

NAG Library Routine Document

C06PVF

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

C06PVF computes the two-dimensional discrete Fourier transform of a bivariate sequence of real data values.

2 Specification

SUBROUTINE C06PVF (M, N, X, Y, IFAIL)

INTEGER M, N, IFAIL
 REAL (KIND=nag_wp) X(M*N)
 COMPLEX (KIND=nag_wp) Y((M/2+1)*N)

3 Description

C06PVF computes the two-dimensional discrete Fourier transform of a bivariate sequence of real data values $x_{j_1 j_2}$, for $j_1 = 0, 1, \dots, m-1$ and $j_2 = 0, 1, \dots, n-1$.

The discrete Fourier transform is here defined by

$$\hat{z}_{k_1 k_2} = \frac{1}{\sqrt{mn}} \sum_{j_1=0}^{m-1} \sum_{j_2=0}^{n-1} x_{j_1 j_2} \times \exp\left(-2\pi i \left(\frac{j_1 k_1}{m} + \frac{j_2 k_2}{n}\right)\right),$$

where $k_1 = 0, 1, \dots, m-1$ and $k_2 = 0, 1, \dots, n-1$. (Note the scale factor of $\frac{1}{\sqrt{mn}}$ in this definition.)

The transformed values $\hat{z}_{k_1 k_2}$ are complex. Because of conjugate symmetry (i.e., $\hat{z}_{k_1 k_2}$ is the complex conjugate of $\hat{z}_{(m-k_1)k_2}$), only slightly more than half of the Fourier coefficients need to be stored in the output.

A call of C06PVF followed by a call of C06PWF will restore the original data.

This routine calls C06PQF and C06PRF to perform multiple one-dimensional discrete Fourier transforms by the fast Fourier transform (FFT) algorithm in Brigham (1974) and Temperton (1983).

4 References

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

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

5 Parameters

- 1: M – INTEGER *Input*
On entry: m , the first dimension of the transform.
Constraint: $M \geq 1$.
- 2: N – INTEGER *Input*
On entry: n , the second dimension of the transform.
Constraint: $N \geq 1$.

- 3: $X(M \times N)$ – REAL (KIND=nag_wp) array *Input*
On entry: the real input dataset x , where $x_{j_1 j_2}$ is stored in $X(j_2 \times m + j_1 + 1)$, for $j_1 = 0, 1, \dots, m - 1$ and $j_2 = 0, 1, \dots, n - 1$. That is, if X is regarded as a two-dimensional array of dimension $(0 : M - 1, 0 : N - 1)$, then $X(j_1, j_2)$ must contain $x_{j_1 j_2}$.
- 4: $Y((M/2 + 1) \times N)$ – COMPLEX (KIND=nag_wp) array *Output*
On exit: the complex output dataset \hat{z} , where $\hat{z}_{k_1 k_2}$ is stored in $Y(k_2 \times (m/2 + 1) + k_1 + 1)$, for $k_1 = 0, 1, \dots, m/2$ and $k_2 = 0, 1, \dots, n - 1$. That is, if Y is regarded as a two-dimensional array of dimension $(0 : M/2, 0 : N - 1)$, then $Y(k_1, k_2)$ contains $\hat{z}_{k_1 k_2}$. Note the first dimension is cut roughly by half to remove the redundant information due to conjugate symmetry.
- 5: 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, $M = \langle value \rangle$.
 Constraint: $M \geq 1$.

IFAIL = 2

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

IFAIL = 3

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

IFAIL = -999

Dynamic memory allocation failed.

7 Accuracy

Some indication of accuracy can be obtained by performing a forward transform using C06PVF and a backward transform using C06PWF, and comparing the results with the original sequence (in exact arithmetic they would be identical).

8 Further Comments

The time taken by C06PVF is approximately proportional to $mn \log(mn)$, but also depends on the factors of m and n . C06PVF is fastest if the only prime factors of m and n are 2, 3 and 5, and is particularly slow if m or n is a large prime, or has large prime factors.

Workspace is internally allocated by C06PVF. The total size of these arrays is approximately proportional to mn .

9 Example

This example reads in a bivariate sequence of real data values and prints their discrete Fourier transforms as computed by C06PVF. Inverse transforms are then calculated by calling C06PWF showing that the original sequences are restored.

9.1 Program Text

```
! C06PVF Example Program Text
! Mark 24 Release. NAG Copyright 2012.

Module c06pvfe_mod

! C06PVF Example Program Module:
! Parameters and User-defined Routines

! .. Use Statements ..
Use nag_library, Only: nag_wp
! .. Implicit None Statement ..
Implicit None
! .. Parameters ..
Integer, Parameter :: nin = 5, nout = 6
Contains
Subroutine readx(nin,x,n1,n2)
! Read 2-dimensional real data

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

Subroutine writx(nout,x,n1,n2)
! Print 2-dimensional real data

! .. Scalar Arguments ..
Integer, Intent (In) :: n1, n2, nout
! .. Array Arguments ..
Real (Kind=nag_wp), Intent (In) :: x(n1,n2)
! .. Local Scalars ..
Integer :: i, j
! .. Executable Statements ..
Do i = 1, n1
Write (nout,*)
Write (nout,99999) 'Real ', (x(i,j),j=1,n2)
End Do
Return

99999 Format (1X,A,3F10.3)
End Subroutine writx

Subroutine writy(nout,y,n1,n2)
```

```

!      Print 2-dimensional complex data

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

99999  Format (1X,A,7F10.3/(6X,7F10.3))
End Subroutine writy
End Module c06pvfe_mod

Program c06pvfe

!      C06PVF Example Main Program

!      .. Use Statements ..
Use nag_library, Only: c06pvf, c06pwf, nag_wp
Use c06pvfe_mod, Only: nin, nout, readx, writx, writy
!      .. Implicit None Statement ..
Implicit None
!      .. Local Scalars ..
Integer                        :: ieof, ifail, m, n
!      .. Local Arrays ..
Complex (Kind=nag_wp), Allocatable :: y(:)
Real (Kind=nag_wp), Allocatable   :: x(:)
!      .. Executable Statements ..
Write (nout,*) 'C06PVF Example Program Results'
!      Skip heading in data file
Read (nin,*)
loop: Do
  Read (nin,*,Iostat=ieof) m, n
  If (ieof<0) Exit loop
  Allocate (x(m*n),y((m/2+1)*n))

  Call readx(nin,x,m,n)
  Write (nout,*)
  Write (nout,*) 'Original data values'
  Call writx(nout,x,m,n)

!      ifail: behaviour on error exit
!      =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
  ifail = 0
!      -- Compute transform
  Call c06pvf(m,n,x,y,ifail)

  Write (nout,*)
  Write (nout,*) 'Components of discrete Fourier transform'
  Call writy(nout,y,m/2+1,n)

!      -- Compute inverse transform
  x = 0._nag_wp
  Call c06pwf(m,n,y,x,ifail)

  Write (nout,*)
  Write (nout,*) 'Original sequence as restored by inverse transform'
  Call writx(nout,x,m,n)

```

```

        Deallocate (x,y)
    End Do loop
End Program c06pvfe

```

9.2 Program Data

```

C06PVF Example Program Data
5 2                               : m, n
  0.010
  0.346
  1.284
  1.960
  1.754
  0.855
  0.089
  0.161
  1.004
  1.844                           : x

```

9.3 Program Results

C06PVF Example Program Results

Original data values

```

Real      0.010      0.346
Real      1.284      1.960
Real      1.754      0.855
Real      0.089      0.161
Real      1.004      1.844

```

Components of discrete Fourier transform

```

Real      2.943      -0.324
Imag      0.000      0.000

Real      -0.024     -0.466
Imag     -0.558     -0.230

Real     -1.167      0.362
Imag      0.636      0.262

```

Original sequence as restored by inverse transform

```

Real      0.010      0.346
Real      1.284      1.960
Real      1.754      0.855
Real      0.089      0.161
Real      1.004      1.844

```
