C06PRF – 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

C06PRF computes the discrete Fourier transforms of m sequences, each containing n complex data values.

2 Specification

```
SUBROUTINE C06PRF(DIRECT, M, N, X, WORK, IFAIL)
CHARACTER*1       DIRECT
INTEGER           M, N, IFAIL
complex           X(M*N), WORK(M*N+2*N+15)
```

3 Description

Given m sequences of n complex data values z_j^p, for $j = 0, 1, \ldots, n-1$ and $p = 1, 2, \ldots, m$, this routine simultaneously calculates the (**forward** or **backward**) discrete Fourier transforms of all the sequences defined by

$$\hat{z}_k^p = \frac{1}{\sqrt{n}} \sum_{j=0}^{n-1} z_j^p \times \exp\left(\pm i \frac{2\pi jk}{n}\right), \qquad k = 0, 1, \ldots, n-1; \quad p = 1, 2, \ldots, m.$$

(Note the scale factor $\frac{1}{\sqrt{n}}$ in this definition.) The minus sign is taken in the argument of the exponential within the summation when the forward transform is required, and the plus sign is taken when the backward transform is required. A call of the routine with DIRECT = 'F' followed by a call with DIRECT = 'B' will restore the original data.

The routine uses a variant of the fast Fourier transform (FFT) algorithm (Brigham [1]) known as the Stockham self-sorting algorithm, which is described in Temperton [2]. Special code is provided for the factors 2, 3, 4 and 5.

4 References

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

[2] Temperton C (1983) Self-sorting mixed-radix fast Fourier transforms *J. Comput. Phys.* **52** 1–23

5 Parameters

1: DIRECT — CHARACTER*1 *Input*

On entry: if the **F**orward transform as defined in Section 3 is to be computed, then DIRECT must be set equal to 'F'. If the **B**ackward transform is to be computed then DIRECT must be set equal to 'B'.

Constraint: DIRECT = 'F' or 'B'.

2: M — INTEGER *Input*

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

Constraint: M ≥ 1.

3: N — INTEGER *Input*

On entry: the number of complex values in each sequence, n.

Constraint: N ≥ 1.

4: X(M∗N) — ***complex*** array *Input/Output*

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

On exit: X is overwritten by the complex transforms.

5: WORK(M∗N+2∗N+15) — ***complex*** 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: the real part of WORK(1) contains the minimum workspace required for the current values of M and N with this implementation.

6: 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

On entry, DIRECT not equal to one of 'F' or 'B'.

IFAIL = 4

On entry, N has more than 30 prime factors.

IFAIL = 5

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 complex data values and prints their discrete Fourier transforms (as computed by C06PRF with DIRECT set to 'F'). Inverse transforms are then calculated using C06PRF with DIRECT set to 'B' and printed out, showing that the original sequences are restored.

9.1 Program Text

```
*     C06PRF 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 ..
      complex          WORK((MMAX+2)*NMAX+15), X(MMAX*NMAX)
*     .. External Subroutines ..
      EXTERNAL         C06PRF
*     .. Intrinsic Functions ..
      INTRINSIC        real, imag
*     .. Executable Statements ..
      WRITE (NOUT,*) 'C06PRF 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-1)
   40    CONTINUE
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Original data values'
         DO 60 J = 1, M
            WRITE (NOUT,*)
            WRITE (NOUT,99999) 'Real ', (real(X(I*M+J)),I=0,N-1)
            WRITE (NOUT,99999) 'Imag ', (imag(X(I*M+J)),I=0,N-1)
   60    CONTINUE
         IFAIL = 0
*
         CALL C06PRF('F',M,N,X,WORK,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Discrete Fourier transforms'
         DO 80 J = 1, M
            WRITE (NOUT,*)
            WRITE (NOUT,99999) 'Real ', (real(X(I*M+J)),I=0,N-1)
            WRITE (NOUT,99999) 'Imag ', (imag(X(I*M+J)),I=0,N-1)
   80    CONTINUE
*
         CALL C06PRF('B',M,N,X,WORK,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Original data as restored by inverse transform'
         DO 100 J = 1, M
            WRITE (NOUT,*)
```

```
            WRITE (NOUT,99999) 'Real ', (real(X(I*M+J)),I=0,N-1)
            WRITE (NOUT,99999) 'Imag ', (imag(X(I*M+J)),I=0,N-1)
  100    CONTINUE
         GO TO 20
      ELSE
         WRITE (NOUT,*) 'Invalid value of M or N'
      END IF
  120 CONTINUE
      STOP
*
99999 FORMAT (1X,A,6F10.4)
      END
```

9.2 Program Data

```
C06PRF Example Program Data
     3     6
     (0.3854,0.5417)
     (0.6772,0.2983)
     (0.1138,0.1181)
     (0.6751,0.7255)
     (0.6362,0.8638)
     (0.1424,0.8723)
     (0.9172,0.9089)
     (0.0644,0.3118)
     (0.6037,0.3465)
     (0.6430,0.6198)
     (0.0428,0.2668)
     (0.4815,0.1614)
     (0.1156,0.6214)
     (0.0685,0.8681)
     (0.2060,0.7060)
     (0.8630,0.8652)
     (0.6967,0.9190)
     (0.2792,0.3355)
```

9.3 Program Results

```
 C06PRF Example Program Results

 Original data values

 Real      0.3854    0.6772    0.1138    0.6751    0.6362    0.1424
 Imag      0.5417    0.2983    0.1181    0.7255    0.8638    0.8723

 Real      0.9172    0.0644    0.6037    0.6430    0.0428    0.4815
 Imag      0.9089    0.3118    0.3465    0.6198    0.2668    0.1614

 Real      0.1156    0.0685    0.2060    0.8630    0.6967    0.2792
 Imag      0.6214    0.8681    0.7060    0.8652    0.9190    0.3355

 Discrete Fourier transforms

 Real      1.0737   -0.5706    0.1733   -0.1467    0.0518    0.3625
 Imag      1.3961   -0.0409   -0.2958   -0.1521    0.4517   -0.0321

 Real      1.1237    0.1728    0.4185    0.1530    0.3686    0.0101
```

```
Imag     1.0677    0.0386    0.7481    0.1752    0.0565    0.1403

Real     0.9100   -0.3054    0.4079   -0.0785   -0.1193   -0.5314
Imag     1.7617    0.0624   -0.0695    0.0725    0.1285   -0.4335

Original data as restored by inverse transform

Real     0.3854    0.6772    0.1138    0.6751    0.6362    0.1424
Imag     0.5417    0.2983    0.1181    0.7255    0.8638    0.8723

Real     0.9172    0.0644    0.6037    0.6430    0.0428    0.4815
Imag     0.9089    0.3118    0.3465    0.6198    0.2668    0.1614

Real     0.1156    0.0685    0.2060    0.8630    0.6967    0.2792
Imag     0.6214    0.8681    0.7060    0.8652    0.9190    0.3355
```