NAG Library Function Document

nag_zload (f16hbc)

1 Purpose

nag_zload (f16hbc) broadcasts a scalar into a complex vector.

2 Specification

```
#include <nag.h>
#include <nagf16.h>
```

```
void nag_zload (Integer n, Complex alpha, Complex x[], Integer incx,
     NagError *fail)
```

3 Description

nag_zload (f16hbc) performs the operation

$$x \leftarrow (\alpha, \alpha, \ldots, \alpha)^{\mathrm{T}},$$

where x is an n-element complex vector and α is a complex scalar.

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 *Input*

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

Constraint: $\mathbf{n} \geq 0$.

2: **alpha** – Complex *Input*

On entry: the scalar α.

3: $\mathbf{x}[\mathit{dim}]$ – Complex *Output*

Note: the dimension, *dim*, of the array **x** must be at least $\max(1, 1 + (\mathbf{n} - 1)|\mathbf{incx}|)$.

On exit: the scalar α scattered with a stride of **incx**. Intermediate elements of **x** are unchanged.

4: **incx** – Integer *Input*

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

Constraint: $\mathbf{incx} \neq 0$.

5: **fail** – NagError * *Input/Output*

The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_BAD_PARAM

On entry, argument ⟨*value*⟩ had an illegal value.

NE_INT

On entry, **incx** = ⟨*value*⟩.
Constraint: **incx** $\neq 0$.

On entry, **n** = ⟨*value*⟩.
Constraint: **n** ≥ 0.

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

Not applicable.

9 Further Comments

None.

10 Example

The scalar $0.5 - 0.3i$ is loaded into a vector of length 4, stored in **x** with increment 2 (**incx** = 2).

10.1 Program Text

```
/* nag_zload (f16hbc) Example Program.
 *
 * Copyright 2005 Numerical Algorithms Group.
 *
 * Mark 8, 2005.
 */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagf16.h>

int main(void)
{

  /* Scalars */
  Complex  alpha;
  Integer  exit_status, i, incx, n, xlen;

  /* Arrays */
  Complex  *x = 0;

  /* Nag Types */
  NagError fail;

  exit_status = 0;
  INIT_FAIL(fail);

  printf("nag_zload (f16hbc) Example Program Results\n\n");

  /* Skip heading in data file */
  scanf("%*[^\n] ");

  /* Read length of vector and increment. */
  scanf("%ld%ld%*[^\n] ", &n, &incx);

  /* Read scalar parameter */
  scanf(" ( %lf , %lf ) %*[^\n] ", &alpha.re, &alpha.im);

  xlen = MAX(1, 1 + (n - 1)*ABS(incx));
  if (n > 0)
    {
      /* Allocate memory */
      if (!(x = NAG_ALLOC(xlen, Complex)))
        {
          printf("Allocation failure\n");
          exit_status = -1;
          goto END;
        }
    }
  else
    {
      printf("Invalid n\n");
      exit_status = 1;
      return exit_status;
    }

  /* nag_zload (f16hbc).
   * Broadcast a complex scalar to a complex vector.
   *
   */
  nag_zload(n, alpha, x, incx, &fail);
  if (fail.code != NE_NOERROR)
    {
      printf("Error from nag_zload.\n%s\n", fail.message);
      exit_status = 1;
      goto END;
    }

  /* Print x. */
  printf("Loaded vector x:\n\n");
  for (i = 0; i < xlen; i = i + incx)
    printf(" x[%1ld] = (%5.2f, %5.2f)\n", i, x[i].re, x[i].im);
 END:
  NAG_FREE(x);

  return exit_status;
}
```

10.2 Program Data

```
nag_zload (f16hbc) Example Program Data
  4  2                      : n, incx the length and increment of x
  ( 0.5,-0.3)               : alpha
```

10.3 Program Results

```
nag_zload (f16hbc) Example Program Results

Loaded vector x:

 x[0] = ( 0.50, -0.30)
 x[2] = ( 0.50, -0.30)
 x[4] = ( 0.50, -0.30)
 x[6] = ( 0.50, -0.30)
```