10.4 Multi-Dimensional Real Fourier Transform

A. Purpose

This subroutine computes Real Fourier Transforms for real data in up to six dimensions using the fast Fourier transform. In ND dimensions, the relations between the values x and the Fourier coefficients ξ have the form

$$x(j_1, j_2, ..., j_{ND}) = \sum_{k_1=0}^{N_1-1} \cdots \sum_{k_{ND}=0}^{N_{ND}-1} \xi(k_1, k_2, ..., k_{ND}) \times W_1^{j_1 k_1} \cdots W_{ND}^{j_{ND} k_{ND}},$$

where $N_\ell = 2^{\mathrm{M}(\ell)}$, $W_\ell = e^{2\pi i/N_\ell}$, $0 \leq j_\ell \leq N_\ell - 1$, x is real and ξ is complex.

B. Usage

B.1 Program Prototype, Single Precision

REAL $\mathbf{A}(N_1,\ N_2,\ ...,\ \geq N_{ND})$ $[N_k = 2^{\mathrm{M}(k)}]$

REAL $\mathbf{S}(\geq \max(\nu_1,\ \nu_2,\ ...,\ \nu_{ND}) - 1)$ $[\nu_k = 2^{\mathrm{M}(k)-2}]$

INTEGER M($\geq$ ND)**, ND, MS**

CHARACTER MODE

On the initial call set MS to 0 to indicate the array S() does not yet contain a sine table. Assign values to A(), MODE, M(), and ND.

$$\boxed{\textbf{CALL SRFT(A, MODE, M, ND, MS, S)}}$$

On return A() will contain computed results. S() will contain the sine table used in computing the Fourier transform. MS may have been changed.

B.2 Argument Definitions

A() [inout] If the argument MODE selects Analysis, A() contains values x on entry and the Fourier coefficients ξ on exit. If MODE selects Synthesis, A() contains the Fourier coefficients ξ on entry and the values x on exit. The Functional Description below describes the way x and ξ are stored in A().

MODE [in] The character variable MODE selects Analysis or Synthesis.

'A' or 'a' selects Analysis, transforming x's to $\xi's$.

'S' or 's' selects Synthesis, transforming ξ's to $x's$.

M() [in] Defines $N_k = 2^{\mathrm{M}(k)}$, the number of real data points in the k^{th} dimension. Require $0 \leq \mathrm{M}(k) \leq 31$ for all k, and $\mathrm{M}(1) = 0$ only if $\mathrm{M}(k) \equiv 0$ for all k. No action is taken with respect to dimensions for which $\mathrm{M}(k) = 0$.

ND [in] Number of dimensions. Require $1 \leq \mathrm{ND} \leq 6$.

MS [inout] Gives the state of the sine table in $S()$. Let MS_{in} and MS_{out} denote the values of MS on entry and return respectively. If the sine table has not previously been computed, set $\mathrm{MS}_{in} = 0$ or -1 before the call. Otherwise the value of MS_{out} from the previous call using the same S() array can be used as MS_{in} for the current call.

Certain error conditions described in Section E cause the subroutine to set $\mathrm{MS}_{out} = -2$ and return. Otherwise, with $\max_i\{\mathrm{M}(i)\} > 0$, the subroutine sets $\mathrm{MS}_{out} = \max(\mathrm{M}(1),\ \mathrm{M}(2),\ ...,\ \mathrm{M}(\mathrm{ND}),\ \mathrm{MS}_{in})$.

If $\mathrm{MS}_{out} > \max(2, \mathrm{MS}_{in})$, the subroutine sets $\mathrm{NT} = 2^{\mathrm{MS}_{out}-2}$ and fills S() with NT $-$ 1 sine values.

If $\mathrm{MS}_{in} = -1$, the subroutine returns after the above actions, not transforming the data in A(). This is intended to allow the use of the sine table for data alteration before a subsequent Fourier transform, as discussed in Section G of Chapter 16.0.

S() [inout] When the sine table has been computed, $\mathrm{S}(j) = \sin \pi j/(2 \times NT)$, $j = 1,\ 2,\ ...,\ \mathrm{NT} - 1$, see MS above.

B.3 Modifications for Double Precision

Change SRFT to DRFT and the REAL type statements to DOUBLE PRECISION.

C. Examples and Remarks

A "smooth" function that approximates

$$A(i,j) = \begin{cases} 0 & \text{if } |9-i| + |9-j| > 4 \\ 1 & \text{if } |9-i| + |9-j| \leq 4 \end{cases} \quad 1 \leq i,\ j \leq 16.$$

is desired. The example at the end of the chapter does this by computing the two dimensional transform of A, applying sigma factors (see Section G of Chapter 16.0), and then transforming back. Results are printed only for $1 \leq i,\ j \leq 9$ since, to within round-off limitations, $\mathrm{A}(9+m, 9-n) = A(9-m, 9-n)$, $1 \leq 9 \pm m \leq 16$ and $1 \leq 9 \pm n \leq 16$.

D. Functional Description

The multi-dimensional real Fourier transform is done by changing it to a problem in complex variables, doing a multi-dimensional complex Fourier transform, and then adjusting the results to obtain the solution for the original problem.

Taking complex conjugates in Eq. (1) and using the fact that x is real, it can be verified that

$$\xi(N_1 - k_1,\ N_2 - k_2, ..., N_d - k_d) = \overline{\xi}(k_1, k_2, ..., k_d),$$

0,0	0,1	$\cdots$	0,J_2-1	0,J_2	J_1,J_2+1	$\cdots$	J_1,N_2-1
J_1, 0	I	$\cdots$	I	J_1,J_2	I	$\cdots$	I
1,0	1,1	$\cdots$	1,J_2-1	1,J_2	1,J_2+1	$\cdots$	1,N_2-1
I	I	$\cdots$	I	I	I	$\cdots$	I
$\cdots$				$\cdots$			$\cdots$
J_1-1,0	J_1-1,1	$\cdots$	J_1-1,J_2-1	J_1-1, J_2	J_1-1,J_2+1	$\cdots$	J_1-1,N_2-1
I	I	$\cdots$	I	I	I	$\cdots$	I

Table 1: Storage of ξ in A() for Two Dimensions

where $N_i - k_i$ is interpreted modulo N_i. (Thus $N_i - k_i = 0$ if k_i is 0.) Using the above, it is possible to pack the nonredundant $\xi's$ in the same space as is required for x.

Storage of ξ in A() for the case $d = 2$ is illustrated in Table 1. The rows and columns in the table correspond to rows and columns in the array A(). Only the subscripts k_1, k_2 of the ξ's are given. The symbols J_1 and J_2 are used as abbreviations for $N_1/2$ and $N_2/2$ respectively. The coefficients with subscripts (0,0), $(J_1, 0)$, $(0, J_2)$, and (J_1, J_2) are real and occupy single array elements as shown. For the other coefficients, the subscript "k_1, k_2" identifies the location of the real part of $\xi(k_1, k_2)$, and an "I" immediately below such a subscript gives the location of the imaginary part of $\xi(k_1,\, k_2)$.

The first column (with the second subscript ignored) gives the storage scheme for the case $d = 1$. For $d > 2$, and with ν = the smallest value of $i(> 1)$ for which $k_i \neq 0$ and $k_i \neq N_\nu/2$, the storage scheme generalizes as follows.

If $1 < k_1 < N_1$ (A($2k_1 + 1$, $k_2 + 1$, ..., $k_d + 1$), A($2k_1 + 2$, $k_2 + 1$, ..., $k_d + 1$)) contains $\xi(k_1, k_2, ..., k_d)$. Else (A(1, $k_2 + 1$, ..., $k_d + 1$), A(2, $k_2 + 1$, ..., $k_d + 1$)) contains

$$\begin{array}{ll} \xi(0, k_2, ..., k_d) & 1 \leq k_\nu < N_\nu/2 \\ \xi(N_1/2, k_2, ..., k_d) & N_\nu/2 < k_\nu < N_\nu \end{array}$$

and when k_i is either 0 or $N_i/2$ for all i, (A(1, $k_2 + 1$, ..., $k_d + 1$), A(2, $k_2 + 1$, ..., $k_d + 1$)) contains ($\xi(0, k_2, ..., k_d)$, $\xi(N_1/2, k_2, ..., k_d)$). Note that in this last case both ξ's are real.

More details can be found in [1].

References

1. Fred T. Krogh, **RFT Multi-dimensional Real Fourier Transform**. TU Doc. CP-2309, NPO 11648, Jet Propulsion Laboratory, Pasadena, CA (1970).

E. Error Procedures and Restrictions

Require $0 \leq$ M(k) ≤ 31 and $1 \leq$ ND ≤ 6. MODE must contain one of the allowed values. If any of these conditions are violated the subroutine will issue an error message using the error processing procedures of Chapter 19.2 with a severity level of 2 to cause execution to stop. A return is made with MS $= -2$ instead of stopping if the statement "CALL ERMSET(-1)" is executed before calling this subroutine.

If the sine table does not appear to have valid data, an error message is printed, and the sine table and then the transform are computed.

F. Supporting Information

The source language is ANSI Fortran 77.

Entry	Required Files
DRFT	DFFT, DRFT, ERFIN, ERMSG, IERM1, IERV1
SRFT	ERFIN, ERMSG, IERM1, IERV1, SFFT, SRFT

Subroutine designed and written by: Fred T. Krogh, JPL, October 1969, revised January 1988.

DRSRFT

```
      program DRSRFT
c>> 1996-06-19 DRSRFT Krogh  Minor change for C conversion.
c>> 1994-10-19 DRSRFT Krogh  Changes to use M77CON
c>> 1994-08-09 DRSRFT WVS Remove '0' from format
c>> 1993-02-04 DRSRFT CLL
c>> 1989-05-07 DRSRFT FTK, CLL
c>> 1989-05-04 DRSRFT FTK, CLL
c     Demo driver for SRFT -- Multi-dimensional real Fourier transform
c     ------------------------------------------------------------------
c--S replaces "?": DR?RFT, ?RFT
c     ------------------------------------------------------------------
      integer J, J1, J2, K, L, M(2), MS, N, N2, N4, ND
      real          A(16, 16), ONE, PI, S(3), SIG, SIGD, TEMP, ZERO
      parameter (PI = 3.14159265358979323846E0)
      parameter (ONE = 1.E0)
      parameter (ZERO = 0.E0)
      data M / 4, 4 /, ND / 2 /
c     ------------------------------------------------------------------
c                        Start of code -- Construct A
      N = 2 ** M(1)
      N2 = N /2
      N4 = N2 / 2
      SIGD = PI / N2
      do 20 J1 = 1, N
         do 10 J2 = 1, N
            A(J1, J2) = ZERO
            if (abs(J1-N2-1) + abs(J2-N2-1) .le. N4) A(J1, J2) = ONE
   10    continue
   20 continue
c     ------------------------------------------------------------------
c                Compute Fourier transform and apply sigma factors
      MS = 0
      call SRFT (A, 'A', M, ND, MS, S)
      do 50 J1 = 1, N, 2
         A(J1, N2+1) = ZERO
         A(J1+1, N2+1) = ZERO
         do 40 J2 = 1, N2
            SIG = ONE
            if (J1 .EQ. 1) then
c                             No change in SIG due to J
               if (J2 .NE. 1) then
                  J = J2 - 1
                  K = 1
               else
                  A(2, 1) = ZERO
                  go to 40
               end if
            else
               J = J1 / 2
               K = 0
            end if
c                          Get nontrivial sigma factors * SIG
   30       continue
            if (J .LT. N4) then
               TEMP = S(J)
            else if (J .EQ. N4) then
```

```
                TEMP = ONE
             else
                TEMP = S(N2-J)
             end if
             SIG = SIG * TEMP / (SIGD * real(J))
             if (K .EQ. 0) then
                if (J2 .NE. 1) then
                   J = J2 - 1
                   K = 1
                   go to 30
                end if
             else
c                                              Apply sigma factors
                if (J1 .EQ. 1) then
                   A(1, N-J2+2) = ZERO
                   A(2, N-J2+2) = ZERO
                else
                   A(J1, N-J2+2) = SIG * A(J1, N-J2+2)
                   A(J1+1, N-J2+2) = SIG * A(J1+1, N-J2+2)
                end if
             end if
             A(J1, J2) = SIG * A(J1, J2)
             A(J1+1, J2) = SIG * A(J1+1, J2)
   40     continue
   50 continue
      call SRFT (A, 'S', M, ND, MS, S)
      print '(/'' Smoothed A'')'
      do 60 L = 1, 9
         print '(9f8.4)', (A(L,N), N = 1, 9)
   60 continue
      stop
      end
```

ODSRFT

```
Smoothed A
  0.0002 -0.0003  0.0004 -0.0013 -0.0009  0.0000  0.0001  0.0000 -0.0022
 -0.0003  0.0003 -0.0004  0.0013  0.0009  0.0000 -0.0002  0.0001  0.0031
  0.0004 -0.0004  0.0005 -0.0016 -0.0009 -0.0003  0.0007 -0.0014 -0.0085
 -0.0013  0.0013 -0.0016  0.0032 -0.0007  0.0033 -0.0075  0.0372  0.1110
 -0.0009  0.0009 -0.0009 -0.0007 -0.0037  0.0018  0.0159  0.2898  0.5621
  0.0000  0.0000 -0.0003  0.0033  0.0018  0.0211  0.2748  0.7259  0.9482
  0.0001 -0.0002  0.0007 -0.0075  0.0159  0.2748  0.7255  0.9780  1.0060
  0.0000  0.0001 -0.0014  0.0372  0.2898  0.7259  0.9780  1.0020  0.9977
 -0.0022  0.0031 -0.0085  0.1110  0.5621  0.9482  1.0060  0.9977  1.0027
```