NAG Library Function Document

**nag_iamax_val (f16dqc)**

1 Purpose

`nag_iamax_val (f16dqc)` computes, with respect to absolute value, the largest component of an integer vector, along with the index of that component.

2 Specification

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

3 Description

`nag_iamax_val (f16dqc)` computes, with respect to absolute value, the largest component, $i$, of an $n$-element integer vector $x$, and determines the smallest index, $k$, such that

$$ i = |x_k| = \max_j |x_j|.$$

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 Integer

   *Input*

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

   **On entry:** the $n$-element vector $x$.

   If $\text{incx} > 0$, $x_i$ must be stored in $x[(i - 1) \times \text{incx}]$, for $i = 1, 2, \ldots, n$.

   If $\text{incx} < 0$, $x_i$ must be stored in $x[(n - i) \times \text{incx}]$, for $i = 1, 2, \ldots, n$.

   Intermediate elements of $x$ are not referenced. If $n = 0$, $x$ is not referenced and may be NULL.

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 $-1$. 
f16dqc

5: \[i\] – Integer * 

*Output*

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

6: \[\text{fail} \] – NagError * 

*Input/Output*

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 \(\langle\text{value}\rangle\) had an illegal value.

**NE_INT**

On entry, \(\text{incx} = \langle\text{value}\rangle\).

Constraint: \(\text{incx} \neq 0\).

On entry, \(n = \langle\text{value}\rangle\).

Constraint: \(n \geq 0\).

**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

\[x = (1, 10, 11, -2, 9)^T.\]
10.1 Program Text

/* nag_iamax_val (f16dqc) Example Program.
* Copyright 2014 Numerical Algorithms Group.
* Mark 9, 2009.
*/

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

int main(void)
{
  /* Scalars */
  Integer exit_status, i, incx, j, k, n, xlen;
  /* Arrays */
  Integer *x = 0;
  /* Nag Types */
  NagError fail;

  exit_status = 0;
  INIT_FAIL(fail);

  printf("nag_iamax_val (f16dqc) 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, Integer)))
      {
        printf("Allocation failure\n");
        exit_status = -1;
        goto END;
      }
  }
  else
  {
    printf("Invalid n\n");
    exit_status = 1;
    goto END;
  }

  /* Input vector x */
  for (j = 0; j < xlen; j = j + incx)
    #ifdef _WIN32
      scanf_s("%"NAG_IFMT"", &x[j]);
    #else
      scanf("%"NAG_IFMT"", &x[j]);
    #endif
    #ifdef _WIN32
      scanf_s("%*[\n ] ");
    #else
      scanf("%*[\n ] ");
    #endif
/* nag_iamax_val (f16dqc).
 * Get absolutely maximum value (i) and location of that value (k)
 * of Integer vector */
 nag_iamax_val(n, x, incx, &k, &i, &fail);

if (fail.code != NE_NOERROR)
{
    printf("Error from nag_iamax_val (f16dqc).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
}

/* Print the absolutely maximum value */
printf("Absolutely maximum element of x is %12"NAG_IFMT"\n", i);
/* Print its location */
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_iamax_val (f16dqc) Example Program Data
5  1
1 10 11 -2 9 : n and incx

10.3 Program Results

nag_iamax_val (f16dqc) Example Program Results

Absolutely maximum element of x is 11
Index of absolutely maximum element of x is 2