CS32

MINIMUM MULTIPLICATION FOURIER ANALYSIS

BY

R. W. HOCKNEY

TECHNICAL REPORT CS32
DECEMBER 14, 1965

COMPUTER SCIENCE DEPARTMENT
School of Humanities and Sciences
STANFORD UNIVERSITY

MINIMUM MULTIPLICATION FOURIER ANALYSIS

BY

R. W. Hockney

Abstract: Fourier analysis and synthesis is a frequently used tool in applied mathematics but is found to be a time consuming process to apply on a digital computer and this fact may prevent the practical application of the technique. This paper describes an algorithm which uses the symmetries of the sine and cosine functions to reduce the number of arithmetic operations by a factor between 10 and 30. The algorithm is applicable to a finite fourier (or harmonic) analysis on $12 \otimes 2^q$ values, where q is any integer ≥ 0 and is applicable to a variety-of end conditions. A complete and tested B5000 Algol program known as FOURIER12 is included.

During the period of this research the author collaborated closely with personnel of Triservice Contract Nonr-225(24) and part of his salary was paid by this contract.

CONTENTS

1. FOURIER ANALYSIS

Suppose we are given a sequence of (n+1) numbers

$$\varphi_0,\varphi_1,\ldots,\varphi_s,\ldots,\varphi_n \tag{1}$$

then these may be expressed exactly by a **finite Fourier series**

$$\varphi_s = \sum_{k=nmin}^{k=nmax} V(s,k)\bar{\varphi}_k \tag{2}$$

where $\bar{\varphi}_k$ is the amplitude of the kth harmonic and $V(s,k)$ is the kth harmonic function. This is the process of Fourier synthesis.

To obtain the values of the $\bar{\varphi}_k$ from the φ_s we form a summation over φ_s with a second set of harmonic functions $W(k,s)$

$$\bar{\varphi}_k = \sum_{s=nmin}^{s=nmax} W(k,s)\varphi_s \tag{3}$$

This inverse transformation is made possible because the functions $V(k,s)$ and $W(s,k)$ satisfy the biorthogonality relations.

$$\left.\begin{array}{ll} & \displaystyle\sum_{s=nmin}^{nmax} W(k,s)\ V(s,k') = \delta_{kk'} \\ \text{and} & \displaystyle\sum_{k=nmin}^{nmax} V(s,k)\ W(k,s') = \delta_{ss'} \end{array}\right\} \tag{4}$$

Thus far we have not specified the extent of the summation (Le., value of nmax) nor explicitly stated the functions V(k,s) and $W(s,k)$, because these depend on the boundary conditions which are imposed on the sequence (1). We shall consider various cases which are distinguished in the Algol procedure FOURIER12 (BC,Q,X,Y) by the value of the input parameter BC (meaning boundary condition).

1.1 The periodic case BC = 3 and 4

This is the usual situation that is meant when the term Fourier or Harmonic analysis is used. In this case the values φ_0 to φ_{n-1} are repeated periodically such that $\varphi_{s+pn} = \varphi_s$ for any integer p. Thus we want an expression for the infinite sequence

$$\ldots,\varphi_0,\ldots,\varphi_{n-1},\varphi_0,\ldots\ \varphi_{n-1},\varphi_0,\ldots\ \varphi_{n-1},\varphi_0,\ldots,\varphi_{n-1},\ldots$$

In this case the harmonic functions are

$$V(s,k) = W(k,s) = \begin{cases} P_k(n/2)\sqrt{\frac{2}{n}}\cos\frac{2\pi s\,k}{n} & .\ 0 \le k \le n/2,\ 0 \le s \le n-1 \\ \sqrt{\frac{2}{n}}\sin\frac{2\pi s(k-\frac{n}{2})}{n} & .\ n/2 < k \le n-1,\ 0 \le s \le n-1 \end{cases} \qquad (5)$$

where

$$P_k(j) = \begin{cases} 1/\sqrt{2} & \text{if } k = 0 \text{ or } j \\ 1 & \text{otherwise} \end{cases}$$

and

The procedure call

FOURIER12(3,Q,X,Y);

performs a periodic Fourier analysis on $n = 12 \times 2^Q$ points $X_0, X_1, \ldots, X_{n-1}$ as follows:

$$Y_k = \sum_{s=0}^{n-1} W(k,s)X_s \quad -- \quad 0 \le k, s \le n-1$$

The Fourier synthesis is performed by the call

FOURIER12(4,Q,Y,X);

which synthesises the harmonic components $Y_0, Y_1, \ldots, Y_{n-1}$ into the point values $X_0, X_1, \ldots, X_{n-1}$ according to

$$X_s = \sum_{k=0}^{n-1} V(s,k)Y_k \quad . \quad 0 \le k, s \le n-1$$

1.2 Sine expansion or zero value case BC = 1

In this case we insist that the value of φ at the two ends is zero, and there are only (n-1) active points $\varphi_1, \varphi_2, \ldots, \varphi_{n-1}$ i.e.,

$$\varphi_0 = \varphi_n = 0 \qquad (6)$$

This is achieved by the image conditions

$$\varphi_{-p} = -\varphi_p$$

and $$\varphi_{n-p} = -\varphi_{n+p}$$

As a result, if the origin is taken at s = 0, only sine terms appear in the series expansion and we get

$$V(s,k) = W(k,s) = \sqrt{\frac{2}{n}} \sin \frac{\pi sk}{n} \qquad 1 \leq k < n-1, \; 1 \leq s < n-1 \tag{7}$$

and nmin = 1, nmax = n-1.

In this case we note that $V(s,k) = W(s,k)$ and the processes of Fourier analysis and synthesis are identical. Both Fourier analysis and synthesis are performed by the same procedure call

FOURIER12(1,Q,X,Y)

according to

$$Y_k = \sum_{s=1}^{s=n-1} \sqrt{\frac{2}{n}} \sin \frac{\pi sk}{n} X_s \qquad 1 < s,\; k \leq n-1$$

1.3 Cosine expansion or zero slope case BC = 2

If we have a mathematical problem with the boundary condition of zero slope, the finite difference form leads to the end conditions

$$\left.\begin{aligned} \varphi_{-p} &= \varphi_p \\ \text{and} \quad \varphi_{n+p} &= \varphi_{n-p} \end{aligned}\right. \tag{8}$$

In this case the Fourier expansion with origin at s = 0 contains only cosine terms and the harmonic functions are:

$$\left.\begin{aligned} V(s,k) &= \sqrt{\frac{2}{n}}\, P_k^2(n) \cos \frac{\pi sk}{n} \\ W(k,s) &= \sqrt{\frac{2}{n}}\, P_s^2(n) \cos \frac{\pi sk}{n} \end{aligned}\right\} \tag{9}$$

Here again $V(s,k) = W(s,k)$ and the process of Fourier analysis and synthesis is **identical.** There are however (n+1) active points since there are now two end values to include and nmin = 0, nmax = n.

Both Fourier analysis and synthesis are performed by the same procedure call namely

```
FOURIER12(2,Q,X,Y);
```

according to

$$Y_k = \sum_{s=0}^{s=n} \sqrt{\frac{2}{n}}\, P_s^2 \cos \frac{\pi sk}{n}\, X_s$$

20 SOLUTION OF EQUATIONS USING EIGENVECTORS

Consider the set of linear equation

$$A\underset{\sim}{\varphi} = \underset{\sim}{b} \tag{10}$$

Let the eigenvectors of A be u_i with eigenvalue λ_i. Then

$$A\, \underset{\sim}{u}_i = \lambda_i u_i \tag{11}$$

If $Q = (\underset{\sim}{u}_{nmin},\ldots,\underset{\sim}{u}_{nmax})$ is the matrix of eigenvectors and A is diagonalisable then we know

$$\left.\begin{aligned} Q^{-1}AQ &= A \\ \text{and} \qquad Q\,A\,Q^{-1} &= A \end{aligned}\right\} \tag{12}$$

where A is the diagonal matrix $A_{ij} = \lambda_i \delta_{ij}$.

Substituting (12) into (10)

$$Q \Lambda Q^{-1} \underset{\sim}{\varphi} = \underset{\sim}{b}$$

(13)

or $$Q^{-1} \underset{\sim}{\varphi} = \Lambda^{-1} Q^{-1} \underset{\sim}{b}$$

If we define the transform of a vector $\underset{\sim}{a}$ to be

$$\bar{a} = Q^{-1} \underset{\sim}{a}$$

with the inverse relation

$$\underset{\sim}{a} = Q \underset{\sim}{\bar{a}}$$

then (13) becomes

$$\underset{\sim}{\bar{\varphi}} = \Lambda^{-1} \underset{\sim}{\bar{b}} \qquad (14)$$

The equations (10) may therefore be solved in 3 stages

1) 'Fourier' analysis of the right hand side into its transform

$$\underset{\sim}{\bar{b}} = Q^{-1} \underset{\sim}{b} \qquad (15)$$

2) Solution by simple division for the transform of φ

$$\underset{\sim}{\bar{\varphi}} = \Lambda^{-1} \underset{\sim}{\bar{b}} \qquad (16)$$

3) 'Fourier synthesis' of the transform of φ into φ

$$\underset{\sim}{\varphi} = Q \underset{\sim}{\bar{\varphi}} \qquad (17)$$

If Q is a full (nxn) matrix without symmetry stages 1) and 3) require $2n^2$ arithmetic -operations and stage 2) n operations giving a total of

$$4n^2 + n \text{ arithmetic operations} \tag{18}$$

If (10) had been solved by Gauss elimination it would have required about

$$\frac{n^3}{2} + 2n^2 \text{ arithmetic operations.} \tag{19}$$

This suggests that if Q is full and the eigenvectors are known they should be used. On the other hand for a band matrix with m diagonals Gauss elimination requires

$$\frac{m^2n}{2} + 2mn \text{ arithmetic operations} \tag{20}$$

and there will be some m below which Gauss elimination is advantageous and above which the eigenfunction method should be used.

Of course in the general case the eigenvectors of A will not be known and it would be a more difficult job to find them than to solve the equation (10) by Gauss elimination.

In certain cases however where the matrix A is the finite difference approximation to a linear differential equation the eigenvectors are known and the method of solution by eigenvectors becomes attractive.

Furthermore if the eigenvectors are sines and cosines the symmetry of these functions can be used to reduce further the numbers of operations required in steps 1) and 2) of the solution of equation (10).

3. SPECIAL CASES WITH KNOWN EIGENVECTORS

Consider the linear differential operator

$$L(x) = \sum_{\ell=0}^{\ell=p} a_{2\ell} \frac{d^{2\ell}}{dx^{2\ell}} \tag{21}$$

which has constant coefficients and in which derivatives only appear to an even order.

Then the finite difference form of

$$L\,\varphi(x) = b(x) \tag{22}$$

at the s th point on a uniform mesh will be expressable as

$$\sum_{j=-p}^{j=+p} g_j\, \varphi_{s+j} = b_s \tag{23}$$

where $g_j = +g_{-j}$ due to the even condition on the derivatives, and the g_j are the same for'all points s on the mesh.

In conventional matrix form equations (23) would be written in the form of equation (10) as

$$A\,\underset{\sim}{\varphi} = \underset{\sim}{b} \tag{24}$$

with

$$A = \begin{bmatrix} g_0 & \cdot & g_1 & \cdot & g_p & & \bigcirc \\ g_1 & \bullet & & \bullet & & \cdot & g_p \\ \cdot & & \cdot & & \cdot & & \cdot \\ g_p & & \cdot & \bullet & & \cdot & g_1 \\ \bigcirc & & g_p & \cdots & g_1 & & g_0 \end{bmatrix} \qquad \underset{\sim}{\varphi} = \begin{bmatrix} \varphi_0 \\ \vdots \\ \vdots \\ \varphi_n \end{bmatrix} \qquad \underset{\sim}{b} = \begin{bmatrix} b_0 \\ \vdots \\ \vdots \\ b_n \end{bmatrix} \tag{25}$$

To solve equation (24) by the eigenvector technique we must first determine the eigenvectors of the matrix A, as these constitute the columns of the transformation matrix Q.

To determine these vectors it is more convenient to return to the equations in the form given in (23).

If the eigenvector is $\underset{\sim}{u}$ and the eigenvalue λ then we need to solve:

$$\sum_{j=-p}^{j=+p} g_j u_{s+j} = \lambda u_s \tag{26}$$

Let us try the complex function $e^{i\theta_k s}$ which by taking real, or imaginary parts can be used to cover both the case of $u_s = \cos \theta_k s$ and $u_s = \sin \theta_k s$

$$\sum_{j=-p}^{j=+p} g_j e^{i\theta_k(s+j)} = g_0 e^{i\theta_k s} + \sum_{j=1}^{p} g_j \{e^{i\theta_k(s+j)} + e^{i\theta_k(s-j)}\}$$

$$= \{g_0 + \sum_{j=1}^{p} g_j (e^{i\theta_k j} + e^{-i\theta_k j})\} e^{i\theta_k s}$$

$$= \{g_0 + 2 \sum_{j=1}^{p} g_j \cos \theta_k j\} e^{i\theta_k s}$$

Therefore

$$\boxed{\lambda_k = g_0 + 2 \sum_{j=1}^{p} g_j \cos \theta_{kj}} \tag{27}$$

and the corresponding eigenvector is

$$u_s = A \sin \theta_k s \quad \text{or} \quad B \cos \theta_k s$$

where A, B, and θ_k are determined from the boundary conditions. We consider the same cases as in section 1.

3.1 The periodic case

$$u_{s+pn} = u_s \quad p \text{ an integer}$$

The vector has a period equal to n . . . $\theta_k = \frac{2\pi k}{n}$ where k is an integer. The value at s = 0 is arbitrary therefore both sine and cosine vectors are permissible

$$\therefore \quad u_s = A \sin \frac{2\pi ks}{n} \quad \text{or} \quad B \cos \frac{2\pi ks}{n} \qquad k = 0, 1, \dots, n/2$$

when suitably ordered and normalized these functions are identical with V(s,k) defined in section 1.1.

3.2 Sine case

$$u_0 = u_n = 0 \text{ and } u_{-p} = -u_p \quad u_{n+p} = -u_{n-p}$$

$u_0 = 0$, therefore only the sine vector is permissible. $u_n = 0$ therefore the sine must go through an integral number of π as s runs from 0 to n and $\theta_k = \frac{\pi k}{n}$.

When normalized these are the functions V(s,k) of section 1.2.

3.3 Cosine case

$$u_{-p} = u_p$$

$$u_{n+p} = u_{n-p}$$

By the symmetry of the end condition only the cosine vector exists and furthermore an integral number of π must be covered as s runs from 0 to n. Therefore $\theta_k = \frac{\pi k}{n}$ and the functions when normalized are the V(s,k) of section 1.3.

We therefore conclude that the eigenvectors of the matrix A are the functions V(s,k) previously defined, with the choice of the V(s,k) being determined by the particular boundary conditions imposed at $s = 0$ and $s = n$.

Forming the transformation matrix Q from the eigenvectors V(s,k) we have for the s,kth element

$$Q_{sk} = V(s,k) \tag{28}$$

Recalling (17) Fourier synthesis of the function $\bar{\underset{\sim}{\varphi}}$ is the operation

$$\underset{\sim}{\varphi} = Q\,\bar{\underset{\sim}{\varphi}} \tag{29}$$

or in terms of the V(s,k)

$$\varphi_s = \sum_k V(s,k)\bar{\varphi}_k \tag{30}$$

To find the inverse of Q we make use of the orthogonality relations (4) when it is clear that

$$(Q^{-1})_{ks} = W(k,s)$$

because then the relations (4) become in matrix form

$$Q^{-1}Q = I$$

$$Q\,Q^{-1} = I \qquad (31)$$

as required.

Recalling (15) we have that Fourier analysis is the operation

$$\bar{\underset{\sim}{\varphi}} = Q^{-1}\varphi \qquad (32)$$

or alternatively

$$\bar{\varphi}_k = \sum_s W(k,s)\varphi_s \qquad (33)$$

4. SOME COMMON CASES

Let $L(x) = \frac{d^2}{dx^2}$ (34)

then the usual 3 point finite difference approximation is

$$L(x)\varphi(x) \cong \sum_{j=-1}^{+1} g_j \phi_{s+j} = \varphi_{s-1} - 2\varphi_s + \varphi_{s+1}$$

thus $g_{-1} = g_1 = 1$ $g_0 = -2$.

The matrix form of L(x) has the following forms according to the boundary conditions

4.1 Periodic

$$A = \begin{bmatrix} -2 & 1 & 0 & \cdot & 0 & 1 \\ 1 & \ddots & \ddots & & \bigcirc & 0 \\ 0 & \ddots & \ddots & \ddots & & \cdot \\ \cdot & & \ddots & \ddots & \ddots & 0 \\ 0 & \bigcirc & & \ddots & \ddots & 1 \\ 1 & 0 & \cdot & 0 & 1 & -2 \end{bmatrix} \qquad d = \begin{vmatrix} d_0 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ d_{n-1} \end{vmatrix} \tag{35}$$

Eigenvector or transformation matrix is defined by

$$Q_{sk} = \begin{cases} P_k \sqrt{\frac{2}{n}} \cos \frac{2\pi sk}{n} & 0 \leq k \leq n/2, \quad 0 \leq s \leq n-1 \\ \\ \sqrt{\frac{2}{n}} \sin \frac{2\pi\ (k-n/2)}{n} & n/2 < k \leq n-1, \quad 0 \leq s \leq n-1 \end{cases} \tag{36}$$

A is symmetric therefore the inverse of Q is its transpose i.e.,

$$(Q^{-1})_{k,s} = Q_{sk}$$

This may also be seen directly from (5). Note that $Q_{sk} \neq (Q^{-1})_{sk}$ and analysis and synthesis are different operations. The eigenvalues are

$$\lambda_k = g_0 + 2\sum_{j=1}^{j=1} g_1 \cos \theta_k \ , \ \theta_k = \frac{2\pi k}{n}$$

$$= -2 + 2 \cos \frac{2\pi k}{n} = -2(1-\cos \frac{2\pi k}{n})$$

$$= -4 \sin^2 \frac{\pi k}{n}$$

Thus the diagonal matrix of eigenvalues Λ is defined by

$$A_{ik} = -4 \sin^2 \frac{\pi k}{n} \delta_{ik} \tag{37}$$

With these definitions

$$A = Q\Lambda Q^{-1} \text{ and } A = Q^{-1}AQ \ .$$

4.2 Sine case

$$A = \begin{vmatrix} -2 & 1 & & \bigcirc \\ 1 & \ddots & \ddots & \\ & \ddots & \ddots & 1 \\ \bigcirc & & 1 & -2 \end{vmatrix} \qquad d = \begin{vmatrix} d_0 \\ | \\ d_n \end{vmatrix} \tag{38}$$

$$Q_{sk} = (Q^{-1})_{ks} = \sqrt{\frac{2}{n}} \sin \frac{\pi ks}{n} \qquad 1 \leq k \leq n-1, \quad 1 \leq s \leq n-1$$

Q^{-1} is symmetric $\therefore$ $Q_{sk} = (Q^{-1})_{sk}$ and the operations of Fourier synthesis and analysis are identical

$$A_{ik} = -4 \sin^2 \frac{\pi k}{n} \delta_{ik} \tag{39}$$

4.3 Cosine case

$$A = \begin{bmatrix} -2 & 2 & & & \bigcirc \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ \bigcirc & & & 2 & -2 \end{bmatrix} \qquad d = \begin{vmatrix} d_0 \\ | \\ d_n \end{vmatrix} \tag{40}$$

$$Q_{sk} = (Q^{-1})_{sk} = \sqrt{\frac{2}{n}}\, P_k^2 \cos\frac{\pi sk}{n} \qquad (41)$$

note that (Q^{-1}) is <u>not</u> symmetric because of the P_k's. However, $Q_{sk} = (Q^{-1})_{sk}$ and analysis and synthesis are identical.

$$A_{ik} = -4\sin^2\frac{\pi k}{n}\,\delta_{ik} \qquad (42)$$

Sometimes one meets the matrix

$$A = \begin{bmatrix} -1 & 1 & & & \\ 1 & -2 & 1 & & \bigcirc \\ & \ddots & \ddots & \ddots & \\ & & & & 1 \\ \bigcirc & & & -2 & 1 \\ & & & 1 & -1 \end{bmatrix} \qquad \text{with} \qquad d = \begin{vmatrix} d_0 \\ \vdots \\ d_n \end{vmatrix} \qquad (43)$$

as the finite difference form of $\frac{d^2}{dx^2}$ with the zero slope conditions at the boundaries expressed as

$$u_s = u_{-s-1} \quad \text{and} \quad u_{n+s} = u_{n-s-1}$$

This amounts to requiring zero slope at s = -1/2 and n + 1/2 and leads to the eigenvectors

$$Q_{sk} = V(s,k) \propto \cos\frac{\pi k(s+1/2)}{n+1} \qquad 0 \le k \le n$$

$$(Q^{-1})_{sk} = W(s,k) \propto \cos\frac{\pi(k+1/2)s}{n+1}$$

We note that $Q_{sk} \neq (Q^{-1})_{sk}$ are the process if analysis and synthesis are different.

Thus although the matrix A in (43) has the elegance of symmetry over the matrix A in (40) its eigenvectors are not so well suited to Fourier analysis and it is advisable to convert the equations (43) to the form of (40) by multiplying the first and last equations by 2 and then use the eigenvectors (41).

5. MINIMUM MULTIPLICATION FOURIER ANALYSIS

Having described Fourier analysis and its application to the solution of certain equations we come to the main point of this paper, which is to describe an efficient method of performing this operation.

We have observed that the operation of Fourier analysis

$$\bar{b} = Q-4, \tag{44}$$

takes $2n^2$ arithmetic operations if all the elements of Q^{-1} are known and stored in an $(n \times n)$ array.

We will now describe an algorithm which computes $\bar{\underset{\sim}{b}}$ from $\underset{\sim}{b}$ with something like a tenth of these operations. This is done by restricting Fourier analysis to those eigenvectors defined in sections 1.1 to 1.3 and restricting the value of n to be of the form 12×2^q where q is an integer ≥ 0.

That is to say n is from the sequence

$$\{12, 24, 48, 96, 192, 384, 768, \cdots\} \tag{45}$$

The process is recursive and depends on removing a common factor (usually of 2) between the harmonic number k and the number of points n at each recurrence until finallywe are required to do a Fourier analysis on 12 points only. This last case is written out in longhand for program efficiency.

This process has the effect that on the first recurrence all the odd harmonics are calculated, on the second recurrence all the harmonics of the form 2 $\otimes$ an odd number, on the third recurrence all harmonics of the form 4$\otimes$ an odd number, and so on. Furthermore, by the use trigonometric identities it is possible to calculate 8 harmonics at the same time, with very little more work than would be required to calculate one harmonic.

Before describing the algorithm in detail it is necessary to establish some identities.

5.1 Removal of a common factor

Consider $$S(k,n,a_s) = \sum_{s=o}^{s=n} a_s \sin \frac{\pi}{2} \frac{sk}{n} \qquad (46)$$

and let there be a common factor of 2 between k and n so that

$$\left.\begin{aligned} k &= 2 \otimes g \\ n &= 2 \otimes h \end{aligned}\right\} \qquad (47)$$

$$S(k,n,a_s) = \sum_{s=o}^{s=h-1} a_s \sin \pi/2 \frac{sg}{h} + a_{n/2} \sin \pi/2 \quad g + \sum_{s=h+1}^{s=n} a_s \sin \pi/2 \frac{sg}{h} \qquad (48)$$

the third term in (48) may be expressed as

$$\sum_{s=0}^{h-1} a_{n-s} \sin \pi/2 \frac{g}{h} (2h-s) = \sum_{s=0}^{h-1} a_{n-s}\{\underset{=0}{\sin \pi g} \cos \frac{\pi}{2}\frac{gs}{h} - \cos \pi g \sin \frac{\pi}{2} \frac{gs}{h}\}$$

$$= \sum_{s=0}^{h-1} -(-1)^g a_{n-s} \sin \frac{\pi}{2}\frac{gs}{h} \qquad (49)$$

Substituting (49) in (48) we get

$$S(k,n,a_s) = \sum_{s=0}^{s=h-1} \{a_s-(-1)^g a_{n-s}\} \sin \frac{\pi}{2}\frac{gs}{h} + a_{n/2} \sin \pi/2\ g \qquad (50)$$

$$= \sum_{s=0}^{s=h} {}^2F_s(a) \sin \frac{\pi}{2}\frac{gs}{h} = \sum_{s=0}^{s=n/2} {}^2F_s(a) \sin \frac{\pi}{2}\, s \frac{k/2}{n/2} \quad (51)$$

where $F_s(a)$ stands for the s^{th} element of a new array of elements obtained from the original array by folding as follows:

$$\left\{\begin{array}{ll} {}^2F_s(a) = a_s - (-1)^g a_{n-s} & s = 0,1,\ldots,n/2-1 \\ {}^2F_{n/2}(a) = a_{n/2} & \end{array}\right. \qquad (52)$$

In equation (51) we have reduced the original summation up to n to a summation up to $h = \frac{n}{2}$ but over a set of folded values ${}^2F_s(a)$.

Using the S notation this is

$$S(k,n,a_s) = S(k/2,\ n/2,\ {}^2F_s(a)) \qquad (53)$$

and a common factor of 2 has been removed.

To take account of the two possibilities of g being odd or even we introduce the 'Twofold' operation on an array, which is a procedure of the algorithm.

5.2 Twofold (L,N,A);

The numbers $A_\ell, A_{\ell+1}, \ldots, A_{\ell+n}$ may be imagined to be folded as follows into a new array B

$$\left.\begin{array}{ll} {}^2F^-(A) = B_{\ell+i} = A_{\ell+i} - A_{\ell+n-i} & \\ {}^2F^+(A) \left\{ \begin{array}{l} B_{\ell+n-i} = A_{\ell+i} + A_{\ell+n-i} \\ \\ B_{\ell+\frac{n}{2}} = A_{\ell+\frac{n}{2}} \end{array}\right. & \end{array}\right\} i = 0,1,\ldots,\frac{n}{2}-1 \qquad (54)$$

Here ℓ is simply a common origin and may be ignored as regards understanding the process,

The elements $B_\ell, B_{\ell+1}, \ldots, B_{\ell+\frac{n}{2}}$ will be seen to be the elements ${}^2F_s(a)$ for g even (denoted by ${}^2F_s^-(a)$) while the elements $B_{\ell+n}, B_{\ell+n-1}, \ldots, B_{\ell+\frac{n}{2}}$ are the elements of $F_s(a)$ for g odd (denoted by ${}^2F_s^+(a)$). The purpose of writing the elements for g odd in reverse order is to enable the new elements B to overwrite immediately the corresponding element in the array A, so that in fact there is no need to introduce the array B in the procedure.

To see the effect of the operation Twofold we will apply it to a few typical harmonics.

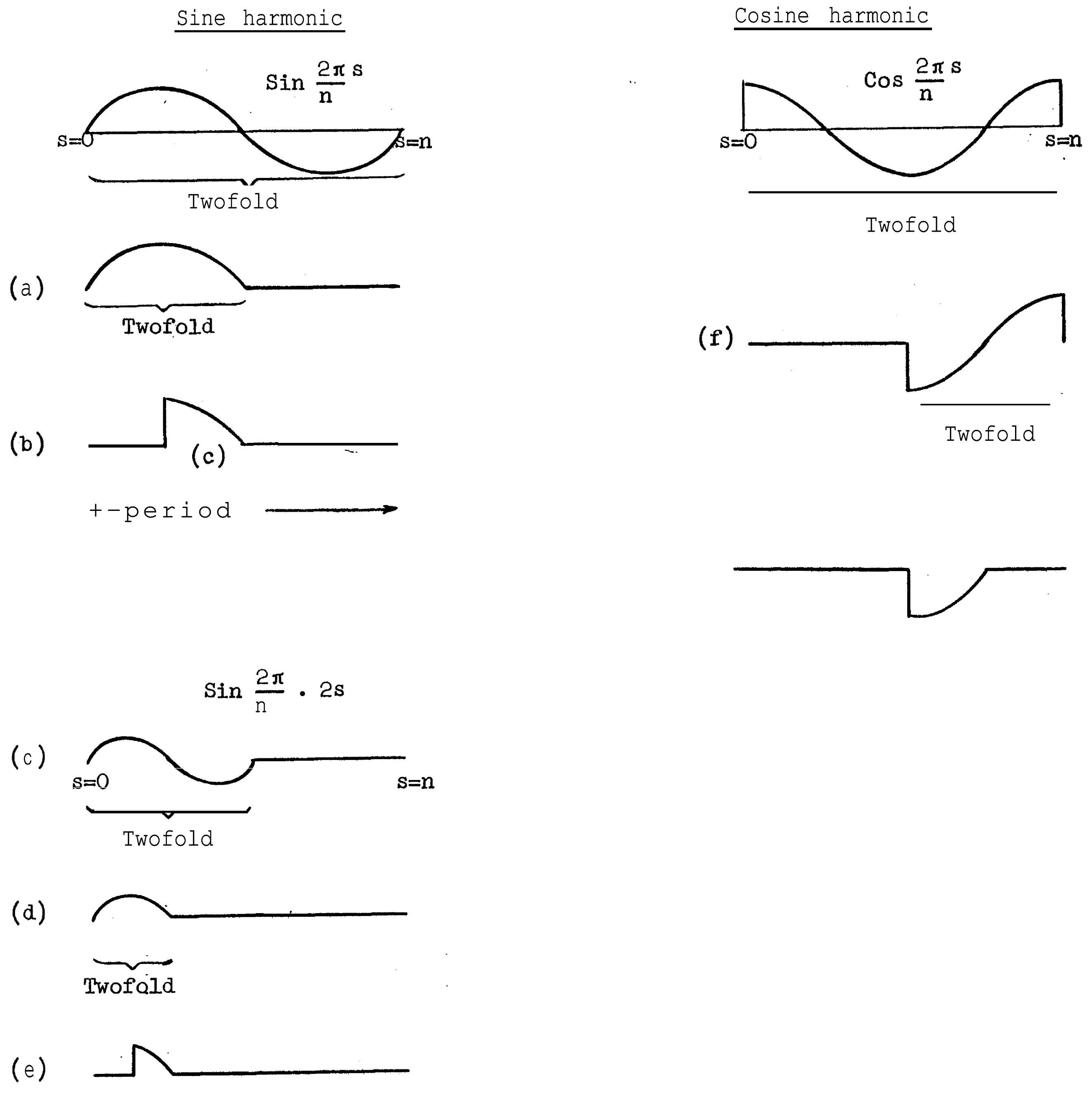


Fig. 1 - The Twofold operation applied to some typical harmonics.

It can be seen from Fig. 1 that one operation of Twofold on the original data immediately separates the sine from the cosine harmonics, the sines appearing on the left side of the fold and the cosines to the right side (see Fig. 1 (a) and (f)).

A further fold applied to the sine harmonic will separate out all the harmonics with an odd number of π^s in the half period. These harmonics will appear in the right half of the two fold (see Fig, 1 (b)). All other harmonics appear on the left of the last two fold (i.e., in the first quarter period).

The process is continued by applying a further twofold to the first quarter period when all harmonics with an odd number of π^s in the first quarter period will separate to the right of the twofold (see Fig. 1 (e)) and all others to the left.

Similar considerations apply to the separating off of the cosine harmonics and in both cases Twofolding can continue until only 12 points are left when the last 12 harmonics are calculated by a special routine.

5.3 Threefold (L,N,A)

As 3 is always a factor of the number n it is also possible to remove a common factor of 3 between k and n by a folding process as follows.

$$S(k,n,a_s) = \sum_{s=o}^{s=n} a_s \sin \frac{\pi}{2} \frac{sk}{n} \qquad s = 0,1,\ldots,n \qquad (55)$$

now let

$$k = 3 \otimes g$$

$$n = 3 \otimes h$$

$$S(k,n,a) = \sum_{s=0}^{s=h-1} a_s \sin \frac{\pi}{2} \frac{sg}{h} + a_h \sin \frac{\pi}{2} g$$

$$+ \sum_{s}^{h-1} a_{2h-s} \sin \frac{\pi}{2} \frac{g}{h} (2h-s) + a_{2h} \sin \pi g$$

$$+ \sum_{s=0}^{h-1} a_{2h+s} \sin \frac{\pi}{2} \frac{g}{h} (2h+s) + a_{3h} \sin \frac{3\pi}{2} g$$

$$S(k,n,a) = \sum_{s=0}^{h-1} a_s \sin \frac{\pi}{2} \frac{sg}{h} + ah \sin \frac{\pi}{2} g$$

$$+ \sum_{s=0}^{h-1} a_{2h-s} (-1)^g \sin \frac{\pi}{2} \frac{sg}{h}$$

$$+ \sum_{s=0}^{h-1} -a_{2h+s} (-1)^g \sin \frac{\pi}{2} \frac{sg}{h} + (-1)^g a_{3h} \sin \frac{\pi}{2} g \quad (56)$$

The three fold will only be used on the right hand section following a Twofold for which it is known that g is odd. Making use of this fact we have

$$S(k,n,a) = \sum_{s=0}^{h} {}^3F_s(a) \sin \frac{\pi}{2} \frac{sg}{h} = \sum_{s=0}^{n/3} {}^3F_s(a) \sin \frac{\pi}{2} s \frac{k/3}{n/3} \quad (57)$$

where ${}^3F_s(a)$ are the elements of the Threefold applied to a_s, defined as follows

$$\begin{cases} {}^3F_s(a) = a_s + a_{2h-s} - a_{2h+s} & s = 1,2,\dots,h-1 \\ \quad\dots & \\ {}^3F_h(a) = a_h - a_n & \end{cases} \qquad (58)$$

Thus $S(k,n,a) = S(k/3, n/3, {}^3F(a))$ and a common factor 3 has been removed.

In view of the fact that the right hand section of a Twofold has its elements written in reverse order the procedure Threefold must reverse the indexing and is defined as follows

$$\begin{cases} A_{\ell+n-i} \leftarrow A_{\ell+n-i} + A_{\ell+n-2h+i} - A_{\ell+n-2h-i} & i = 1,2,\dots,h-1 \\ A_{\ell+n-h} \leftarrow A_{\ell+n-h} - A_{\ell} & \end{cases} \qquad (59)$$

5.4 <u>Eightk</u> (k,N,A)

Having eliminated all common factors by the operation of Two or Threefolding, processes incidentally that require only additions, we are left with a reduced number of points from which we must compute the harmonic amplitudes. We will restrict consideration to the evaluation of the sine summation $S(k,n,a)$ when given k,n,a because we shall see in section 5.5 how this can also be used to evaluate the corresponding cosine summation.

$$\text{Let } \overline{\varphi}_\ell = S(2\ell,n,a) = \sum_{s=0}^{s=n} a_s \sin \frac{\pi s \ell}{n} \qquad s = 0,1,\dots,n \qquad (60)$$

where now it happens to be more convenient to let k = 2ℓ and ℓ takes the values $0,1,\ldots,n$. As mentioned earlier it is possible to calculate eight harmonic amplitudes at a time and these are

$$\bar{\varphi}_{\ell},\ \bar{\varphi}_{n/4-\ell},\ \bar{\varphi}_{n/4+\ell},\ \bar{\varphi}_{n/2-\ell},\ \bar{\varphi}_{n/2+\ell},\ \bar{\varphi}_{3n/4-\ell},\ \bar{\varphi}_{3n/4+\ell},\ \bar{\varphi}_{n-\ell} \quad (70)$$

The harmonics being picked off an equal distance ℓ to either side of the key values $\bar{\varphi}_0, \bar{\varphi}_{n/4}, \bar{\varphi}_{n/2}, \bar{\varphi}_{3n/4}, \bar{\varphi}_n$.

Let us define the quantities

$$A^{(\ell)} = \sum_{s \bmod 4=2} a_s \sin\frac{\pi s\ell}{n}$$

$$B^{(\ell)} = \sum_{s \bmod 4=0} a_s \sin\frac{\pi s\ell}{n}$$

$$E^{(\ell)} = \sum_{s \bmod 4=1 \text{ or } 3} a_s \sin\frac{\pi s\ell}{n}$$

$$F^{(\ell)} = \sum_{s \bmod 4=1} a_s \cos\frac{\pi s\ell}{n} - \sum_{s \bmod 4=3} a_s \cos\frac{\pi s\ell}{n} \quad (71)$$

where each sum is taken over all s in the range 0 to n that satisfies the modulo condition given.

Then

$$\bar{\varphi}_{\ell} = \sum_{s=0}^{n} a_s \sin\frac{\pi s\ell}{n} = (A^{(\ell)}+B^{(\ell)}) + E^{(\ell)} \quad (72)$$

$$\bar{\varphi}_{n-\ell} = \sum_{s=o}^{n} a_s \sin\frac{\pi s(n-\ell)}{n} = -\sum_{s=o}^{n} - a_s \cos \pi s \sin\frac{\pi s\ell}{n}$$

(73)

$$= -\sum_{s \bmod 2=0} a_s \sin\frac{\pi s\ell}{n} + \sum_{s \bmod 2=1} a_s \sin\frac{\pi s\ell}{n} = -(A^{(\ell)}+B^{(\ell)}) + E^{(\ell)}$$

$$\bar{\varphi}_{n/2\mp\ell} = \sum_{s=o} a_s \sin \frac{\pi s}{n}\left(\frac{n}{2} \mp \ell\right) = \sum_{s=o}^{n} a_s \left\{\sin \frac{\pi}{2} s \cos \frac{\pi s \ell}{n} \mp \cos \frac{\pi}{2} s \sin \frac{\pi s \ell}{n}\right\}$$

$$= \mp \sum_{s \bmod 2=0} a_s \cos \frac{\pi}{2} s \sin \frac{\pi s \ell}{n} + \sum_{s \bmod 2=1} a_s \sin \frac{\pi}{2} s \cos \frac{\pi s \ell}{n}$$

$$= \mp \sum_{s \bmod 4=0} a_s \sin \frac{\pi s \ell}{n} \pm \sum_{s \bmod 4=2} a_s \sin \frac{\pi s \ell}{n} + \sum_{s \bmod 4=1} a_s \cos \frac{\pi s \ell}{n}$$

$$\sum_{s \bmod 4=3} a_s \cos \frac{\pi s \ell}{n}$$

$$\therefore \quad \bar{\varphi}_{n/2\mp\ell} = \pm \left(B^{(\ell)} - A^{(\ell)}\right) + F^{(\ell)} \qquad (74)$$

To obtain the remaining harmonics we determine expressions for $A^{(n/4-\ell)}$, $B^{(n/4-\ell)}$, $E^{(n/4-\ell)}$ and $F^{(n/4-\ell)}$ and use these values again in equations (72) to (74)

$$A^{(n/4-\ell)} = \sum_{s \bmod 4=2} a_s \sin \frac{\pi s(n/4-\ell)}{n} = \sum_{s \bmod 4=2} a_s \left\{\sin \frac{\pi s}{4} \cos \frac{\pi s \ell}{n}\right.$$

$$\left. - \cos \frac{\pi s}{4} \sin \frac{\pi s \ell}{n}\right\}$$

$$= \sum_{s \bmod 4=2} a_s \sin \frac{\pi s}{4} \cos \frac{\pi s \ell}{n} = \sum_{s \bmod 8=2} a_s \cos \frac{\pi s \ell}{n}$$

$$\sum_{s \bmod 8=6} a_s \cos \frac{\pi s \ell}{n} \qquad (75)$$

similarly

$$B^{(n/4-\ell)} = \sum_{s \bmod 4=0} a_s \sin \frac{\pi s}{n} (n/4-\ell) = - \sum_{s \bmod 8=0} a_s \sin \frac{\pi s \ell}{n}$$

$$+ \sum_{s \bmod 8=4} a_s \sin \frac{\pi s \ell}{n} \qquad (76)$$

Introducing now the notation

$$S_j^{(\ell)} = \sum_{s \bmod 8=j} a_s \sin\frac{\pi s \ell}{n}$$

$$C_j^{(\ell)} = \sum_{s \bmod 8=j} a_s \cos\frac{\pi s \ell}{n} \tag{77}$$

We can express the results of equations (71) as

$$A^{(\ell)} = S_2^{(\ell)} + S_6^{(\ell)}$$

$$B^{(\ell)} = S_0^{(\ell)} + S_4^{(\ell)}$$

$$E^{(\ell)} = (S_1^{(\ell)}+S_5^{(\ell)}) + (S_3^{(\ell)}+S_7^{(\ell)}) \tag{78}$$

$$F^{(\ell)} = (C_1^{(\ell)}+C_5^{(\ell)}) - (C_3^{(\ell)}+C_7^{(\ell)})$$

from which we compute

$$\bar{\varphi}_\ell = (A+B) + E \qquad \bar{\varphi}_{n/2+\ell} = -(A-B) + F$$

$$\bar{\varphi}_{n-\ell} = -(A+B) + E \qquad \bar{\varphi}_{n/2-\ell} = (A-B) + F \tag{79}$$

Equations (75) and (76) become

$$A^{(n/4-\ell)} = C_2^{(\ell)} - C_6^{(\ell)}$$

$$B^{(n/4-\ell)} = -S_0^{(\ell)} + S_4^{(\ell)} \tag{80}$$

and one may also show that .

$$E^{(n/4-\ell)} = \frac{1}{\sqrt{2}} [(C_1^{(\ell)}-C_5^{(\ell)}) - (S_1^{(\ell)}-S_5^{(\ell)}) + (C_3^{(\ell)}-C_7^{(\ell)}) + (S_3^{(\ell)}-S_7^{(\ell)})]$$

$$F^{(n/4-\ell)} = \frac{1}{\sqrt{2}} [(C_1^{(\ell)}-C_5^{(\ell)}) + (S_1^{(\ell)}-S_5^{(\ell)}) + (C_3^{(\ell)}-C_7^{(\ell)}) - (S_3^{(\ell)}-S_7^{(\ell)})]$$

where $\frac{1}{\sqrt{2}}$ comes in as the value of sin $\pi/4$ and cos $\pi/4$.

From these new A, B, E, F one calculates

$$\bar{\varphi}_{n/4-\ell} = (A+B) + E , \qquad \bar{\varphi}_{3n/4-\ell} = -(A+B) + F$$
$$\bar{\varphi}_{3n/4+\ell} = -(A+B) + E , \qquad \varphi_{n/4+\ell} = (A-B) + F \tag{81}$$

Formulae (77) to (80) constitute the algorithm for calculating the eight harmonic amplitudes.

5.5 The cosine harmonics

Cosine summations of the form

$$C(g,h,a_s) = \sum_{s=0}^{s=h} a_s \cos \frac{\pi}{2}\frac{sg}{h} \qquad g = 0,1,\dots,h \tag{82}$$

are required for the cosine harmonics. These may be calculated using a sine summing routine such as EIGHTK by applying it to the coefficients in the reverse order and making a simple alteration of sign.

$$C(g,h,A) = \sum_{s=0}^{s=h} a_{h-s} \cos \frac{\pi}{2} \frac{(h-s)}{h} g$$

$$= \sum_{s=0}^{s=h} a_{h-s} \{\cos \frac{\pi}{2} g \cos \frac{\pi}{2} \frac{sg}{h} + \sin \frac{\pi}{2} g \sin \frac{\pi}{2} \frac{sg}{h}\}$$

It will be observed in the algorithm that the harmonics are only evaluated after Twofolding has reduced g to be an odd number. Using this fact we have

$$C(g,h,a_s) = \sin \frac{\pi}{2} g \sum_{s=0}^{s=h} a_{h-s} \sin \frac{\pi}{2} \frac{sg}{h}$$

$$C(g,h,a_s) = \sin \frac{\pi}{2} g \; S(g,h,a_{h-s}) \tag{83}$$

$$= (-1)^{\frac{g-1}{2}} S(g,h,a_{h-s}) \qquad g \text{ odd}$$

5.6 <u>The procedure value</u> - VAL(L,N,A,M,Y,SI)

The procedure VAL is the ALGOL procedure which evaluates all the summations of the type $S(g,h,a)$.

It performs a sine analysis on the input values $Z_{\ell+1}, Z_{\ell+2}, \ldots, Z_{\ell+n\ 1}$ and stores the harmonic amplitudes in the array $Y_{m+1}, Y_{m+2}, \ldots, Y_{m+n-1}$, using as eigenvectors the contents of the array SI, which is assumed to contain the first quarter period of the first sine harmonic.

The procedure calculates

$$Y_{m+k} = \sum_{s=0}^{s=n} Z_{\ell+s} SI_{ks}$$

One use of VAL is in the call of the procedure FOURIER12 with BC = 1, which performs the Fourier sine analysis of synthesis defined in section 1.2 and used in section 3.2 and 4.2.

In this case SI is filled with the normalized function:

$$SI_t = \sqrt{\frac{2}{n}} \sin \frac{\pi t}{n} \tag{84}$$

and the procedure VAL is entered

VAL(O,N,Z,O,Y,SI)

when the Fourier sine analysis will be performed on $Z_1,\ldots,Z_{n-1}$ with the harmonic amplitudes in $Y_1,\ldots,Y_{n\ 1}$

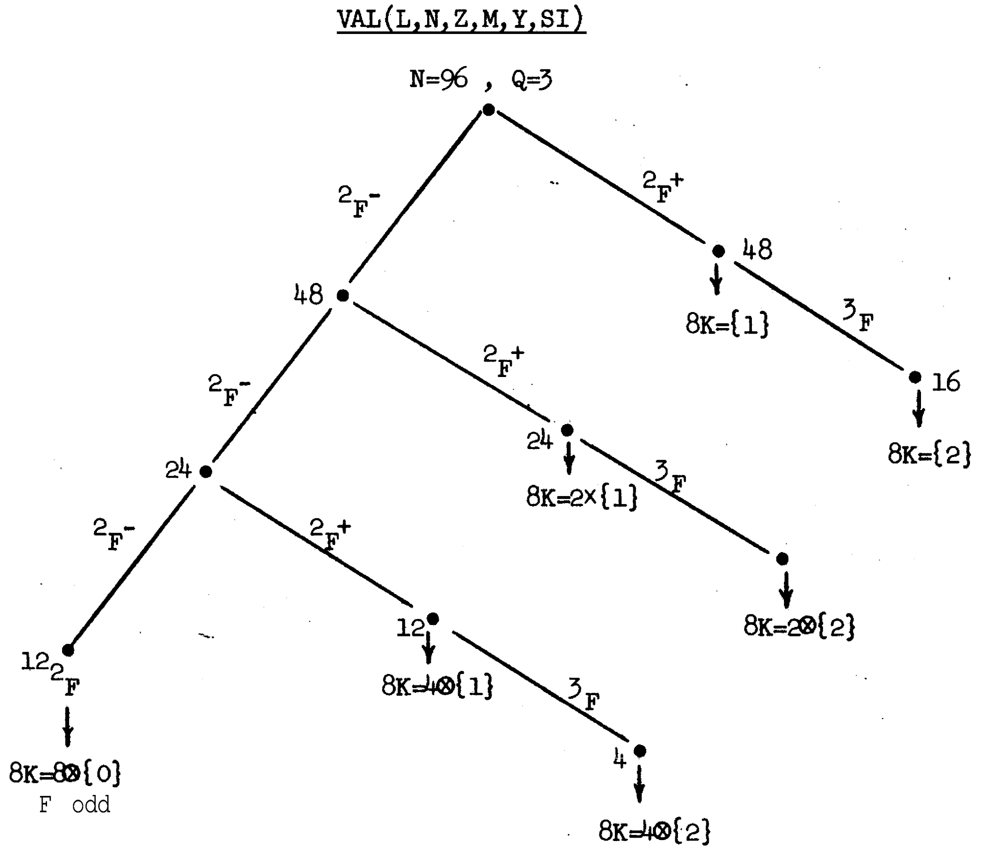
$$\text{where}\qquad Y_k = \sum_{s=1}^{s=n-1} Z_s\sqrt{\frac{2}{n}}\,\sin\frac{\pi sk}{n} \tag{85}$$

The procedure may be understood by reference to Fig., 2 where the case of 96 points is illustrated.

The basic recurrence of the process starts with a Twofold, initially on the original 96 points. The 48 values obtained by subtraction on the left side of the Twofold, and indicated by the shorthand ${}^{2}F^{-}$, are the input points for the next stage of the recurrence.

The 48 values obtained by addition on the right side of the Twofold and indicated by the shorthard ${}^{2}F^{-}$, are used to compute all the harmonics with odd **k**. First all odd harmonics that are not multiples **of** 3 are found, eight at a time, by entry to the procedure EIGHTK. This is defined as the set of numbers {1}. Secondly all odd harmonics that are multiples of 3 are found, by first performing a Threefold (${}^{3}F$) and then entering EIGHTK with the reduced number of **points**. This set of numbers is defined as {2}. In both cases EIGHTK need only be entered for values of k less than **N/8** because the remaining values are filled in automatically by the procedure.

The next stage of the recurrence works on the 48 points from the subtractive side of the previous Twofold and computes similarly all



FOURIER12(1,Q,X,Y);

$$X_0 \leftarrow X_n \leftarrow 0 \;;$$

$$SI_t \leftarrow \sqrt{\frac{2}{n}}\ \sin \frac{\pi t}{n} \;;$$

VAL(0,N,X,0,Y,SI);

$$Y_0 \leftarrow Y_n \leftarrow 0 \;;$$

$$Y_0, \ldots, Y_n$$

FIG2 - The procedure **VAL** and SINE **ANAL/SYNTH BC=1**

{1} - the set of all odd **numbers** which are not multiples of three
{2} - the set of **all** odd numbers which are **multiples** of three
{0} - the set of integers 1 to 12.
8K= - the procedure 8K is entered to obtain-the harmonics from the **indicated** set

2_F^- - the subtractive or left 'side of a Twofold

2_F^+ - the additive or right side of a Twofold

3_F - a threefold

24● - the number of points involved

harmonics of the form $2 \otimes$ an odd number. The process repeats until we are left with 12 points from which the harmonics $k = 8 \otimes \{1,2,\ldots,11\}$ are computed. This calculation is written out in full within EIGHTK and is selected by the artifice of making the input parameter F an odd number, in all other circumstances it happens that F is even.

It should be clear that the increase of the number of points by a factor of 2 simply adds a further limb to the recurrence tree of Fig. 2 and that the number of points may be increased in this manner until some machine limit is reached. On the B5000 using one dimensional arrays which have a maximum length of 1023 this limit is soon reached, namely when n = 768.

5.7 The procedure SLOPE(L,N,Z,M,Y,SI)

The procedure SLOPE is the ALGOL procedure which evaluates all the summations $C(g,h,a_s)$. It performs a cosine analysis on the input values $Z_\ell, Z_{\ell+1}, \ldots, Z_{\ell+n}$ and stores the harmonic amplitudes in $Y_m, Y_{m+1}, \ldots, Y_{m+n}$, using as eigenvector the array SI.

The procedure calculates

$$Y_{m+k} = \sum_{s=0}^{s=n} Z_{\ell+s} \, SI_{ks+n/2} \qquad (86)$$

One use of SLOPE is in the call of the procedure FOURIER12 with BC = 2, which performs the Fourier cosine analysis or synthesis defined in section 1.3 and used in section 3.3 and 4.3.

In this case the array SI is filled with the function

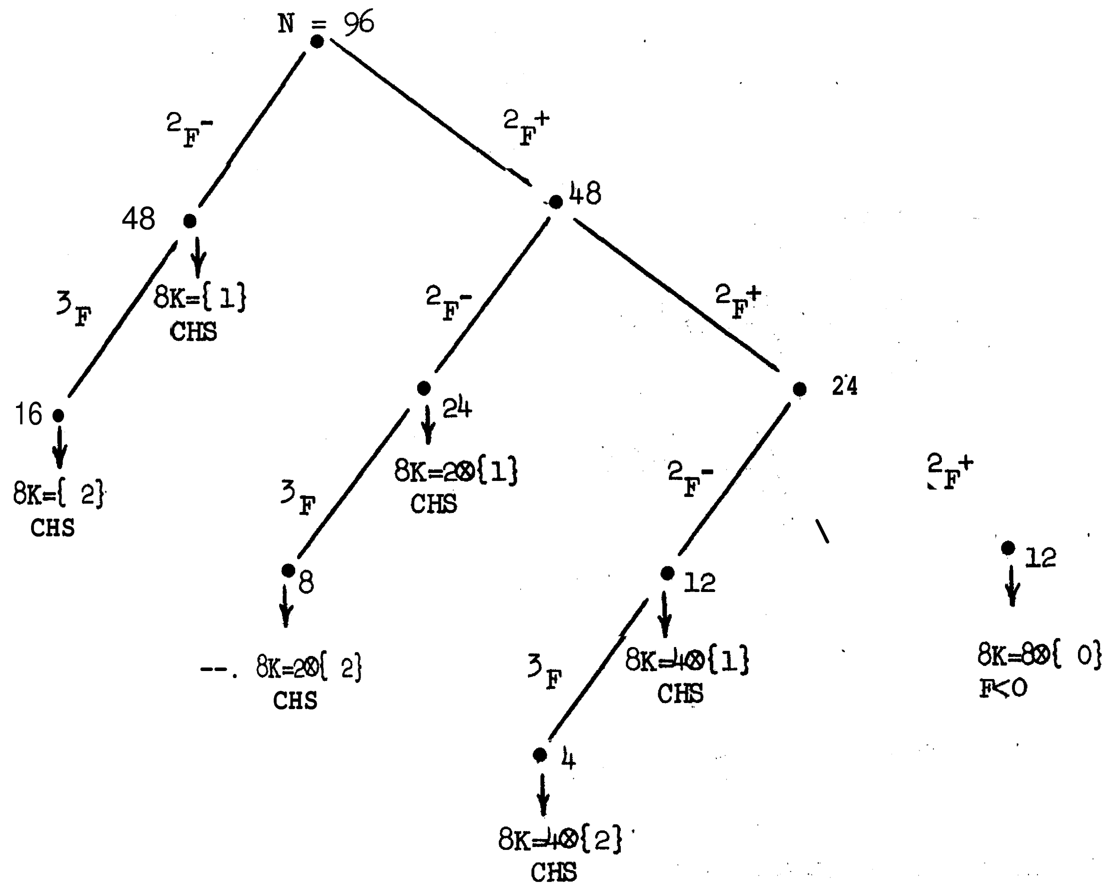
$$SI_t = \sqrt{\frac{2}{n}} \sin \frac{\pi t}{n} , \qquad (87)$$

The end values Z_0 and Z_n are halved to take account of the factor P_s^2 in the definition of the normalized function given in equation (9) and the procedure slope is entered .

```
SLOPE(O,N,Z,O,Y,SI) ;
```

when a Fourier cosine analysis is performed on the elements $Z_0,\ldots,Z_n$ with the harmonic amplitudes in $Y_0,\ldots,Y_n$. However, due to the fact that the input values were not reversed in order before the procedure EIGHTK is applied, as is required by equation (83) it is necessary finally to **reverse the** sign of all odd harmonics to get the correct result.

The detailed operation of slope should be clear from Fig. 3. It is only necessary to say that CHS stands for the change of sign required by equation (83) in order that EIGHTK calculate cosine harmonics instead of sine harmonics, and that the special case of a 12 point cosine analysis is selected in the routine EIGHTK by making the input parameter F negative. In all other circumstances it happens that F is positive, SLOPE(L,N,Z,M,Y,SI);



FOURIER12(2,Q,X,Y);

$$X_0 \leftarrow X_0/2 \;,\; X_n \leftarrow X_n/2 \;;$$

$$SI_t \leftarrow \sqrt{\frac{2}{n}} \sin\frac{\pi t}{n} \;;$$

SLOPE(0,N,X,0,Y,SI);

change sign odd;

$Y_0, \ldots, Y_n$

FIG 3 The procedure SLOPE and COSINE ANAL/SYNTH BC=2

Notation as FIG 2

CHS - change at sign required by equation (83)

5.8 Periodic analysis and-synthesis

In the periodic analysis defined in section 1.1 the cosine harmonics are given by

$$\bar{\varphi}_k = \sum_{s=0}^{s=n-1} P_k\sqrt{\frac{2}{n}} \cos\frac{2\pi sk}{n}\,\varphi_s \qquad \text{for} \quad 0 \leq k \leq n/2 \qquad (88)$$

$$= \sum_{s=1}^{n/2-1} P_k\sqrt{\frac{2}{n}} \cos\frac{2\pi sk}{n}\,\varphi_s + \sum_{s=1}^{n/2-1} P_k\sqrt{\frac{2}{n}} \cos\frac{2\pi k}{n}\,(n-s)\varphi_s +$$

$$+ P_k\sqrt{\frac{2}{n}} \cos \pi k\, \varphi_{n/2} + P_k\sqrt{\frac{2}{n}}\,\varphi_0$$

$$k = P_k\sqrt{\frac{2}{n}}\,\varphi_0 + \sum_{s=1}^{n/2-1} \{\varphi_s+\varphi_{n\ s}\}\, P_k\sqrt{\frac{2}{n}} \cos\frac{2\pi k}{n} + P_k\sqrt{\frac{2}{n}} \cos \pi k\, \varphi_{n/2}$$

φ_s is initially given in the range s = 0 to n-1. If however we now introduce $\varphi_n = \varphi_{0/2}$ and also make $\varphi_0 = \varphi_{0/2}$ then

$$\bar{\varphi}_k = \sum_{s=0}^{n/2-1} \{\varphi_s+\varphi_{n-s}\}\, P_k\sqrt{\frac{2}{n}} \cos\frac{2\pi k}{n} + P_k\sqrt{\frac{2}{n}} \cos \pi k\, \varphi_{n/2} \qquad (89)$$

Both terms in equation (89) may be neatly combined if we recall the definition of the Twofold in section **5.2** equation (54) whence

$$\left.\begin{array}{ll} {}^2F^+_{n-s}(\varphi) = \varphi_s + \varphi_{n-s} & s = 0,1,\ldots,n/2-1 \\ {}^2F^+_{n/2}(\varphi) = \varphi_{n/2} & \end{array}\right\} \qquad (90)$$

and

$$\bar{\varphi}_k = \sum_{s=0}^{n/2} {}^2F^+_{n-s}(\varphi)\, P_k\sqrt{\frac{2}{n}} \cos\frac{2\pi k}{n}$$

$$\bar{\varphi}_k = P_k\sqrt{\frac{2}{n}}\, C(2k,\ n/2,\ {}^2F^+_{n\ s}(\varphi)) \qquad (91)$$

Equation (91) shows that the cosine terms of the periodic analysis may be computed from the additive side of a twofold on the original data, as is shown in Fig. 4.

The sine harmonics **amplitudes** are defined by

$$\bar{\varphi}_{n/2+k} = \sum_{s=1}^{s=n-1} \varphi_s\sqrt{\frac{2}{n}} \sin\frac{2\pi sk}{n} \qquad 1 \le k \le n/2-1 \qquad (92)$$

$$= \sum_{s=1}^{s=n/2-1} \varphi_s\sqrt{\frac{2}{n}} \sin\frac{2\pi sk}{n} + \sum_{s=1}^{s=n/2-1} \varphi_s\sqrt{\frac{2}{n}} \sin\frac{2\pi k}{n}(n-s) + \varphi_{n/2} \otimes 0$$

$$= \sum_{s=0}^{n/2} \{\varphi_s - \varphi_{n-s}\}\sqrt{\frac{2}{n}} \sin\frac{2\pi sk}{n}$$

$$\bar{\varphi}_{n/2+k} = \sqrt{\frac{2}{n}}\ S(2k,\ n/2,\ {}^2F^-_s(\varphi)) \qquad (93)$$

Thus the sine harmonics can be computed from the subtractive side of a Twofold on the original data,

Periodic Fourier synthesis is defined by

$$\varphi_s = \sum_{k=0}^{k=n/2} P_k\sqrt{\frac{2}{n}} \cos\frac{2\pi sk}{n}\, \bar{\varphi}_k + \sum_{k=n/2+1}^{k=n} \sqrt{\frac{2}{n}} \sin\frac{2\pi s}{n}(k-n/2)\varphi_k \quad (94)$$

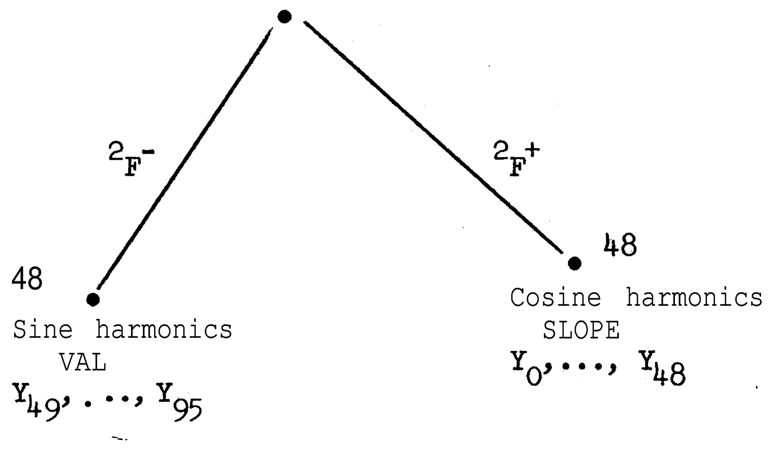
$$s = 0,1,\ldots,n$$

FOURIER12(3,Q,X,Y);

$$SI_t \leftarrow \sqrt{\frac{2}{n}}\sin\frac{2\pi t}{n} \;;$$

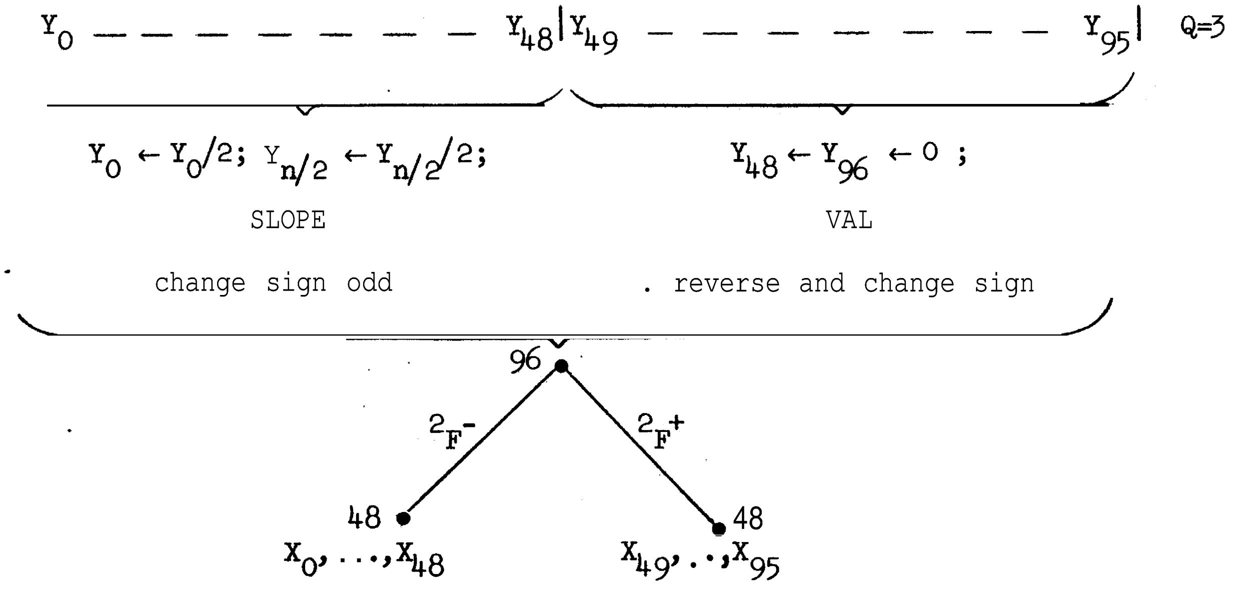
$$X_0 \leftarrow X_n \leftarrow X_0/2 \;;$$

$$N = 96 \;,\; Q = 3$$



FOURIER12(4,Q,Y,X);

$$SI_t \leftarrow \sqrt{\frac{2}{n}}\; \sin\frac{2\pi t}{n} \;;$$



Y_0 — — — — — — — — Y_{48} | Y_{49} — — — — — — — — Y_{95} |
Q=3
$Y_0 \leftarrow Y_0/2;\; Y_{n/2} \leftarrow Y_{n/2}/2;$
SLOPE
change sign odd
$Y_{48} \leftarrow Y_{96} \leftarrow 0 \;;$
VAL
reverse and change sign
96
$^2F^-$
$^2F^+$
48
48
$X_0, \ldots, X_{48}$
$X_{49}, \ldots, X_{95}$

PERIODIC **SYNTHESIS BC=4**

FIG. 4.

in order to reduce the 1st and 2nd sums to those that can be performed by the procedures VAL and SLOPE it is necessary to reduce the range of variation of s to that of k.

Consider

$$\varphi_{n-s} = \sum_{k=0}^{k=n/2} P_k \sqrt{\frac{2}{n}} \cos \frac{2\pi k}{n} (n-s)\ k + \sum_{k=n/2-1}^{k=n} \sqrt{\frac{2}{n}} \sin \frac{2\pi}{n} (k-n/2)(n-s)\ \bar{\varphi}_k$$

$$s = 0,1,\ldots,n/2$$

$$\varphi_{n-s} = \sum_{k=0}^{k=n/2} P_k \sqrt{\frac{2}{n}} \cos \frac{2\pi sk}{n} \bar{\varphi}_k - \sum_{k=n/2-1}^{k=n} \sqrt{\frac{2}{n}} \sin \frac{2\pi s}{n} (k-n/2)\ \bar{\varphi}_k$$

and (94) becomes

$$\left.\begin{aligned} \varphi_{n-s} &= \sqrt{\frac{2}{n}}\, C(2s,\ n/2,\ P_k\bar{\varphi}_k) - \sqrt{\frac{2}{n}}\, S(2s,\ n/2,\ \bar{\varphi}_k) \\ \varphi_s &= \sqrt{\frac{2}{n}}\, C(2s,\ n/2,\ P_k\bar{\varphi}_k) + \sqrt{\frac{2}{n}}\, S(2s,\ n/2,\ \bar{\varphi}_k) \end{aligned}\right\} \quad (95)$$

*where the function S must be understood to be operating on the variables $\varphi_{n/2+1},\ldots,\varphi_{n-1}$ in contradiction to its definition in equation (55).

The summations C are performed by the procedure call

$$Y_0 \leftarrow Y_0/2\ ; \quad Y_{n/2} \leftarrow Y_{n/2}/2\ ;$$
SLOPE(0, n/2, Y,0,X,SI) ; (96)
change sign odd ;

acting on the 1st n/2 harmonic components $Y_0,\ldots,Y_{n/2}$ with the results placed in $Y_0,\ldots,Y_{n/2}$.

The summations S are performed by the call

$$Y_{n/3} \leftarrow Y_n \leftarrow 0 \qquad (97)$$

VAL(n/2, n/2, Y, n/2,Y,SI) ;

with the results placed in $Y_{n/2+1},\ldots,Y_{n-1}$.

We note from (95) that φ_s $(s=0,1,\ldots,n/2)$ is obtained by an addition of C and S and that φ_{n-s} is obtained by a subtraction. A Twofold operating on the whole sequence $\varphi_0,\ldots,\varphi_n$ has this effect if the sine summation terms are first reversed and have their signs changed. This process is illustrated in the lower part of Fig. 4.

6. OPERATION COUNTS

Table I gives the complete information on the number of arithmetic operations required for different values of Q and BC. The operations counted are only those used directly in the arithmetic of the summations and do not include any additions or multiplications which are concerned with indexing and 'housekeeping' operations. They therefore represent the best that can be achieved in an efficiently written program in machine code.

For comparison purposes the number of operations is compared with that which would be performed in the direct evaluation of the summations in for example

$$\bar{\varphi}_k = \sum_{s=0}^{s=N} \varphi_s \sin\frac{\pi ks}{N} \qquad k = 0,1,\ldots,N \qquad (98)$$

Such an evaluation requires n^2 additions and n^2 multiplications for all values of k.

We introduce the factor of simplication

$$F = \frac{2N^2}{\text{total number of operations}} \qquad (99)$$

and the weighted comparison for 7090, using 15μs for addition and 25μs for multiplication as follows

$$F7090 = \frac{15\ N^2 + 25\ N^2}{15 \times (\#\text{ of adds}) + 25 \otimes (\#\text{ of mults})} \qquad (100)$$

The results for the total number of operations are shown in Fig. 5 and 6 together with-the theoretical asymptote for large N.

It will be observed that the periodic analysis requires asymptotically only half the number of operations of a sine or cosine analysis and that asymptotically the number of operations increases as N^2.

The following empirical fits have been made

Sine or Cosine analysis

$$\text{total \# operations} = \frac{2\ N^2}{18} + 5N \qquad (101)$$

-Periodic analysis

$$\text{total \# operations} = \frac{2\ N^2}{36} + 6.3\ N \qquad (102)$$

* Sine Analysis BC = 1

Q	N	Adds	Mults	Total Ops	F	F7090	max error $\times 10^{11}$	Time Secs
0	12	51	27	78	3.7	4.0	15	0.8
1	24	135	57	192	6.0	6.7	15	1.0
2	48	339	150	489	9.0	9.9	17	1.3
3	96	977	565	1,542	11.9	12.8	27	2.4
4	192	3,016	2,157	5,173	14.2	14.9	25	5.2
5	384	10,231	8,477	18,708	15.8	16.1	23	1404
6	768	37,206	33,661	70,867	16.6	16.8	22	43

* Cosine Analysis BC = 2

0	12	44	21	65	4.4	4.9	10	0.5
1	24	128	51	179	6.4	7.2	14	0.55
2	48	345	156	501	9.2	10.2	14	0.9
3	96	970	559	1,529	12.0	12.9	15	1.8
4	192	3,009	2,151	5,160	14.3	14.9	18	5.0
5	384	10,224	8,471	18,695	15.8	16.1	21	15
6	768	37,199	33,655	70,854	16.6	16.8	24	43

* Periodic Analysis BC = 3 or 4

1	24	110	37	147	7.8	8.9	2	0.80
2	48	302	98	400	11.5	13.2	12	0.85
3	96	780	305	1,085	16.9	19.1	21	1.5
I4	192	2,130	1,113	3,243	22.7	24.6	18	3.4
5	384	6,400	4,297	10,697	27.5	29.0	22	8.j
6	768	21,214	16,937	38,151	30.9	31.8	20	26

TABLE I

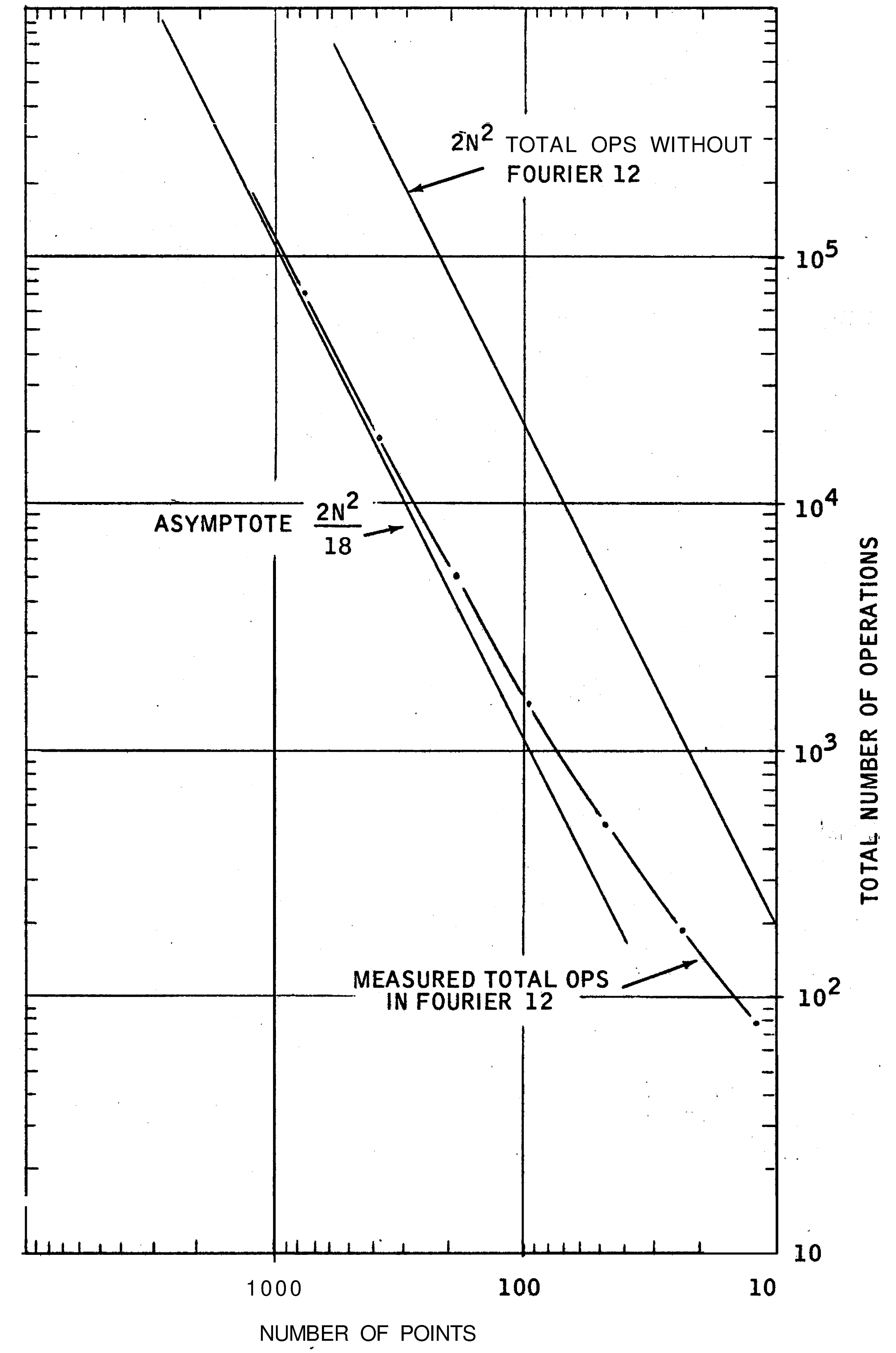


FIG. 5 - OPERATION COUNTS SINE OR COSINE ANALYSIS
BC = 1.2

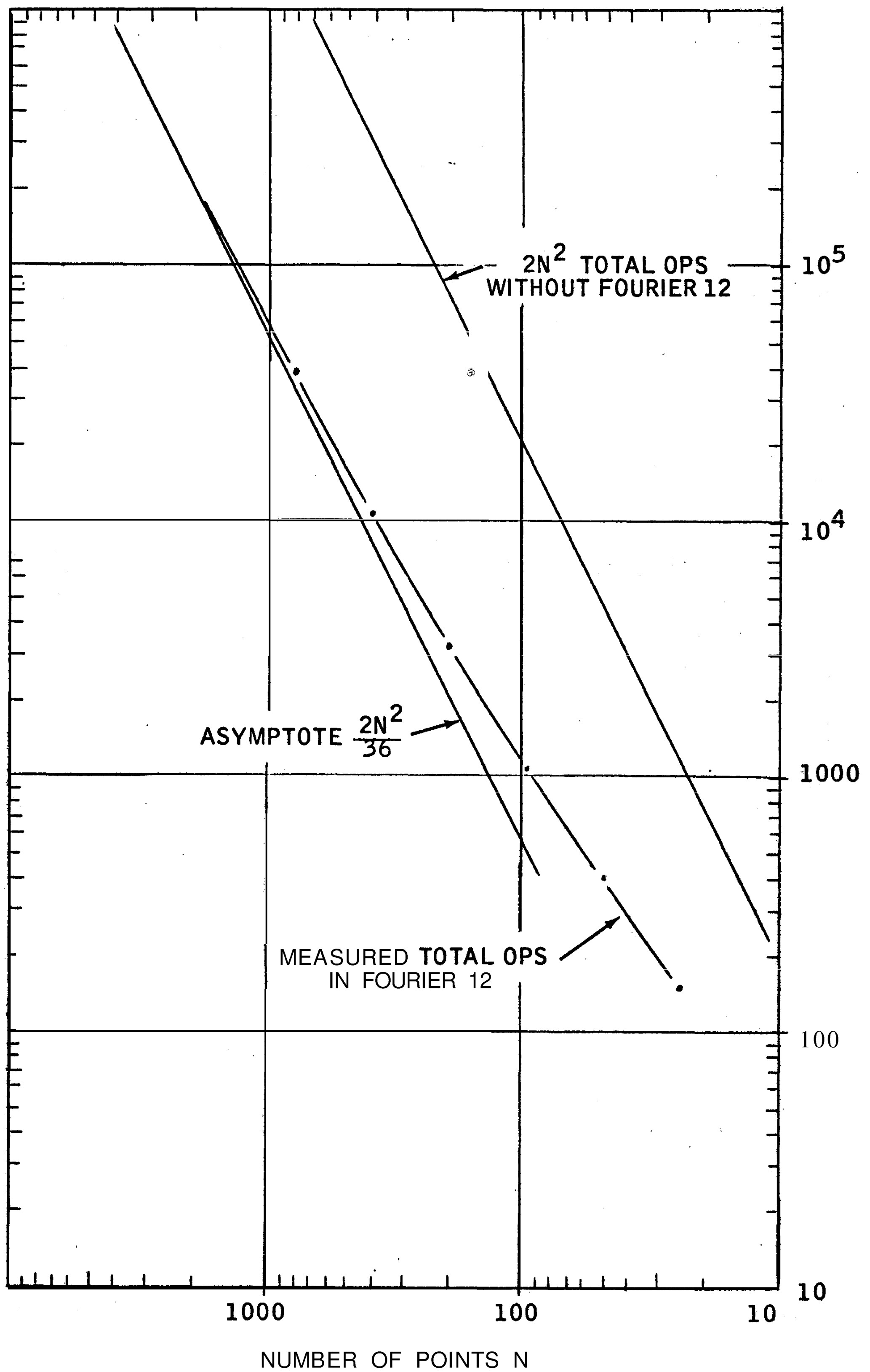


FIG. 6 - **OPERATION COUNTS** PERIODIC ANALYSIS
BC =3, 4

7. RUN TIME COMPARISON

The relations (101) and (102) show that for very large N the Fourier 12 program with periodic conditions can be, at the most, 36 times faster than a program which evaluates the summations directly.

The Fourier 12 program is however complicated logically and contains a large amount of indexing and the time to perform these operations has been neglected in the counts of arithmetic operations given in section 6. In order to get a more realistic view of the possible time savings we have compared FOURIER12 with the following alternative Algol programs?

FOURIERDEF - This program evaluates the Fourier amplitudes directly from their defining summations, It evaluates explicitly a sine or cosine for every term of the sum.

FOURIERE - This program is CACM Algorithm #157[1] modified for an even number of points as described in a remark by G. Schubert[2]. This program considers only periodic analysis corresponding to BC = 3. It does not evaluate the sine and cosine for each term and contains $2N^2$ arithmetic operations for large N. It corresponds therefore to the direct evaluation considered in section 6 where no allowance was made for the evaluation of the sine function. The results of the comparison are shown in Fig. 7 where it can be seen that for large N, and periodic conditions Algorithm 157 is 7 1/2 times faster than Fourierdef, and Fourier 12 is 9 1/2 times faster than Algorithm 157.

Thus about 1/4 of the potential saving of 36 is obtained from an efficiently written B5000 Algol program., It is to be expected that a

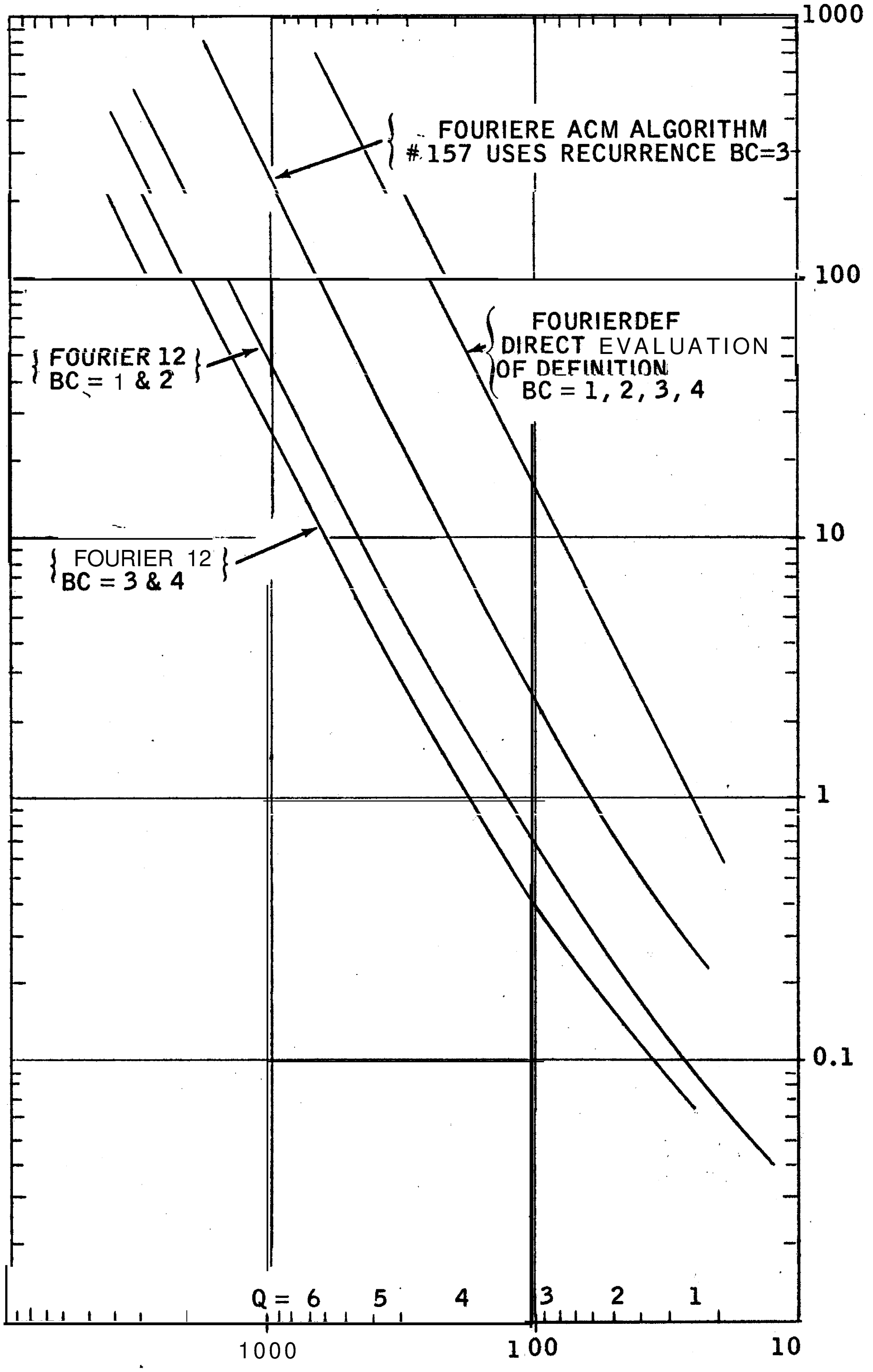


FIG. 7 - TIME COMPARISON

well written machine, code program could realize a good deal more of the potential saving perhaps up to 70%.

We also observe that both the sine and cosine analysis (BC = 1 or 2) are slower than the periodic by a factor of 2 as expected from the formulae (101) and (102).

8. ERROR COMPARISON

The accuracy of the Fourier 12 program has been checked in two ways.

In the first place we have compared the harmonic amplitudes produced by Fourierdef, Algorithm 157 and Fourier 12 for a random input vectors and the results for the periodic case (BC = 3) are shown in Table 2. The results for BC = 1, 2 and 4 have been obtained and are qualitatively similar to the case BC = 3 except that we have no comparison with Algorithm 157.

BC = 3 PERIODIC

Q	N	ALG # 157	FOURIER 12
1	24	8×10^{-11}	$< 10^{-11}$
2	48	4×10^{-10}	1×10^{-11}
3	96	$y \times 10^{-8}$	1×10^{-11}
4	192	4×10^{-7}	2×10^{-11}
5	384	2×10^{-5}	1×10^{-10}
6	768	1×10^{-5}*	

TABLE 2

Maximum deviation from FOURIERDEF result for a random input vector ranging in magnitude from -1/2 to +1/2.

* In this case to avoid excessive machine time that would be required to evaluate Fourierdef we have assumed that Fourier 12 is correct.

Table 2 shows that the error in Fourier 12 does not increase significantly with increasing N and is of the order of the truncation error of the B5000 machine, which is $\sim 10^{-11}$. Thus technique of folding used in Fourier 12 appears to be a stable process numerically.

It can be seen however that the error using Algorithm 157 increases with N such that for N > 100 the calculation of Fourier amplitudes by the recurrence techniques suggested by Goertzel[3] and used in Algorithm 157 is probably not a suitable method.

As a further confirmation of numerical instability in Algorithm 157 we have used as input the test vector

$$x_i = (-1)^i \qquad \text{for } i = 0,1,\ldots,N-1$$

which is $\sqrt{N} \otimes$ the highest cosine harmonic,

Table 3 shows the relative error in the amplitude of the highest cosine harmonic when calculated by the various routines. Again there is a steady increase in the relative error in Algorithm 157 as N increases, whereas there is virtually no increase in the error when Fourier 12 or Fourierdef is used.

BC = 3 PERIODIC

Q	N	FOURIERDEF	ALG 157	FOURIER 12
1	24	4×10^{-11}	3×10^{-11}	1×10^{-11}
2	48	1×10^{-10}	3×10^{-10}	1×10^{-11}
3	96	1×10^{-10}	2×10^{-7}	c 10^{-11}
4	192	3×10^{-10}	1×10 -6	1×10^{-11}
5	384	8×10^{-9}	3×10^{-5}	1×10^{-10}
6	768	--	8×10^{-5}	c 10^{-11}

TABLE 3

Relative error in the highest cosine harmonic amplitude.

The second check on the accuracy of Fourier 12 was a self consistency check performed as follows. A random input vector is analyzed into Fourier harmonics by Fourier 12 and afterwards the harmonics are synthesized by Fourier 12. The final vector obtained should be identical with the initial vector and the greatest deviation between the two is recorded in Table 4. The test was performed for three different random vectors in each case and the maximum derivation of the three cases is recorded. The random vector varied in magnitude from -1/2 to +1/2.

Q	N	SINE CASE BC=1 then BC=1	COSINE BC=2 then BC=2	PERIODIC BC=3 then BC=4
0	12	1.5×10^{-10}	1.0×10^{-10}	---
1	24	1.5×10^{-10}	1.4×10^{-10}	2.0×10^{-11}
2	48	1.7×10^{-10}	1.4×10^{-10}	102×10^{-10}
3	96	2.7×10^{-10}	1.5×10^{-10}	2.1×10^{-10}
4	192	2.5×10^{-10}	1.8×10^{-10}	1.8×10^{-10}
5	384	$--2.3 \times 10^{-10}$	2.1×10^{-10}	2.2×10^{-10}
6	768	2.2×10^{-10}	2.4×10^{-10}	2.0×10^{-10}

TABLE 4

REFERENCES

[1] Mifsud, C. J., "Algorithm 157 Fourier Series Approximation", Corn. ACM Vol. 6, No. 3 March 1963, p. 103.

[2] Schubert, G. R., "Remark on Algorithm 157", Com. ACM Vol. 6, No. 10 October 1963, p. 618.

[3] Goertzel, "Mathematical methods for digital computers" John Wiley and Sons, Inc. 1960.

```
PROCEDURE FOURIER12 (BC,Q,X,Y);
VALUE BC,Q;
INTEGER BC, Q;        ARRAY X,Y[0];
BEGIN
      OWN INTEGER N3,N4,N5,N6,N7,N8,N9,N10,N11;  .
      N4←N3←12×2*Q;
      BEGIN . .
 OWN  INTEGER K,F1,F,I,K1,N2,I1,I2,J,K2,H2,H,L1,L2,L3;
   OWN HEAL A,B,E,F,TS,TC,G,R,A1,A2,A3,A4,A5,TERM,B1,B2,B3,B4,B5,B6;
    LABEL  END1;
      OWN  ARRAY  Z[0:N3],SI[0:N3/2];
      OWN  ARRAY  S,C[0:8];
      OWN  HEAL  NO;
      REAL  PRCCEDURE  SIN1(X);
      VALUE  X;    INTEGER X;
      BEGIN    X←X  MCD (N10);
      SIN1←IF    X≤N7              THEN SI[X]    ELSE
              IF X>N7    AND X≤N3 THEN SI[N3-X]  ELSE
              IF X>N3   AND X≤N11      THEN -SI[X-N3] ELSE
                                           -SI[N10-X];
      END SIN1;
PROCEDURE TWOFOLD(L,N);
VALUE  L,N;         INTEGER  L,N;
      BEGIN
   H2  ←  N/2; .
   FOR  I ← 0  STEP 1 UNTIL   H2-1 0 0




   BEGI N     I1←I+L;    I2←N-I+L;
      A←Z[I1];          B←Z[I2];
      Z[I1]←  A-B;        Z[I2]←A+B;   .
        ADO  ← ADD+2;
  ENOJ
 END TWOFOLD;
PROCEDURE THREEFOLD(L,N);
VALUE  L,N;                    INTEGER L,N;
      BEGIN
   H←N DIV 3;  I1←L+N;  I2←L+N-H-H;
        FOR  I←1  STEP  1 UNTIL H-1 00
          Z[I1-I]←Z[I1-I]+Z[I2+I]-Z[I2-I];
           Z[I1-H]←Z[I1-H]-Z[L];
          ADD ← ADD + 2 ×   (H-1)    +1;
END THREEFOLD;
PROCEDURE EIGHTK(L,N,K,M,Y);
      VALUE L,N,K,M;        INTEGER L,N,K,M;    ARRAY  Y[0];
      BEGIN
      FUR I←0 STEP 1 UNTIL 8 D O S[I]←C[I]←0;
      I F K<0 THEN
      BEGIN COMMENT  1  2 POINT COSINES I  h LONGHAND FOR EFFICIENCY;
        TWOFOLD(L+6,6);
        A + Z[L]×B6+Z[L+4]×B2;     B← Z[L+2]×B4;           . .
        A1←Z[L+3]×B3 J
        F← Z[L+1]×B1-A1+Z[L+5]×B5;     E← Z[L+1]×B5+A1+Z[L+5]×B1;
        G←A+B;    R←A-B;   K2←-K/2;
      I←M+K2;    J←M-K2;
      Y[I]←-G-E;   Y[N7+I]←F-R;   Y[N3+J]←E-G;   Y[N7+J]←-R-F;
        G←B6×(Z[L]-Z[L+4]);  E←B3×(  Z[L+1]-Z[L+3]-Z[L+5]);  K2←3×K2;
        Y[K2+M]←-G-E;  Y[N3-K2+M]←E-G;
        G←-Z[L+6]×B6-Z[L+8]×B2;  E←-Z[L+7]×B4;  K2←-K;
      I←M+K2;    J←M-K2;
     Y [I]←G+E;    Y[N3+J]←G-E;    Y[N7+M]←B6×(-Z[L+6]+Z[L+8]);
                 A←Z[L+12]+Z[L+10];  B←Z[L+11]+Z[L+9];
        Y[M]←B6×(A+B);  Y [N3+M ]←B6×(A-B);
        F←Z[L+11]×B2-Z[L+9]×B6;  R←Z[L+12]×B6-Z[L+10]×B2;
        Y[N7+J]←R+F;    Y[N7+I]←R-F;
            ADD←ADD+28;    MULT←MULT+20;
    END
    ELSE
    IF  K  MOO  2=1 THEN
   BEGIN
    K←K+1;
      A1←Z[L+9]×B3;
      A←Z[L+10]×B2+Z[L+6]×B6;    B←Z[L+8]×B4;
      E←Z[L+11]×B1+Z[L+7]×B5+A1;    F←Z[L+11]×B5+Z[L+7]×B1−A1;
      G←A+B;      R←A−B;     K2←K/2;
      I←M+K2;     J←M−K2;
      Y[I]←E+G;    Y[N7+I]←F−R;    Y[N3+J]←E−G;    Y[N7+J]←F+R;
      G←B6×(Z[L+10]−Z[L+6]);   E←B3×(Z[L+11]+Z[L+9]−Z[L+7]); K2←3×K2;
    Y[K2+M]←E+G;       Y [N3−K2+M ]←E−G;
    Z[L+6]←C;
      TWOFOLD(L,6);
    E←B2×Z[L+5]+B6×Z[L+3]];    G←B4×Z[L+4]];   I←M+K;    J←M−K;
     Y[I]←E+G;      Y[N3+J]←E−G;    Y[N7+M]←B6×(Z[L+5]−Z[L+3]);




       Y[N7+J]←B4×(Z[L+1]+Z[L+2]);    Y[N7+I]←B4×(Z[L+1]−Z[L+2]);
        ADD←ADD+22;     MULT←MULT+16;
  END
ELSE
BEGIN
 K2←  K/ 21
 FOR J ← 1 STEP 1 WHILE  J≤8  AND  J≤N  DO
BEGIN
    FOR 1 +    J  STEP 8 WHILE I  ≤ N    DO
   BEGIN I1 ←  I×K2;    L3←L+N−I;
          TERM←Z[L3]×SIN1(I1);
          TS←IF I=J  THEN  TERM  ELSE  TS+TERM;
          If I≠J  THEN  ADD←ADD+1;
          MULT←MULT+1;
          IF  J   MOC  4  ≠ 0 THEN
          BEGI N
            TERM←Z[L3]×SIN1(I1+N7);
            TC←IFI=J THEN TERM  ELSE  TC+TERM;
          IF I≠J THEN A D O ←ADD+1;
          MULT←MULT+1;
          END TC;
    END I;
         S[J] ←  TS  ; C[J]  +  TC ;
 END ;
 A   ← S[2] +   SC61   ; B ← S[8] +   St43   ;
 E ← (S[1] + S[5])  t  (S[3] + S[7]);
 F ← (C[1] + C[5]) − (  C  C  3  1 + C[7]);
 G ← A+B ; R  +  A−B;    I←M+K2;   J←M−K2;
 Y[I]←E+G;    Y[N7+I]←F−R;    Y[N3+J]←E−G;    Y[N7+J]←F+R;
    ADD ← ADD +  14  ;
 IF  h  ≠N5 THEN
 BEGIN
    A + C[2]  − C[6]  j  B  +  −S[8] + S[4]];
    E  +  C[1]−C[5] ;  F  ← S[1] − S[5]];
    Al  +  E+F ;  A2  +  E−F;
    E ← C[3]−C[7]];  F  +  S[3] − S[7]];
    A3  ← E−F;  A4  ← E+F;
      E  ←  A5×(A2+A4);
      F  ←  A5  ×(A1+A3);
    G  ← A+B  ;  R  ←  A−B;
    Y[N5+J]←E+G;    Y[N6+J]←F−R;    Y[N6+I]←E−G;    Y[N5+I]←F+R;
          ADD←ADD+18; MULT←MULT +2;
      END;
     END;
END EIGHTK;
 PROCEDURE VAL (L,M,Y );
 VALUE  L,M;             INTEGER  L,M;          ARRAY  Y[0];
        BEGIN
        N2 ← N3; F1←  21
        FOR P←1 STEP 1  WHILE  N2>12 DO
        BEGIN
            TWOFOLD(L,N2);
            N2←N2/2;            L1←L+N2;
            FOR  K←1 STEP  2  WHILE  F1×K≤N5 DO
            IF K MOD 3≠0 THEN


              EIGHTK(L1,N2,F1×K,M,Y);
            THREEFOLD(L1,N2);
            FOR K←  3 STEP 6 WHILEF1×K≤N5 DO
          BEGIN N9←N2DIV3 1   EIGHTK(L1+2×N9,N9,F1×K,M,Y);  END;
            Z[L+N2]←0;          F1←F1+F1;
        END Pi
            TWOFOLD(L,N2);
            EIGHTK(L,N2,F1−1,M,Y);
 END VAL;
 PROCEDURE SLOPE (L,M,Y);
 VALUE  L,M;                     INTEGER L,M; ARRAY Y[0];
 BEGIN
        N2←N3;   F1←2;          L2←L+N3;
        FOR P←1  STEP  1  WHILE  N2>12  00
        BEGIN
          TWOFOLD(L2−N2,N2);
          N2←N2/2;          L1←L2−N2−N2;
            FUR  K←1  STEP  2  WHILE F1×K≤N5 DO
            IF  K  MOD  3  ≠0 THEN
              EIGHTK(L1,N2,F1×K,M,Y);
            THREEFOLD(L1,N2);
            FOR K←3  STEP  6  WHILE F1×K≤N5 DO
        BEGIN N9←N2 OIV 3;   EIGHTK(L2−4×N9,N9,F1×K,M,Y);  END;
            FOR K←1  STEP  4  WHILE F1×K≤N10 DO
            BEGIN  I←M+F1×K/2;   Y[I]←−Y[I];  END;
        F1←F1+F1;
          END;
        L1←L2−N2;
          TWOFOLD(L1,N2);
          EIGHTK(L1,N2,−F1,M,Y);
        EN0 SLOPE;
    IF Q≥7 OR Q<0 THEN  GO  TO  END1;
        IF (BC=3 OH BC=4) AND Q=0  THEN  GO  TO  END1;
    FOR  I ←  0  STEP  1  UNTIL  N3  00  Z[I]←X[I];
    PI  ←  3.141592653591        A5←1/SQRT(2);
        N0←SQRT(2/N3);
        IF  BC=3  THEN
        BEGIN
          Z[N3]←Z[0]←Z[0]/2;
          TWOFOLD(0,N3);
          N3←N3 DIV 21
        END;
        IF BC=4  THEN  N3←N3 OIV 2;
        N5←N3  DIV 4;   N6←3×N5;   N7←N3 DIV 21    N10←2×N3;   N11←3×N7;
        FOR I ←  0  STEF  1 UNTIL N7  00  SIC11 ←N0×SIN(PI×I/N3));
        B1←SI[N3 O I V 12];B2←SI[N3 DIV 6 1 1   B3←SI[N5];
        B4←SI[N3 DIV  311   B5←SI[5×N3  OIV  12];  B6←SI[N7];
    IF BC=1  THEN
    BEGIN
        Z[0] ←  Z[N3] ← 0;
        VAL (0,0,Y );
        Y[0]←Y[N3]←0;
    END;
    IF  BC=2  THEN
    BEGIN
```

```
        Z[0]←Z[0]/2;             Z[N3]←Z[N3]/2;
        MULT←MULT+2;
        SLOPE (0,0,Y);
          FOR K←1 STEP 2 UNTIL N3-1 DO Y[K]←-Y[K];
    END;
    IF BC=3 THEN
    BEGIN
        SLOPE (N3,0,Y);
        Y[0]←Y[0]×A5;        Y[N3]←Y[N3]×A5;
     MULT←MULT+2;
        Z[0]←Z[N3]←0;
        VAL (0,N3,Y);
        Y[N4]←0;
    END ;
        IF BC=4 THEN
        BEGIN
        Z[0]←Z[0]×A5;            Z[N3]←Z[N3]×A5;
        MULT←MULT+2;
        SLOPE(0,0,Y);
        FOR K←1 STEP 2 UNTIL N3-1 DO    Y[K]←-Y[K];
        Z[N3]←Z[N4]←0;
        VAL(N3,N3,Y);
        FOR I←1 STEP 1 UNTIL N7 DO
            BEGIN J←N3+I; K←N4-I; A←Y[J]; Y[J]←-Y[K]; Y[K]←-A  END;
        Y[N4]←0;
        FOR I←1 STEP 1 UNTIL N3-1 DO
BEGIN J←N4-I; A←Y[I]; B←Y[J]; Y[I]←A-B; Y[J]←A+B; ADD←ADD+2;END;
        END;
        END;
END  END  FOURIER12;
```