C06PXF – 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

C06PXF computes the three-dimensional discrete Fourier transform of a trivariate sequence of complex data values (using complex data type).

2 Specification

```
SUBROUTINE C06PXF(DIRECT, N1, N2, N3, X, WORK, IFAIL)
CHARACTER*1       DIRECT
INTEGER           N1, N2, N3, IFAIL
complex           X(N1*N2*N3), WORK(N1*N2*N3+N1+N2+N3+45)
```

3 Description

This routine computes the three-dimensional discrete Fourier transform of a trivariate sequence of complex data values $z_{j_1 j_2 j_3}$, where $j_1 = 0,1,\ldots,n_1-1$, $j_2 = 0,1,\ldots,n_2-1$ and $j_3 = 0,1,\ldots,n_3-1$.

The discrete Fourier transform is here defined by

$$\hat{z}_{k_1 k_2 k_3} = \frac{1}{\sqrt{n_1 n_2 n_3}} \sum_{j_1=0}^{n_1-1} \sum_{j_2=0}^{n_2-1} \sum_{j_3=0}^{n_3-1} z_{j_1 j_2 j_3} \times \exp\left(\pm 2\pi i \left(\frac{j_1 k_1}{n_1} + \frac{j_2 k_2}{n_2} + \frac{j_3 k_3}{n_3}\right)\right),$$

where $k_1 = 0,1,\ldots,n_1-1$, $k_2 = 0,1,\ldots,n_2-1$ and $k_3 = 0,1,\ldots,n_3-1$.

(Note the scale factor of $\frac{1}{\sqrt{n_1 n_2 n_3}}$ 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.

This routine calls C06PRF to perform multiple one-dimensional discrete Fourier transforms by the fast Fourier transform (FFT) algorithm (Brigham [1]).

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: N1 — INTEGER *Input*

On entry: the first dimension of the transform, n_1.

Constraint: N1 ≥ 1.

3: N2 — INTEGER *Input*

On entry: the second dimension of the transform, n_2.

Constraint: N2 $\geq$ 1.

4: N3 — INTEGER *Input*

On entry: the third dimension of the transform, n_3.

Constraint: N3 $\geq$ 1.

5: X(N1∗N2∗N3) — ***complex*** array *Input/Output*

On entry: the complex data values. If X is regarded as a three-dimensional array of dimension (0:N1−1,0:N2−1,0:N3−1), then X(j_1, j_2, j_3) must contain $z_{j_1 j_2 j_3}$.

On exit: the corresponding elements of the computed transform.

6: WORK(N1∗N2∗N3+N1+N2+N3+45) — ***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 N1, N2 and N3 with this implementation.

7: 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, N1 < 1.

IFAIL = 2

On entry, N2 < 1.

IFAIL = 3

On entry, N3 < 1.

IFAIL = 4

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

IFAIL = 5

On entry, N1 has more than 30 prime factors.

IFAIL = 6

On entry, N2 has more than 30 prime factors.

IFAIL = 7

On entry, N3 has more than 30 prime factors.

IFAIL = 8

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 $n_1 n_2 n_3 \times \log(n_1 n_2 n_3)$, but also depends on the factorization of the individual dimensions n_1, n_2 and n_3. The routine is somewhat faster than average if their only prime factors are 2, 3 or 5; and fastest of all if they are powers of 2.

9 Example

This program reads in a trivariate sequence of complex data values and prints the three-dimensional Fourier transform. It then performs an inverse transform and prints the sequence so obtained, which may be compared to the original data values.

9.1 Program Text

```
*     C06PXF Example Program Text.
*     Mark 19 Release. NAG Copyright 1999.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          N1MAX, N2MAX, N3MAX, NMAX, LWORK
      PARAMETER        (N1MAX=16,N2MAX=16,N3MAX=16,
     +                 NMAX=N1MAX*N2MAX*N3MAX,LWORK=N1MAX+N2MAX+N3MAX+
     +                 NMAX+45)
*     .. Local Scalars ..
      INTEGER          IFAIL, N, N1, N2, N3
*     .. Local Arrays ..
      complex          WORK(LWORK), X(NMAX)
*     .. External Subroutines ..
      EXTERNAL         C06PXF, READX, WRITX
*     .. Executable Statements ..
      WRITE (NOUT,*) 'C06PXF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
   20 CONTINUE
      READ (NIN,*,END=40) N1, N2, N3
      N = N1*N2*N3
      IF (N.GE.1 .AND. N.LE.NMAX) THEN
         CALL READX(NIN,X,N1,N2,N3)
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Original data values'
         CALL WRITX(NOUT,X,N1,N2,N3)
         IFAIL = 0
*
*        -- Compute transform
         CALL C06PXF('F',N1,N2,N3,X,WORK,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Components of discrete Fourier transform'
         CALL WRITX(NOUT,X,N1,N2,N3)
*
*        -- Compute inverse transform
         CALL C06PXF('B',N1,N2,N3,X,WORK,IFAIL)
```

```
*
         WRITE (NOUT,*)
         WRITE (NOUT,*)
     +     'Original sequence as restored by inverse transform'
         CALL WRITX(NOUT,X,N1,N2,N3)
         GO TO 20
      ELSE
         WRITE (NOUT,*) ' ** Invalid value of N1, N2 or N3'
      END IF
   40 CONTINUE
      STOP
      END
*
      SUBROUTINE READX(NIN,X,N1,N2,N3)
*     Read 3-dimensional complex data
*     .. Scalar Arguments ..
      INTEGER          N1, N2, N3, NIN
*     .. Array Arguments ..
      complex          X(N1,N2,N3)
*     .. Local Scalars ..
      INTEGER          I, J, K
*     .. Executable Statements ..
      DO 40 I = 1, N1
         DO 20 J = 1, N2
            READ (NIN,*) (X(I,J,K),K=1,N3)
   20    CONTINUE
   40 CONTINUE
      RETURN
      END
*
      SUBROUTINE WRITX(NOUT,X,N1,N2,N3)
*     Print 3-dimensional complex data
*     .. Scalar Arguments ..
      INTEGER          N1, N2, N3, NOUT
*     .. Array Arguments ..
      complex          X(N1,N2,N3)
*     .. Local Scalars ..
      INTEGER          I, J, K
*     .. Intrinsic Functions ..
      INTRINSIC        real, imag
*     .. Executable Statements ..
      DO 40 I = 1, N1
         WRITE (NOUT,*)
         WRITE (NOUT,99998) 'z(i,j,k) for i =', I
         DO 20 J = 1, N2
            WRITE (NOUT,*)
            WRITE (NOUT,99999) 'Real ', (real(X(I,J,K)),K=1,N3)
            WRITE (NOUT,99999) 'Imag ', (imag(X(I,J,K)),K=1,N3)
   20    CONTINUE
   40 CONTINUE
      RETURN
*
99999 FORMAT (1X,A,7F10.3,/(6X,7F10.3))
99998 FORMAT (1X,A,I6)
      END
```

9.2 Program Data

```
C06PXF Example Program Data
2 3 4  : values of N1, N2, N3
    ( 1.000, 0.000) ( 0.999,-0.040) ( 0.987,-0.159) ( 0.936,-0.352)
    ( 0.994,-0.111) ( 0.989,-0.151) ( 0.963,-0.268) ( 0.891,-0.454)
    ( 0.903,-0.430) ( 0.885,-0.466) ( 0.823,-0.568) ( 0.694,-0.720)
    ( 0.500, 0.500) ( 0.499, 0.040) ( 0.487, 0.159) ( 0.436, 0.352)
    ( 0.494, 0.111) ( 0.489, 0.151) ( 0.463, 0.268) ( 0.391, 0.454)
    ( 0.403, 0.430) ( 0.385, 0.466) ( 0.323, 0.568) ( 0.194, 0.720)
```

9.3 Program Results

```
 C06PXF Example Program Results

 Original data values

 z(i,j,k) for i =     1

 Real      1.000     0.999     0.987     0.936
 Imag      0.000    -0.040    -0.159    -0.352

 Real      0.994     0.989     0.963     0.891
 Imag     -0.111    -0.151    -0.268    -0.454

 Real      0.903     0.885     0.823     0.694
 Imag     -0.430    -0.466    -0.568    -0.720

 z(i,j,k) for i =     2

 Real      0.500     0.499     0.487     0.436
 Imag      0.500     0.040     0.159     0.352

 Real      0.494     0.489     0.463     0.391
 Imag      0.111     0.151     0.268     0.454

 Real      0.403     0.385     0.323     0.194
 Imag      0.430     0.466     0.568     0.720

 Components of discrete Fourier transform

 z(i,j,k) for i =     1

 Real      3.292     0.051     0.113     0.051
 Imag      0.102    -0.042     0.102     0.246

 Real      0.143     0.016    -0.024    -0.050
 Imag     -0.086     0.153     0.127     0.086

 Real      0.143    -0.050    -0.024     0.016
 Imag      0.290     0.118     0.077     0.051

 z(i,j,k) for i =     2

 Real      1.225     0.355     0.000    -0.355
 Imag     -1.620     0.083     0.162     0.083

 Real      0.424     0.020     0.013    -0.007
```

```
Imag      0.320    -0.115    -0.091    -0.080

Real     -0.424     0.007    -0.013    -0.020
Imag      0.320    -0.080    -0.091    -0.115

Original sequence as restored by inverse transform

z(i,j,k) for i =     1

Real      1.000     0.999     0.987     0.936
Imag      0.000    -0.040    -0.159    -0.352

Real      0.994     0.989     0.963     0.891
Imag     -0.111    -0.151    -0.268    -0.454

Real      0.903     0.885     0.823     0.694
Imag     -0.430    -0.466    -0.568    -0.720

z(i,j,k) for i =     2

Real      0.500     0.499     0.487     0.436
Imag      0.500     0.040     0.159     0.352

Real      0.494     0.489     0.463     0.391
Imag      0.111     0.151     0.268     0.454

Real      0.403     0.385     0.323     0.194
Imag      0.430     0.466     0.568     0.720
```