NAG Library Function Document

nag_5pt_summary_stats (g01alc)

1 Purpose

nag_5pt_summary_stats (g01alc) calculates a five-point summary for a single sample.

2 Specification

```
#include <nag.h>
#include <nagg01.h>
void nag_5pt_summary_stats (Integer n, const double x[], double res[],
     NagError *fail)
```

3 Description

nag_5pt_summary_stats (g01alc) calculates the minimum, lower hinge, median, upper hinge and the maximum of a sample of n observations.

The data consist of a single sample of n observations denoted by x_i and let z_i, for $i = 1, 2, \ldots, n$, represent the sample observations sorted into ascending order.

Let $m = \dfrac{n}{2}$ if n is even and $\dfrac{(n+1)}{2}$ if n is odd,

and $k = \dfrac{m}{2}$ if m is even and $\dfrac{(m+1)}{2}$ if m is odd.

Then we have

$$
\begin{array}{lll}
\text{Minimum} & = z_1, & \\
\text{Maximum} & = z_n, & \\
\text{Median} & = z_m & \text{if } n \text{ is odd,} \\
 & = \dfrac{z_m + z_{m+1}}{2} & \text{if } n \text{ is even,} \\
\text{Lower hinge} & = z_k & \text{if } m \text{ is odd,} \\
 & = \dfrac{z_k + z_{k+1}}{2} & \text{if } m \text{ is even,} \\
\text{Upper hinge} & = z_{n-k+1} & \text{if } m \text{ is odd,} \\
 & = \dfrac{z_{n-k} + z_{n-k+1}}{2} & \text{if } m \text{ is even.}
\end{array}
$$

4 References

Erickson B H and Nosanchuk T A (1985) *Understanding Data* Open University Press, Milton Keynes

Tukey J W (1977) *Exploratory Data Analysis* Addison–Wesley

5 Arguments

1: **n** – Integer *Input*

On entry: n, number of observations in the sample.

Constraint: **n** ≥ 5.

2: **x**[**n**] – const double *Input*

On entry: the sample observations, $x_1, x_2, \ldots, x_n$.

3: **res**[**5**] – double *Output*

On exit: **res** contains the five-point summary.

res[0]
The minimum.

res[1]
The lower hinge.

res[2]
The median.

res[3]
The upper hinge.

res[4]
The maximum.

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

The NAG error argument (see Section 2.7 in How to Use the NAG Library and its Documentation).

6 Error Indicators and Warnings

NE_ALLOC_FAIL

Dynamic memory allocation failed.
See Section 2.3.1.2 in How to Use the NAG Library and its Documentation for further information.

NE_BAD_PARAM

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

NE_INT_ARG_LT

On entry, **n** = ⟨*value*⟩.
Constraint: **n** $\geq$ 5.

NE_INTERNAL_ERROR

An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please contact NAG for assistance.

An unexpected error has been triggered by this function. Please contact NAG.
See Section 2.7.6 in How to Use the NAG Library and its Documentation for further information.

NE_NO_LICENCE

Your licence key may have expired or may not have been installed correctly.
See Section 2.7.5 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

The computations are stable.

8 Parallelism and Performance

nag_5pt_summary_stats (g01alc) is not threaded in any implementation.

9 Further Comments

The time taken by nag_5pt_summary_stats (g01alc) is proportional to n.

10 Example

This example calculates a five-point summary for a sample of 12 observations.

10.1 Program Text

```
/* nag_5pt_summary_stats (g01alc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 *
 */

#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nagg01.h>

int main(void)
{

  Integer exit_status = 0, i, n;
  NagError fail;
  double *res = 0, *x = 0;

  INIT_FAIL(fail);

  printf("nag_5pt_summary_stats (g01alc) Example Program Results\n");
  /* Skip heading in data file */
#ifdef _WIN32
  scanf_s("%*[^\n] ");
#else
  scanf("%*[^\n] ");
#endif
#ifdef _WIN32
  scanf_s("%" NAG_IFMT " ", &n);
#else
  scanf("%" NAG_IFMT " ", &n);
#endif
  if (n >= 5) {
    if (!(x = NAG_ALLOC(n, double)) || !(res = NAG_ALLOC(5, double)))
    {
      printf("Allocation failure\n");
      exit_status = -1;
      goto END;
    }
  }
  else {
    printf("Invalid n.\n");
    exit_status = 1;
    return exit_status;
  }
  for (i = 1; i <= n; ++i)
#ifdef _WIN32
    scanf_s("%lf ", &x[i - 1]);
#else
```

```
    scanf("%lf ", &x[i - 1]);
#endif
  /* nag_5pt_summary_stats (g01alc).
   * Five-point summary (median, hinges and extremes)
   */
  nag_5pt_summary_stats(n, x, res, &fail);
  if (fail.code != NE_NOERROR) {
    printf("Error from nag_5pt_summary_stats (g01alc).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
  }

  printf("\n");
  printf(" Maximum     %16.4f\n", res[4]);
  printf(" Upper Hinge %16.4f\n", res[3]);
  printf(" Median      %16.4f\n", res[2]);
  printf(" Lower Hinge %16.4f\n", res[1]);
  printf(" Minimum     %16.4f\n", res[0]);
END:
  NAG_FREE(x);
  NAG_FREE(res);

  return exit_status;
}
```

10.2 Program Data

```
nag_5pt_summary_stats (g01alc) Example Program Data
12
12.0  9.0  2.0  5.0  6.0  8.0  2.0  7.0  3.0  1.0  11.0  10.0
```

10.3 Program Results

```
nag_5pt_summary_stats (g01alc) Example Program Results

 Maximum             12.0000
 Upper Hinge          9.5000
 Median               6.5000
 Lower Hinge          2.5000
 Minimum              1.0000
```