

NAG Library Routine Document

C06PXF

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)
  INTEGER          N1, N2, N3, IFAIL
  COMPLEX (KIND=nag_wp) X(N1*N2*N3), WORK(*)
  CHARACTER(1)    DIRECT
```

3 Description

C06PXF computes the three-dimensional discrete Fourier transform of a trivariate sequence of complex data values $z_{j_1 j_2 j_3}$, for $j_1 = 0, 1, \dots, n_1 - 1$, $j_2 = 0, 1, \dots, n_2 - 1$ and $j_3 = 0, 1, \dots, n_3 - 1$.

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

(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 C06PXF with DIRECT = 'F' followed by a call with DIRECT = 'B' will restore the original data.

This routine performs multiple one-dimensional discrete Fourier transforms by the fast Fourier transform (FFT) algorithm (see Brigham (1974)).

4 References

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

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 forward transform as defined in Section 3 is to be computed, then DIRECT must be set equal to 'F'.

If the backward transform is to be computed then DIRECT must be set equal to 'B'.

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

- 2: N1 – INTEGER *Input*
On entry: n_1 , the first dimension of the transform.
Constraint: $N1 \geq 1$.
- 3: N2 – INTEGER *Input*
On entry: n_2 , the second dimension of the transform.
Constraint: $N2 \geq 1$.
- 4: N3 – INTEGER *Input*
On entry: n_3 , the third dimension of the transform.
Constraint: $N3 \geq 1$.
- 5: X($N1 \times N2 \times N3$) – COMPLEX (KIND=nag_wp) array *Input/Output*
On entry: the complex data values. Data values are stored in X using column-major ordering for storing multidimensional arrays; that is, $z_{j_1 j_2 j_3}$ is stored in $X(1 + j_1 + n_1 j_2 + n_1 n_2 j_3)$.
On exit: the corresponding elements of the computed transform.
- 6: WORK(*) – COMPLEX (KIND=nag_wp) array *Workspace*
Note: the dimension of the array WORK must be at least $N1 \times N2 \times N3 + N1 + N2 + N3 + 45$.
 The workspace requirements as documented for C06PXF may be an overestimate in some implementations.
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. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction 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 parameter, 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, $N1 < 1$.

IFAIL = 2

On entry, $N2 < 1$.

IFAIL = 3

On entry, $N3 < 1$.

IFAIL = 4

On entry, DIRECT \neq 'F' or 'B'.

IFAIL = 8

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

IFAIL = -99

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.8 in the Essential Introduction for further information.

IFAIL = -399

Your licence key may have expired or may not have been installed correctly.

See Section 3.7 in the Essential Introduction for further information.

IFAIL = -999

Dynamic memory allocation failed.

See Section 3.6 in the Essential Introduction 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

C06PXF is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

C06PXF makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

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 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 . C06PXF is faster if the only prime factors are 2, 3 or 5; and fastest of all if they are powers of 2.

10 Example

This example 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.

10.1 Program Text

```

!   C06PXF Example Program Text
!   Mark 25 Release. NAG Copyright 2014.

Module c06pxfe_mod

!   C06PXF Example Program Module:
!       Parameters and User-defined Routines

!   .. Use Statements ..
Use nag_library, Only: nag_wp
!   .. Implicit None Statement ..
Implicit None
!   .. Accessibility Statements ..
Private
Public                               :: readx, writx
!   .. Parameters ..
Integer, Parameter, Public           :: nin = 5, nout = 6
Contains
Subroutine readx(nin,x,n1,n2,n3)
!   Read 3-dimensional complex data

!   .. Scalar Arguments ..
Integer, Intent (In)                 :: n1, n2, n3, nin
!   .. Array Arguments ..
Complex (Kind=nag_wp), Intent (Out)  :: x(n1,n2,n3)
!   .. Local Scalars ..
Integer                               :: i, j, k
!   .. Executable Statements ..
Do i = 1, n1
  Do j = 1, n2
    Read (nin,*)(x(i,j,k),k=1,n3)
  End Do
End Do
Return
End Subroutine readx

Subroutine writx(nout,x,n1,n2,n3)
!   Print 3-dimensional complex data

!   .. Scalar Arguments ..
Integer, Intent (In)                 :: n1, n2, n3, nout
!   .. Array Arguments ..
Complex (Kind=nag_wp), Intent (In)   :: x(n1,n2,n3)
!   .. Local Scalars ..
Integer                               :: i, j, k
!   .. Intrinsic Procedures ..
Intrinsic                             :: aimag, real
!   .. Executable Statements ..
Do i = 1, n1
  Write (nout,*)
  Write (nout,99998) 'z(i,j,k) for i =', i
  Do j = 1, n2
    Write (nout,*)
    Write (nout,99999) 'Real ', (real(x(i,j,k)),k=1,n3)
    Write (nout,99999) 'Imag ', (aimag(x(i,j,k)),k=1,n3)
  End Do
End Do
Return

99999  Format (1X,A,7F10.3/(6X,7F10.3))
99998  Format (1X,A,I6)
End Subroutine writx
End Module c06pxfe_mod

Program c06pxfe

!   C06PXF Example Main Program

!   .. Use Statements ..

```

```

Use nag_library, Only: c06pxf, nag_wp
Use c06pxfe_mod, Only: nin, nout, readx, writx
! .. Implicit None Statement ..
Implicit None
! .. Local Scalars ..
Integer :: ieof, ifail, n, n1, n2, n3
! .. Local Arrays ..
Complex (Kind=nag_wp), Allocatable :: work(:), x(:)
! .. Executable Statements ..
Write (nout,*) 'C06PXF Example Program Results'
! Skip heading in data file
Read (nin,*)
loop: Do
  Read (nin,*,Iostat=ieof) n1, n2, n3
  If (ieof<0) Exit loop
  n = n1*n2*n3
  Allocate (x(n),work(n+n1+n2+n3+45))
  Call readx(nin,x,n1,n2,n3)
  Write (nout,*)
  Write (nout,*) 'Original data values'
  Call writx(nout,x,n1,n2,n3)

!   ifail: behaviour on error exit
!   =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
  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)
  Deallocate (x,work)
End Do loop

End Program c06pxfe

```

10.2 Program Data

C06PXF Example Program Data

```

  2  3  4                                     : 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) : x

```

10.3 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
