C06RBF – NAG Fortran Library Routine Document

Note. Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

C06RBF computes the discrete Fourier cosine transforms of m sequences of real data values.

2 Specification

```
SUBROUTINE C06RBF(M, N, X, WORK, IFAIL)
INTEGER           M, N, IFAIL
real              X(M*(N+3)), WORK(M*N+2*N+15)
```

3 Description

Given m sequences of $n+1$ real data values x_j^p, for $j = 0, 1, \ldots, n$ and $p = 1, 2, \ldots, m$, this routine simultaneously calculates the Fourier cosine transforms of all the sequences defined by

$$\hat{x}_k^p = \sqrt{\frac{2}{n}}\left(\frac{1}{2}x_0^p + \sum_{j=1}^{n-1} x_j^p \times \cos\left(jk\frac{\pi}{n}\right) + \frac{1}{2}(-1)^k x_n^p\right), \qquad k = 0, 1, \ldots, n; \quad p = 1, 2, \ldots, m.$$

(Note the scale factor $\sqrt{\frac{2}{n}}$ in this definition.)

Since the Fourier cosine transform is its own inverse, two consecutive calls of this routine will restore the original data.

The transform calculated by this routine can be used to solve Poisson's equation when the derivative of the solution is specified at both left and right boundaries (Swarztrauber [2]).

The routine uses a variant of the fast Fourier transform (FFT) algorithm (Brigham [1]) known as the Stockham self-sorting algorithm, described in Temperton [4], together with pre- and post-processing stages described in Swarztrauber [3]. Special coding is provided for the factors 2, 3, 4 and 5.

4 References

[1] Brigham E O (1973) *The Fast Fourier Transform* Prentice–Hall

[2] Swarztrauber P N (1977) The methods of cyclic reduction, Fourier analysis and the FACR algorithm for the discrete solution of Poisson's equation on a rectangle *SIAM Rev.* **19 (3)** 490–501

[3] Swarztrauber P N (1982) Vectorizing the FFT's *Parallel Computation* (ed G Rodrique) Academic Press 51–83

[4] Temperton C (1983) Fast mixed-radix real Fourier transforms *J. Comput. Phys.* **52** 340–350

5 Parameters

1: M — INTEGER *Input*

On entry: the number of sequences to be transformed, m.

Constraint: M $\geq$ 1.

2: N — INTEGER *Input*

On entry: one less than the number of real values in each sequence, i.e., the number of values in each sequence is $n+1$.

Constraint: N $\geq$ 1.

3: X(M∗(N+3)) — ***real*** array *Input/Output*

On entry: the data must be stored in X as if in a two-dimensional array of dimension (1:M,0:N+2); each of the m sequences is stored in a **row** of the array. In other words, if the $(n+1)$ data values of the pth sequence to be transformed are denoted by x_j^p, for $j = 0, 1, \ldots, n$ and $p = 1, 2, \ldots, m$, then the first $m(n+1)$ elements of the array X must contain the values

$$x_0^1, x_0^2, \ldots, x_0^m,\ x_1^1, x_1^2, \ldots, x_1^m, \ldots,\ x_n^1, x_n^2, \ldots, x_n^m.$$

The $(n+2)$th and $(n+3)$th elements of each row x_{n+2}^p, x_{n+3}^p, for $p = 1, 2, \ldots, m$, are required as workspace. These $2m$ elements may contain arbitrary values as they are set to zero by the routine.

On exit: the m Fourier cosine transforms stored as if in a two-dimensional array of dimension (1:M,0:N+2). Each of the m transforms is stored in a **row** of the array, overwriting the corresponding original data. If the $(n+1)$ components of the pth Fourier cosine transform are denoted by $\hat{x}_k^p$, for $k = 0, 1, \ldots, n$ and $p = 1, 2, \ldots, m$, then the $m(n+3)$ elements of the array X contain the values

$$\hat{x}_0^1, \hat{x}_0^2, \ldots, \hat{x}_0^m,\ \hat{x}_1^1, \hat{x}_1^2, \ldots, \hat{x}_1^m, \ldots,\ \hat{x}_n^1, \hat{x}_n^2, \ldots, \hat{x}_n^m, 0, 0, \ldots, 0 \text{ (}2m\text{ times)}.$$

4: WORK(M∗N+2∗N+15) — ***real*** array *Workspace*

The workspace requirements as documented for this routine may be an overestimate in some implementations. For full details of the workspace required by this routine please refer to the Users' Note for your implementation.

On exit: WORK(1) contains the minimum workspace required for the current values of M and N with this implementation.

5: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = 1

On entry, M < 1.

IFAIL = 2

On entry, N < 1.

IFAIL = 3

An unexpected error has occurred in an internal call. Check all subroutine calls and array dimensions. Seek expert help.

7 Accuracy

Some indication of accuracy can be obtained by performing a subsequent inverse transform and comparing the results with the original sequence (in exact arithmetic they would be identical).

8 Further Comments

The time taken by the routine is approximately proportional to $nm \times \log n$, but also depends on the factors of n. The routine is fastest if the only prime factors of n are 2, 3 and 5, and is particularly slow if n is a large prime, or has large prime factors.

9 Example

This program reads in sequences of real data values and prints their Fourier cosine transforms (as computed by C06RBF). It then calls the routine again and prints the results which may be compared with the original sequence.

9.1 Program Text

```
*     C06RBF Example Program Text.
*     Mark 19 Release. NAG Copyright 1999.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          MMAX, NMAX
      PARAMETER        (MMAX=5,NMAX=20)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, J, M, N
*     .. Local Arrays ..
      real             WORK(MMAX*NMAX+2*NMAX+15), X((NMAX+3)*MMAX)
*     .. External Subroutines ..
      EXTERNAL         C06RBF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'C06RBF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
   20 CONTINUE
      READ (NIN,*,END=120) M, N
      IF (M.LE.MMAX .AND. N.LE.NMAX) THEN
         DO 40 J = 1, M
            READ (NIN,*) (X(I*M+J),I=0,N)
   40    CONTINUE
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Original data values'
         WRITE (NOUT,*)
         DO 60 J = 1, M
            WRITE (NOUT,99999) (X(I*M+J),I=0,N)
   60    CONTINUE
         IFAIL = 0
*
*        -- Compute transform
         CALL C06RBF(M,N,X,WORK,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Discrete Fourier cosine transforms'
         WRITE (NOUT,*)
         DO 80 J = 1, M
            WRITE (NOUT,99999) (X(I*M+J),I=0,N)
   80    CONTINUE
*
*        -- Compute inverse transform
         CALL C06RBF(M,N,X,WORK,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Original data as restored by inverse transform'
         WRITE (NOUT,*)
         DO 100 J = 1, M
            WRITE (NOUT,99999) (X(I*M+J),I=0,N)
  100    CONTINUE
```

```
         GO TO 20
      ELSE
         WRITE (NOUT,*) 'Invalid value of M or N'
      END IF
  120 CONTINUE
      STOP
*
99999 FORMAT (6X,7F10.4)
      END
```

9.2 Program Data

```
C06RBF Example Program Data
3  6 : Number of sequences, M,   (number of values in each sequence)-1, N
 0.3854  0.6772  0.1138  0.6751  0.6362  0.1424  0.9562 : X, sequence 1
 0.5417  0.2983  0.1181  0.7255  0.8638  0.8723  0.4936 : X, sequence 2
 0.9172  0.0644  0.6037  0.6430  0.0428  0.4815  0.2057 : X, sequence 3
```

9.3 Program Results

```
 C06RBF Example Program Results

 Original data values

          0.3854    0.6772    0.1138    0.6751    0.6362    0.1424    0.9562
          0.5417    0.2983    0.1181    0.7255    0.8638    0.8723    0.4936
          0.9172    0.0644    0.6037    0.6430    0.0428    0.4815    0.2057

 Discrete Fourier cosine transforms

          1.6833   -0.0482    0.0176    0.1368    0.3240   -0.5830   -0.0427
          1.9605   -0.4884   -0.0655    0.4444    0.0964    0.0856   -0.2289
          1.3838    0.1588   -0.0761   -0.1184    0.3512    0.5759    0.0110

 Original data as restored by inverse transform

          0.3854    0.6772    0.1138    0.6751    0.6362    0.1424    0.9562
          0.5417    0.2983    0.1181    0.7255    0.8638    0.8723    0.4936
          0.9172    0.0644    0.6037    0.6430    0.0428    0.4815    0.2057
```