

# NAG Library Routine Document

## C06RGF

**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

C06RGF computes the discrete quarter-wave Fourier sine transforms of  $m$  sequences of real data values. The elements of each sequence and its transform are stored contiguously.

### 2 Specification

```
SUBROUTINE C06RGF (IDIR, M, N, X, IFAIL)
  INTEGER          IDIR, M, N, IFAIL
  REAL (KIND=nag_wp) X(N,M)
```

### 3 Description

Given  $m$  sequences of  $n$  real data values  $x_j^p$ , for  $j = 1, 2, \dots, n$  and  $p = 1, 2, \dots, m$ , C06RGF simultaneously calculates the quarter-wave Fourier sine transforms of all the sequences defined by

$$\hat{x}_k^p = \frac{1}{\sqrt{n}} \left( \sum_{j=1}^{n-1} x_j^p \times \sin\left(j(2k-1)\frac{\pi}{2n}\right) + \frac{1}{2}(-1)^{k-1} x_n^p \right), \quad \text{if IDIR} = 1,$$

or its inverse

$$x_k^p = \frac{2}{\sqrt{n}} \sum_{j=1}^n \hat{x}_j^p \times \sin\left((2j-1)k\frac{\pi}{2n}\right), \quad \text{if IDIR} = -1,$$

where  $k = 1, 2, \dots, n$  and  $p = 1, 2, \dots, m$ .

(Note the scale factor  $\frac{1}{\sqrt{n}}$  in this definition.)

A call of C06RGF with IDIR = 1 followed by a call with IDIR = -1 will restore the original data.

The two transforms are also known as type-III DST and type-II DST, respectively.

The transform calculated by this routine can be used to solve Poisson's equation when the solution is specified at the left boundary, and the derivative of the solution is specified at the right boundary (see Swarztrauber (1977)).

The routine uses a variant of the fast Fourier transform (FFT) algorithm (see Brigham (1974)) known as the Stockham self-sorting algorithm, described in Temperton (1983), together with pre- and post-processing stages described in Swarztrauber (1982). Special coding is provided for the factors 2, 3, 4 and 5.

### 4 References

Brigham E O (1974) *The Fast Fourier Transform* Prentice-Hall

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

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

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

## 5 Arguments

- 1: IDIR – INTEGER *Input*  
*On entry:* indicates the transform, as defined in Section 3, to be computed.  
 IDIR = 1  
     Forward transform.  
 IDIR = -1  
     Inverse transform.  
*Constraint:* IDIR = 1 or -1.
- 2: M – INTEGER *Input*  
*On entry:*  $m$ , the number of sequences to be transformed.  
*Constraint:*  $M \geq 1$ .
- 3: N – INTEGER *Input*  
*On entry:*  $n$ , the number of real values in each sequence.  
*Constraint:*  $N \geq 1$ .
- 4: X(N,M) – REAL (KIND=nag\_wp) array *Input/Output*  
*On entry:* the data values of the  $p$ th sequence to be transformed, denoted by  $x_j^p$ , for  $j = 1, 2, \dots, n$  and  $p = 1, 2, \dots, m$ , must be stored in  $X(j, p)$ .  
*On exit:* the  $n$  components of the  $p$ th quarter-wave sine transform, denoted by  $\hat{x}_k^p$ , for  $k = 1, 2, \dots, n$  and  $p = 1, 2, \dots, m$ , are stored in  $X(k, p)$ , overwriting the corresponding original values.
- 5: IFAIL – INTEGER *Input/Output*  
*On entry:* IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation for details.  
 For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this argument, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**  
*On exit:* IFAIL = 0 unless the routine detects an error or a warning has been flagged (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 or warnings detected by the routine:

IFAIL = 1

    On entry,  $M = \langle \text{value} \rangle$ .  
     Constraint:  $M \geq 1$ .

IFAIL = 2

    On entry,  $N = \langle \text{value} \rangle$ .  
     Constraint:  $N \geq 1$ .

IFAIL = 3

On entry, IDIR =  $\langle value \rangle$ .  
Constraint: IDIR = -1 or 1.

IFAIL = 4

An internal error has occurred in this routine. Check the routine call and any array sizes. If the call is correct then please contact NAG for assistance.

IFAIL = -99

An unexpected error has been triggered by this routine. Please contact NAG.  
See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL = -399

Your licence key may have expired or may not have been installed correctly.  
See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL = -999

Dynamic memory allocation failed.  
See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

## 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 Parallelism and Performance

C06RGF is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this routine. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

## 9 Further Comments

The time taken by C06RGF is approximately proportional to  $nm \log(n)$ , but also depends on the factors of  $n$ . C06RGF 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. Workspace of order  $O(n)$  is internally allocated by this routine.

## 10 Example

This example reads in sequences of real data values and prints their quarter-wave sine transforms as computed by C06RGF with IDIR = 1. It then calls the routine again with IDIR = -1 and prints the results which may be compared with the original data.

### 10.1 Program Text

```

Program c06rgfe
!      C06RGF Example Program Text
!
!      Mark 26 Release. NAG Copyright 2016.
!
!      .. Use Statements ..
!      Use nag_library, Only: c06rgf, nag_wp

```

```

!      .. Implicit None Statement ..
      Implicit None
!      .. Parameters ..
      Integer, Parameter          :: nin = 5, nout = 6
!      .. Local Scalars ..
      Integer                      :: idir, ieof, ifail, j, m, n
!      .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable :: x(:, :)
!      .. Executable Statements ..
      Write (nout,*) 'C06RGF Example Program Results'
!      Skip heading in data file
      Read (nin,*)
loop: Do
      Read (nin,*,Iostat=ieof) m, n
      If (ieof<0) Then
         Exit loop
      End If

      Allocate (x(n,m))
      Read (nin,*)(x(1:n,j),j=1,m)
      Write (nout,*)
      Write (nout,*) 'Original data values'
      Write (nout,*)
      Write (nout,99999)(x(1:n,j),j=1,m)

!      ifail: behaviour on error exit
!      =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
      ifail = 0
!      -- Compute transform
      idir = 1
      Call c06rgf(idir,m,n,x,ifail)

      Write (nout,*)
      Write (nout,*) 'Discrete quarter-wave Fourier sine transforms'
      Write (nout,*)
      Write (nout,99999)(x(1:n,j),j=1,m)

!      -- Compute inverse transform
      idir = -1
      Call c06rgf(idir,m,n,x,ifail)

      Write (nout,*)
      Write (nout,*) 'Original data as restored by inverse transform'
      Write (nout,*)
      Write (nout,99999)(x(1:n,j),j=1,m)
      Deallocate (x)
End Do loop

99999 Format (1X,6F10.4)
      End Program c06rgfe

```

## 10.2 Program Data

```

C06RGF Example Program Data
  3      6
0.3854  0.6772  0.1138  0.6751  0.6362  0.1424      : m, n
0.5417  0.2983  0.1181  0.7255  0.8638  0.8723
0.9172  0.0644  0.6037  0.6430  0.0428  0.4815      : x

```

## 10.3 Program Results

C06RGF Example Program Results

Original data values

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

Discrete quarter-wave Fourier sine transforms

0.7304	0.2078	0.1150	0.2577	-0.2869	-0.0815
0.9274	-0.1152	0.2532	0.2883	-0.0026	-0.0635
0.6268	0.3547	0.0760	0.3078	0.4987	-0.0507

Original data as restored by inverse transform

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

---