NAG Library Routine Document

F16GCF (BLAS_ZAXPBY)

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

F16GCF (BLAS_ZAXPBY) computes the sum of two scaled vectors, for complex scalars and vectors.

2 Specification

```
SUBROUTINE F16GCF (N, ALPHA, X, INCX, BETA, Y, INCY)

INTEGER                 N, INCX, INCY
COMPLEX (KIND=nag_wp) ALPHA, X(1+(N-1)*ABS(INCX)), BETA,                    &
                        Y(1+(N-1)*ABS(INCY))
```

The routine may be called by its BLAST name ***blas_zaxpby***.

3 Description

F16GCF (BLAS_ZAXPBY) performs the operation

$$y \leftarrow \alpha x + \beta y,$$

where x and y are n-element complex vectors, and α and β are complex scalars. If n is less than or equal to zero, or if α is equal to zero and β is equal to 1, this routine returns immediately.

4 References

Basic Linear Algebra Subprograms Technical (BLAST) Forum (2001) *Basic Linear Algebra Subprograms Technical (BLAST) Forum Standard* University of Tennessee, Knoxville, Tennessee http://www.netlib.org/blas/blast-forum/blas-report.pdf

5 Parameters

1: N – INTEGER *Input*

On entry: n, the number of elements in x and y.

2: ALPHA – COMPLEX (KIND=nag_wp) *Input*

On entry: the scalar α.

3: X($1 + (\mathrm{N} - 1) \times |\mathrm{INCX}|$) – COMPLEX (KIND=nag_wp) array *Input*

On entry: the n-element vector x.

If INCX > 0, x_i must be stored in X($1 + (i - 1) \times$ INCX), for $i = 1, 2, \ldots, \mathrm{N}$.

If INCX < 0, x_i must be stored in X($1 - (\mathrm{N} - i) \times$ INCX), for $i = 1, 2, \ldots, \mathrm{N}$.

Intermediate elements of X are not referenced.

4: INCX – INTEGER *Input*

On entry: the increment in the subscripts of X between successive elements of x.

Constraint: INCX $\neq 0$.

5: BETA – COMPLEX (KIND=nag_wp) *Input*

On entry: the scalar β.

6: Y$(1 + (\text{N} - 1) \times |\text{INCY}|)$ – COMPLEX (KIND=nag_wp) array *Input/Output*

On entry: the n-element vector y.

If INCY > 0, y_i must be stored in Y$(1 + (i - 1) \times \text{INCY})$, for $i = 1, 2, \ldots, \text{N}$.

If INCY < 0, y_i must be stored in Y$(1 - (\text{N} - i) \times \text{INCY})$, for $i = 1, 2, \ldots, \text{N}$.

Intermediate elements of Y are not referenced.

On exit: the updated vector y stored in the array elements used to supply the original vector y.

Intermediate elements of Y are unchanged.

7: INCY – INTEGER *Input*

On entry: the increment in the subscripts of Y between successive elements of y.

Constraint: INCY $\neq 0$.

6 Error Indicators and Warnings

If INCX $= 0$ or INCY $= 0$, an error message is printed and program execution is terminated.

7 Accuracy

The BLAS standard requires accurate implementations which avoid unnecessary over/underflow (see Section 2.7 of Basic Linear Algebra Subprograms Technical (BLAST) Forum (2001)).

8 Further Comments

None.

9 Example

This example computes the result of a scaled vector accumulation for

$$\alpha = 3 + 2i, \qquad x = (-4 + 2.1i, 3.7 + 4.5i, -6 + 1.2i)^{\text{T}},$$
$$\beta = -i, \qquad y = (-3 - 2.4i, 6.4 - 5i, -5.1)^{\text{T}}.$$

9.1 Program Text

```
    Program f16gcfe

!     F16GCF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: blas_zaxpby, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter              :: nin = 5, nout = 6
!     .. Local Scalars ..
      Complex (Kind=nag_wp)           :: alpha, beta
      Integer                         :: incx, incy, n, nx, ny
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: x(:), y(:)
!     .. Intrinsic Procedures ..
      Intrinsic                       :: abs
!     .. Executable Statements ..
```

```
      Write (nout,*) 'F16GCF Example Program Results'

!     Skip heading in data file
      Read (nin,*)

      Read (nin,*) n
      Read (nin,*) incx, incy

      nx = 1 + (n-1)*abs(incx)
      ny = 1 + (n-1)*abs(incy)
      Allocate (x(nx),y(ny))

      Read (nin,*) alpha, beta
      Read (nin,*) x(1:nx:abs(incx))
      Read (nin,*) y(1:ny:abs(incy))

!     Compute Y = alpha*X + beta*Y

      Call blas_zaxpby(n,alpha,x,incx,beta,y,incy)

      Write (nout,*)
      Write (nout,99999)
      Write (nout,99998) y(1:ny:abs(incy))

99999 Format (1X,'Result of scaled vector addition is')
99998 Format (1X,'Y = ( ',2('(',F9.4,',',F9.4,'), '),'(',F9.4,',',F9.4,') )')
    End Program f16gcfe
```

9.2 Program Data

```
F16GCF Example Program Data
    3                                       : n
   -1           -1                          : incx and incy
  ( 3., 2.0)   ( 0.0,-1.0)                  : alpha and beta
  (-4., 2.1)   ( 3.7, 4.5)   (-6.0, 1.2)    : x
  (-3.,-2.4)   ( 6.4,-5.0)   (-5.1, 0.0)    : y
```

9.3 Program Results

```
 F16GCF Example Program Results

 Result of scaled vector addition is
 Y = ( ( -18.6000,   1.3000), (  -2.9000,  14.5000), ( -20.4000,  -3.3000) )
```