

## 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  $i = 1, 2, \dots, n$ , represent the sample observations sorted into ascending order.

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

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

Then we have

Minimum	= $z_1$ ,	
Maximum	= $z_n$ ,	
Median	= $z_m$	if $n$ is odd,
	= $\frac{z_m + z_{m+1}}{2}$	if $n$ is even,
Lower hinge	= $z_k$	if $m$ is odd,
	= $\frac{z_k + z_{k+1}}{2}$	if $m$ is even,
Upper hinge	= $z_{n-k+1}$	if $m$ is odd,
	= $\frac{z_{n-k} + z_{n-k+1}}{2}$	if $m$ is even.

#### 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:*  $n \geq 5$ .

- 2: **x[n]** – const double *Input*  
*On entry:* the sample observations,  $x_1, x_2, \dots, x_n$ .
- 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 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  $\langle value \rangle$  had an illegal value.

### NE\_INT\_ARG\_LT

On entry, **n** =  $\langle value \rangle$ .  
 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 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 computations are stable.

## 8 Parallelism and Performance

Not applicable.

## 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.
 *
 * Copyright 2014 Numerical Algorithms Group.
 *
 * Mark 4, 1996.
 * Mark 8 revised, 2004.
 */

#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

---