Acta Cryst. (1952). **5**, 109

The Computation of Fourier Syntheses with a Digital Electronic Calculating Machine

BY J. M. BENNETT* AND J. C. KENDREW

Mathematical Laboratory and Cavendish Laboratory, Cambridge, England

(*Received* 28 *July* 1951)

Programmes have been devised for computing Patterson and Fourier syntheses in two and three dimensions with the EDSAC. An outline of the methods used is given and future possibilities are discussed. At present a two-dimensional summation of about 400 independent terms for about 2000 points takes $1\frac{1}{2}$ hr.; a three-dimensional summation of 2000 terms for 18,000 points takes 9 hr. A method is described whereby the EDSAC, without special modification, can be made to print results directly in contour form with considerable economy in time.

1. Introduction

In this paper are described some methods which have been developed for carrying out crystallographic Fourier syntheses with the Cambridge Electronic Delay Storage Automatic Calculator (EDSAC). In this machine the simple arithmetical operations of addition and subtraction are carried out in a mere $1\frac{1}{2}$ milliseconds, and multiplication in 6 milliseconds; it should thus be capable of handling crystallographic calculations much more rapidly than hand or punch-card machines.

We do not propose to give an account of the EDSAC itself; we shall assume that the reader has some general acquaintance with it, such as can be obtained, for example, from papers by Wilkes (1949*a*, *b*) and by Wilkes & Renwick (1949) which deal respectively with the mode of operation of the EDSAC and with its construction. A more detailed account of programming for the EDSAC is given by Wilkes, Wheeler & Gill (1951).

It may, however, be useful to point out that the EDSAC works in the binary scale, i.e. it uses only the digits 0 and 1; and that it is provided with a 'store' consisting at present of 512 'short storage locations' each capable of retaining 17 binary digits. These locations are used to hold sequences of binary digits which may represent either numbers required or produced in the course of a computation, or orders—coded in binary numerical form—instructing the machine to carry out particular operations in a predetermined sequence. These numbers are brought into the 'accumulator' or 'multiplier register', before they are operated on, by means of appropriate orders. The machine can carry out directly only the simplest arithmetical operations—addition, subtraction, multiplication, and 'shift' (equivalent to multiplication or division by an integral power of 2; e.g. the number 24 in binary form is 11000—a right shift of two places gives 00110 or 6, i.e. $24 \div 2^2$, and a similar left shift gives 1100000 or 96, i.e. 24×2^2); more complex operations are carried out by using appropriate combinations of the simple ones. Provision is also made for testing the sign of the number in the accumulator, and for using the result of this test to determine whether the next order executed by the machine is that immediately following the sign-testing order, or that in some different predetermined position in the programme; this facility may be used as a 'switch' to select alternative paths in the computation. The numerical coding of orders has the corollary that orders themselves, like numbers, may be subjected to arithmetical operations and converted into new orders during a computation, thus enabling the programme to be altered during its course in a sense determined by the result of the computation so far. The storage capacity of the EDSAC is shortly to be doubled, giving 1024 locations in all; and it is expected that in future machines a magnetic drum auxiliary store with a capacity of several thousand locations will also be available.

So far we have constructed programmes only for the most simple types of Fourier summation, viz. for two- and three-dimensional Patterson syntheses. These programmes may be used unaltered for Fourier syntheses of the same symmetry, and require only simple modifications for other symmetries. We anticipate little difficulty in applying similar methods to other problems, especially since nearly all crystallographic computations can easily be reduced to Fourier summations. In particular, the calculation of structure factors, to which we hope to proceed later, should be quite straightforward.

Some of the first methods to be described were developed by one of us (J.M.B.) in collaboration with Mr H. E. Huxley of the Cavendish Laboratory.

* Present address: Ferranti Ltd., Moston, Manchester 10, England.

2. One-dimensional Fourier synthesis

In order to indicate in general terms the methods used we shall first give an outline of the programme required for a simple one-dimensional Fourier synthesis. The example chosen is the summation of

$$\sum_{0}^{h_{\max.}} F_h \cos 2\pi hx/a$$

at equal intervals $(x/a)_{\text{basic}}$ from $x=0$ to $x=a$. We shall suppose that the $(h_{\max.}+1)$ values of F_h are contained in storage positions m, $(m+1), \ldots, (m+h_{\max.})$.* Consider the situation at some point during the synthesis when the variables have the values h, x. We arrange to retain in convenient storage locations three 'counters' which will contain at this point

(a) x/a; (b) hx/a, (c) $[-(h_{\max.}+1)+h]$.

Then the next instructions given to the machine are the following:

(1) Take the current value of hx/a from (b) and find $\cos 2\pi hx/a$.

(2) Find F_h (by means of an order which selects the contents of storage positions $(m+h)$).

(3) Multiply F_h by $\cos 2\pi hx/a$ and add the result to a running total kept somewhere in the store.

(4) Add (a) to (b), i.e. add x/a to hx/a, giving $(h+1)\,x/a$, and place in (b) instead of hx/a.

(5) Change the order at (2) so that in the next cycle of operations it selects storage position $(m+h+1)$, thus finding F_{h+1}.

(6) Increase (c) by unity, so that it becomes $-(h_{\max.}+1)+h+1$. Test the sign of this number. If it is still negative, return to (1) and repeat the cycle. If it is zero (as it will be after the $(h_{\max.}+1)$th cycle) the summation has been completed; in this case proceed to

(7) Print out running total.

(8) Re-set (b) to zero and (c) to $-(h_{\max.}+1)$. Set the order at (2) to select storage position m. Clear running total (i.e. set to zero).

* We use the symbol F throughout this paper as the coefficient of a Fourier term irrespective of whether it represents the amplitude or the intensity of an X-ray reflexion.

(9) Add $(x/a)_{\text{basic}}$ to (a). Test (a). If it is still <1, return to (1) and repeat cycles. If $(a)=1$, the programme is finished.

This sequence of operations is illustrated schematically in Fig. 1.

An actual routine designed to carry out this programme would consist of about 100 orders. In addition the store must contain a short routine to take the F_h's from teleprinter tape and place them in a suitable form into positions m, $(m+1)$, ...; and also a print routine which takes the sums, converts them into decimal notation, and prints out the result (with sign) to the desired number of significant figures. These routines need about 25 and 50 orders respectively. There remains to be discussed one important matter, namely the method of calculating $\cos 2\pi hx/a$.

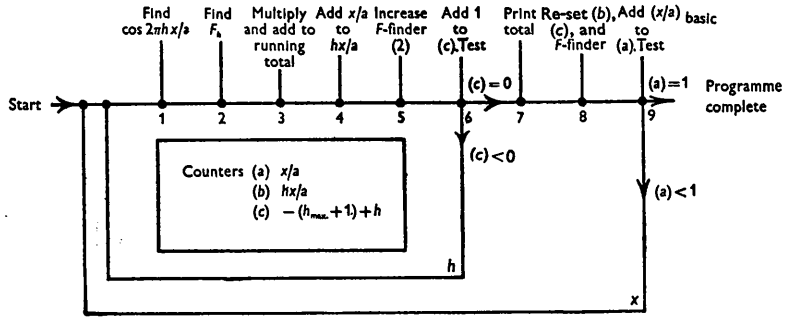


Fig. 1. Scheme for computing one-dimensional Fourier synthesis.

3. Calculation of cosines (and sines)

Three different methods of computation were tried:

(1) Each term of the type $\overset{\cos}{\underset{\sin}{\cdot}}\, 2\pi hx/a$ was obtained as required from the previous term $\overset{\cos}{\underset{\sin}{\cdot}}\, 2\pi(h-1)\,x/a$ by means of the standard addition formulae for $\overset{\cos}{\underset{\sin}{\cdot}}\,(A+B)$. $\overset{\text{Cos}}{\underset{\text{Sin}}{\cdot}}\, 2\pi x/a$ must be known and can be obtained in each outer (x) cycle by the same formulae (from the previous value and $\overset{\cos}{\underset{\sin}{\cdot}}\, 2\pi(x/a)_{\text{basic}}$).

(2) By means of a short sub-routine introduced at the beginning of the programme, a table of values of $\cos 2\pi x/a$ from 0° to 90° was prepared and stored, the intervals chosen being those normally used in crystallography, i.e. $(90/15)°$ or $(90/30)°$. This method has the disadvantage that these intervals do not readily fit in with techniques using binary-scale counters—in fact two separate counters are required, one taking account of the quadrant, and the other of the position of the angle in the quadrant. The necessity for re-setting these counters is avoided in the method described in (3).

(3) A similar sub-routine was used to compute a table of cosines from 0° to 90°, but this time at intervals of $(90/2^m)°$ where m is an integer (in our programmes

the value $m=5$ was chosen; i.e. $2^m=32$ and there are 33 entries in the table). In this case the hx/a counter merely records the number of basic intervals (of $(90/32)^\circ$); then the ordinal number of the position in the table at which the required cosine is to be found $(=|hx/a| \bmod 32)$ can be obtained from the five least significant digits, and the quadrant from the next two. For example, if

$$hx/a=36 \; [=36\times(90/32)^\circ=90^\circ+(4/32)\times 90^\circ],$$

the counter will contain 36 in binary form, viz. 0100100: the five least significant digits are 00100, giving 4 for the position in the table, and the next two are 0100000, which on right shift gives 0000001 or 1, meaning second quadrant ('00' represents first quadrant). In the first quadrant the table position is used direct; in the second it is subtracted from 33 and the value obtained from this position in the table is given a minus sign (since $\cos(90+\theta)=-\cos(90-\theta)$). Cosines in other quadrants, and sines, can be found by minor modifications of the basic procedure.

As an example of the relative speeds of the three methods, a two-dimensional Patterson synthesis for which $(h_{\text{max.}}+1)\times(k_{\text{max.}}+1)=176$ was computed at the following rates (by the first programme described in §4 below):

Calculation of cosines by method (1): 15 sec. per (x, y) value.

Calculation of cosines by method (2): 12 sec. per (x, y) value.

Calculation of cosines by method (3): 10·5 sec. per (x, y) value.

Method (1) is more economical of storage space than (2) and (3) since in the latter a table of cosine values must be stored throughout the programme. Part of this space may be recovered since, once the table has been prepared, the sub-routines introduced for the purpose of computing it may be removed, and the space occupied by it used for other purposes (e.g. the print sub-routine). The routine which reads the F_h's from the tape and stores them away may similarly be over-written once all the F_h's have been taken in.

4. Two-dimensional syntheses

First method

In two dimensions a Fourier summation of

$$\sum_0^{h_{\text{max.}}}\sum_0^{k_{\text{max.}}} F_{hk}\cos 2\pi(hx/a+ky/b)$$

may be carried out by methods analogous to those described in §2; a system of four loops is used (see Fig. 2).

Additional counters will of course be required so that, for each F_{hk}, the hx/a and ky/b counters are added together and the result is used to find appropriate cosine.

Second method

A great deal of time can however be saved by separating the x and y variables and dividing the synthesis into two stages, computing in stage I a set of $(k_{\text{max.}}+1)$ terms:

$$A(k,x)=\sum_0^{h_{\text{max.}}} F_{hk}\cos 2\pi hx/a$$

which are stored and used in stage II to compute values of:

$$\sum_0^{k_{\text{max.}}} A(k,x)\cos 2\pi ky/b$$

for a whole row of points having the same x co-ordinate. This of course is the method proposed by Lipson & Beevers (1936) and commonly used for hand computation of two-dimensional Fourier syntheses. The method is shown schematically in Fig. 3.

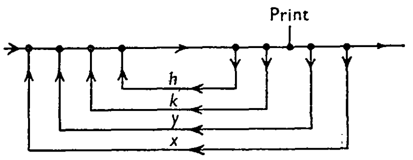


Fig. 2. Scheme for computing two-dimensional synthesis: first method.

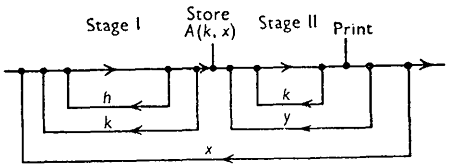


Fig. 3. Scheme for computing two-dimensional synthesis by second method, the variables being separated.

The saving in time can be estimated, in terms of this scheme, from the fact that, if the computation is to be made for XY points, the total number of times inner cycles (which are repeated most frequently and make up the bulk of the computation) have to be traversed in the whole computation is

$$X[(k_{\text{max.}}+1)(h_{\text{max.}}+1)+Y(k_{\text{max.}}+1)],$$

whereas in the first method the inner cycle is traversed $XY(k_{\text{max.}}+1)(h_{\text{max.}}+1)$ times—a number greater in the ratio $[1/Y+1/(h_{\text{max.}}+1)]$. In practice the saving of time is not so great as this because the printing time, which is the same in the two methods, is a considerable proportion of the whole, being limited in the EDSAC by the standard rate of operation of a teleprinter.

In the example mentioned at the end of §3, the rates of operation with the two methods (using the fastest method of cosine computation) were:

First method	10·5 sec. per point.
Second method	2·5 sec. per point and 10 sec. for calculation of each set of intermediate totals (or $10/Y$ sec. per point).

(These times include in each case 1·33 sec. per point for printing.)

The second method of course requires more storage space than the first, since ($k_{\text{max.}}+1$) storage positions (or a small integral multiple thereof in lower symmetries—see §7) must be reserved for intermediate totals. In our earlier work the first method was used because storage capacity was a serious limitation; more recently a larger store has become available so the second method has been adopted.

5. Methods of economizing storage capacity

In a machine such as the EDSAC, which at present has only a fairly small fast store, it is impossible to accommodate all the terms of a typical two-, and more especially of a three-, dimensional synthesis if each short storage position (of 17 binary digits) is used to store only one term. We have adopted or contemplated various economy measures which will now be described.

(*a*) *Zero elements*

In most crystallographic syntheses there frequently occur sequences of two or more zero terms. For Patterson syntheses (where all terms are positive) we have indicated such gaps by using negative numbers. Thus a row of terms

$$+5 \quad +16 \quad 0 \quad 0 \quad 0 \quad +4$$

would appear in the store as

$$+5 \quad +16 \quad -3 \quad +4$$

(*b*) *Packing*

Often all the terms to be stored are very much smaller than the maximum capacity of a short storage position ($2^{17}-1$). It is generally possible to pack more than one term into each position, though some increase in computing time is inevitable since packing and unpacking routines are required (we shall indicate the magnitude of this increase below). We have used two methods:

(i) *Packing two terms in one short storage position.* If the 17 digits of a short storage position are represented by $A, \quad Bbbbbbbb, \quad Ccccccc$

then A is normally 0, and $b\ldots b$ and $c\ldots c$ are a pair of terms, B and C indicating signs (0 if positive, 1 if negative). If A is 1 the indication to the machine is that the rest of the position contains a code combination (meaning, e.g., gap, end of row) instead of a pair of terms. The method is especially useful when, as often happens, terms occur in pairs, e.g. $(F_{hk}+F_{h\bar{k}})$, $(F_{hk}-F_{h\bar{k}})$. The largest term which can be accommodated is (2^7-1), or 127.

(ii) *Packing three terms in one long storage location.* One long storage location consists of two adjacent short ones together with a single intermediate digit (used in the EDSAC to separate storage locations). This enables three terms to be packed as follows

$$\alpha Aaaaaaaaaa, \quad S, \quad \beta Bbbbbbbbbb, \quad S, \quad \gamma Ccccccccc$$

where S, S are 'sandwich digits' merely separating the terms and useful in simplifying the unpacking routine. The three terms are $a\ldots a$, $b\ldots b$, $c\ldots c$ and their signs are indicated by A, B and C as before; the largest number which can be accommodated is now (2^9-1) or 255. We have used this method for intermediate totals in three-dimensional syntheses (see §8). These totals may be much larger than 255, so we have provided the additional facility that, if the term in binary form exceeds 9 digits, its 9 least significant digits are placed in one position and the 9 (or less) more significant digits in the neighbouring one. That a term has been split in this way is indicated by a 1 (instead of 0) at α, β, or γ (before its most significant half only); and we can now accommodate terms up to $(2^{18}-1)$ or 262,143. As an example the terms +528, +7 would be stored as

```
10,000010000 | 1 | 00,000000010 | 1 | 00,000000111
↑↑   ↑            ↑            ↑      ↑          ↑
|+   16           +           512     +          7
|     \_________________________/
|                528
Indicating split number to follow.
```

The sandwich digits S are used during unpacking, in which the basic operation is to extract a term from the left-hand end of the storage location. The sandwich digit lying to the right of the term just extracted is then tested; if it is present, another term is shifted up to the left-hand end ready for the next unpacking operation, but if it is absent the storage location must be empty and arrangements are made to extract from the next storage location in the succeeding unpacking operation.

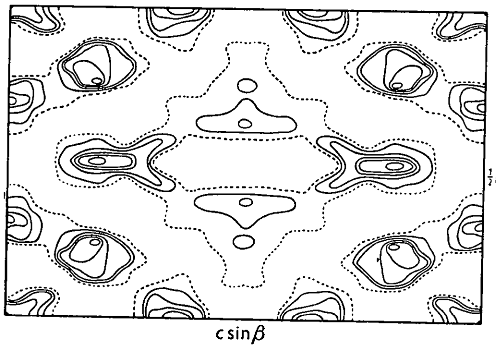
(*c*) *Logarithmic representation of terms*

In many cases X-ray intensities are measured by visual comparison with a standard step wedge whose steps represent exposures in the ratios $1:\sqrt{2}:2:2\sqrt{2}\ldots$, say. In this instance the accuracy of measurement can only be of the order $\pm 20\%$. The method of measurement is such that a logarithmic representation of terms seems appropriate.

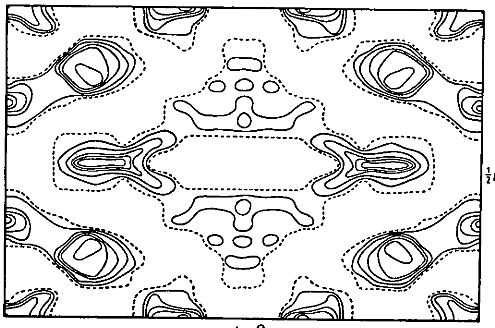
If, for example, we require to handle terms in the range 0–127, normal binary representation requires 7 digits; on the other hand, if all values are smoothed off to steps of $\sqrt{2}$, four digits suffice; if to steps of 2, three are sufficient. Thus the binary equivalent of 49 is 0110001. Smoothed off to the nearest power of $\sqrt{2}$, which is eleven, 49 may be represented as 1011; smoothed off to the nearest power of 2 (i.e. 6), 49 may be represented as 110.

It is not suggested that this smoothing is always possible, but in many cases the synthesis obtained from such smoothed-off data is not significantly different from that compounded of unsmoothed data. For example Fig. 4(*a*) is the *a* Patterson projection of horse methaemoglobin (computed from new unpublished data obtained by H. E. Huxley); Figs. 4(*b*) and 4(*c*) are the same projection computed from intensities smoothed off to $(\sqrt{2})^m$ and 2^n respectively.

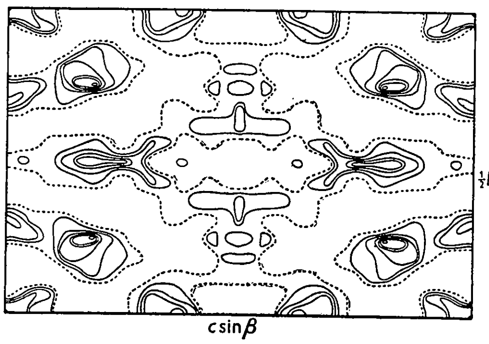
The former type of packing would make it possible to place three, and the latter four, terms in a short storage position (allowing for sign digits).



(a)



(b)



(c)

Fig. 4. a Patterson projection of horse methaemoglobin. (a) Computed from measured intensities. (b) Computed from intensities smoothed off to nearest integral powers of $\surd 2$. (c) Computed from intensities smoothed off to nearest integral powers of 2. (Contours have not been normalized.)

The interpretation of information stored in this way is merely a matter of shifting the coefficient of the term (i.e. the cosine or sine) to the left by a number of places equal to the power of two. If the gradations are in steps of $\surd 2$ the result must be multiplied by a constant equal to $\surd 2$ if the last digit of the stored term is 1 (indicating an odd power of $\surd 2$).

6. Checks

Correct operation of the arithmetical units of the machine can be checked by means of special test routines before the start of a programme. It is, however, necessary to include checks in the programme to verify the continued correct operation of the machine during computation, the most likely sources of error being those due to the loss or gain of digits in the store. One such check is to add the value of each term as it is taken out of the store to a running check total; once in each outer cycle, i.e. once per block of values for a given x, this check total is printed out. This printed total should of course remain the same throughout the computation. This method checks that the terms have been correctly stored and also gives some check on the performance of the Arithmetical Unit. If through a partial store failure *orders* are distorted, the result will almost certainly be that the machine will stop or will print obviously absurd results, so no special checks are needed to meet this danger.

If an error occurs at some point during a computation, it can easily be arranged that on repetition only that part of the computation is carried out which follows the point at which the error took place on the first attempt.

7. Crystal symmetry: Patterson and Fourier summations

We have so far discussed the computation of a two-dimensional summation of the form

$$\sum_{0}^{h_{\text{max.}}}\sum_{0}^{k_{\text{max.}}} F_{hk}\cos 2\pi hx/a\cos 2\pi ky/b.$$

Provided that factors of $\frac{1}{2}$ and $\frac{1}{4}$ have been applied to terms F_{h0} (and F_{0k}; $h,k\neq 0$) and F_{00} respectively—exactly as is done in the Beevers–Lipson method—we see that this summation corresponds to a projection having a centre and a plane of symmetry. In practice the symmetry is often lower. To take a case common in Patterson syntheses, the projection may possess only a centre of symmetry. Here we must prepare two sets of intermediate terms in the first stage.

Stage I:

$$\left.\begin{aligned} A(k,x) &= \sum_{0}^{h_{\text{max.}}}(F_{hk}+F_{h\bar{k}})\cos 2\pi hx/a \\ B(k,x) &= \sum_{0}^{h_{\text{max.}}}(F_{hk}-F_{h\bar{k}})\sin 2\pi hx/a \end{aligned}\right\}\begin{gathered}\text{having } 2(k_{\text{max.}}+1)\\ \text{terms in all;}\end{gathered}$$

and similarly we must compute in the second stage:

Stage II:

$$C(x,y)=\sum_{0}^{k_{\text{max.}}} A\cos 2\pi ky/b;\quad D(x,y)=\sum_{0}^{k_{\text{max.}}} B\sin 2\pi ky/b.$$

The final value at (x,y) is now $(C-D)$, and at $(x,\bar{y})$ is $(C+D)$. It is convenient to calculate pairs of A and B, and pairs of C and D, together. Terms F_{hk} and $F_{h\bar{k}}$

are entered alternately on the input tape: as it takes them in, the machine computes $(F_{hk}+F_{h\bar{k}})$ and $(F_{hk}-F_{h\bar{k}})$ and packs the pair in a single short storage location as described in § 5 (*b*) (i). The intermediate totals occupy $2(k_{\max.}+1)$ locations. At the end C and D are computed, the machine printing out first $(C-D)$ and then $(C+D)$.

Other symmetries, including those where the phase angles need not be 0 or π, can be dealt with similarly by trivial modifications of a few standard routines; it is convenient to break up terms with general phase into cosine and sine components, rather than to introduce the phase angle directly. Thus both Patterson and Fourier syntheses of any symmetry can be handled, though the lower the symmetry the more lengthy the computation and the more the storage space required for intermediate terms.

We now give examples of the time taken for computations of two standard types. Let the total number of independent terms entered on the tape be t, and the number of points at which the value of the function is computed be XY (X and Y are each of the form 2^m+1—see § 3 (3)); $H=h_{\max.}+1$ and $K=k_{\max.}+1$.

Case A. Projection with centre and plane of symmetry

Total time

$$=30+t/12+X[(1\cdot33+0\cdot053K)\,Y+3+t/20]\text{ sec.*}$$

In this programme the store is occupied as follows: main routine, 148 locations; print routine, 50; table of cosines, 33; $(k_{\max.}+1)$ intermediate totals, say 20—in all 251 locations, leaving room for 261 terms (773 when the total storage capacity is increased from 512 to 1024 locations). Terms are not packed.

Of the total time, $1\cdot33XY$ sec. are occupied in printing.

Example. Patterson projection of whale myoglobin (*c* projection of $P2_12_12$). 260 terms, computed for $17\times33=561$ points (intervals of $a/64$, $b/32$). Total time, 30 min. (of which 14 min. are printing time).

Case B. Projection with centre of symmetry

Total time

$$=45+t/10+X[(3+0\cdot093K)\,Y/2+3+0\cdot044t]\text{ sec.}$$

In the store the main routine occupies 186 locations; the print routine, 48; the table of cosines, 33; $2(k_{\max.}+1)$ intermediate totals, say 40—in all 307 locations, leaving room for 205 packed pairs, or 410 terms in all (1434 terms when the total storage capacity is doubled).

The increase in time due to packing is about 0·075 sec. per term—$0\cdot075t$ sec. in all; unpacking does not involve an appreciable delay.

Of the total time, $1\cdot33XY$ sec. are occupied in printing, as before.

* It will be noted that in this and the similar expression for case B the quantity H does not appear explicitly, since it is implicit in t. (In the expression for a three-dimensional summation on p. 115, L is absent for the same reason.)

Example. Patterson projection of horse haemoglobin (*b* projection of $C2$). 410 terms (205 pairs of F_{hk} and $F_{h\bar{k}}$), computed for $65\times33=2145$ points (intervals of $a/64$, $c/64$). Total time, 97 min. (of which 48 min. are printing time). The time taken to pack the terms is only 30 sec. in all.

8. Three-dimensional summations

We shall now describe a programme devised for the computation of a three-dimensional Patterson synthesis in the space group $P2_1$. In three dimensions the arguments in favour of using Beevers–Lipson methods are even stronger than in two; the stages in the computation in this particular space group are then the following:

Stage I:

$$\left.\begin{aligned}A(h,k,z)&=\sum_0^{l_{\max.}}(F_{hkl}+F_{hk\bar{l}})\cos2\pi lz/c\\ B(h,k,z)&=\sum_0^{l_{\max.}}(F_{hkl}-F_{hk\bar{l}})\sin2\pi lz/c\end{aligned}\right\}$$

No. of intermediate terms to be stored
$=2(h_{\max.}+1)\,(k_{\max.}+1)$.

Stage II:

$$\left.\begin{aligned}C(x,k,z)&=\sum_0^{h_{\max.}}A\cos2\pi hx/a\\ D(x,k,z)&=\sum_0^{h_{\max.}}B\sin2\pi hx/a\end{aligned}\right\}$$

No. of intermediate terms to be stored
$=2(k_{\max.}+1)$.

Stage III:

$$E(x,y,z)=\sum_0^{k_{\max.}}C\cos2\pi ky/b;$$

$$F(x,y,z)=\sum_0^{k_{\max.}}D\cos2\pi ky/b;$$

$$P(x,y,z)=E-F;\quad P(x,\bar{y},z)=E+F.$$

It will be seen that storage problems are now acute; the routines necessary to carry out the computation are longer than in two dimensions, and the intermediate totals after Stage I alone may in a typical case (where $h_{\max.}+1=k_{\max.}+1=20$, say) number 800. Thus even if they are packed three to a long storage location (§ 5 (*b*) (ii)), the space occupied is more than $800\times\frac{2}{3}$ (i.e. 533) short storage locations. With the present capacity of the EDSAC even these intermediate totals can be accommodated only if $h_{\max.}$ and $k_{\max.}$ are rather small. Even when the anticipated increase to 1024 locations has taken place there will still be too little space for the table of F values. In fact it will only become practicable to store the latter (perhaps 10,000 in number) when a magnetic store is available.

In the meantime we have adopted a method in which the F values need not be held in the store at all; they are fed in anew for the computation of each z layer of the synthesis. The stages in the computation of each layer are as follows:

(1) A routine for one-dimensional summation (along z) is placed in the store, together with a constant equal to the required value of z/c. A packing routine and a table of cosines are also stored.

(2) A tape punched alternately with F_{hkl} and $F_{hk\bar{l}}$ is passed into the machine. $(F_{hkl}+F_{hk\bar{l}})$ and $(F_{hkl}-F_{hk\bar{l}})$ are computed and the multiplications required for calculating the contributions to A and B are carried out. The A and B values are packed, three to a long storage location.

(3) When all the F's have been passed into the machine, the tables of A and B are complete for the current value z/c of Z. The original routines are now overwritten by routines for carrying out stages II and III and by an unpacking routine (to unpack the A's and B's). The machine prints out $P(x, y, z)$ and $P(x, \bar{y}, z)$ for a particular z/c and for all x, y; i.e. for one z layer of the synthesis.

For other values of z/c (1), (2) and (3) are repeated, using in each case the appropriate z/c in (1).

In order to test this method we have repeated the three-dimensional Patterson summation of horse haemoglobin described by Perutz (1949), but have omitted all terms of spacing less than 6·3 Å; then the $2(h_{\max.}+1)(k_{\max.}+1)$ terms of the first intermediate totals can be accommodated in the present store. The space group of horse methaemoglobin is $C2$ instead of $P2_1$, and, in order to avoid modification of the routine (which was devised for another purpose), no special arrangements were made to allow for space-group absences; x and z were, however, interchanged (so that in stage I the summation along x was carried out), alternate pairs of terms being zero.

The number of independent terms (including zeros) was 2188. Here

$$h_{\max.}=16, \quad k_{\max.}=10, \quad l_{\max.}=8;$$
$$XYZ = 17\times 33\times 33 = 18{,}513.$$

The computing times were as follows:

For each x, time to take in terms and carry out stage I, 5 min; time to carry out stages II and III, 27 min. Total machine time for whole computation $=17(5+27)$ min. $=9$ hr. 4 min. (of which about 5 hr. 40 min. are occupied in printing, and about 50 min. in packing and unpacking).

In the general case the computing time would be

$$Z\{0{\cdot}12t+110+X[0{\cdot}18HK + (0{\cdot}14K+1{\cdot}1)(Y+1)]\}\ \text{sec.}$$

If the terms could be held in an auxiliary magnetic store throughout the computation the total computing time would be reduced approximately in the ratio $1 : 0{\cdot}12t(Z-1)$, $0{\cdot}12t$ being the time taken to insert the terms once into the machine. In a synthesis of many terms, when $0{\cdot}12t$ might amount to half an hour, this saving would be very considerable.

9. Contour plotting

For many purposes it is unnecessary to know accurately the values of the function at every point in the final summation; all that is required is to draw approximate contours of electron or vector density in positive regions. It was suggested to us by Mr R. A. Brooker that the EDSAC could be made to print out the results in contour form directly, to the required degree of accuracy.

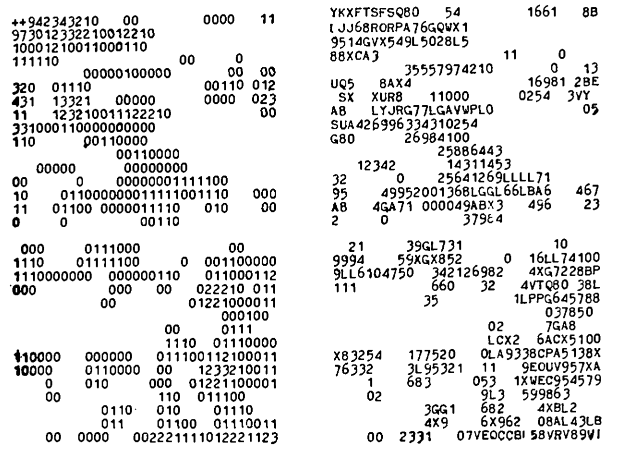
Fig. 5 (a) shows the b Patterson projection of whale myoglobin printed in this way. It has been arranged that

(a) for every negative value of the function the machine prints a space,

(b) for every positive value between 0 and 31 the number 0 is printed,

(c) 1 is printed for values 32–63, 2 for 64–95, etc., up to 9 for values 288–319,

(d) '+' is printed for values exceeding 319.



```
++942343210   00        0000    11
973012332210012210
10001210011000110
111110               00      0
         00000100000        00  00
320  01110               00110 012
431  13321   00000       0000  023
11   123210011122210            00
331000110000000000
110      00110000
             00110000
   00000      00000000
00      0    00000001111100
10    011000000011111001110    000
11   01100 0000011110    010    00
0     0         00110

 000     0111000            00
1110    01111100      0  001100000
1110000000  000000110    011000112
000         000   00    022210 011
           00          0122100011
                           000100
                    00     0111
                    1110  01110000
110000   000000   011100112100011
10000    0110000  00    1233210011
    0    010     000   01221100001
    00             110  011100
           0110   010     01110
           011    01100   01110011
    00  0000   002221111012221123
```

(a)

```
YKXFTSFSQ80   54         1661   8B
ιJJ68RORPA76GQWX1
9514GVX549L5028L5
88XCA3                11     0
         35557974210        0   13
UQ5   8AX4               16981 2BE
 SX  XUR8    11000      0254  3VY
A8   LYJRG77LGAVWPLO            05
SUA426996334310254
G80      26984100
             25886443
   12342       14311453
32      0    25641269LLLL71
95    49952001368LGGL66LBA6    467
A8   4GA71  000049ABX3   496    23
2     0          379E4

  21      39GL731           10
9994    59XGX852      0  16LL74100
9LL6104750   342126982   4XG7228BP
111          660    32  4VTQ80 38L
            35         1LPPG645788
                           037850
                    02     7GA8
                    LCX2  6ACX5100
X83254    177520   0LA9338CPA5138X
76332     3L95321  11   9EOUV957XA
     1    683     053  1XWEC954579
     02            9L3  599863
            3GG1   682    4XBL2
            4X9    6X962  08AL43LB
     00  2331   07VEOCCBI 58VRV89VI
```

(b)

Fig. 5. Patterson projection of whale myoglobin: 'contours' printed by the EDSAC. (a) Contours at intervals of 32. (b) Contours at intervals of 4.

Contours may rapidly be drawn in at intervals of 32. Moreover, the programme is so arranged that, by changing two parameters, contours at any other interval 2^m (where m is a positive integer) may be printed.

If we use this simple scheme, the accuracy with which contours can be drawn is less than in the conventional method of plotting the actual values of the function on a grid. However, greater accuracy may be obtained by printing at closer intervals, making use of letters in addition to the ten digits 0–9. With the teleprinter code used in the EDSAC it is easy to make use of 18 or more letters; for example, Fig. 5 (b) shows the same projection as Fig. 5 (a), the contour values now being printed at intervals of 4, and symbols being alloted in such a way that for values lying between $4n$ and $4n+3$, when $n=0, 1, \ldots, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21,$

22, 23, 24, 25, the machine prints 0, 1, ..., 9, $L, X, G, A, B, C, V, P, Q, W, E, R, T, Y, U, I$. In this example it is unnecessary to *draw* contours at intervals of 4; intervals of 32 are quite adequate. But the fact that the machine has *printed* at intervals of 4 makes it possible to insert contours at intervals of 32 with an accuracy almost as great as is obtained by the conventional method, and certainly as great as the accuracy of the original data justifies.

Of course the 'cell dimensions' in the machine plot are set by the space and shift distances of the teleprinter. The simplest way to convert the contours obtained from this plot to the correct cell dimensions (and angles in the case of a non-rectangular space group) would be by means of a mechanical linkage device.

Incidentally the method is much more rapid than those we have hitherto described. As previously indicated, printing time represents a large proportion of the whole; in printing values of the function by normal methods, the teleprinter has to record 4 digits (say), sign (+ or −), space, and to execute a line shift and carriage return—in all 8 operations each taking $\frac{1}{6}$ sec. In contour plotting it prints one symbol only for each value, and line shift and carriage return are required only once for each row. Thus by contour plotting the overall machine time in a typical case was reduced from 49 min. to 29 min.

It must be emphasized that the method just described involves no modification of the standard EDSAC teleprinter output. It might be worth while in future applications to consider the design of a special output system (either quite separate from the computer or auxiliary to it) designed specifically for crystallographic purposes, e.g. with cathode ray tube presentation; and in this case the method could be improved greatly.

10. Conclusion

The computations described in this paper belong to the class which requires the retention of a large amount of information (i.e. the F values) in the store throughout; the terms lie, however, in a restricted range of magnitudes and are usually known to a relatively low degree of accuracy. The amount of printing involved is moreover so considerable as to warrant special consideration.

These features slow down the speed of operation achieved with the EDSAC, as it is constituted at present, and, although the results described here represent a considerable increase in speed over existing methods, a further great improvement could be effected by such simple measures as increasing the storage capacity and incorporating a faster output system.

At the present stage it does not seem possible to assess the relative values for crystallographic purposes of general-purpose digital machines, such as the EDSAC, and of special-purpose analogue machines (see, for example, Pepinsky, 1947; Pepinsky & Sayre, 1948), though it would appear to us that some crystallographic problems may be more economically tackled by the one and some by the other type. Digital machines have the great advantages of accuracy (which is becoming increasingly important in crystallography), and of versatility, which enables the capital cost and running expenses of a general-purpose digital machine to be shared among a number of users with very different problems to solve. On the other hand, analogue machines such as the Pepinsky machine may have special advantages for some crystallographic problems (see Bunn, 1951).

The authors wish to express their thanks to the Medical Research Council and to the Institution of Electrical Engineers for financial support; and to the Director and Staff of the Mathematical Laboratory, University of Cambridge, for their hospitality and enthusiastic co-operation.

References

Bunn, C. W. (1951). Reported by Wilson, A. J. C., *Brit. Jour. Appl. Phys.*, **2**, 61.

Lipson, H. & Beevers, C. A. (1936). *Proc. Phys. Soc.* **48**, 772.

Pepinsky, R. (1947). *J. Appl. Phys.* **18**, 601.

Pepinsky, R. & Sayre, D. (1948). *Nature, Lond.*, **162**, 22.

Perutz, M. F. (1949). *Proc. Roy. Soc.* A, **195**, 474.

Wilkes, M. V. (1949*a*). *J. Sci. Instrum.* **26**, 217.

Wilkes, M. V. (1949*b*). *Nature, Lond.*, **164**, 557.

Wilkes, M. V. & Renwick, W. (1949). *J. Sci. Instrum.* **26**, 217.

Wilkes, M. V., Wheeler, D. J. & Gill, S. (1951). *The Preparation of Programmes for an Electronic Digital Computer, with Special Reference to the EDSAC and the use of a Library of Sub-Routines.* Cambridge, Mass.: Addison-Wesley Press.