THE FAST FOURIER TRANSFORM IN FOURIER SPECTROSCOPY

BY

THOMAS E. MICHELS

GPO PRICE $__________

CFSTI PRICE(S) $__________

Hard copy (HC) $2.00

Microfiche (MF) $0.50

JULY 1966

GODDARD SPACE FLIGHT CENTER
GREENBELT, MARYLAND
THE
FAST FOURIER TRANSFORM
IN
FOURIER SPECTROSCOPY

by

Thomas E. Michels

July 1966

Goddard Space Flight Center
Greenbelt, Maryland
# CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>v</td>
</tr>
<tr>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>DESCRIPTION OF THE PROBLEM</td>
<td>2</td>
</tr>
<tr>
<td>NUMERICAL QUADRATURE OF THE INTERFEROGRAM INTEGRAL</td>
<td>4</td>
</tr>
<tr>
<td>THE &quot;FAST FOURIER TRANSFORM&quot;</td>
<td>6</td>
</tr>
<tr>
<td>1. Description of the Method</td>
<td>6</td>
</tr>
<tr>
<td>2. Application to the Interferogram Integral</td>
<td>9</td>
</tr>
<tr>
<td>3. Summary of the Application</td>
<td>13</td>
</tr>
<tr>
<td>4. Computational Timing</td>
<td>13</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>15</td>
</tr>
<tr>
<td>APPENDIX I</td>
<td>17</td>
</tr>
<tr>
<td>APPENDIX II</td>
<td>23</td>
</tr>
</tbody>
</table>
THE
FAST FOURIER TRANSFORM
IN
FOURIER SPECTROSCOPY

by

Thomas E. Michels

ABSTRACT

The "Fast Fourier Transform" (suggested by Drs. J. W. Cooley and J. W. Tukey) is presented with special application to solving the interferogram integral obtained in Fourier Spectroscopy. Computational timing is tabulated and an explanation of a computer routine using this method to a binary base is presented.
THE
FAST FOURIER TRANSFORM
IN
FOURIER SPECTROSCOPY

INTRODUCTION

The problem of digitally reducing interferogram data obtained from an interferometer spectrometer is two fold; (1) numerical quadrature of the interferogram integral to obtain the spectral magnitudes, and (2) computation of the phase of the spectral magnitudes to determine the direction of radiation flow.

A third requirement can be placed on the computation if a great number of spectrums are to be analysed; that is fast calculation on a high speed digital computer. This is the problem with which this paper is primarily concerned.

Many methods of fast calculations of the interferogram integral have been suggested; however, the "Fast Fourier Transform" (Cooley, J. W. and Tukey, J. W., 1965) method offers many advantages in speed and accuracy which others do not have.

The application of this method to discrete interferogram data is discussed and the use of an existing computer program written by Dr. J. W. Cooley in FORTRAN IV using the "Fast Fourier Transform" to compute a Fourier transform or series is explained in Appendicies I and II. Only a small amount of interferometer theory is discussed so as to keep the general theme of the report.

It should be mentioned that the Fast Fourier Transform has been implemented with much success into the Infrared Interferometer Spectrometer (IRIS) Experiment Data Reduction Program as well as the theoretical simulating the IRIS experiment. Computational timings have been tabulated using these programs and are given in a following section. The reader should also be aware that now with the computational speeds obtained using the "Fast Fourier Transform," problems in Fourier Spectroscopy which previously were overly time consuming on the computer are now realistic for computer solution. These include taking convolutions for truncated Fourier Integrals, shown by Dr. J. W. Cooley, co-author of the Fast Fourier Transform (in an unpublished report) and correction of asymmetric interferograms (Forman, M. L., 1965).
The author is greatly indebted to Drs. B. Conrath and R. R. Hanel for many hours of helpful discussions on interferometer theory and Fourier analysis, and to Dr. J. W. Cooley for the use of his computer program.

DESCRIPTION OF THE PROBLEM

The instrument used to obtain interferogram data in the IRIS experiment is essentially a two beam Michelson interferometer, sketched in Figure 1.

![Figure 1 - Sketch of Michelson Interferometer](image)

The incoming heterochromatic radiation is split at the beam splitter, A, into two waves of equal amplitude; one directed toward a moveable mirror at C, and the other toward a fixed mirror at B. They are reflected back to the beam splitter, recombined, and directed to the detector at D. The recombined signal received at the detector is the interference pattern of the two beams called the interferogram and, ideally, can be defined as a function of path difference traveled by the two beams by

\[ I(\delta) = \int_{-\infty}^{\infty} B_\nu \, e^{i2\pi\nu \delta} \, d\nu \]  

(1)

The path difference, δ, is defined with the aid of Figure 1 by

\[ \delta = 2 \, (CA - BA) \]
The multiplying factor of two times the distance (CA-BA) arises from the fact that the radiation travels to the mirror and back again to transverse the added distance ($\delta$ can be given as a function of time and mirror velocity; however, we will use the path difference relationship here).

$B_{\nu}$ is the spectral distribution of the incoming radiation and is written here in terms of wave number, $\nu$. It has been extended to include negative wave numbers by the definition

$$B_{-\nu} = B_{\nu}$$

Clearly, $B_{\nu}$ is real, and $B_{\nu}$ and $I(\delta)$ are transform pairs; thus one is able to obtain the spectrum $B_{\nu}$ by taking the Fourier transform of equation (1) which is

$$B_{\nu} = \int_{-\infty}^{\infty} I(\delta) e^{-i 2\pi \nu \delta} \, d\delta. \quad (2)$$

If $I(\delta)$ were symmetric, which in the theoretical case is true, equation (2) reduces to

$$B_{\nu} = 2 \int_{0}^{\infty} I(\delta) \cos (2\pi \nu \delta) \, d\delta$$

However, in actual interferometer use a phase shift, $\phi_{\nu}$, is introduced by the instrument components to the incoming radiation and equation (1) should be re-written as

$$I(\delta) = \int_{-\infty}^{\infty} B_{\nu} \, e^{i \phi_{\nu}} \, e^{i 2\pi \nu \delta} \, d\nu \quad (3)$$

$I(\delta)$ in this case will be an asymmetric function about zero path difference, $\delta_{c}$. Upon taking the Fourier transform of equation (3), one obtains

$$B_{\nu} e^{i \phi_{\nu}} = \int_{-\infty}^{\infty} I(\delta) e^{-i 2\pi \nu \delta} \, d\delta. \quad (4)$$
Equation (4) is of the form

\[ B_\nu e^{i\varphi_\nu} = C_\nu + i S_\nu \]

and one merely takes the absolute value of the right hand side to obtain the spectral magnitudes.

The phase shift \( \varphi_\nu \) can, of course, be computed from

\[ \varphi_\nu = \tan^{-1} \left( \frac{S_\nu}{C_\nu} \right) \]  

(5)

The phase spectrum can be a useful tool in interpreting the amplitude spectrum. The direction of the net energy transfer between the detector and the target (determining whether the detector is warmer or colder than the target) changes direction from one spectral region to another and is indicated by an abrupt phase shift of 180°.

Therefore, the problem of digitally reducing interferogram data requires solution of the interferogram integral in equation (4) and computation of the spectral amplitude phases by equation (5).

NUMERICAL QUADRATURE OF THE INTERFEROGRAM INTEGRAL

The numerical calculation of the integral in equation (4) can be done by various quadrature methods. The Gaussian quadrature is perhaps the most accurate, but this method requires unequal spacing in the sampled data.

It is suggested that in practical use, when the truncated range \((-\delta_1, \delta_2)\) of the integral (see equation (6)) is large enough, many advantages are offered by use of the trapezoidal rule.

One advantage in accuracy is easily seen by looking at Eulers summation formula: (Scarborough, 1955)

\[
\int_a^b f(x) \, dx = \frac{h}{2} \left[ \frac{f(x_0)}{2} + f(x_1) + \ldots + f(x_{n-1}) + \frac{f(x_n)}{2} \right] - \frac{h}{12} [f'(b) - f'(a)] \\
+ \frac{h^3}{720} [f''(b) - f''(a)] - \frac{h^5}{30240} [f'''(b) - f'''(a)] + \cdots
\]
This states that if the odd derivatives at the endpoints of an integration are small or equal, the trapezoidal rule is an excellent approximation to the integral.

In many cases, the derivatives at the endpoints of the interferogram are equal, and in the worst cases observed the derivatives were always of the same order. In these cases, the order of the error would be $h$, the separation distance ($\Delta \delta$ in the interferogram integral), and for the interferogram integral, this is small.

Therefore, the trapezoidal rule appears to be a satisfactory quadrature method and, as one will see, is a suitable form for the "Fast Fourier Transform" to obtain speed of calculation.

Rewriting equation (4) as a truncated inverse over the interval $(-\delta_1, \delta_2)$, and, for convenience, setting $A_\nu = B_\nu e^{i\varphi_\nu}$, we have

$$A_\nu = \int_{-\delta_1}^{\delta_2} I(\delta) e^{-i2\pi\nu\delta} \, d\nu \quad (6)$$

Since the interferogram values sampled by the instrument are an average over an interval, the trapezoidal rule takes the form

$$A_\nu = B_\nu \, e^{i\varphi_\nu} = \Delta \delta \sum_{j=m-n}^{n} I(\delta_j) \, e^{-i2\pi\nu\delta_j} \quad (7)$$

where the subscript, $j$, ranges over $N$ sampled intervals with $N$ defined as

$$N = m + n + 1$$

The point $\delta_0$ is of course the sampled interval which normally will not correspond exactly to the point of zero path difference. Correction methods have been studied for correction of asymmetric interferograms of this type (Forman, M. L.); however, this paper will not concern itself with these methods.

Solution of equation (7) to obtain the spectral magnitudes and phase angles is easily accomplished using the "Fast Fourier Transform," and, as one will see, yields a very high degree of accuracy for computer solution.
1. Description of the Method

The form of the Fourier series equation required for the "Fast Fourier Transform" is

\[
I(j) = \sum_{k=0}^{N-1} B(k)e^{\frac{2\pi j k}{N}}
\]  

\(j = 0, 1, 2 \ldots N-1\)

and its transform is

\[
B(k) = \frac{1}{N} \sum_{j=0}^{N-1} I(\delta_j) e^{-i\frac{2\pi j k}{N}}
\]  

Normal solution of equation (8) using a decimal based summation requires an order of \(N^2\) operations where an operation is defined as one multiplication and addition. However, the number of operations using a different base, say \(r\), is of the order \(8N \log_r N\). A simple example will illustrate the saving.

If \(N\) is composite such that \(N = r_1 \cdot r_2\), and \(j\) and \(k\) are written

\[j = r_1 j_1 + j_0, \quad j_0 = 0, 1, 2 \ldots r_1 - 1; \quad j_1 = 0, 1, \ldots r_2 - 1\]

and

\[k = r_2 k_1 + k_0, \quad k_0 = 0, 1, \ldots r_2 - 1; \quad k_1 = 0, 1, \ldots r_1 - 1\]

then equation (8) can be written
\[ I(j) = \sum_{k_0} \sum_{k_1} B(k) e^{i \frac{2\pi}{N} j (r_2 k_1 + k_0)} \]

\[ = \sum_{k_0} \sum_{k_1} B(k) e^{i \frac{2\pi}{N} j r_2 k_1} e^{i \frac{2\pi}{N} j k_0} \]

But

\[ e^{i \frac{2\pi}{N} j r_2 k_1} = e^{i \frac{2\pi}{N} k_1 r_2 (r_1 j_1 + j_0)} = e^{i \frac{2\pi}{N} k_1 j_0 r_2} \]

Therefore, the inner sum over \( k_1 \), depends only on \( j_0 \) and \( k_0 \), which can be written as

\[ B_1(j_0, k_0) = \sum_{k_1} B(r_2 k_1 + k_0) e^{i \frac{2\pi}{N} k_1 j_0 r_2} \]

and equation (8) now becomes

\[ I(r_1 j_1 + j_0) = \sum_{k_0} B_1(j_0, k_0) e^{i \frac{2\pi}{N} j k_0} \]

The total number of operations for this case is now \( N(r_1 + r_2) \) as opposed to \( N^2 \) before.

It has been shown (Colley, J. W., Tukey, J. W., 1965) that the most saving is obtained when \( N \) is written as some number raised to an integer power, or

\[ N = r^m \]
and, further, if \( r = e \), one uses the least operations in solving equation (8). However, if \( r = 2 \), the saving is approximately the same, and from a digital computer standpoint, certain advantages in programming are obtained.

With this in mind, if one writes

\[
N = 2^m
\]

and

\[
j = 2^{m-1} j_{m-1} + 2^{m-2} j_{m-2} + \ldots + j_0
\]

and

\[
k = 2^{m-1} k_{m-1} + 2^{m-2} k_{m-2} + \ldots + k_0
\]

where each \( j_\ell \) and \( k_\ell \), \( \ell = 0, \ldots, m-1 \) take on the values 0, 1, then equation (8) takes the form

\[
I(2^{m-1} j_{m-1} + 2^{m-2} j_{m-2} + \ldots + j_0) = B_m (j_0 + 2j_1 + \ldots + 2^{m-1} j_{m-1})
\]

where

\[
B_1(j_0, 2^{m-2} k_{m-2} + \ldots + k_0) = \sum_{k_{m-1}} B(2^{m-1} k_{m-1} + \ldots + k_0) e^{i \frac{2\pi}{N} j_0 k_{m-1} 2^{m-1}}
\]
The routine listed in Appendix I uses the base two and is written such that either the Fourier series coefficients or their transform can be computed. That is, one is able to compute either equation (8) or equation (9).

2. Application to the Interferogram Integral.

We saw that the spectral magnitudes and phase angles can be obtained from equation (7),

\[ A_\nu = B_\nu e^{i\Phi_\nu} = \Delta \delta \sum_{j=-m}^{n} I(\delta_j) e^{-12\pi \nu \delta_j}. \]

If one sets,

\[ \nu_k = k \Delta \nu, \quad k = 0, 1, \ldots, N - 1 \]

and

\[ \delta_j = j \Delta \delta, \quad j = 0, 1, \ldots, N - 1 \]
where $\nu_0 = \delta_0 = 0$, we have

$$A_{\nu_k} = \Delta \delta \sum_{j = -n}^{m} I(\delta_j) e^{-i2\pi j k \Delta \nu \Delta \delta}. \quad (10)$$

Further, if one makes the substitutions

$$j = j - n$$

$$\Delta \delta \Delta \nu = N^{-1} = \text{(Number of sampled intervals)}^{-1},$$

one can write the summation in the form suitable for the "Fast Fourier Transform." That is,

$$A_{\nu_k} = \Delta \delta e^{i \frac{2\pi k n}{N}} \sum_{j = 0}^{N - 1} I(\delta_j) e^{-i \frac{2\pi}{N} j k} \quad (11)$$

where

$$N = m + n + 1.$$

If the number of sampled intervals, $N$, does not equal a power of two and the interferogram has zero mean (no DC component), one is able to add values equal to zero to either or both ends of the interferogram without loss of accuracy and compute spectral magnitudes corresponding to $\nu = k \Delta \nu$. Here,

$$\Delta \nu = \frac{1}{\Delta \delta \cdot N'}$$

where

$$N' = N + \ell$$

such that $N'$ is equal to two raised to some integral power.
Further, if the interferogram is positioned in the area of $N'$ intervals as shown in Figure 2, such that $\delta_0$ is a distance $N'/2$ from the beginning, a particularly simple case arises. Keeping in mind that $A_\nu = B_\nu e^{i\varphi_\nu}$, equation (11) takes the following form for this case,

$$A_{\nu_k} = \delta \sum_{j=0}^{N-1} I(\delta_j) e^{i \frac{2\pi j k}{N}}$$  \hspace{1cm} (12)

and the spectral magnitudes are computed as before from

$$B_{\nu_k} = |A_{\nu_k}|$$  \hspace{1cm} (13)

Re-writing equation (12) as

$$B_{\nu_k} e^{i \varphi_{\nu_k}} = e^{i k \pi} (C_{\nu_k} - i S_{\nu_k})$$

or

$$B_{\nu_k} e^{i (\varphi_{\nu_k} - k \pi)} = C_{\nu_k} - i S_{\nu_k}$$

the phase angle can then be computed from

$$\varphi_{\nu_k} = \tan^{-1} \left( \frac{S_{\nu_k}}{C_{\nu_k}} \right) + k \pi$$  \hspace{1cm} (14)
or

\[ \varphi_{\nu_k} = \tan^{-1} \left[ \frac{(-1)^k S_{\nu_k}}{(-1)^k C_{\nu_k}} \right] \]

For the case when the interval \( \delta_0 \) is not positioned in the center of the array, one still can use equation (12) to compute the spectral magnitudes, however, one must use an equation similar to equation (11) to compute the phase angles. That is,

\[ \varphi_{\nu_k} = \tan^{-1} \left( \frac{S_{\nu_k}}{C_{\nu_k}} \right) + \frac{2\pi kn}{N} \quad (15) \]

where, \( n \), is the number of intervals from the beginning of the array to \( \delta_0 \).

In using the routine in Appendix I, one must be careful applying it to compute the Interferogram integral. From experience it proved easier and faster to compute the Fourier series coefficients rather than the Fourier transform. This means that for the real input data, one obtains the complex conjugate of the vector \( B_{\nu_k} e^{i\varphi_{\nu_k}} \) as output from the "Fast Fourier Transform."

Clearly, this makes no difference in computing the spectral magnitudes other than a factor of two, however, the phase angles must now be computed from

\[ \varphi_{\nu_k} = k\pi - \tan^{-1} \left( \frac{S_{\nu_k}}{C_{\nu_k}} \right) \quad (16) \]

or

\[ \varphi_{\nu_k} = \tan^{-1} \left[ \frac{(-1)^{k+1} S_{\nu_k}}{(-1)^{k+1} C_{\nu_k}} \right] \quad (17) \]

or, in the case where \( \delta_0 \) is not positioned in the center of the interferogram,

\[ \varphi_{\nu_k} = \frac{2\pi kn}{N} - \tan^{-1} \left( \frac{S_{\nu_k}}{C_{\nu_k}} \right) \quad (18) \]

The method of computing the spectral magnitudes and phase angles can be summarized as follows:

1. Read the interferogram data into the "Fast Fourier Transform" (FFT) and compute a Fourier series yielding $C_{\nu_k}$ and $S_{\nu_k}$ as output.

$$\text{FFT}(\text{series}) = C_{\nu_k} + i S_{\nu_k}$$

2. Compute the spectral magnitudes from

$$B_{\nu_k} = 2 \cdot \Delta \delta \cdot |C_{\nu_k} + i S_{\nu_k}|$$

where

$$\nu_k = k \Delta \nu, \ k = 0, \ 1, \ \ldots \ N - 1$$

and

$$\Delta \nu = \frac{1}{\Delta \delta \cdot N'}$$

3. Compute the phase angles from either equation (17) or (18) depending upon the position of the interferogram data in the array of size $N' = 2^n$.

In using the routine in Appendix I, one automatically obtains the same number of points of output as he has read in as input.

4. Computational Timing

Figure 3 shows the computational times obtained using the FORTRAN IV version of the "Fast Fourier Transform" routine (HARM) listed in Appendix I. The line indicating the times obtained on the IBM 360/65 system were obtained with the HARM subroutine, compiled with the optimization option. These times were obtained by Mr. Guy Marcot, Laboratory for Space Sciences, GSFC. A comparison with conventional times obtained with the algorithm by Goertzel indicates the saving, especially with a large number of data points.
Figure 3—Computation Times using the FORTRAN IV version of HARM on the IBM 7094 and 360/65 Computers vs. the Goertzel algorithm.

Another version of the HARM subroutine, written in FORTRAN II and FAP, which is used in the theoretical simulation of the IRIS experiment was timed for $2^{11}$ and $2^{12}$ points on the IBM 7094 computer. These results are tabulated below.

<table>
<thead>
<tr>
<th>No. of Points</th>
<th>Time for Fourier series calculation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2^{11}$</td>
<td>1.3 sec</td>
</tr>
<tr>
<td>$2^{12}$</td>
<td>2.6 sec</td>
</tr>
</tbody>
</table>

The times tabulated above include the time obtained to compute the sine and cosine transform and do not include taking the absolute value or obtain the phase angle. This has been timed using the theoretical IRIS simulation. To compute 722 spectral magnitudes and phase angles out of the $2^{11}$ or $2^{12}$ available took approximately 3/10 seconds.
REFERENCES


APPENDIX I

Sample Program Using the "Fast Fourier Transform" Routine "HARM"

The following program has been written exemplifying the use of the "HARM" subroutine to compute the transform of a set of real data and take the inverse of the transform to compare with the original data.

"HARM" is written in FORTRAN IV and is set up to accept complex input as well as to perform a three-dimensional sum in directions M(1), M(2), M(3). The return from "HARM" has the complex vector computed and stored in the array A, where

\[ A(1) + i A(2) = C_{\nu_0} + i S_{\nu_0} \]
\[ A(3) + i A(4) = C_{\nu_1} + i S_{\nu_1} \]
\[ \text{and so on.} \]

The array M and option IFS must be set prior to entry, and if one wishes to use the routine in one-dimension, as in the sample program, one sets

\[ M(1) = \log_2 N \]
\[ M(2) = 0 \]
\[ M(3) = 0 \]

The call to the "HARM" subroutine is

CALL HARM (A, M, INV, S, IFS, IFERR)

A = Array of complex input, where the real and imaginary parts are in consecutive storage locations. A must be dimensioned 2N.
M = Array containing the logarithm to the base 2 of the length of the summations in directions M(1), M(2), M(3). M is dimensioned 3.
INV = Array computed in "HARM" for bit inverting. INV is dimensioned N/8.

S = Array computed in "HARM" containing the array of sines. S is dimensioned N/8.

IFS =  0 compute INV and S tables
        +1 compute INV and S tables and do Fourier series
        -1 compute INV and S tables and do Fourier transform
        +2 do Fourier series only
        -2 do Fourier transform only

IFERR = Error option. (See Appendix II)

Further amplification of the use of "HARM" can be obtained from the program write-up on file in the Laboratory for Atmospheric and Biological Sciences.
C. TEST OF FAST FOURIER TRANSFORM ROUTINE (HARM) .

C. THIS PROGRAM PERFORMS A TRANSFORM ON A SET OF DISCRETE REAL DATA,
C. THEN TAKES THE INVERSE OF THE TRANSFORM TO RECOVER THE ORIGINAL
C. DATA. THE INPUT DATA REPRESENTS A BLACK BODY TEMPERATURE SPECTRUM
C. COMPUTED IN THE 'IRIS' EXPERIMENT.

DIMENSION A(16384), M(3), INV(2048), S(2048)

C. SET UP DIMENSIONS
M(1)=6
M(2)=0
M(3)=0
N1 = 2**M(1)
N2=2**M(2)
N3=2**M(3)
NTOT=N1*N2*N3

DO 10 I=1,NTOT
10 A(I)=0.
READ (2,15) (A(2*I-1), I=1,NTOT)

C. WRITE (3,17) (A(2*I-1), I=1,NTOT)
15 FORMAT (6E12.5)

WRITE (3,17) (A(2*I-1), I=1,NTOT)

17 FORMAT (26H1ORIGINAL DATA- REAL PART // (5F13.5))

WRITE (3,18) (A(2*I), I=1,NTOT)

18 FORMAT (26H1ORIGINAL DATA- IMAG PART // (5F13.5))

C. EVALUATE FOURIER TRANSFORM

IF IFS = -1, SETUP INV AND S TABLES AND COMPUTE FOURIER SERIES
IFS = -1
CALL HARM (A,M, INV, S, IFS, IFERR)

WRITE (3,42) (A(2*I-1), I=1,NTOT)

25 FORMAT (23H1REAL PART OF TRANSFORM // (5F13.5))

WRITE (3,44) (A(2*I), I=1,NTOT)

42 FORMAT (15H1REAL PART // (5F13.5))

44 FORMAT (15H1IMAGINARY PART // (5F13.5))

C. TAKE INVERSE

IF IFS = +2, COMPUTE FOURIER SERIES
IFS = +2
CALL HARM (A,M, INV, S, IFS, IFERR)
WRITE (3,70) (A(2*I-1), I=1,NTOT)

70 FORMAT (60H1REAL PART OF ORIGINAL INPUT OBTAINED BY INVERTING TRANS
1FORM // (5F13.5))

WRITE (3,71) (A(2*I), I=1,NTOT)

71 FORMAT (15H1IMAGINARY PART // (5F13.5))

PAUSE 77777

END
<table>
<thead>
<tr>
<th>REAL PART</th>
<th>230.09000</th>
<th>229.99000</th>
<th>229.99000</th>
<th>230.01000</th>
<th>230.09000</th>
</tr>
</thead>
<tbody>
<tr>
<td>230.03000</td>
<td>230.08000</td>
<td>230.08000</td>
<td>230.02000</td>
<td>229.90000</td>
<td></td>
</tr>
<tr>
<td>230.03000</td>
<td>230.10000</td>
<td>229.96000</td>
<td>229.95000</td>
<td>229.99000</td>
<td></td>
</tr>
<tr>
<td>230.10000</td>
<td>229.94000</td>
<td>229.98000</td>
<td>230.03000</td>
<td>230.08000</td>
<td></td>
</tr>
<tr>
<td>229.87000</td>
<td>229.84000</td>
<td>229.99000</td>
<td>230.07000</td>
<td>229.98000</td>
<td></td>
</tr>
<tr>
<td>229.85000</td>
<td>229.85000</td>
<td>230.00000</td>
<td>229.90000</td>
<td>229.74000</td>
<td></td>
</tr>
<tr>
<td>229.05000</td>
<td>229.88000</td>
<td>229.84000</td>
<td>229.47000</td>
<td>229.73000</td>
<td></td>
</tr>
<tr>
<td>229.92000</td>
<td>229.58000</td>
<td>229.89000</td>
<td>230.04000</td>
<td>230.48000</td>
<td></td>
</tr>
<tr>
<td>229.85000</td>
<td>229.74000</td>
<td>229.61000</td>
<td>229.21000</td>
<td>229.80000</td>
<td></td>
</tr>
<tr>
<td>229.52000</td>
<td>229.27000</td>
<td>229.63000</td>
<td>229.69000</td>
<td>229.69000</td>
<td></td>
</tr>
<tr>
<td>229.10000</td>
<td>229.71000</td>
<td>229.63000</td>
<td>229.24000</td>
<td>229.05000</td>
<td></td>
</tr>
<tr>
<td>229.42000</td>
<td>229.57000</td>
<td>229.59000</td>
<td>229.95000</td>
<td>229.88000</td>
<td></td>
</tr>
<tr>
<td>230.00000</td>
<td>229.87000</td>
<td>229.67000</td>
<td>229.64000</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IMAG PART</th>
<th>0.0</th>
<th>0.0</th>
<th>0.0</th>
<th>0.0</th>
<th>0.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>-0.00000</td>
<td>-0.00000</td>
<td>-0.00000</td>
<td>-0.00000</td>
<td>-0.00000</td>
<td>-0.00000</td>
</tr>
<tr>
<td>-0.00000</td>
<td>-0.00000</td>
<td>-0.00000</td>
<td>-0.00000</td>
<td>-0.00000</td>
<td>-0.00000</td>
</tr>
<tr>
<td>-0.00000</td>
<td>-0.00000</td>
<td>-0.00000</td>
<td>-0.00000</td>
<td>-0.00000</td>
<td>-0.00000</td>
</tr>
<tr>
<td>-0.00000</td>
<td>-0.00000</td>
<td>-0.00000</td>
<td>-0.00000</td>
<td>-0.00000</td>
<td>-0.00000</td>
</tr>
<tr>
<td>-0.00000</td>
<td>-0.00000</td>
<td>-0.00000</td>
<td>-0.00000</td>
<td>-0.00000</td>
<td>-0.00000</td>
</tr>
</tbody>
</table>
### Real Part of Transform

|   | 0.81067 | 0.00526 | 0.04921 | 0.03292 | -0.01639 | -0.01372 | -0.00677 | 0.01196 | 0.00546 | 0.00945 | 0.00975 | -0.00713 | 0.01539 | -0.01854 | 0.00375 | 0.00945 | 0.01425 | 0.02524 | 0.01149 | 0.00676 | -0.00713 | -0.02518 | 0.01359 | -0.02211 | 0.00301 | -0.01639 | 0.03292 | 0.04921 | 0.00526 |
|---|---------|---------|---------|---------|----------|----------|----------|---------|---------|---------|---------|------------|---------|-----------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|------------|---------|-----------|---------|---------|---------|---------|---------|

### Imaginary Part

<p>|   | -0.13424 | -0.03590 | 0.02090 | 0.00646 | -0.03926 | -0.01928 | -0.00230 | 0.00504 | -0.04073 | -0.00435 | 0.00575 | -0.01536 | 0.00598 | -0.00435 | 0.00210 | -0.00565 | -0.00637 | 0.03250 | 0.03489 | -0.00272 | 0.01345 | -0.00215 | -0.00504 | -0.00449 | -0.00575 | -0.00557 | -0.00215 | -0.02996 | 0.03590 | -0.13424 |</p>
<table>
<thead>
<tr>
<th>Real Part of Original Input Obtained by Inverting Transform</th>
</tr>
</thead>
<tbody>
<tr>
<td>230.08999 229.98999 229.98999 230.09999 230.08999</td>
</tr>
<tr>
<td>230.02999 230.02999 230.07999 230.01999 229.89999</td>
</tr>
<tr>
<td>230.09999 230.09999 229.95999 229.94999 229.89999</td>
</tr>
<tr>
<td>230.09999 230.09999 229.97999 230.02999 230.07999</td>
</tr>
<tr>
<td>230.09999 230.09999 229.98999 230.06999 229.97999</td>
</tr>
<tr>
<td>229.86999 229.86999 229.99999 229.89999 229.73999</td>
</tr>
<tr>
<td>229.91999 229.88000 229.83999 229.46999 229.72999</td>
</tr>
<tr>
<td>229.84999 229.73999 229.60999 229.21000 229.79999</td>
</tr>
<tr>
<td>229.84999 229.73999 229.60999 229.68999 229.68999</td>
</tr>
<tr>
<td>229.84999 229.73999 229.60999 229.23999 229.04999</td>
</tr>
<tr>
<td>229.84999 229.73999 229.60999 229.94999 229.87999</td>
</tr>
<tr>
<td>229.99999 229.86999 229.66999 229.64000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Imaginary Part</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00000</td>
</tr>
<tr>
<td>0.00000</td>
</tr>
<tr>
<td>0.00000</td>
</tr>
<tr>
<td>0.00000</td>
</tr>
<tr>
<td>0.00000</td>
</tr>
<tr>
<td>0.00000</td>
</tr>
<tr>
<td>-0.00000</td>
</tr>
<tr>
<td>-0.00000</td>
</tr>
<tr>
<td>0.00000</td>
</tr>
</tbody>
</table>
APPENDIX II
"FAST FOURIER TRANSFORM" - "HARM"

HARM DISCRETE FOURIER TRANSFORM. BASIC FORTRAN IV, IBM.

SUBROUTINE HARM(NA, INV, N, IFS, IFERR).
DIMENSION A(1), INV(1), S(1), N(3), M(3), NP(3), WI(2), W2(2), W3(2)
EQUIVALENCE (N1, N(1)), (N2, N(2)), (N3, N(3))

INPUT PARAMETERS TO BE SET BY USER BEFORE ENTERING HARM-

A IS A 3-DIMENSIONAL ARRAY OF COMPLEX COEFFICIENTS.
OF DIMENSION(N1, N(2), N(3)).
THE A'S ARE STORED WITH REAL PART OF A(I1, I2, 13) IN THE LOCATION
WITH INDEX 2*(13*N(1)+I2*N(1)+I1)*1 AND THE IMAGINARY PART
IN THE LOCATION IMMEDIATELY FOLLOWING.

IF THE FOURIER SERIES IS REQUESTED, ARRAY A IS REPLACED BY
X(1, I1, I2, J3)*SUM A(K1, K2, K3)*W1**((K1*I1)*W2**((K2*I2)*W3**((K3*I3)
SUMMED OVER K1=0, N(1)-1, K2=0, N(2)-1, K3=0, N(3)-1
WHERE W1=N(I1)-TH ROOT OF UNITY.

M(1, 1, 2, 3) WHERE N(1)=2**M(1) IS THE NO. OF PTS IN THE I-TH DIM.
THE DIMENSION OF A IN THE CALLING PROGRAM SHOULD BE TWICE THE
NUMBER OF COMPLEX ELEMENTS OF THE LARGEST A ARRAY TO BE PROCESS-
FD.
THE COMPLEX X'S ARE STORED IN THE SAME MANNER AS A.

IF THE FOURIER TRANSFORM IS REQUESTED, THE ARGUMENT A IS TAKEN
TO BE X AND IS REPLACED BY THE ARRAY A SATISFYING THE FOURIER
SERIES.

LET MT=MAX(M(1), M(2), M(3))-2, NT=2**MT, WITH M BEING THE M
GIVEN WHEN THE TABLES ARE SET.

S(IJ)=SIN((I*PI/(2*NT)))*J=1, 2, 3, 4, ..., NT-1.

INV(I+1)=WORD CONTAINING BITS OF J IN INVERTED ORDER IN ITS
RIGHTMOST MT BIT POSITIONS. FOR J=0, 1, 2, ..., NT-1.

LET IFS=0 TO SET UP SIN AND INV TABLES.
IFS=1 TO SET UP SIN AND INV TABLES AND DO FOURIER SERIES.
IFS=-1 TO SET UP SIN AND INV TABLES AND DO FOURIER TRANSFORM.
IFS=2 TO DO FOURIER SERIES ONLY.
IFS=-2 TO DO FOURIER TRANSFORM ONLY.

ONE DOES NOT HAVE TO REPEAT THE CALL TO *HARM* WITH IFS=0, +1, -1
IF ONE DOES NOT CHANGE THE MAXIMUM M.
IFS=0 IF THE ARGUMENTS M ARE C, K.
IFS=1 IF THERE IS AN ERROR IN CALLING *HARM*.
IFS=0, +1, -1, IT MEANS THAT THE MAXIMUM M IS GREATER THAN 20
OR LESS THAN 3.
IFS=+2, IT MEANS THAT A SUFFICIENTLY LARGE SIN AND INV TABLE
HAS NOT BEEN COMPUTED. ONE MUST CALL *HARM* WITH IFS=0, +1 AND
WITH A MAX M(I) GREATER THAN OR EQUAL TO THE MAX M(I) FOR WHICH A
FOURIER TRANSFORM IS TO BE COMPUTED.
IFS=-1 IF ONE IS CALLING CN *HARM* WITH IFS=0, +1 TO COMPUTE
SIN. INV TABLES WHICH IT ALREADY HAS COMPUTED ON A PREVIOUS

CALL TO HARM WITH THE SAME MAXIMUM M

10 IF (IABS (IFS)-1) 900, 930, 12
12 MTT = MAXC (M1, M2, M3) - 2
   R0T2 = SORT (2, )
   IF (MTT-MT) 14, 14, 13
13 IFFRR = 1
14 RETURN
15 IFFRR = 0
16 M1 = M(1)
17 M2 = M(2)
18 M3 = M(3)
19 N1 = 2**M1
20 N2 = 2**M2
21 N3 = 2**M3
   IF (IFS) 16, 1, 20
   TO CALCULATE TRANSFORM REPLACE A BY CONJG (A)/N
22 N10T = N1*N2*N3
23 FN = NTO
24 DO 18 I = 1, N10T
   25 A (2*I-1) = A (2*I-1)/F
26 A (2*I) = -A (2*I)/F
27 NP(1) = N1*2
28 NP(2) = NP(1)*N2
29 NP(3) = NP(2)*N3
   DO 750 ID = 1, 3
   30 IL = NP (ID)-NP (ID)
   31 MI = M(ID)
   32 IF (MI) 250, 250, 30
   33 IDIF = NP (ID)
   34 KRIT = NP (ID)
   35 MEV = 2*(MT/2)
   36 IF (MI-MEV) 16, 60, 40
   37 N IS ODD, DC L = 1 CASE
   38 K1 = KRIT-2
   39 DO 5C I = 1, IL1, IDIF
   40 KLAST = K1+I
   41 DO 5C K = 1, KLAST+2
   42 KD = K+KRIT
   43 DO ONE STEP WITH L = 1, J = 0
   44 A(K) = A(K)+A(KD)
   45 A(KD) = A(K)-A(KD)
   46 T = A(KD)
   47 A(KD+1) = A(K)+T
   48 A(KD+1) = A(K+1)+T
   49 IF (MT-11250) 52, 52, 52
   50 IF (FIRST) 2
   51 JLAST = 2** (L-2)-1
   52 JLAST = 1
GO TO 70
M IS EVEN
60 IFIRST = 2
JLAST = 0
70 DO 240 L = IFIRST, MI + 2
      JIDIF = KIDIF
      KIDIF = KIDIF / 4
      KL = KIDIF - 2
      DO FOR J = 0
         DO AC I = 1, IL1, IDIF
            KLAST = I + KL
         DO AC K = I, KLAST, 2
            K1 = K + KIDIF
            K2 = K1 + KIDIF
            K3 = K2 + KIDIF
         END DO
         DO TWO STEPS WITH J = 0
            A(K1) = A(K1) + A(K2)
            A(K2) = A(K1) = A(K2)
            A(K1) = A(K1) + A(K3)
            A(K3) = A(K1) - A(K3)
            A(K) = A(K1) + A(K1)
            A(K1) = A(K1) - A(K1)
            A(K2) = A(K2) + A(K3) + T
            A(K3) = A(K2) - A(K3) + T
            T = A(K2)
            A(K2) = A(K) = T
            A(K1) = A(K) + T
            T = A(K2 + 1)
            A(K3 + 1) = A(K1 + 1) - T
            A(K1 + 1) = A(K1 + 1) + T
         END DO
         DO A(K3) = A(K1) - T
            A(K1) = A(K1) + T
            T = A(K3 + 1)
            A(K3 + 1) = A(K1 + 1) - T
            A(K1 + 1) = A(K1 + 1) + T
         END DO
         DO A(K1) = A(K1) - T
            A(K) = A(K) + T
            T = A(K1 + 1)
            A(K1 + 1) = A(K1 + 1) - T
            A(K1 + 1) = A(K1 + 1) + T
         END DO
      END DO
      R = -A(K3 + 1)
      T = A(K3)
      A(K3) = A(K2) - R
      A(K2) = A(K2) + R
      A(K3 + 1) = A(K2 + 1) - T
      80 A(K2 + 1) = A(K2 + 1) + T
      IF (JLAST).LE.235, 235, 82
      82 JJ = JJ + IF + 1
DO FOR J=1
ILAST = IL + JJ
DO 85 I = JJ, ILAST, IDIF
KLAST = KL+I
DO 85 K=I,KLAST,2
K1 = K+KBIT
K2 = K1+KBIT
K3 = K2+KBIT
C IFTTING W=(1+I)/ROOT2, W3=(-1+I)/ROOT2, W2=1,
C A(K) = A(K1)+A(K2)*I
C A(K2) = A(K1)-A(K2)*I
C A(K1) = A(K1)*W+A(K3)*W3
C A(K3) = A(K1)*W-A(K3)*W3
C
C A(K1) = A(K1)+A(K1)
C A(K1) = A(K1)-A(K1)
C A(K2) = A(K2)+A(K3)*I
C A(K3) = A(K2)-A(K3)*I
C
R = -A(K2+1)
T = A(K2)
A(K21) = A(K1)+R
A(K1) = A(K1)+R
A(K21) = A(K1)-T
A(K1) = A(K1)+T

C AWR = A(K1)-A(K1+1)
C AWI = A(K1+1)+A(K1)
C R = A(K3)-A(K3+1)
C T = A(K3)-A(K3+1)
A(K3) = (AWR-R)/ROOT2
A(K3+1) = (AWR-T)/ROOT2
A(K1) = (AWR+R)/ROOT2
A(K1+1) = (AWI+T)/ROOT2
T = A(K1)
A(K1) = A(K1)+T
A(K1) = A(K1)+T
A(K1+1) = A(K1+1)-T
A(K1+1) = A(K1)+T
R = A(K3+1)
T = A(K3)
A(K3) = A(K2)-R
A(K21) = A(K2)+R
A(K3+1) = A(K21)-T
85 A(K21) = A(K21)+T
IF(JLAST-1) 235,235,90
90 JJ = JJ + J.IDIF

C NOW DO THE REMAINING J'S
DO 230 J=2, JLAST
C
C FFTCH W'S
C DEF- W=W**INF(J) , W2=W**2 , W3=W**3
96 T=INV(J+1)
98. IC=NT-I
   W(1)=S(1C)
   W(2)=S(1)
   T?=2+I
   I2C=NT-I2
   IF(I2C<120.110.100
   G0 TO 130
   100 W2(1)=S(I2C)
   W2(2)=S(I2)
   G0 TO 130
   110 W2(1)=0n.
   W2(2)=1n.
   G0 TO 130
   C
   2*I IS IN FIRST QUADRANT
   120 I2CC=I2C+NT
   I2C=-I2C
   W2(1)=S(I2CC)
   W2(2)=S(I2CC)
   130 I3=I+I2
   I3C=NT-I3
   IF(I3C<160.150.140
   C
   13. IN FIRST QUADRANT
   140 W3(1)=S(I3C)
   W3(2)=S(I2)
   G0 TO 200
   150 W3(1)=0n.
   W3(2)=1n.
   G0 TO 200
   C
   160 I3CC=I3C+NT
   IF(I3CC<190.180.170
   C
   170 I3 IN SECOND QUADRANT
   180 I3C=-I3C
   W3(1)=S(I3C)
   W3(2)=S(I3CC)
   G0 TO 200
   190 I3CC=NT+I3CC
   I3CC=-I3CC
   W3(1)=S(I3CC)
   W3(2)=S(I3CC)
   200 TLAST=TL+JJ
   DO 220 J=JL,ILAST,1DIF
   KLAST=KL+1
   DO 220 K=1,KLAST,2
   K1=K+K0T
   K2=K1+K0T
   K3=K2+K0T
   DO 220
DO TWO STEPS WITH J NOT G

A(K) = A(K) + A(K2) * W2
A(K2) = A(K) - A(K2) * W2
A(K1) = A(K1) + W * A(K3) * W3
A(K3) = A(K1) * W - A(K3) * W3

A(K) = A(K) + A(K1)
A(K1) = A(K) - A(K1)
A(K2) = A(K2) + A(K3) * I
A(K3) = A(K2) - A(K3) * I

R = A(K2) * W2(1) - A(K2 + 1) * W2(2)
T = A(K2) * W2(2) + A(K2 + 1) * W2(1)
A(K2) = A(K) - R
A(K) = A(K) + R
A(K2 + 1) = A(K + 1) - T
A(K + 1) = A(K + 1) + T

R = A(K3) * W3(1) - A(K3 + 1) * W3(2)
T = A(K3) * W3(2) + A(K3 + 1) * W3(1)
A(K3) = A(K) - R
A(K) = A(K) + R
A(K3 + 1) = A(K1) - T
A(K1) = A(K1) + T
A(K1) = A(K1) - T
A(K + 1) = A(K + 1) + T
R = A(K3 + 1)
T = A(K3)
A(K3) = A(K2) - R
A(K) = A(K2) + R
A(K3 + 1) = A(K2 + 1) - T
A(K2 + 1) = A(K2 + 1) + T

END OF I AND K LOOPS

JJ = JJ + 1
END OF J-LOOP

JLAST = 4 * JLAST + 3
CONTINUE

END OF L LOOP
CONTINUE
END OF IN LOOP

WE NOW HAVE THE COMPLEX FOURIER SUMS BUT THEIR ADDRESSES ARE BIT-REVERSED. THE FOLLOWING ROUTINE PUTS THEM IN ORDER

NTSQ = NT * NT
M3MT = M3 - MT
370 IF(M3MT) 370, 360, 360
M3 = GR OR FO, MT
360 IGN3 = 1
N3VNT = N3 / NT

28
MINN3=NT
GO TO 38C
C M3 LESS THAN MT
370 IG03=2
N3VNT=1
NTVN3=NT/N3
MINN3=N3
380 JJD3 = NTSQ/N3
M2MT=M2-MT
450 IF (M2MT+147C.46C.46G
C M2 = GR. OR EQ. MT
460 IG02=1
N2VNT=N2/NI
MINN2=NT
GO TO 48C
C M2 LESS THAN MT
470 IG02 = 2
N2VNT=1
NTVN2=NI/N2
MINN2=N2
480 JJD2=NTSC/N2
M1MT=M1-MT
550 IF (M1MT+157C.56C.56G
C M1= GR. OR EQ. MT
560 IG01=1
N1VNT=N1/NT
MINN1=NT
GO TO 58C
C M1 LESS THAN MT
570 IG01=2
N1VNT=1
NTVN1=NT/N1
MINN1=N1
580 JJD1=NTSC/N1
600 JJ3=1
J=1
DO 880 JPP3=1*N3VNT
I PP3=INV(IJJ3)
DO 87C JP3=1*MINN3
GO TO (610.62C.1, IG03
610 IP3=INV(JP3)*N3VNT
GO TO 630
620 IP3=INV(JP3)/NTVN3
630 I3=(TPP3+IP3)*N2
700 JJ2=1
DO 870 JPP2=1*N2VNT
I PP2=INV(IJJ2)+I3
DO 86C JP2=1*MINN2
GO TO (.71C.72C.1, IG02
710 IP2=INV(JP2)*N2VNT
GO TO 73C
720 IP2=INV(JP2)/NTVN2
730 I2=(TPP2+IP2)*N1
800 JJ1=1
DO 86C JPP1=1*N1VNT
TPP1=INV(IJJ1)+I2
DO 85C JP1=1,MINN1
GO TO (P17*820)*10G1
81C IP1=INV(JP1)*NIVNT
GO TO 83C
82C IP1=INV(JP1)/NTVN1
83C I=I(I+1)+I
IF (I-I) 84C,845,845
84C T=A(I)
A(I)=A(J)
A(J+1)=T
I=A(I)+1
A(I+1)=A(J+1)
A(J+1)=T
845 CONTINUE
85C J=J+2
86C J1=J1+JJD1
C END OF JPP1 AND JP2
87C J2=J2+JJD2
C END OF JPP2 AND JP3 LOOPS
88C J3=J3+JJD3
C END OF JPP3 LOOP
IF (IFS) 882,1,1
C DOING TRANSFORM, REPLACE A BY CONJG(A)
882 DO 884 J=1,NTQI
884 A(2*1)=-A(2*1)
GO TO 1
C RETURN
C THE FOLLOWING PROGRAM COMPUTES THE SIN AND INV TABLES.
90C MT=MAXC(M(1),M(2),M(3))-2
MT=MAXC(2,MT)
904 IF (MT-2)
905 IFERP=1
GO TO 1
C RETURN
906 IFERRR=0
MT=2**MT
NT2=NT/2
C SET UP SIN TABLE
C THETA=PIE/2**(L+1) FOR L=1
910 THETA(I)=7F#35A1634
C JSTEP=2**((MT-I+1)) FOR L=1
JSTEP=NT
C JDIF=2**((MT-L)) FOR L=1
JDIF=NT2
S(JDIF)=SIN(THETA)
DO 950 L=2,MT
THETA=THETA/2.
JSTEP=JSTEP
JSTEP=MDIF
JDIF=JSTEP/2
S(JDIF)=SIN(THETA)
JC1=NT-JDIF
S(JC1)=COS(THETA)

30
JLAST = NT - JSTEP2

IF (JLAST - JSTFP) 950, 920, 920

920 DO 940 J = JSTFP, JLAST, JSTFP

JC = NT - J

JD = J * JDIF

940 S(JD) = S(J) * S(JC) + S(JDIF) * S(JC)

CONTINUE

C SET UP INV(J) TABLE

C

MTLEXP = NTV2

C MTLFXP = 2**(MT-L)**, FOR L = 1

LMTEXP = 1

C IM1FXP = 2**(L-1)**, FOR L = 1

INV(1) = C

DO 980 L = 1, MT

INV(1M1FXP + 1) = MTLEXP

DO 970 J = 2, LMTEXP

JJ = J * LMTEXP

970 INV(J,J) = INV(J) + MTLEXP

MTLEXP = MTLEXP / 2

980 IM1EXP = LMTEXP**2

IF (IFSR) 12, 1, 12

C RETURN

END