NAG Library Function Document

nag_zamax_val (f16jsc)

1 Purpose
nag_zamax_val (f16jsc) computes, with respect to absolute value, the largest component of a complex vector, along with the index of that component.

2 Specification

```c
#include <nag.h>
#include <nagf16.h>
void nag_zamax_val (Integer n, const Complex x[], Integer incx, Integer *k, 
double *r, NagError *fail)
```

3 Description
nag_zamax_val (f16jsc) computes, with respect to absolute value, the largest component, \( r \), of an \( n \)-element complex vector \( x \), and determines the smallest index, \( k \), such that

\[
|Re x_k| + |Im x_k| = \max_j |Re x_j| + |Im x_j|.
\]

4 References


5 Arguments

1: \( n \) – Integer

- **Input**
- On entry: \( n \), the number of elements in \( x \).
- Constraint: \( n \geq 0 \).

2: \( x[dim] \) – const Complex

- **Input**
- Note: the dimension, \( dim \), of the array \( x \) must be at least \( \max(1, 1 + (n - 1) \times |\text{incx}|) \).
- On entry: the vector \( x \). Element \( x_i \) is stored in \( x[(i - 1) \times |\text{incx}|] \), for \( i = 1, 2, \ldots, n \).

3: \( \text{incx} \) – Integer

- **Input**
- On entry: the increment in the subscripts of \( x \) between successive elements of \( x \).
- Constraint: \( \text{incx} \neq 0 \).

4: \( k \) – Integer *

- **Output**
- On exit: \( k \), the index, from the set \( \{0, |\text{incx}|, \ldots, (n - 1) \times |\text{incx}|\} \), of the largest component of \( x \) with respect to absolute value. If \( n = 0 \) on input then \( k \) is returned as \(-1\).

5: \( r \) – double *

- **Output**
- On exit: \( r \), the largest component of \( x \) with respect to absolute value. If \( n = 0 \) on input then \( r \) is returned as 0.0.

Mark 25
The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_ALLOC_FAIL
Dynamic memory allocation failed.
See Section 3.2.1.2 in the Essential Introduction for further information.

NE_BAD_PARAM
On entry, argument <value> had an illegal value.

NE_INT
On entry, <value> <value>.
Constraint: <value>.
On entry, <value> <value>.
Constraint: <value>.

NE_INTERNAL_ERROR
An unexpected error has been triggered by this function. Please contact NAG.
See Section 3.6.6 in the Essential Introduction for further information.

NE_NO_LICENCE
Your licence key may have expired or may not have been installed correctly.
See Section 3.6.5 in the Essential Introduction for further information.

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
This example computes the largest component with respect to absolute value and index of that component for the vector

\[ x = (-4 + 2.1i, 3.7 + 4.5i, -6 + 1.2i)^T. \]

10.1 Program Text
/* nag_zamax_val (f16jsc) Example Program. */
* Copyright 2014 Numerical Algorithms Group.
* * Mark 9, 2009.
*/
```c
#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagf16.h>

int main(void)
{
    /* Scalars */
    Integer exit_status, i, incx, k, n, xlen;
    double r;
    /* Arrays */
    /* Nag Types */
    NagError fail;

    exit_status = 0;
    INIT_FAIL(fail);

    printf("nag_zamax_val (f16jsc) Example Program Results\n\n");

    /* Skip heading in data file */
    #ifdef _WIN32
        scanf_s("%*[\n"]
    #else
        scanf("%*[\n"]
    #endif
    /* Read the number of elements and the increment */
    #ifdef _WIN32
        scanf_s("%"NAG_IFMT"%"NAG_IFMT"%*[\n"] , &n, &incx);
    #else
        scanf("%"NAG_IFMT"%"NAG_IFMT"%*[\n"] , &n, &incx);
    #endif
    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;
        goto END;
    }

    /* Input vector x */
    for (i = 0; i < xlen; i = i + incx)
    #ifdef _WIN32
        scanf_s(" ( %lf , %lf ) ", &x[i].re, &x[i].im);
    #else
        scanf(" ( %lf , %lf ) ", &x[i].re, &x[i].im);
    #endif
    #ifdef _WIN32
        scanf_s("%*[\n"]
    #else
        scanf("%*[\n"]
    #endif

    /* nag_zamax_val (f16jsc). */
    /* Get absolutely maximum value (r) and location of that value (k) */
    /* of Complex array */
    nag_zamax_val(n, x, incx, &k, &r, &fail);
    if (fail.code != NE_NOERROR)
    {
        Mark 25
```
printf("Error from nag_zamax_val (f16jsc).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
}

/* Print the absolutely maximum value */
/* Print its location */
printf("Absolutely maximum element of x is %12.5f\n", r);
printf("Index of absolutely maximum element of x is %3"NAG_IFMT"\n", k);

END:
    NAG_FREE(x);
    return exit_status;
}

10.2 Program Data

nag_zamax_val (f16jsc) Example Program Data

3 1
(-4., 2.1) ( 3.7, 4.5) (-6., 1.2) : n and incx
(-4., 2.1) ( 3.7, 4.5) (-6., 1.2) : Array x

10.3 Program Results

nag_zamax_val (f16jsc) Example Program Results

Absolutely maximum element of x is 8.20000
Index of absolutely maximum element of x is 1