

NAG Library Routine Document

F16EHF (BLAS_DWAXPBY)

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

F16EHF (BLAS_DWAXPBY) computes the sum of two scaled vectors, preserving input, for real scalars and vectors.

2 Specification

```
SUBROUTINE F16EHF (N, ALPHA, X, INCX, BETA, Y, INCY, W, INCW)
INTEGER          N, INCX, INCY, INCW
REAL (KIND=nag_wp) ALPHA, X(1+(N-1)*ABS(INCX)), BETA,      &
                  Y(1+(N-1)*ABS(INCY)), W(1+(N-1)*ABS(INCW))
```

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

3 Description

F16EHF (BLAS_DWAXPBY) performs the operation

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

where x and y are n -element real vectors, and α and β are real scalars.

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 Arguments

- | | | |
|----|---|--------------|
| 1: | N – INTEGER | <i>Input</i> |
| | <i>On entry:</i> n , the number of elements in x , y and w . | |
| 2: | ALPHA – REAL (KIND=nag_wp) | <i>Input</i> |
| | <i>On entry:</i> the scalar α . | |
| 3: | X(1 + (N – 1) × INCX) – REAL (KIND=nag_wp) array | <i>Input</i> |
| | <i>On entry:</i> the n -element vector x . | |
| | If INCX > 0, x_i must be stored in X(($i - 1$) × INCX + 1), for $i = 1, 2, \dots, N$. | |
| | If INCX < 0, x_i must be stored in X((N – i) × INCX + 1), for $i = 1, 2, \dots, N$. | |
| | Intermediate elements of X are not referenced. If N = 0, X is not referenced. | |
| 4: | INCX – INTEGER | <i>Input</i> |
| | <i>On entry:</i> the increment in the subscripts of X between successive elements of x . | |
| | <i>Constraint:</i> INCX ≠ 0. | |

- 5: BETA – REAL (KIND=nag_wp) Input
On entry: the scalar β .
- 6: Y(1 + (N – 1) × |INCY|) – REAL (KIND=nag_wp) array Input
On entry: the n -element vector y .
 If INCY > 0, y_i must be stored in Y(($i - 1$) × INCY + 1), for $i = 1, 2, \dots, N$.
 If INCY < 0, y_i must be stored in Y((N – i) × |INCY| + 1), for $i = 1, 2, \dots, N$.
 Intermediate elements of Y are not referenced. If $\beta = 0.0$ or $N = 0$, Y is not referenced.
- 7: INCY – INTEGER Input
On entry: the increment in the subscripts of Y between successive elements of y .
Constraint: INCY \neq 0.
- 8: W(1 + (N – 1) × |INCW|) – REAL (KIND=nag_wp) array Output
On exit: the n -element vector w .
 If INCW > 0, w_i is in W(($i - 1$) × INCW + 1), for $i = 1, 2, \dots, N$.
 If INCW < 0, w_i is in W((N – i) × |INCW| + 1), for $i = 1, 2, \dots, N$.
 Intermediate elements of W are not referenced.
- 9: INCW – INTEGER Input
On entry: the increment in the subscripts of W between successive elements of w .
Constraint: INCW \neq 0.

6 Error Indicators and Warnings

If INCX = 0 or INCY = 0 or INCW = 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 Parallelism and Performance

F16EHF (BLAS_DWAXPBY) is not threaded in any implementation.

9 Further Comments

None.

10 Example

This example computes the result of a scaled vector accumulation for

$$\begin{aligned} \alpha &= 3, & x &= (-6, 4.5, 3.7, 2.1, -4)^T, \\ \beta &= -1, & y &= (-5.1, -5, 6.4, -2.4, -3)^T. \end{aligned}$$

x and y , and also the sum vector w , are stored in reverse order.

10.1 Program Text

```

Program f16ehfe

!      F16EHF Example Program Text
!
!      Mark 26 Release. NAG Copyright 2016.
!
!      .. Use Statements ..
!      Use nag_library, Only: blas_dwaxpby, nag_wp
!      .. Implicit None Statement ..
!      Implicit None
!      .. Parameters ..
!      Integer, Parameter          :: nin = 5, nout = 6
!      .. Local Scalars ..
!      Real (Kind=nag_wp)         :: alpha, beta
!      Integer                    :: i, incw, incx, incy, ix, iy, n
!      .. Local Arrays ..
!      Real (Kind=nag_wp), Allocatable :: w(:), x(:), y(:)
!      .. Intrinsic Procedures ..
!      Intrinsic                  :: abs
!      .. Executable Statements ..
!      Write (nout,*) 'F16EHF Example Program Results'

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

!      Read (nin,*) n
!      Read (nin,*) incx, incy, incw
!      Allocate (w(1+(n-1)*abs(incw)),x(1+(n-1)*abs(incx)),y(1+(n-1)*abs(incy)))
!
!      Read (nin,*) alpha, beta

!      Read the vectors x and y and store forwards or backwards
!      as determined by incx (resp. incy).
!      If (incx>0) Then
!          ix = 1
!      Else
!          ix = 1 - (n-1)*incx
!      End If

!      Do i = 1, n
!          Read (nin,*) x(ix)
!          ix = ix + incx
!      End Do

!      If (incy>0) Then
!          iy = 1
!      Else
!          iy = 1 - (n-1)*incy
!      End If

!      Do i = 1, n
!          Read (nin,*) y(iy)
!          iy = iy + incy
!      End Do

!      Compute w = alpha*x + beta*y

!      Call blas_dwaxpby(n,alpha,x,incx,beta,y,incy,w,incw)

!      Display the vector w forwards or backwards
!      as determined by incw.
!      Write (nout,*)
!      Write (nout,99999)
!      If (incw>0) Then
!          Write (nout,99998) w(1:1+(n-1)*incw:incw)
!      Else
!          Write (nout,99998) w(1-(n-1)*incw:1:incw)

```

```
      End If
99999 Format (1X,'Result of scaled vector addition is')
99998 Format (1X,'w =',5F9.4)
      End Program f16ehfe
```

10.2 Program Data

F16EHF Example Program Data

```
5
-1   -1   -1           : n
3.0  -1.0              : incx, incy and incw
-6.0
4.5
3.7
2.1
-4.0                   : Vector x
-5.1
-5.0
6.4
-2.4
-3.0                   : Vector y
```

10.3 Program Results

F16EHF Example Program Results

```
Result of scaled vector addition is
w = -12.9000  18.5000  4.7000  8.7000  -9.0000
```
