selected solution is the one yielding the smaller  $\phi$ . If the other choice is desired for a particular cycle and a particular vector, this may be accomplished by inserting one Fortran statement for each such exception immediately following statement number 959 in the main program LS Phases. For this purpose, note that cycles are indexed in the program by N, and vectors by L. Suppose as an example that a first running of LS Phases indicates a multiple solution in which the alternative choice for  $\phi_{\bar{h}}$  is preferred; and suppose that this occurs in cycle 5 and that the vector in question is the twelfth one in the input E deck. Then the Fortran statement would be

```
IF(N .EQ. 5 .AND. L .EQ. 12) MULT SOLN = 2
```

As many such statements as desired may be included for the same or different cycles and vectors. The exceptional choice of phase is made only for the specific cycle(s) and vector(s) covered by the special statement(s); after each such execution, the program automatically resets MULT SOLN to 1.

The need for the exceptional choice seldom occurs more than once or twice in an entire run, if at all, and usually only in an early cycle.

The input required for LS Phases will be specified in detail later. An essential part of this input is, of course, the cosine invariants. The calculation of these invariants, not covered here, may be based on matching triples obtained by letting all vectors corresponding to permissible phases interact with each other and rejecting all triples whose A values are adjudged too small; or it may be based solely on the matching triples output from Buildup. If one wishes to limit the triples output from Buildup to those whose cosine invariants are actually to be used by LS Phases, it is essential that the E deck input to Buildup include only vectors in the final basic set.

Another item of input concerns an option which the user of LS Phases has of requesting that any particular phase(s) be "forced" to prespecified value(s). By this is meant that the value for the phase to be carried forward from the cycle for subsequent phase calculations is set equal to the prespecified forcing value, although the actually computed one will be printed and will also be used in the calculation of figures of merit (to be described). The procedure for requesting such forcing will be detailed in Section 8. Usually, at least the three origin-determining phases will be forced to their original values.

If for any reason the user wishes to output, at the conclusion of the run, the  $\phi_{\bar{h}}$  which would have been carried forward to subsequent cycles, he will find these phases stored in the array PHI in consecutive locations starting in the first, and the corresponding  $h\ k\ l$  values stored in the same order in the arrays JB 1, JB 2, and JB 3 respectively. The phases stored in PHI include the result of any forcing.

As already noted, in each cycle not only are the phases for new pickups computed, but also the phases brought forward into the cycle are recomputed ("refined") on the basis of phase information brought forward. It is usually desirable to include several refinement cycles at the end of the LS Phases run, i. e., after all pickups are complete, in which only previously approximated phases are computed. Also as many refinement cycles as desired may be sandwiched in between the pickup cycles. In the case that one or more refinement cycles are added at the end, it is important in the running of Program Buildup that one such cycle be included as the final cycle of the run, since otherwise some contributors, and hence some needed invariants, would not be identified. For refinement cycles the number of new pickups is, of course, indicated as 0 in the input.

Since the quantity  $\phi$  as defined by (4.3) is minimized as a function of  $\phi_{\bar{h}}$ , and since it gives a weighted average of the squared deviations from the invariants  $c_{\bar{h}}$ , its square

root may be taken as a sort of rms value of the residuals in the least-squares process, and its relative smallness may be taken as indicative of the reliability of the  $\phi_{\bar{h}}$  determination. The value of  $\sqrt{\Phi}$  is accordingly included as part of the computer output for each  $\phi_{\bar{k}}$ . Furthermore, an overall cycle Figure of Merit is obtained by taking the square root of the weighted average of all the  $\Phi$ 's obtained for the cycle, wherein the weight for each  $\Phi$  is the associated  $\Sigma A_{\bar{k}}$  as defined by (3.2) and following.

A second Figure of Merit for each cycle is obtained on the basis of the  $h\ 0\ \ell$  phases calculated during the cycle. Since such phases for the space group  $P2_1$  are known to be either 0 or  $\pi$ , it is meaningful to consider the squared deviation of the least-squares result  $\phi_{\bar{h}}$  from the nearer of 0 or  $\pi$ ; alternatively for the case of any  $h\ 0\ \ell$  which is forced to a prespecified value, consider the squared deviation from such value. Calculate the square root of the weighted average of all such squared deviations for the cycle to obtain a second Figure of Merit; the weights used for the averaging are again the  $\Sigma A_{\bar{k}}$  quantities. The result is output for each cycle, together with the number of  $h\ 0\ \ell$ 's contributing.

At the conclusion of the cycle-by-cycle printout of phases, a "summary print" is output to present in a single table the results of the phase calculations. Each line of the table contains the  $h\ k\ \ell$  indices for some one of the phases, followed by the  $\phi_{h\ k\ \ell}$  output cycle by cycle, starting with the last cycle and working backward through the cycle in which the phase was first calculated, or until the input value ("Cycle 0" value) is reached. The maximum total number of phases printed per line is 25, and the user may reduce this by changing the dimensioning as explained later. If the number of cycles, including "Cycle 0", exceeds the dimensioning, one or more of the earlier cycles (1, 2, 3, ...) will be omitted from the print, but the input phases will always be included.

The computer input and output for LS Phases will be described in Section 8. Appendices C and D contain a listing of the Fortran program and information on a very brief sample run; this run is a follow-up on the sample run used for Program Buildup. The example chosen is hypothetical, without physical significance, and is intended to be illustrative and tutorial only.

## 6. CAPACITY AND ARRAY DIMENSIONS

Programs Buildup and LS Phases were written for the CDC-3800 computer and assume two memory banks of 32K each. In each program a BANK statement is used to control partially the storage allocations. These statements may be modified as expedient, or omitted for other computers.

Program Buildup is currently dimensioned to allow up to 1500 reflections in the E deck, 100 cycles, and 9000 unique matching vector triples, order not considered, for the buildup and for subsequent use in the computation of the cosine invariants. The program contains an array JSUB which is currently dimensioned at (12, 29, 18). This dimensioning restricts the maximum value of the index  $h$  to 11, and of  $k$  to 28; for the index  $\ell$  it restricts the difference between the extreme positive and extreme negative values, i. e., the sum of the magnitudes of the two extremes, to 17.

Each of the above limitations may be changed, subject only to the availability of storage, as follows:

- (a) To change the maximum number of reflections allowed, in Common Block /11/ change each 1500 in the dimensioning to the new value.
- (b) To change the maximum number of cycles allowed, in Common Block /22/ change 100 to the new value.

(c) To change the maximum number of unique vector triples allowed, in Common Block /44/ change each 9000 to the new value. (The program has a built-in safeguard which prevents overflow in the case of the Common Block /44/ arrays. The excess invariants, however, are not output in the invariants print and punch options. In uses of Buildup in which the invariants are not to be output, e. g. , when merely exploring the buildup or establishing the buildup for use with the tangent formula, storage may be saved if need be by drastically reducing the dimensioning in Common Block /44/.)

(d) To change the bounds on the indices  $h$ ,  $k$ , and  $l$ , in Common Block /33/ the dimensions of the three-dimensional array JSUB may be changed subject to the conditions that

1. The first dimension, currently set at 12, must be at least as large as one more than the maximum value of  $h$ ,
2. The second dimension, currently set at 29, must be at least as large as one more than the maximum value of  $k$ ,
3. The third dimension, currently set at 18, must be at least as large as one more than the difference between the extreme positive and the extreme negative values of  $l$ .

LS Phases is currently dimensioned to allow up to 300 reflections in the E deck, 100 cycles, and 9000 cosine invariants for use in the phase determinations. The program contains an array JSUB (12, 29, 18), which imposes the same restrictions on the index ranges as the like-named array in Buildup, and an array SMRY PRINT (300, 25), which limits to 25 the number of columns in the body of the table summarizing the phases as computed cycle by cycle. Twenty-five is an upper bound set by the 136-character line of the line printers at the NRL computing facility. This dimensioning of 25 columns may be reduced to as low as 1 if need be to save storage, while perhaps sacrificing some columns of print.

In judging the adequacy of the allowance for 300 reflections, bear in mind that those for which phases are to be determined are assumed to have been identified by a prior use of Buildup, which has also output an E deck including only such reflections. Usually, only a relatively small number of phases, e. g. , not more than 300, will be sought by the least-squares technique.

The above limitations may be changed, subject only to the availability of storage, as follows:

(a) To change the maximum number of reflections allowed, in Common Blocks /111/ and /222/ change each 300 to the new value.

(b) To change the maximum number of cycles allowed, change the dimensioning of the NR OUT TBL in the DIMENSION statement from 100 to the desired value.

(c) To change the maximum number of invariants to be used in the phase determinations, in Common Block /333/ change each 9000 to the new value. Note that the maximum number applies only to the number used and not to the number read, since no invariant is retained in the computer memory unless it is usable, i. e. , unless the associated matching vector triple contains only vectors involved in the phase determinations.

(d) To change the bounds on the indices  $h$ ,  $k$ , and  $l$ , in Common Block /444/ the dimensions of the three-dimensional array JSUB may be changed subject to the same conditions as those itemized above for the JSUB array in Buildup.

## 7. PROGRAM BUILDUP INPUT AND OUTPUT

Data Input to Program Buildup1. Control card

FORMAT(13I5, F10)

## Columns

- 1-5 NR E, the number of cards in the E deck.
- 6-10 MILLR 1, an upper bound on the first Miller index h.
- 11-15 MILLR 2, an upper bound on the second index k.
- 16-20 MILLR 3 LO, a lower bound on the third index l.
- 21-25 MILLR 3 HI, an upper bound on the third index l.  
Suggestion—Verify the adequacy of the dimensioning for the JSUB array in Common Block /33/:  
 First dimension  $\geq$  MILLR 1 + 1  
 Second dimension  $\geq$  MILLR 2 + 1  
 Third dimension  $\geq$  MILLR 3 HI - MILLR 3 LO + 1.
- 26-30 NR BV, the number of base vectors initially (i. e., the number whose phases are assumed known).
- 31-35 NR QDS, the number of quadruples; these will be described below. Since there is one quadruple for each cycle, NR QDS is also the number of cycles.
- 36-40 MAX NR INV, the maximum number of invariants which storage allocations permit. This entry is the same as the dimensioning for the JSER and A VALS arrays in Common Block /44/ (presently set at 9000).
- 41-45 K 1, used to indicate whether certain contributors, whose cosine invariants cannot be reliably computed, are to be disallowed. If K 1 = 1, they will be disallowed; if K 1 = 0, they will not.
- 46-50 K 2, the invariants print indicator. If K 2 = 1, the matching triples whose cosine invariants will be needed are printed; if K 2 = 0, they will not be printed.
- 51-55 K 3, the invariants punch indicator. If K 3 = 1, information similar to that for the K 2 print option (above) will be output onto punched cards.
- 56-60 K 4, the base vectors punch indicator. If K 4 = 1, the vectors in the original basic set and those added, in the order of pickup, are punched, together with their |E| values, for use with the program LS Phases.
- 61-65 K 5, the E-deck punch indicator. If K 5 = 1, an E deck is punched, consisting only of those vectors in the basic set at the end of the final cycle (the same set as the one punched in the K4 option, but in the E-deck format).
- 66-75 EEE FCTR, the factor by which the product of the three E magnitudes must be multiplied to produce A.

2. E Deck FORMAT(5(3I3, F7.4))

Includes all  $h, k, \ell, |E|$  values (five sets per card) which are in the original basic set or are available for pickup.

3. Original base vectors FORMAT(3I5)

One base vector per card (one  $h k \ell$  triple)

4. Quadruples FORMAT(4I5)

One card for each cycle that is contemplated; on each card are specified four items:

- a. Cycle number. This will be printed to identify the cycle. Cycles may be numbered 1, 2, 3, . . . , or to suit the user.
- b. Number of pickups to be sought in the cycle.
- c. Minimum number of contributors which a vector must have before being considered for pickup.
- d. Minimum  $\Sigma A_i$  for pickups.

Output from Program Buildup

For each cycle, results are output as follows:

A. A two- or three-line summary giving

Cycle number (CY\_\_).

The number of old base vectors, i. e. , vectors in the basic set at the start of the cycle (OLD BV\_\_).

The number of new base vectors, i. e. , vectors picked up during the cycle (NEW BV\_\_).

The number of newly found contributors whose three vectors are all in the E deck (NEW IN E\_\_).

The corresponding cumulative number (CUM IN E\_\_).

The number of contributors to the old base vectors (CTRB OLD BV\_\_).

The number of contributors to the new base vectors (CTRB NEW BV\_\_).

The sum of the two previous numbers (CTRB BOTH\_\_).

The cumulative number of unique cosine invariants, order not considered, which will be needed. This is omitted unless at least one of the "invariant options" is indicated, i. e. , unless at least one of the indicators K 1, K 2, or K 3 has been set to 1. If the number of invariants exceeds the maximum number permitted by the dimensioning, i. e. , exceeds the input item MAX NR INV, this maximum number is printed, followed by "PLUS". (CUM NR OF COSINE INVARIANTS\_\_).

B. The body of the printout for the cycle, including one line of printout for each vector in the basic set at the end of the cycle; new pickups are flagged by two asterisks. Listed

are  $h$ ,  $k$ ,  $\ell$ ,  $|E|$ , number of the cycle in which picked up (0 denotes original basic set), average  $A$ , i. e.,  $\sum A_k / (\text{number of contributors})$ , number of contributors, and  $\sum A_k$ .

Optional outputs, each of which may be independently chosen, include:

A. A printout of the distinct matching vector triples, i. e., vector triples with distinct canonical forms, order not considered, together with the three  $|E|$  values and  $A$ . This listing is ordered on decreasing  $A$ , and the successive triples are numbered sequentially in the final columns of print.

B. A deck of punched cards, FORMAT(9I4, 4F8.4, F9.5), giving  $h_1, k_1, \ell_1, h_2, k_2, \ell_2, h_3, k_3, \ell_3, |E_1|, |E_2|, |E_3|, |E_1 E_2 E_3|$ , and  $A$ ; one card per triple. The information in this deck is that needed by programs to determine the cosine invariants.

C. A deck of punched cards, FORMAT(3I5, F10.4), giving  $h, k, \ell$ , and  $|E|$  for the vectors in the basic set at the end of the computation, i. e., original base vectors plus pickups; one vector per card. This deck is for use as input to the LS Phases program, after available phase information is added in a manner to be described.

D. An E deck, FORMAT(5(3I3, F7.4)), giving  $h, k, \ell$ , and  $|E|$ , five sets per card, for the vectors in the basic set at the end of the computation. This deck is intended primarily as a replacement for the presumably more inclusive original E deck, in the event that it is decided to rerun the program to punch the invariants needed; since the new E deck would contain only the vectors in the final basic set, no unneeded invariants would be punched. The vectors and  $|E|$  values are listed in the order in which they are added to the basic set.

## 8. PROGRAM LS PHASES INPUT AND OUTPUT

### Data Input to Program LS Phases

#### 1. Control card                      FORMAT(7I5, F10)

##### Columns

1-5	NR CY TOT, the number of cycles to be run.
6-10	NR V ORIG, the number of vectors in the original basic set.
11-15	MILLR 1, an upper bound on the first Miller index $h$ .
16-20	MILLR 2, an upper bound on the second index $k$ .
21-25	MILLR 3 LO, a lower bound on the third index, $\ell$ .
26-30	MILLR 3 HI, an upper bound on the third index, $\ell$ . <u>Suggestion</u> —Verify the adequacy of the dimensioning for the JSUB array in Common Block /444/: First dimension $> \text{MILLR } 1 + 1$ Second dimension $> \text{MILLR } 2 + 1$ Third dimension $> \text{MILLR } 3 \text{ HI} - \text{MILLR } 3 \text{ LO} + 1$ .
31-35	N COL SMRY, the number of columns for which the SMRY PRNT array in Common Block /222/ is actually dimensioned, i. e., the second subscript in the dimensioning. <i>Must not exceed 25.</i>

36-45           EEE FCTR, the factor by which the product of the three E magnitudes must be multiplied to produce A.

2. Number out table                   FORMAT(16I5)

For each cycle in turn, a number is specified to indicate how many vectors will be in the basic set at the end of the cycle, as previously determined by Program Buildup. Punched 16 per card.

3. Vectors, E magnitudes, and phase information                   FORMAT(3I5, F10.4, 2F15)

These are listed in the order in which the vectors are added to the basic set. The following information is listed for each vector in the final basic set:

a.  $h, k, \ell, |E|$ .

b. Forcing information. If the phase  $\phi_h$  as computed by the program is always to be set (forced) to a specified value, enter that value, in radians and on the range  $-\pi$  to  $\pi$ . If the phase  $\phi_h$  is not to be so forced, enter 9.

c. If  $h k \ell$  is in the original basic set, enter the input phase  $\phi_h$ , in radians and on the range  $-\pi$  to  $\pi$ . Otherwise leave blank. Note that this deck is an optional output (the K 4 option) from Program Buildup, except that the forcing information and the input phases must be added.

4. Cosine invariants                   FORMAT(3(3I3, 1X), 24X, F9.5, F6, F8.4)

The three matching vectors are entered, one after the other, followed by A, the number of contributors to the average upon which the cosine computation is based, and the cosine itself. *The invariants deck must be followed by a card containing only a 9 punch in col. 1.*

Output from Program LS Phases

A. A preliminary output lists the input  $h, k, \ell, |E|, \phi$ , and forcing information. This output is included mainly to show the two last-named items, as a matter of record and for checking purposes.

B. Following is a cycle-by-cycle printout which lists for each phase the indices  $h k \ell$ , the E magnitude, the number of contributors to the phase determination,  $\Sigma A_k$ , average A, i. e.,  $\Sigma A_k / (\text{number of contributors})$ , and  $\phi$  and  $\sqrt{\Phi}$ . In the case of two solutions, the  $\phi$  not selected and the corresponding  $\sqrt{\Phi}$  are listed at the end of the line. Any new additions to the basic set are flagged by two asterisks. The final items in the cycle printout are the cycle Figure of Merit and the  $h k \ell$  Figure of Merit, the latter followed by the number of  $h k \ell$  phases computed in the cycle and hence serving as contributors to the Figure of Merit.

C. A "summary print" is output, following the cycle-by-cycle printout of phases, to present in a single table the results of the phase calculations. This is described in some detail near the end of Section 5.

D. The last output item reports on the cosine invariants. The total number read and the number actually used in the computations are printed. This printout is followed by a listing of the vector triples (each vector in canonical form) whose cosine invariants could have been used in the calculations but which were missing from the input. The list

will ordinarily be limited to those few triples whose corresponding cosines cannot be reliably computed for the present space group; here it is of some interest in that it indicates the number of such triples. It has, however, also proved useful in detecting unintentional omissions from the invariants deck.

#### REFERENCES

1. Hauptman, H. , and Karle, J. , Acta Cryst. , 9:45 (1956).
2. Hauptman, H. , "The Encyclopedia of X-Rays and Gamma Rays," p. 749, New York: Reinhold, 1963.
3. Hauptman, H. , Fisher, J. , Hancock, H. , and Norton, D. , Acta Cryst. , B25:811 (1969).
4. Hauptman, H. , Abstract B8, New Orleans Meeting of the Am. Cryst. Assoc. , March 2, 1970.
5. Hauptman, H. , and Karle, J. , "Solution of the Phase Problem, I. The Centrosymmetric Crystal," Am. Cryst. Assoc. Monograph No. 3, 1953.
6. Karle, J. , and Hauptman, H. , Acta Cryst. 9:635 (1956).

APPENDIX A  
LISTING OF PROGRAM BUILDUP

```

PROGRAM BUILDUP
BANK,(0),/44/
COMMON/11/LGEN(1500,3),EGEN(1500),LBSC(1500,3),EBSC(1500),
X LBIG(1500,3),BIG SG A(1500),SUM A(1500),KNT(1500)
COMMON/22/JQDS(100,4)
COMMON/33/JSUB(12,29,18)
COMMON/44/JSER(9000), A VALS(9000)
DIMENSION JV(4,3),JVV(4,3),JVC(3),JVVC(3),JS(3)
150 FORMAT(13I5,F10)
160 FORMAT(5(3I3,F7.4))
170 FORMAT(3I5)
180 FORMAT(4I5)
C**** READ DATA AND INITIALIZE--
C (1) READ THE CONTROL CARD. COMPUTE UPPER BOUNDS ON THE NUMBER OF
C DISTINCT VALUES FOR EACH OF THE THREE MILLER INDICES. THESE WILL
C GIVE THE RANGES ON THE SUBSCRIPTS TO BE USED FOR THE JSUB ARRAY.
C COMPUTE ALSO JADD, WHICH WILL BE ADDED TO THE THIRD INDEX TO
C ENSURE A POSITIVE RESULT FOR SUBSCRIPTING.
C (2) READ THE GENERAL E DECK, GIVING H, K, L, AND THE MAGNITUDE OF
C E. USING EACH H, K, L AS THE BASIS FOR SUBSCRIPTING, STORE IN
C JSUB ARRAY THE SUBSCRIPT INDICATING WHERE THE H,K,L,AND E VALUES
C ARE TO BE FOUND.
C (3) READ THE INPUT BASIC VECTORS INTO LBSC. SUPPLY THE CORRE-
C SPONDING E VALUES TO EBSC.
C (4) READ THE QUADRUPLES GIVING FOR EACH CYCLE THE PICKUP QUOTAS
C AND CRITERIA
C (5) INITIALIZE, INCLUDING SOME FLAGGING IN THE KNT ARRAY. NOTE
C THAT THE KNT ARRAY WILL BE USED FOR STORING THE COUNTS ON THE NR OF
C CONTRIBUTORS, BUT ALSO FLAGS WILL BE USED IN THIS ARRAY TO DENOTE
C OLD BASE VECTORS AND CURRENT ADDITIONS TO THE BASIC SET, WITH NR
C OF CYCLE WHEN PICKED. II WILL INDEX THE CYCLE NR.
READ 150,NR E, MILLR1, MILLR2, MILLR3 LO, MILLR3 HI, NR BV,NR QDS,
X MAX NR INV, K1, K2, K3, K4, K5, EEE FCTR
N1 = MILLR1 + 1
N2 = MILLR2 + 1
N3 = MILLR3 HI - MILLR3 LO + 1
JADD = -MILLR3 LO + 1
READ 160, ((LGEN(I,J), J=1,3), EGEN(I), I=1, NR E)
DO 185 I=1, NR E
185 SUM A(I) = KNT(I) = 0
DO 186 I=1, N1
DO 186 J=1, N2
DO 186 K=1, N3
186 JSUB(I,J,K) = 0
DO 190 I=1, NR E
NX1 = LGEN(I,1) + 1
NX2 = LGEN(I,2) + 1
NX3 = LGEN(I,3) + JADD
190 JSUB(NX1,NX2,NX3) = I
READ 170, ((LBSC(I,J), J=1,3), I=1, NR BV)
READ 180, ((JQDS(I,J),J=1,4), I=1, NR QDS)
JCUMNR = 0
JBSO = 1 00 000 000 000B $JBSN = 1 000 000B
MSK1 = 7777 000 000B $MSK2 = 777 777B
DO 200 I=1, NR BV
JTS = JSUB(LBSC(I,1)+1,LBSC(I,2)+1,LBSC(I,3)+JADD)
EBSC(I)=EGEN(JTS)
200 KNT(JTS)=JBSO

```

	II = 1	5900
	JHI = 1	6000
	NR INV = 0	6100
210	N CY CUR = JQDS(II,1)	6200
	NRPK = JQDS(II,2)	6300
	MINKNT = JQDS(II,3)	6400
	XMINSIG = JQDS(II,4)	6500
	NTBIGE = NNEWBV = NCOLDBV = NCNEWBV = 0	6600
	INV OVFL0 = 5H	6700
	DO 220 I = 1, NR E	6800
220	BIGSGA(I)=0	6900
		7000
C****	ALLOW THE BASIC SET TO INTERACT WITH ITSELF (LOOP EXTENDS THROUGH	7100
C	STATEMENT 490). THIS IS DONE, FOR THE FIRST CYCLE, BY LETTING	7200
C	EACH VECTOR INTERACT WITH THE FOUR (OR TWO) SYMMETRY-RELATED FORMS	7300
C	OF EACH VECTOR THAT DOES NOT PRECEDE IT IN THE LIST OF BASE	7400
C	VECTORS, TO PRODUCE IN EACH CASE A THIRD VECTOR. IF THE CANONICAL	7500
C	FORM OF THE LATTER IS IN THE E DECK THE VECTOR TRIPLE IS A CON-	7600
C	TRIBUTOR TO THE THIRD VECTOR, WHICH IS EITHER ALREADY IN THE	7700
C	BASIC SET OR IS A POTENTIAL PICKUP. HENCE THE LOCATIONS RESERVED	7800
C	FOR STATISTICS ON THIS THIRD VECTOR IN THE KNT AND SUM A ARRAYS	7900
C	ARE INCREASED BY UNITY AND A RESPECTIVELY. (NOTE THAT IF THE	8000
C	THIRD VECTOR IS NOT THREE-DIMENSIONAL, THEN THERE MAY BE TWO	8100
C	EQUALLY EFFECTIVE SYMMETRY TRANSFORMATIONS WHICH WILL PUT IT INTO	8200
C	CANONICAL FORM.)	8300
C	FOR CYCLES AFTER THE FIRST CYCLE THE PROCEDURE IS SIMILAR EXCEPT	8400
C	THAT ONLY THE PICKUPS DURING THE IMMEDIATELY PREVIOUS CYCLE INTER-	8500
C	ACT WITH THE BASIC SET--SINCE THESE INTERACTIONS YIELD THE ONLY	8600
C	CONTRIBUTORS NOT ALREADY DISCOVERED AND TABULATED.	8700
	DO 490 I=1, NR BV	8800
	IF(I .GT. JHI) JHI = I	8900
	DO 490 J=JHI, NR BV	9000
	JV1 = LBSC(I,1)	9100
	JV2 = LBSC(I,2)	9200
	JV3 = LBSC(I,3)	9300
	JV(1,1) = LBSC(J,1)	9400
	JV(1,2) = LBSC(J,2)	9500
	JV(1,3) = LBSC(J,3)	9600
	JV(2,1)=JV(4,1)=-JV(1,1)	9700
	JV(3,1)=JV(1,1)	9800
	JV(4,2)=JV(1,2)	9900
	JV(2,2)=JV(3,2)=-JV(1,2)	10000
	JV(3,3)=JV(1,3)	10100
	JV(2,3)=JV(4,3)=-JV(1,3)	10200
	JVV(1,1)=JVV(3,1)=-JV1-JV(1,1)	10300
	JVV(2,1)=JVV(4,1)=-JV1+JV(1,1)	10400
	JVV(1,2)=JVV(4,2)=-JV2-JV(1,2)	10500
	JVV(2,2)=JVV(3,2)=-JV2+JV(1,2)	10600
	JVV(1,3)=JVV(3,3)=-JV3-JV(1,3)	10700
	JVV(2,3)=JVV(4,3)=-JV3+JV(1,3)	10800
	IF(JV2 .EQ. 0 .OR. JV(1,2) .EQ. 0 .OR.	10900
X	JV1 .EQ. 0 .AND. JV3 .EQ. 0 .OR.	11000
X	JV(1,1) .EQ. 0 .AND. JV(1,3) .EQ. 0) 230, 240	11100
230	NV = 2	11200
	GO TO 250	11300
240	NV = 4	11400
250	DO 490 K = 1, NV	11500
	IF(JVV(K,1)) 290, 260	11600
260	IF(JVV(K,2) .LT. 0) 270, 280	11700

270	IF(JVV(K,3) .LT. 0) 340, 360	11800
280	IF(JVV(K,3) .LT. 0) 380, 320	11900
290	IF(JVV(K,1) .LT. 0) 300, 310	12000
300	IF(JVV(K,2) .LT. 0) 340, 380	12100
310	IF(JVV(K,2) .LT. 0) 360, 320	12200
320	JTR = 1	12300
	JVC1=JV1	12400
	JVC2=JV2	12500
	JVC3=JV3	12600
	DO 330 KDX=1,3	12700
	JVC(KDX)=JV(K,KDX)	12800
330	JVVC(KDX)=JVV(K,KDX)	12900
	GO TO 400	13000
340	JTR = 2	13100
	JVC1=-JV1	13200
	JVC2=-JV2	13300
	JVC3=-JV3	13400
	DO 350 KDX=1,3	13500
	JVC(KDX)=-JV(K,KDX)	13600
350	JVVC(KDX)=-JVV(K,KDX)	13700
	GO TO 400	13800
360	JTR = 3	13900
	JVC1=JV1	14000
	JVC2=-JV2	14100
	JVC3=JV3	14200
	DO 370 KDX=1,3,2	14300
	JVC(KDX)=JV(K,KDX)	14400
370	JVVC(KDX)=JVV(K,KDX)	14500
	JVC(2)=-JV(K,2)	14600
	JVVC(2)=-JVV(K,2)	14700
	GO TO 400	14800
380	JTR = 4	14900
	JVC1=-JV1	15000
	JVC2=JV2	15100
	JVC3=-JV3	15200
	DO 390 KDX=1,3,2	15300
	JVC(KDX)=-JV(K,KDX)	15400
390	JVVC(KDX)=-JVV(K,KDX)	15500
	JVC(2)=JV(K,2)	15600
	JVVC(2)=JVV(K,2)	15700
400	CONTINUE	15800
	NX1=JVVC(1)+1	15900
	NX2=JVVC(2)+1	16000
	NX3=JVVC(3)+JADD	16100
	IF(NX3 .LE. 0 .OR. NX1 .GT. N1 .OR. NX2 .GT. N2 .OR. NX3 .GT. N3)	16200
X	GO TO 490	16300
	IF(JSUB(NX1,NX2,NX3) .LE. 0) GO TO 490	16400
	IF(K1 .NE. 1) GO TO 410	16500
	IF(JV2 .EQ. 0 .AND. JV(K,2) .EQ. 0 .AND. JVV(K,2) .EQ. 0) GO TO 410	16600
	IF(JV1 .EQ. JV(K,1) .AND. JV2 .EQ. -JV(K,2) .AND.	16700
X	JV3 .EQ. JV(K,3)) GO TO 490	16800
	IF(JV1 .EQ. JVV(K,1) .AND. JV2 .EQ. -JVV(K,2) .AND.	16900
X	JV3 .EQ. JVV(K,3)) GO TO 490	17000
	IF(JV(K,1) .EQ. JVV(K,1) .AND. JV(K,2) .EQ. -JVV(K,2) .AND.	17100
X	JV(K,3) .EQ. JVV(K,3)) GO TO 490	17200
410	EEE = EGEN(JSUB(NX1,NX2,NX3)) * EBSC(I) * EBSC(J)	17300
	A = EEE FCTR * EEE	17400
C	IF NEITHER A COUNT NOR AN OUTPUT OF INVARIANTS IS CALLED FOR,	17500
C	GO TO 460. OTHERWISE CONSTRUCT A COMPOSITE NUMBER TO REPRESENT	17600

```

C THE PRESENT VECTOR TRIPLE. THIS NUMBER WILL CONTAIN THE THREE 17700
C SUBSCRIPTS DENOTING THE LOCATIONS OF THE (CANONICAL FORMS OF THE) 17800
C THREE MATCHING VECTORS--THE THREE SUBSCRIPTS TO APPEAR IN ORDER OF 17900
C INCREASING SIZE FOR UNIQUENESS. IF THIS COMPOSITE NUMBER IS NOT 18000
C ALREADY IN THE JSER ARRAY, INCREASE THE COUNT ON INVARIANTS, AND 18100
C STORE THE COMPOSITE AND A--EXCEPT DO NOT STORE IF JSER IS ALREADY 18200
C FILLED. INSTEAD, SET INV OVFL0 SO THAT HEREAFTER THE CUMULATIVE 18300
C NUMBER OF INVARIANTS PRINTED WILL BE INDICATED AS XXX PLUS. 18400
IF(K1 .NE. 1 .AND. K2 .NE. 1 .AND. K3 .NE. 1) GO TO 460 18500
JS(1) = JSUB(JV1+1,JV2+1,JV3+JADD) 18600
JS(2) = JSUB(JV(1,1)+1,JV(1,2)+1,JV(1,3)+JADD) 18700
JS(3)=JSUB(JVVC(1)+1,JVVC(2)+1,JVVC(3)+JADD) 18800
DO 430 IX=1,2 18900
LO=IX+1 19000
DO 430 JX=LO,3 19100
IF(JS(JX) .LT. JS(IX)) 420, 430 19200
420 JTS=JS(IX) 19300
JS(IX)=JS(JX) 19400
JS(JX)=JTS 19500
430 CONTINUE 19600
JS COMP=100000000B*JS(1)+10000B*JS(2)+JS(3) 19700
DO 440 L=1, NR INV 19800
IF(JS COMP .EQ. JSER(L)) GO TO 460 19900
440 CONTINUE 20000
NR INV = NR INV + 1 20100
IF(NR INV .LE. MAX NR INV) GO TO 450 20200
NR INV = NR INV - 1 20250
INV OVFL0 = 5H PLUS 20300
GO TO 460 20400
450 JSER(NR INV) = JS COMP 20500
A VALS(NR INV) = A 20600
460 NT BIG E = NT BIG E + 1 20700
SUMA(JSUB(NX1,NX2,NX3)) = SUMA(JSUB(NX1,NX2,NX3)) + A 20800
KNT(JSUB(NX1,NX2,NX3)) = KNT(JSUB(NX1,NX2,NX3)) + 1 20900
IF(JVVC(2) .EQ. 0) 470, 490 21000
470 IF(JTR .EQ. 1 .OR. JTR .EQ. 4) 480, 490 21100
480 IF(JVC2.EQ.0.OR.JVC1.EQ.JVC(1).AND.-JVC2.EQ.JVC(2).AND.
X JVC3.EQ.JVC(3)) GO TO 490 21300
JVC2=-JVC2 21400
JVC(2)=-JVC(2) 21500
JTR=100 21600
GO TO 400 21700
490 CONTINUE 21800
21900
C**** OF THE PROSPECTIVE PICKUPS, ELIMINATE THOSE WHICH DO NOT SATISFY 22000
C THE SPECIFIED CRITERIA FOR NUMBER OF CONTRIBUTORS (MIN KNT) AND 22100
C SIGMA A (XMIN SIG). OF THE SURVIVORS TAKE THE QUOTA OF NEW PICK- 22200
C UPS FROM THOSE WITH LARGEST SUM OF A INsofar AS THERE ARE ENOUGH 22300
C (OTHERWISE THE QUOTA FOR THIS CYCLE IS NOT FULLY MET). 22400
JCUMNR = JCUMNR + NTBIGE 22500
DO 560 I=1, N1 22600
DO 560 J=1,N2 22700
DO 560 K=1,N3 22800
JTMP = KNT(JSUB(I,J,K)) 22900
IF(JTMP .LT. JBSO) 500, 510 23000
500 TMP = SUMA(JSUB(I,J,K)) 23100
IF(JTMP .GE. MINKNT .AND. TMP .GE. XMINSIG) 520, 560 23200
510 JTMP = JTMP .AND. MSK2 23300
NCOLDBV=NCOLDBV + JTMP 23400

```

```

GO TO 560
520 SMALL = 1E9
NSMALL = 1
DO 540 L=1,NRPK
IF(BIGSGA(L) .LT. SMALL) 530, 540
530 SMALL = BIGSGA(L)
NSMALL=L
540 CONTINUE
IF(TMP .GT. SMALL) 550, 560
550 BIGSGA(NSMALL) = TMP
LBIG(NSMALL,1)=I
LBIG(NSMALL,2)=J
LBIG(NSMALL,3) = K
560 CONTINUE
DO 570 L=1,NRPK
IF(BIGSGA(L) .EQ. 0) GO TO 580
570 NNEWBV =NNEWBV + 1

C**** ADD THE NEW PICKUPS TO THE BASIC SET.
580 JLO = NRBV
DO 590 I=1,NNEWBV
JLO = JLO + 1
NX1 = LBIG(I,1)
NX2 = LBIG(I,2)
NX3 = LBIG(I,3)
LBSC(JLO,1) = NX1-1
LBSC(JLO,2) = NX2-1
LBSC(JLO,3) = NX3-JADD
EBSC(JLO) = EGEN(JSUB(NX1,NX2,NX3))
JTMP = KNT(JSUB(NX1,NX2,NX3))
NCNEWBV = NCNEWBV + JTMP
590 KNT(JSUB(NX1,NX2,NX3)) = JTMP +JBSN * N CY CUR

C**** PRINT THE RESULTS FOR THE CURRENT CYCLE.
600 FORMAT( 1X*CY*14,3X*OLD BV*15,3X*NEW BV*14,3X,*NEW IN E*16,3X,
X *CUM IN E*17)
610 FORMAT(6X,*CTRB OLD BV*16,6X,*CTRB NEW BV*16,7X,*CTRB BOTH*17)
620 FORMAT(6X,*CUM NR OF COSINE INVARIANTS* 16,A5)
630 FORMAT(/5X*INDICES E MAG. CY AVG A CTRBS SUM A*)
NCTOT = NCOLDBV + NCNEWBV
PRINT 600, N CY CUR, NR BV, N NEW BV, N T BIG E, JCUM NR
PRINT 610, NCOLDBV,NCNEWBV,NCTOT
IF(K1 .EQ. 1 .OR. K2 .EQ. 1 .OR. K3 .EQ. 1) PRINT 620, NR INV,
X INV OVFL0
PRINT 630
640 FORMAT(1X,3I4,F9.4,I5,F8.2,I6,F8.2, 3H **)
650 FORMAT(1X,3I4,F9.4,I5,F8.2,I6,F8.2)
DO 680 K=1, N3
DO 680 I=1, N1
DO 680 J=1, N2
JTS = JSUB(I,J,K)
JTMP=KNT(JTS)
IF(JTMP .LE. JBSN) GO TO 680
JTS1 = JTMP .AND. MSK2
JTS3 = JTMP .AND. MSK1
JTS3 = JTS3 / JBSN
JH1 = I-1
JH2 = J-1
JH3 = K-JADD

```

	TMP = SUMA(JTS)	29400
	TS2 = TMP / JTS1	29500
	IF(JTMP .GE. JBSO) 660, 670	29600
660	PRINT 650, JH1, JH2, JH3, EGEN(JTS), JTS3, TS2, JTS1, TMP	29700
	GO TO 680	29800
670	PRINT 640, JH1, JH2, JH3, EGEN(JTS), JTS3, TS2, JTS1, TMP	29900
	KNT(JTS) = JTMP + JBSO	30000
680	CONTINUE	30100
		30200
C****	GO TO THE NEXT CYCLE IF APPROPRIATE.	30300
	JHI = NRBV + 1	30400
	NR BV = NR BV + N NEW BV	30500
	IF(N NEW BV .GT. 0 .OR. NR PK .EQ. 0) GO TO 700	30600
690	FORMAT(* NO NEW BASE VECTORS*)	30700
	PRINT 690	30800
	GO TO 710	30900
700	IF(II .EQ. NRQDS) GO TO 710	31000
	II = II + 1	31100
	PRINT 705	31200
705	FORMAT(/////)	31300
	GO TO 210	31400
		31500
C****	IF MATCHING VECTOR TRIPLES ARE TO BE OUTPUT, ORDER THE A VALS AND	31600
C	JSER ARRAYS ON A. UNSCRAMBLE EACH ENTRY IN JSER TO YIELD THE SUB-	31700
C	SCRIPT FOR EACH OF THE THREE VECTORS. BY APPLYING SYMMETRY TRANS-	31800
C	FORMATIONS ADJUST THE SIGNS OF THE VECTOR INDICES TO YIELD THREE	31900
C	VECTORS ADDING TO 0. OUTPUT THESE MATCHING TRIPLES BY PRINTER	32000
C	AND/OR PUNCH TOGETHER WITH E AND A DATA.	32100
710	IF(K2 .NE. 1 .AND. K3 .NE. 1) GO TO 840	32200
	IF(K2 .EQ. 1) PRINT 720	32300
720	FORMAT(1H1,11X,*MATCHING VECTOR TRIPLES WHOSE COSINE INVARIANTS WI	32400
	XLL BE NEEDED*//4X*VECTOR 1*6X*VECTOR 2*6X*VECTOR 3*6X*E 1*5X*E 2*	32500
	X5X*E 3*8X*A* 6X*COUNT*)	32600
	NR INV M = NR INV - 1	32700
	DO 740 I=1, NR INV M	32800
	I PL = I + 1	32900
	DO 730 J=I PL, NR INV	33000
	IF(A VALS(I) .GE. A VALS(J)) GO TO 730	33100
	TS = A VALS(I)	33200
	JTS = JSER(I)	33300
	A VALS(I) = A VALS(J)	33400
	JSER(I) = JSER(J)	33500
	A VALS(J) = TS	33600
	JSER(J) = JTS	33700
730	CONTINUE	33800
740	CONTINUE	33900
	DO 830 I=1, NR INV	34000
	JS 1 = JSER(I) / 100000000B	34100
	JS 2 = JSER(I) / 10000B - JS 1 * 10000B	34200
	JS 3 = JSER(I) .AND. 7777B	34300
	JV(1,1)=JV(3,1)=LGEN(JS2,1)	34400
	JV(2,1)=JV(4,1)= -LGEN(JS2,1) +1 -1	34500
	JV(1,2)=JV(4,2)=LGEN(JS2,2)	34600
	JV(2,2)=JV(3,2)= -LGEN(JS2,2) +1 -1	34700
	JV(1,3)=JV(3,3)=LGEN(JS2,3)	34800
	JV(2,3)=JV(4,3)= -LGEN(JS2,3) +1 -1	34900
	DO 780 J=1,4	35000
	JS(1) = -LGEN(JS1,1) - JV(J,1) +1 -1	35100
	JS(2) = -LGEN(JS1,2) - JV(J,2) +1 -1	35200

```

JS(3) = -LGEN(JS1,3) - JV(J,3) +1 -1          35300
JVVC(1) = JS(1)                                35400
JVVC(2) = IABS(JS(2))                          35500
JVVC(3) = JS(3)                                35600
IF(JS(1)) 750, 760, 770                        35700
750 JVVC(1) = -JVVC(1)                          35800
JVVC(3) = -JVVC(3) + 1 - 1                    35900
GO TO 770                                       36000
760 JVVC(3) = IABS(JVVC(3))                     36100
770 IF(JVVC(1) .NE. LGEN(JS3,1) .OR. JVVC(2) .NE. LGEN(JS3,2) .OR.
X   JVVC(3) .NE. LGEN(JS3,3)) GO TO 780        36200
J USE = J                                       36400
GO TO 790                                       36500
780 CONTINUE                                    36600
790 E1 = EGEN(JS1)                              36700
E2 = EGEN(JS2)                                  36800
E3 = EGEN(JS3)                                  36900
EEE=E1*E2*E3                                    37000
A PRNT=EEE FCTR*EEE                             37100
IF(K2 .NE. 1) GO TO 810                        37200
PRINT 800, (LGEN(JS1,IP),IP=1,3),(JV(JUSE,IP),IP=1,3),
X   (JS(IP), IP=1,3),E1,E2,E3,A PRNT,I        37400
800 FORMAT(1X,3(3I4,2X),3F8.4,F10.4,18)       37500
810 IF(K3 .NE. 1) GO TO 830                    37600
PUNCH 820,(LGEN(JS1,IP),IP=1,3),(JV(J USE,IP),IP=1,3),
X   (JS(IP),IP=1,3),E1,E2,E3,EEE,A PRNT     37800
820 FORMAT(9I4,4F8.4,F9.5)                    37900
830 CONTINUE                                    38000

C**** OUTPUT THE FINAL BASIC VECTORS WITH E VALUES IF CALLED FOR--IN A
C   FORMAT SUITABLE FOR LS PHASES INPUT.      38100
840 IF(K4 .NE. 1) GO TO 860                    38200
PUNCH 850,((LBSC(I,J), J=1,3),EBSC(I), I=1, NR BV) 38300
850 FORMAT(3I5,F10.4)                          38400
38500
38600
38700
C**** OUTPUT AN E DECK CONTAINING ONLY THE FINAL BASIC VECTORS WITH
C   E VALUES IF CALLED FOR.                  38800
860 IF(K5 .NE. 1) GO TO 870                    38900
PUNCH 160,((LBSC(I,J), J=1,3),EBSC(I), I=1, NR BV) 39000
870 END                                         39100
39200

```

APPENDIX B  
HYPOTHETICAL EXAMPLE OF A BUILDUP RUN



2. Computer Output Print for the Example

Note that of the five input vectors, three had no contributors in the first cycle; by Cycle 3, only 7 0 4 had no contributors. The column headed "CY" gives the number of the cycle in which each of the vectors was added to the basic set; CY 0 refers to the input.

CY	1	OLD BV	5	NEW BV	1	NEW IN E	6	CUM IN E	6
		CTR8 OLD BV	2	CTR8 NEW BV	1			CTR8 BOTH	3
		CUM NR OF COSINE INVARIANTS	5		5				
	INDICES	E MAG,	CY	AVG A	CTRBS	SUM A			
3	0 -9	2,9307	0	0,00	0	0,00			
7	0 -4	2,0657	0	0,00	0	0,00			
2	1 -1	3,4692	0	0,00	0	0,00			
4	1 -1	2,6041	1	10,46	1	10,46	**		
2	0 0	5,3961	0	12,77	1	12,77			
4	0 0	2,0433	0	12,77	1	12,77			

CY	2	OLD BV	6	NEW BV	3	NEW IN E	5	CUM IN E	11
		CTR8 OLD BV	6	CTR8 NEW BV	5			CTR8 BOTH	11
		CUM NR OF COSINE INVARIANTS	7		7				
	INDICES	E MAG,	CY	AVG A	CTRBS	SUM A			
1	0 -9	2,8268	2	9,59	1	9,59	**		
3	0 -9	2,9307	0	0,00	0	0,00			
7	0 -4	2,0657	0	0,00	0	0,00			
2	1 -1	3,4692	0	10,46	1	10,46			
4	1 -1	2,6041	1	10,46	1	10,46			
6	1 -1	1,6771	2	3,80	2	7,61	**		
2	0 0	5,3961	0	11,23	3	33,69			
4	0 0	2,0433	0	12,77	1	12,77			
0	1 1	2,0013	2	5,16	2	10,33	**		

CY	3	OLD BV	9	NEW BV	0	NEW IN E	14	CUM IN E	25
		CTR8 OLD BV	25	CTR8 NEW BV	0			CTR8 BOTH	25
		CUM NR OF COSINE INVARIANTS	7		7				
	INDICES	E MAG,	CY	AVG A	CTRBS	SUM A			
1	0 -9	2,8268	2	9,59	1	9,59			
3	0 -9	2,9307	0	9,59	1	9,59			
7	0 -4	2,0657	0	0,00	0	0,00			
2	1 -1	3,4692	0	7,02	3	21,05			
4	1 -1	2,6041	1	5,93	3	17,80			
6	1 -1	1,6771	2	3,80	2	7,61			
2	0 0	5,3961	0	8,68	8	69,48			
4	0 0	2,0433	0	4,49	5	22,44			
0	1 1	2,0013	2	5,16	2	10,33			

MATCHING VECTOR TRIPLES WHOSE COSINE INVARIANTS WILL BE NEEDED

VECTOR 1	VECTOR 2	VECTOR 3	E 1	E 2	E 3	A	COUNT
2 0 0	2 0 0	-4 0 0	5,3961	5,3961	2,0433	12,7680	1
2 0 0	2 1 -1	-4 -1 1	5,3961	3,4692	2,6041	10,4616	2
1 0 -9	2 0 0	-3 0 9	2,8268	5,3961	2,9307	9,5935	3
0 1 1	-2 0 0	2 -1 -1	2,0013	5,3961	3,4692	8,0399	4
2 0 0	4 1 -1	-6 -1 1	5,3961	2,6041	1,6771	5,0574	5
2 1 -1	4 0 0	-6 -1 1	3,4692	2,0433	1,6771	2,5512	6
0 1 1	-4 0 0	4 -1 -1	2,0013	2,0433	2,6041	2,2852	7

### 3. Listing of the Punched Cards as Output by the Computer for the Example (all optional)

The first seven cards comprise the invariants deck; each card gives a matching vector triple, the three  $|E|$  values, the product of the three  $|E|$ 's, and the A value. This deck may serve as input to the cosine-determining program.

The nine cards with four quantities each give  $h, k, l, |E|$  for the vectors in the basic set at the end of the run. This deck, when phase information is added, will serve as a combined E and phases deck for input to LS Phases.

The final two cards comprise an E deck containing only the final set of base vectors in the order in which added to the set and in a format suitable for use in rerunning Buildup to punch only those invariants needed for LS Phases.

2	0	0	2	0	0	-4	0	0	5.3961	5.3961	2.0433	59.4966	12.76797						
2	0	0	2	1	-1	-4	-1	1	5.3961	3.4692	2.6041	48.7491	10.46157						
1	0	-9	2	0	0	-3	0	9	2.8268	5.3961	2.9307	44.7040	9.59348						
0	1	1	-2	0	0	2	-1	-1	2.0013	5.3961	3.4692	37.4646	8.03991						
2	0	0	4	1	-1	-6	-1	1	5.3961	2.6041	1.6771	23.5666	5.05739						
2	1	-1	4	0	0	-6	-1	1	3.4692	2.0433	1.6771	11.8883	2.55123						
0	1	1	-4	0	0	4	-1	-1	2.0013	2.0433	2.6041	10.6488	2.28524						
2	0	0							5.3961										
2	1	-1							3.4692										
3	0	-9							2.9307										
4	0	0							2.0433										
7	0	-4							2.0657										
4	1	-1							2.6041										
0	1	1							2.0013										
1	0	-9							2.8268										
6	1	-1							1.6771										
2	0	0	5.3961	2	1	-1	3.4692	3	0	-9	2.9307	4	0	0	2.0433	7	0	-4	2.0657
4	1	-1	2.6041	0	1	1	2.0013	1	0	-9	2.8268	6	1	-1	1.6771				

APPENDIX C  
LISTING OF PROGRAM LS PHASES

	PROGRAM LS PHASES	100
	BANK,(0),/111/, /222/, /444/, LS PHASES	200
	COMMON/111/JB1(300),JB2(300),JB3(300),E(300),PAR TBL(300),	300
X	PHI FXD(300), PHI(300), SGA(300), NR CTRB(300), SG WK(300),	400
X	S1(300), S2(300), C1(300), C2(300), C(300)	500
	COMMON/222/ SMRY PRNT(300,25)	600
	COMMON/333/ JSER(9000), CK ARA(9000), WK ARA(9000)	700
	COMMON/444/ JSUB(12,29,18)	800
	DIMENSION NR OUT TBL(100),JV TRIP(9), JV(4,3),JVV(4,3),NDX(3),	900
X	JS(3), JVC(3), JVVC(3),RTS(3,2), NR CY PT(25), MSNG SER(100)	1000
200	FORMAT(715, F10)	1100
205	FORMAT(1615)	1200
210	FORMAT(315,F10,4,2F15)	1300
215	FORMAT(3(3I3,1X),24X,F9,5,F6,F8,4)	1400
		1500
C****	BEGIN THE READ OF DATA AND INITIALIZE--	1600
C	(1) COMPUTE UPPER BOUNDS ON THE NUMBER OF DISTINCT VALUES FOR EACH	1700
C	OF THE THREE MILLER INDICES. THESE WILL GIVE RANGES ON THE SUB-	1800
C	SCRIPTS TO BE USED FOR THE JSUB ARRAY. COMPUTE ALSO JADD, WHICH	1900
C	WILL BE ADDED TO THE THIRD INDEX TO ENSURE A POSITIVE RESULT FOR	2000
C	SUBSCRIPTING.	2100
C	(2) FOR EACH VECTOR IN THE FINAL BASIC SET, USE H K L AS THE	2200
C	BASIS FOR SUBSCRIBING TO STORE IN JSUB ARRAY THE SUBSCRIPT INDI-	2300
C	CATING WHERE THE DATA ON THIS VECTOR WILL BE FOUND IN THE VARIOUS	2400
C	ARRAYS IN COMMON BLOCKS /111/ AND /222/.	2500
240	READ 200, NR CY TOT, NR V ORIG, MILLR 1, MILLR2, MILLR3 LO,	2600
X	MILLR3 HI, N COL SMRY, EEE FCTR	2700
	N1 = MILLR1 + 1	2800
	N2 = MILLR2 + 1	2900
	N3 = MILLR3 HI - MILLR3 LO + 1	3000
	JADD = -MILLR3 LO + 1	3100
	READ 205, (NR OUT TBL(I), I=1, NR CY TOT)	3200
	NR V FIN = NR OUT TBL(NR CY TOT)	3300
	READ 210, (JB1(I),JB2(I),JB3(I),E(I),PHI FXD(I), PHI(I),I = 1,	3400
X	NR V FIN)	3500
	DO 250 I=1, N1	3600
	DO 250 J=1, N2	3700
	DO 250 K=1, N3	3800
250	JSUB(I,J,K) = 0	3900
	DO 255 I=1, NR V FIN	4000
255	JSUB(JB1(I)+1,JB2(I)+1,JB3(I)+JADD) = I	4100
		4200
C****	READ INVARIANTS TO COMPLETE DATA INPUT. AFTER THE READ OF EACH	4300
C	CARD, VERIFY THAT EACH OF THE THREE VECTORS IS IN THE E DECK. IF SO	4400
C	CONSTRUCT A COMPOSITE NUMBER TO REPRESENT THE PRESENT VECTOR	4500
C	TRIPLE. THIS NUMBER WILL CONTAIN THE THREE SUBSCRIPTS DENOTING	4600
C	THE LOCATIONS OF THE (CANONICAL FORMS OF THE) THREE MATCHING	4700
C	VECTORS--THE THREE SUBSCRIPTS TO APPEAR IN ORDER OF INCREASING	4800
C	SIZE FOR UNIQUENESS. STORE THE COMPOSITE NUMBER IN JSER. STORE IN	4900
C	CK ARA THE COSINE C SUB K. STORE IN WK ARA THE WEIGHT W SUB K	5000
C	ASSOCIATED WITH THE COSINE VALUE--THE PRODUCT OF A AND THE SQUARE	5100
C	ROOT OF THE NUMBER OF CONTRIBUTORS.	5200
	I = NR CK RD = 0	5300
260	READ 215, (JVTRIP(K),K=1,9),WK ARA TS, XN, CK ARA TS	5400
	IF(JV TRIP(1) .GE. 900) GO TO 275	5500
	NR CK RD = NR CK RD + 1	5600
	DO 264 K=1,3	5700
	DO 261 L=1,3	5800
261	NDX(L) = JV TRIP(3*K-(3-L))	5900

	IF(NDX(2) .LT. 0) NDX(2) = -NDX(2)	6000
	IF(NDX(1)) 262, 263, 2635	6100
262	NDX(1) = -NDX(1)	6200
	NDX(3) = -NDX(3)	6300
	GO TO 2635	6400
263	NDX(3) = IABS(NDX(3))	6500
2635	IF(NDX(1) .GT. MILLR1 .OR. NDX(2) .GT. MILLR2 .OR. NDX(3) .LT. X MILLR3 LO .OR. NDX(3) .GT. MILLR3 HI) GO TO 260	6600
	JS(K) = JSUB(NDX(1)+1, NDX(2)+1, NDX(3)+JADD)	6800
	IF(JS(K) .EQ. 0) GO TO 260	6900
264	CONTINUE	7000
	DO 266 IX=1,2	7100
	LO = IX+1	7200
	DO 266 JX=LO,3	7300
	IF(JS(JX) .LT. JS(IX)) 265, 266	7400
265	JTS = JS(IX)	7500
	JS(IX) = JS(JX)	7600
	JS(JX) = JTS	7700
266	CONTINUE	7800
	JS COMP = 100000000B*JS(1)+10000B*JS(2)+JS(3)	7900
	DO 268 M=1, I	8000
	IF(JS COMP .EQ. JSER(M)) GO TO 260	8100
268	CONTINUE	8200
	I = I + 1	8300
	JSER(I) = JS COMP	8400
	WK ARA(I) = WK ARA TS * SORT(XN)	8500
	CK ARA(I) = CK ARA TS	8600
	IF(CK ARA(I) .GT. 1.0) CK ARA(I) = 1.0	8700
	IF(CK ARA(I) .LT. -1.0) CK ARA(I) = -1.0	8800
	GO TO 260	8900
275	NR CK = I	9000
		9100
C****	ORDER JSER, CK ARA, AND WK ARA ON JSER.	9200
	I UP = NR CK - 1	9300
	DO 280 I=1,IUP	9400
	JLO = I+1	9500
	DO 280 J=J LO, NR CK	9600
	IF(JSER(I) .LE. JSER(J)) GO TO 280	9700
	JTS = JSER(I)	9800
	TSC = CK ARA(I)	9900
	TSW = WK ARA(I)	10000
	JSER(I) = JSER(J)	10100
	CK ARA(I) = CK ARA(J)	10200
	WK ARA(I) = WK ARA(J)	10300
	JSER(J) = JTS	10400
	CK ARA(J) = TSC	10500
	WK ARA(J) = TSW	10600
280	CONTINUE	10700
		10800
C****	FOR EACH VECTOR IN THE E DECK CHECK THE PARITY OF K. FOR ODD	10900
C	PARITY STORE PI IN PAR TBL. FOR EVEN PARITY STORE 0. (USED LATER	11000
C	FOR SYMMETRY TRANSFORMATIONS REQUIRING THE ADDITION OF PI TO THE	11100
C	PHASE IF K IS ODD.)	11200
	DO 330 I=1, NR V FIN	11300
	IF(JB2(I)/2*2 .NE. JB2(I)) 310, 320	11400
310	PAR TBL(I) = 3.14159 2654	11500
	GO TO 330	11600
320	PAR TBL(I) = 0	11700
330	CONTINUE	11800

C****	FILL THE SMRY PRNT ARRAY WITH 99900000.	11900
	NR COL USD = MINO(NR CY TOT+1, N COL SMRY)	12000
	DO 335 I=1, NR V FIN	12100
	DO 335 J=1, NR COL USD	12200
335	SMRY PRNT(I,J) = 99900000.	12300
		12400
		12500
C****	STORE INPUT BASE VECTORS IN SMRY PRNT. PRINT INPUT H K L E AND	12600
C	PHASE INFO. CALL REV 2 TO SET UP ITS TRIG TABLE FOR LATER USE BY	12700
C	REV 2 (ENTRY SRCH) IN FINDING VALUE(S) OF PHI WHICH MINIMIZE	12800
C	CAPITAL PHI.	12900
340	FORMAT(2X,*INDICES E MAG. INPUT PHI FORCING INFO*)	13000
345	FORMAT(3I3, F9.4, F11.4, F15.9)	13100
350	FORMAT(3I3, F9.4, 11X, F15.9)	13200
	DO 355 I=1, NR V ORIG	13300
355	SMRY PRNT(I, NR COL USD) = 100.0*PHI(I)	13400
	PRINT 340	13500
	DO 356 K=1, N3	13600
	DO 356 I=1, N1	13700
	DO 356 J=1, N2	13800
	JP = JSUB(I,J,K)	13900
	IF(JP .EQ. 0) GO TO 356	14000
	IF(JP .LE. NR V ORIG) 3555, 3556	14100
3555	PRINT 345,JB1(JP),JB2(JP),JB3(JP),E(JP),PHI(JP), PHI FXD(JP)	14200
	GO TO 356	14300
3556	PRINT 350, JB1(JP),JB2(JP),JB3(JP),E(JP),PHI FXD(JP)	14400
356	CONTINUE	14500
	NR MSNG = 0	14600
	NR V IN = NR V ORIG	14700
	CALL REV2	14800
	DO 360 I=1, N1	14900
	DO 360 J=1, N2	15000
	DO 360 K=1, N3	15100
360	JSUB(I,J,K) = 0	15200
		15300
C****	EXCEPT FOR FINAL OUTPUTS, RMNDR OF PROGRAM IS N-LOOP FOR CYCLES	15400
C	(THROUGH STATEMENT NR 978).	15500
	DO 978 N=1, NR CY TOT	15600
	NR V OUT = NR OUT TBL(N)	15700
	DO 370 I=1, NR V OUT	15800
	JSUB(JB1(I)+1,JB2(I)+1, JB3(I)+JADD) = I	15900
370	NR CTRB(I) = SG WK(I)=S1(I)=S2(I)=C1(I)=C2(I)=C(I)=SGA(I)=0	16000
		16100
C****	ENTER I AND J LOOPS, WHEREIN I AND J REPRESENT 2 INTERACTING VCTRS	16200
C	LET EACH VECTOR IN THE LIST OF BASE VECTORS INTERACT WITH THE FOUR	16300
C	(OR TWO) SYMMETRY-RELATED FORMS OF EACH VECTOR THAT DOES NOT PRE-	16400
C	CEDE IT IN THE LIST, TO PRODUCE IN EACH CASE A THIRD VECTOR. IF	16500
C	THE LATTER IS IN THE E DECK THE VECTOR TRIPLE IS A CONTRIBUTOR TO	16600
C	THE THIRD VECTOR--WHICH BY ITS PRESENCE IN THE E DECK IS GUARAN-	16700
C	TEED TO BE ALREADY IN THE BASIC SET OR A FUTURE PICKUP. THEN LO-	16800
C	CATIONS RESERVED FOR STATISTICS ON THIS THIRD VECTOR ARE UPDATED.	16900
C	STATISTICS INCLUDE SUMS ACCUMULATING FOR USE IN COMPUTING	17000
C	CONSTANTS IN THE DEFINITION OF CAPITAL PHI.	17100
	DO 952 I=1, NR V IN	17200
	DO 952 J=1, NR V IN	17300
	JV1 = JB1(I)	17400
	JV2 = JB2(I)	17500
	JV3 = JB3(I)	17600
	JV(1,1) = JB1(J)	17700

	JV(1,2) = JB2(J)	17800
	JV(1,3) = JB3(J)	17900
	JV(3,1)=JV(1,1)	18000
	JV(2,1)=JV(4,1)=-JV(1,1)	18100
	JV(4,2)=JV(1,2)	18200
	JV(2,2)=JV(3,2)=-JV(1,2)	18300
	JV(2,3)=JV(4,3)=-JV(1,3)	18400
	JVV(1,1)=JVV(3,1)=-JV1-JV(1,1)	18500
	JV(3,3)=JV(1,3)	18600
	JVV(2,1)=JVV(4,1)=-JV1+JV(1,1)	18700
	JVV(1,2)=JVV(4,2)=-JV2-JV(1,2)	18800
	JVV(2,2)=JVV(3,2)=-JV2+JV(1,2)	18900
	JVV(1,3)=JVV(3,3)=-JV3-JV(1,3)	19000
	JVV(2,3)=JVV(4,3)=-JV3+JV(1,3)	19100
	IF(JV2 .EQ. 0 .OR. JV(1,2) .EQ. 0 .OR.	19200
	X JV1 .EQ. 0 .AND. JV3 .EQ. 0 .OR.	19300
	X JV(1,1) .EQ. 0 .AND. JV(1,3) .EQ. 0) 910,912	19400
910	NV = ?	19500
	GO TO 914	19600
912	NV = 4	19700
		19800
C****	ENTER K LOOP FOR 4(OR 2) VARIANTS OF J-TH VECTOR.	19900
914	DO 952 K = 1,NV	20000
	IF(JVV(K,1)) 922,916	20100
916	IF(JVV(K,2) .LT. 0) 918,920	20200
918	IF(JVV(K,3) .LT. 0) 930,932	20300
920	IF(JVV(K,3) .LT. 0) 936,928	20400
922	IF(JVV(K,1) .LT. 0) 924, 926	20500
924	IF(JVV(K,2) .LT. 0) 930,936	20600
926	IF(JVV(K,2) .LT. 0) 932,928	20700
928	JTR = 1	20800
	PHI I = PHI(I)	20900
	PHI J = PHI(J)	21000
	JVC1=JV1	21100
	JVC2=JV2	21200
	JVC3=JV3	21300
	DO 929 KDX=1,3	21400
	JVC(KDX)=JV(K,KDX)	21500
929	JVVC(KDX)=JVV(K,KDX)	21600
	GO TO 938	21700
930	JTR=2	21800
	PHI I = -PHI(I)	21900
	PHI J = -PHI(J)	22000
	JVC1=-JV1	22100
	JVC2=-JV2	22200
	JVC3=-JV3	22300
	DO 931 KDX=1,3	22400
	JVC(KDX)=-JV(K,KDX)	22500
931	JVVC(KDX)=-JVV(K,KDX)	22600
	GO TO 938	22700
932	JTR=3	22800
	PHI I = -PHI(I) + PAR TBL(I)	22900
	PHI J = -PHI(J) + PAR TBL(J)	23000
	JVC1=JV1	23100
	JVC2=-JV2	23200
	JVC3=JV3	23300
	DO 934 KDX=1,3,2	23400
	JVC(KDX)=JV(K,KDX)	23500
934	JVVC(KDX)=JVV(K,KDX)	23600

	JVC(2)=-JV(K,2)	23700
	JVVC(2)=-JVVC(K,2)	23800
	GO TO 938	23900
936	JTR=4	24000
	PHI I = PHI(I) + PAR TBL(I)	24100
	PHI J = PHI(J) + PAR TBL(J)	24200
	JVC1=-JV1	24300
	JVC2=JV2	24400
	JVC3=-JV3	24500
	DO 937 KDX=1,3,2	24600
	JVC(KDX)=-JV(K,KDX)	24700
937	JVVC(KDX)=-JVVC(K,KDX)	24800
	JVC(2)=JV(K,2)	24900
	JVVC(2)=JVVC(K,2)	25000
938	CONTINUE	25100
	NX1=JVVC(1)+1	25200
	NX2=JVVC(2)+1	25300
	NX3=JVVC(3)+JADD	25400
	IF(NX3 .LE. 0 .OR. NX1 .GT. N1 .OR. NX2 .GT. N2 .OR. NX3 .GT. N3)	25500
X	GO TO 952	25600
C	COMPUTE THE COMPOSITE NUMBER REPRESENTING THE PRESENT VECTOR	25700
C	TRIPLE. SEARCH FOR IT IN THE JSER LIST OF SUCH COMPOSITES BY SUC-	25800
C	CESSIVE HALVING OF THE LIST. IF NOT FOUND THE CORRESPONDING	25900
C	COSINE INVARIANT IS MISSING.	26000
	JS(1) = JSH = JSUB(NX1,NX2,NX3)	26100
	IF(JS .EQ. 0) GO TO 952	26200
	JS(2) = I	26300
	JS(3) = J	26400
	DO 9382 IX = 1,2	26500
	LO = IX + 1	26600
	DO 9382 JX = LO, 3	26700
	IF(JS(JX) .LT. JS(IX)) 9381, 9382	26800
9381	JTS = JS(IX)	26900
	JS(IX) = JS(JX)	27000
	JS(JX) = JTS	27100
9382	CONTINUE	27200
	JS COMP = 100000000B * JS(1) + 10000B*JS(2) + JS(3)	27300
	IF(JS COMP .LT. JSER(1) .OR. JS COMP .GT. JSER(NR CK)) GO TO 941	27400
	LO SUB = 1	27500
	JUP SUB = NR CK	27600
	DO 9405 II=1,20	27700
	MID SUB = (LO SUB + JUP SUB) / 2	27800
	IF(JS COMP - JSER(MID SUB)) 939, 940, 9395	27900
939	JUP SUB = MID SUB - 1	28000
	IF(LO SUB .LE. JUP SUB) 9405, 941	28100
9395	LO SUB = MID SUB + 1	28200
	IF(LO SUB .LE. JUP SUB) 9405, 941	28300
940	CK = CK ARA(MID SUB)	28400
	WK = WK ARA(MID SUB)	28500
	GO TO 942	28600
9405	CONTINUE	28700
941	IF(NR MSNG .EQ. 100) GO TO 952	28800
	DO 9411 ISP=1, NR MSNG	28900
	IF(JS COMP .EQ. MSNG SER(ISP)) GO TO 952	29000
9411	CONTINUE	29100
	NR MSNG = NR MSNG + 1	29200
	MSNG SER(NR MSNG) = JS COMP	29300
	GO TO 952	29400
942	IF(K .EQ. 1) GO TO 945	29500

```

IF(K .EQ. 2) GO TO 943
IF(K .EQ. 3) GO TO 944
ARG = PHI I + PHI J + PAR TBL(J)
GO TO 946
943 ARG = PHI I - PHI J
GO TO 946
944 ARG = PHI I - PHI J + PAR TBL(J)
GO TO 946
945 ARG = PHI I + PHI J
946 SIN ARG = SIN(ARG)
COS ARG = COS(ARG)
NR CTRB(JS H) = NR CTRB(JSH) + 1
SG WK(JS H) = SG WK(JS H) + WK
S1(JSH)=S1(JSH)+WK*CK*SINARG
S2(JSH)=S2(JSH)+WK*2.0*SINARG*COSARG
C1(JSH)=C1(JSH)+WK*CK*COSARG
C2(JSH)=C2 (JSH)+WK*(1.0-2.0*SINARG*SINARG)
C(JSH)=C(JSH)+WK*(0.5+CK*CK)
SGA(JSH)=SGA(JSH)+EEE FCTR*(I)*E(J)*E(JSH)
IF(JVVC(2) .EQ. 0) 950,952
950 IF(JTR .EQ. 1 .OR. JTR .EQ. 4) 951,952
951 IF(JVC2.EQ.0.OR.JVC1.EQ.JVC(1).AND.-JVC2.EQ.JVC(2).AND.
X JVC3.EQ.JVC(3)) GO TO 952
IF(JTR .NE. 1) GO TO 9514
JTR = 3
PHI I = -PHI (I) + PAR TBL(I)
PHI J = -PHI(J) + PAR TBL(J)
GO TO 942
9514 JTR = 2
PHI I = -PHI(I)
PHI J = -PHI(J)
GO TO 942
952 CONTINUE
C END K LOOP. END I AND J LOOPS.

C**** NOW READY TO DO ALL THE PHI COMPUTATIONS FOR THE CURRENT CYCLE.
954 PRINT 956, N
956 FORMAT(///* CYCLE*I3/5X,
X*INDICES E MAG. CTRBS SUM A AVG A*,4X,
X2(*PHI RMS OF RESIDUALS *))
958 FORMAT(3X,3I3,F9.4,I5,F8.2,F8.3,2(F9.4,2X,F12.10,3X))
9582 FORMAT(1X,2H**,3I3,F9.4,I5,F8.2,F8.3,2(F9.4,2X,F12.10,3X))
CY MERIT = SUM SG A = HZL MERIT = NR HZL = S HZL SGA = 0
DO 972 K=1,N3
DO 972 I=1,N1
DO 972 J=1,N2
L = JSUB(I,J,K)
IF(L .EQ. 0) GO TO 972
SUM SG A = SUM SG A + SG A(L)
IF(NR CTRB(L) .EQ. 0) 9584, 9588
9584 A AV = 0
RTS(1,1) = 99000.
RMS1 = 100.0
GO TO 969
9588 A AV = SG A(L) / NR CTRB(L)
CALL SRCH (S1(L)/SGWK(L),S2(L)/SGWK(L),C1(L)/SGWK(L),
X C2(L)/SGWK(L),C(L)/SGWK(L),KNT,RTS)
959 MULT SOLN = 1
C**** IF THERE IS A MULTIPLE SOLUTION TO THE MINIMIZING PROBLEM (I.E.

```

29600  
29700  
29800  
29900  
30000  
30100  
30200  
30300  
30400  
30500  
30600  
30700  
30800  
30900  
31000  
31100  
31200  
31300  
31400  
31500  
31600  
31700  
31800  
31900  
32000  
32100  
32200  
32300  
32400  
32500  
32600  
32700  
32800  
32900  
33000  
33100  
33200  
33300  
33400  
33500  
33600  
33700  
33800  
33900  
34000  
34100  
34200  
34300  
34400  
34500  
34600  
34700  
34800  
34900  
35000  
35100  
35200  
35300  
35400

```

C      TWO PHI VALUES PRODUCING MINIMA OF CAPITAL PHI) THE PROGRAM WILL      35500
C      SELECT ONE--NORMALLY THE ONE YIELDING THE SMALLER CAP PHI IN THE      35600
C      USUAL CASE OF UNEQUAL CAP PHI. IF THE OTHER CHOICE IS DESIRED FOR    35700
C      A PARTICULAR CYCLE (INDEX N) AND A PARTICULAR VECTOR (INDEX L)      35800
C      INSERT FORTRAN STATEMENT HERE AS EXPLAINED IN REPORT.              35900
      IF(KNT .EQ. 1) GO TO 966
      IF(MULT SOLN .EQ. 1) 960, 962
960    TS1 = RTS(1,2)
      TS2 = RTS(2,2)
      GO TO 964
962    TS1 = RTS(2,2)
      TS2 = RTS(1,2)
964    IF(TS1 .LT. TS2) GO TO 966
      IF(TS1 .EQ. TS2 .AND. MULT SOLN .EQ. 2) GO TO 966
      TS1 = RTS(1,1)
      TS2 = RTS(1,2)
      RTS(1,1) = RTS(2,1)
      RTS(1,2) = RTS(2,2)
      RTS(2,1) = TS1
      RTS(2,2) = TS2
966    PHI(L) = RTS(1,1)
      IF(PHI FXD(L) .GT. 5.0) GO TO 9662
      PHI(L) = PHI FXD(L)
      IF(JB2(L) .NE. 0) GO TO 9666
      TS = ABSF(RTS(1,1) - PHI(L))
      IF(TS .GT. 3.141592654) TS = 6.283185307 - TS
      GO TO 9665
9662   IF(JB2(L) .NE. 0) GO TO 9666
      TS = ABSF(RTS(1,1))
      IF(TS .GE. 1.570796327) TS = 3.141592654 - TS
9665   HZL MERIT = HZL MERIT + SGA(L)*TS*TS
      S HZL SGA = S HZL SGA + SGA(L)
      NR HZL = NR HZL + 1
9666   IF(KNT .EQ. 1) GO TO 968
      IF(RTS(1,2) .LT. 0 .AND. RTS(1,2) .GT. -0.00000005) RTS(1,2)=0
      IF(RTS(2,2) .LT. 0 .AND. RTS(2,2) .GT. -0.00000005) RTS(2,2)=0
      RMS1 = SQRT (RTS(1,2))
      RMS2 = SQRT (RTS(2,2))
      IF(L .GT. NR V IN) GO TO 967
      PRINT 958, JB1(L),JB2(L),JB3(L),E(L),NRCTRB(L),SGA(L),AAV,
X     RTS(1,1), RMS1,RTS(2,1),RMS2
      GO TO 970
967    PRINT 9582, JB1(L),JB2(L),JB3(L),E(L),NRCTRB(L),SGA(L),AAV,
X     RTS(1,1), RMS1,RTS(2,1),RMS2
      GO TO 970
968    IF(RTS(1,2) .LT. 0 .AND. RTS(1,2) .GT. -0.00000005) RTS(1,2)=0
      RMS1=SQRTF(RTS(1,2))
      IF(L .GT. NR V IN) 9689, 969
9689   PRINT 9582, JB1(L),JB2(L),JB3(L),E(L),NRCTRB(L),SGA(L),AAV,
X     RTS(1,1), RMS1
      GO TO 970
969    PRINT 958, JB1(L),JB2(L),JB3(L),E(L),NRCTRB(L),SGA(L),AAV,
X     RTS(1,1), RMS1
970    CY MERIT = CY MERIT + SGA(L)*RTS(1,2)
      JTS = NR CY TOT + 1 - N
      IF(JTS .GT. NR COL USD - 1) GO TO 972
      SMRY PRNT(L,JTS) = 100.0 * RTS(1,1)
972    CONTINUE
      CY MERIT = SQRT (CYMERIT/SUM SG A)

```

```

HZL MERIT = SQRT (HZL MERIT/S HZL SGA)
974 CONTINUE
976 FORMAT(1H0, *CYCLE FIG OF MERIT*F9.6)
PRINT 976, CY MERIT
977 FORMAT(1X,*HOL FIG OF MERIT *, F9.6* (*12* HOL CONTRIBUTORS*))
PRINT 977, HZL MERIT, NR HZL
978 NR V IN = NR V OUT

C**** PRINT A SUMMARY OF PHASES COMPUTED.
979 FORMAT(1H1,54X,*SUMMARY OF COMPUTED PHASES*/11X,
X*COLUMN HEADINGS GIVE CYCLE NUMBER (CYCLE 0 DENOTES INPUT), AND
XBULAR ENTRIES GIVE PHI (RADIAN) MULTIPLIED BY 100*/
9795 FORMAT(10X,25(3X,I2))
9796 FORMAT(1H0)
980 FORMAT(1X,I2,2I3,2X,25F5)
PRINT 979
NR CY PT(1) = NR CY TOT
DO 981 I=2, NR COL USD
981 NR CY PT(I) = NR CY PT(I-1) - 1
NR CY PT(NR COL USD) = 0
PRINT 9795, (NR CY PT(I), I=1, NR COL USD)
PRINT 9796
DO 986 K=1, N3
DO 986 I=1, N1
DO 986 J=1, N2
L = JSUB(I,J,K)
IF(L .EQ. 0) GO TO 986
N = 0
DO 982 M=1, NR COL USD
IF(SMRY PRNT(L,M) .GE. 900000000,) GO TO 984
982 N = N + 1
984 PRINT 980, JB1(L),JB2(L),JB3(L),(SMRY PRNT(L,NPT),NPT=1,N)
986 CONTINUE

C**** PRINT SUMMARY RE INVARIANTS--NUMBER OF COSINE INVARIANTS READ,
C NUMBER USABLE AND USED, AND A LIST OF ANY MISSING--THE LATTER
C SPECIFIED BY GIVING CANONICAL FORMS OF THE THREE MATCHING VECTORS
C AND VALUE OF A.
988 FORMAT(1X,////,I7* COSINES WERE READ*/1X,I6
X * COSINES ENTERED INTO THE COMPUTATION*)
PRINT 988, NR CK RD, NR CK
IF(NR MSNG .EQ. 0) GO TO 998
DO 990 I=1, NR MSNG
IF(I .EQ. 1) PRINT 989
989 FORMAT(1H0,*COSINES CORRESPONDING TO THESE VECTOR TRIPLES (FOLLOWE
XD BY A VALUES) WERE MISSING FROM THE INPUT*/18X,* (IF MORE THAN 100
X ARE MISSING, ONLY THE FIRST 100 ARE PRINTED)*)
JS(1) = MSNG SER(I) / 1000000000B
JS(2) = MSNG SER(I) .AND. 777700000B
JS(2) = JS(2) / 100000B
JS(3) = MSNG SER(I) .AND. 7777B
A PRNT = EEE FCTR * E(JS(1))*E(JS(2))*E(JS(3))
9895 FORMAT(3(3I3,5X),F9.6)
PRINT 9895,JB1(JS(1)),JB2(JS(1)),JB3(JS(1)),JB1(JS(2)),JB2(JS(2)),
X JB3(JS(2)),JB1(JS(3)),JB2(JS(3)),JB3(JS(3)), A PRNT
990 CONTINUE
998 END
SUBROUTINE REV 2(S1,S2,C1,C2,CNST,KNT,RTS)
C**** ASSUME A SINGLE PRELIMINARY CALL HAS BEEN MADE TO REV 2 TO SET UP

```



APPENDIX D  
HYPOTHETICAL EXAMPLE OF AN LS PHASES RUN



2. Computer Output for the Example

Asterisks in the summary of computed phases (see next page) indicate that  $\phi$ , in the input basic set, cannot yet be recomputed because it has no contributors. By Cycle 3, only  $\phi_{70\bar{4}}$  is in this category.

The print of missing cosines exhibits the fact that one is missing; this particular invariant could have been computed, but was omitted for illustration.

INDICES	E MAG.	INPUT PHI	FORCING INFO
1 0 -9	2,8268		9,000000000
3 0 -9	2,9307	0,0000	0,000000000
7 0 -4	2,0657	0,0000	0,000000000
2 1 -1	3,4692	0,0000	0,000000000
4 1 -1	2,6041		9,000000000
6 1 -1	1,6771		9,000000000
2 0 0	5,3961	3,1416	9,000000000
4 0 0	2,0433	0,0000	9,000000000
0 1 1	2,0013		9,000000000

CYCLE 1

INDICES	E MAG.	CTRBS	SUM A	AVG A	PHI	RMS OF RESIDUALS	PHI	RMS OF RESIDUALS
3 0 -9	2,9307	0	0,00	0,000	*****	*****		
7 0 -4	2,0657	0	0,00	0,000	*****	*****		
2 1 -1	3,4692	0	0,00	0,000	*****	*****		
** 4 1 -1	2,6041	1	10,46	10,462	-3,1404	0,000000000		
2 0 0	5,3961	0	0,00	0,000	*****	*****		
4 0 0	2,0433	0	0,00	0,000	*****	*****		

CYCLE FIG OF MERIT 0,000000  
 HCL FIG OF MERIT 0,000000 ( 0 HCL CONTRIBUTORS)

CYCLE 2

INDICES	E MAG.	CTRBS	SUM A	AVG A	PHI	RMS OF RESIDUALS	PHI	RMS OF RESIDUALS
** 1 0 -9	2,8268	1	9,59	9,593	2,5726	0,0000332558	-2,5726	0,0000414382
3 0 -9	2,9307	0	0,00	0,000	*****	*****		
7 0 -4	2,0657	0	0,00	0,000	*****	*****		
2 1 -1	3,4692	1	10,46	10,462	0,0017	0,0000053948		
4 1 -1	2,6041	1	10,46	10,462	-3,1404	0,0000000000		
** 6 1 -1	1,6771	2	7,61	3,804	0,0015	0,0000076294		
2 0 0	5,3961	2	20,92	10,462	-3,1404	0,0000053948		
4 0 0	2,0433	0	0,00	0,000	*****	*****		
** 0 1 1	2,0013	2	10,33	5,163	0,2433	0,0149321314	-0,2425	0,0151635613

CYCLE FIG OF MERIT 0,005761  
 HCL FIG OF MERIT 0,329039 ( 2 HCL CONTRIBUTORS)

CYCLE 3

INDICES	E MAG.	CTRBS	SUM A	AVG A	PHI	RMS OF RESIDUALS	PHI	RMS OF RESIDUALS
1 0 -9	2,8268	1	9,59	9,593	2,5714	0,0000396435	-2,5738	0,0000414382
3 0 -9	2,9307	1	9,59	9,593	1,1369	0,0000295485	-0,0012	0,0000373762
7 0 -4	2,0657	0	0,00	0,000	*****	*****		
2 1 -1	3,4692	3	21,05	7,018	-0,0306	0,0003799025		
4 1 -1	2,6041	3	17,80	5,935	-3,0571	0,0055649151		
6 1 -1	1,6771	2	7,61	3,804	0,0017	0,0000076294		
2 0 0	5,3961	7	56,71	8,102	3,1416	0,0043177588		
4 0 0	2,0433	4	9,67	2,418	0,0000	0,0204001892		
0 1 1	2,0013	2	10,33	5,163	0,2425	0,0148173684	-0,2433	0,0152791383

CYCLE FIG OF MERIT 0,007451  
 HCL FIG OF MERIT 0,425845 ( 4 HCL CONTRIBUTORS)

SUMMARY OF COMPUTED PHASES  
 COLUMN HEADINGS GIVE CYCLE NUMBER (CYCLE 0 DENOTES INPUT), AND TABULAR ENTRIES GIVE PHI (RADIAN) MULTIPLIED BY 100

	3	2	1	0
1 0 -9	257	257		
3 0 -9	114*****			0
7 0 -4	*****			0
2 1 -1	.3	0****		0
4 1 -1	-306	-314	-314	
6 1 -1	0	0		
2 0 0	314	-314****		314
4 0 0	0*****			0
0 1 1	24	24		

6 COSINES WERE READ

6 COSINES ENTERED INTO THE COMPUTATION

COSINES CORRESPONDING TO THESE VECTOR TRIPLES (FOLLOWED BY A VALUES) WERE MISSING FROM THE INPUT  
 (IF MORE THAN 100 ARE MISSING, ONLY THE FIRST 100 ARE PRINTED)

2 0 0	2 0 0	4 0 0	12,747969
-------	-------	-------	-----------

Security Classification

## DOCUMENT CONTROL DATA - R &amp; D

*(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)*

1. ORIGINATING ACTIVITY (Corporate author) Naval Research Laboratory Washington, D. C. 20390		2a. REPORT SECURITY CLASSIFICATION <b>Unclassified</b>
		2b. GROUP
3. REPORT TITLE <b>COMPUTER PROGRAM FOR THE LEAST-SQUARES DETERMINATION OF THE PHASES OF THE CRYSTAL STRUCTURE FACTORS</b>		
4. DESCRIPTIVE NOTES (Type of report and inclusive dates) <b>Final</b>		
5. AUTHOR(S) (First name, middle initial, last name) <b>J. Harrison Hancock, Janet E. Fisher, and Herbert A. Hauptman</b>		
6. REPORT DATE <b>December 16, 1970</b>	7a. TOTAL NO. OF PAGES <b>46</b>	7b. NO. OF REFS <b>6</b>
8a. CONTRACT OR GRANT NO. <b>NRL Problem N01-19</b>	9a. ORIGINATOR'S REPORT NUMBER(S) <b>NRL Report 7167</b>	
b. PROJECT NO. <b>RR 002-07-41-5065</b>		
c.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)	
d.		
10. DISTRIBUTION STATEMENT <b>This document has been approved for public release and sale; its distribution is unlimited.</b>		
11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY <b>Dept. of the Navy (Office of Naval Research), Arlington, Va. 22217</b>	
13. ABSTRACT <p>An effective technique, employing the Principle of Least Squares, which leads from the values of the crystal structure invariants to the values of individual phases, has recently been obtained. In the present report the details of the computer program which implements this technique are described. An auxiliary program to identify the required structure invariants has also been written and is contained herein.</p>		

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Crystal structure Least squares method Computer programs Structure invariants						