

## NAG C Library Function Document

### nag\_summary\_stats\_freq (g01adc)

#### 1 Purpose

nag\_summary\_stats\_freq (g01adc) calculates the mean, standard deviation and coefficients of skewness and kurtosis for data grouped in a frequency distribution.

#### 2 Specification

```
void nag_summary_stats_freq (Integer k, const double x[], const Integer ifreq[],
    double *xmean, double *xsd, double *xskew, double *xkurt, Integer *n,
    NagError *fail)
```

#### 3 Description

The input data consist of a univariate frequency distribution, denoted by  $f_i$ , for  $i = 1, 2, \dots, k-1$ , and the boundary values of the classes  $x_i$ , for  $i = 1, 2, \dots, k$ . Thus the frequency associated with the interval  $(x_i, x_{i+1})$  is  $f_i$ , and nag\_summary\_stats\_freq (g01adc) assumes that all the values in this interval are concentrated at the point

$$y_i = (x_{i+1} + x_i)/2, \quad i = 1, 2, \dots, k-1.$$

The following quantities are calculated:

(a) total frequency,

$$n = \sum_{i=1}^{k-1} f_i.$$

(b) mean,

$$\bar{y} = \frac{\sum_{i=1}^{k-1} f_i y_i}{n}.$$

(c) standard deviation,

$$s_2 = \sqrt{\frac{\sum_{i=1}^{k-1} f_i (y_i - \bar{y})^2}{(n-1)}}, \quad n \geq 2.$$

(d) coefficient of skewness,

$$s_3 = \frac{\sum_{i=1}^{k-1} f_i (y_i - \bar{y})^3}{(n-1) \times s_2^3}, \quad n \geq 2.$$

(e) coefficient of kurtosis,

$$s_4 = \frac{\sum_{i=1}^{k-1} f_i (y_i - \bar{y})^4}{(n-1) \times s_2^4} - 3, \quad n \geq 2.$$

The function has been developed primarily for groupings of a continuous variable. If, however, the function is to be used on the frequency distribution of a discrete variable, taking the values  $y_1, \dots, y_{k-1}$ , then the boundary values for the classes may be defined as follows:

(i) for  $k > 2$ ,

$$\begin{aligned} x_1 &= (3y_1 - y_2)/2 \\ x_j &= (y_{j-1} + y_j)/2, \quad j = 2, \dots, k-1 \\ x_k &= (3y_{k-1} - y_{k-2})/2 \end{aligned}$$

(ii) for  $k = 2$ ,

$$x_1 = y_1 - a \quad \text{and} \quad x_2 = y_1 + a \quad \text{for any } a > 0.$$

## 4 References

None.

## 5 Parameters

- 1: **k** – Integer *Input*  
*On entry:* the number of class boundaries, which is one more than the number of classes of the frequency distribution,  $k$ .  
*Constraint:*  $k > 1$ .
- 2: **x[k]** – const double *Input*  
*On entry:* the elements of **x** must contain the boundary values of the classes in ascending order, so that class  $i$  is bounded by the values in **x**[ $i - 1$ ] and **x**[ $i$ ], for  $i = 1, 2, \dots, k - 1$ .  
*Constraint:* **x**[ $i$ ] < **x**[ $i + 1$ ] for  $i = 0, 1, \dots, k - 2$ .
- 3: **ifreq[k]** – const Integer *Input*  
*On entry:* the  $i$ th element of **ifreq** must contain the frequency associated with the  $i$ th class, for  $i = 1, 2, \dots, k - 1$ . **ifreq**[ $k - 1$ ] is not used by the function.  
*Constraint:* **ifreq**[ $i - 1$ ]  $\geq 0$ , for  $i = 1, 2, \dots, k - 1$  and  $\sum_{i=1}^{k-1} \mathbf{ifreq}[i - 1] > 0$ .
- 4: **xmean** – double \* *Output*  
*On exit:* the mean value,  $\bar{y}$ .
- 5: **xsd** – double \* *Output*  
*On exit:* the standard deviation,  $s_2$ .
- 6: **xskew** – double \* *Output*  
*On exit:* the coefficient of skewness,  $s_3$ .
- 7: **xkurt** – double \* *Output*  
*On exit:* the coefficient of kurtosis,  $s_4$ .
- 8: **n** – Integer \* *Output*  
*On exit:* the total frequency,  $n$ .
- 9: **fail** – NagError \* *Input/Output*  
The NAG error parameter (see the Essential Introduction).

## 6 Error Indicators and Warnings

### NE\_INT

On entry, **k** =  $\langle value \rangle$ .  
Constraint: **k** > 1.

### NE\_FREQ\_CONS

Either **ifreq**[ $i$ ] < 0 for some  $i$ , or the sum of frequencies is zero.

### NE\_FREQ\_SUM

The total frequency is less than 2.

**NE\_NOT\_INCREASING**

On entry,  $x[i - 2] > x[i - 1]$ :  $i = \langle value \rangle$ ,  $x[i - 2] = \langle value \rangle$ ,  $x[i - 1] = \langle value \rangle$ .

**NE\_BAD\_PARAM**

On entry, parameter  $\langle value \rangle$  had an illegal value.

**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 consult NAG for assistance.

**7 Accuracy**

The method used is believed to be stable.

**8 Further Comments**

The time taken by `nag_summary_stats_freq (g01adc)` increases linearly with  $k$ .

**9 Example**

In the example program, `nprob` determines the number of sets of data to be analysed. For each analysis, the boundary values of the classes and the frequencies are read. After `nag_summary_stats_freq (g01adc)` has been successfully called, the input data and calculated quantities are printed. In the example, there is one set of data, with 14 classes.

**9.1 Program Text**

```

/* nag_summary_stats_freq (g01adc) Example Program.
 *
 * Copyright 2001 Numerical Algorithms Group.
 *
 * Mark 7, 2001.
 */

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

int main(void)
{
    /* Scalars */
    double xsd, xskew, xkurt, xmean;
    Integer exit_status, i, j, k, kmin1, n, nprob;

    NagError fail;

    /* Arrays */
    double *x=0;
    Integer *ifreq=0;

    INIT_FAIL(fail);
    Vprintf("g01adc Example Program Results\n");

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

    Vscanf("%ld%*[^\\n] ", &nprob);
    for (j = 1; j <= nprob; ++j)
    {
        Vscanf("%ld%*[^\\n] ", &kmin1);
        k = kmin1 + 1;
    }
}

```

```

/* Allocate memory */
if ( !(x = NAG_ALLOC(k, double)) ||
      !(ifreq = NAG_ALLOC(k, Integer)) )
{
  Vprintf("Allocation failure\n");
  exit_status = -1;
  goto END;
}

for (i = 1; i <= kmin1; ++i)
  Vscanf("%lf%ld", &x[i - 1], &ifreq[i - 1]);
Vscanf("%lf*[\n] ", &x[k - 1]);

Vprintf("\nProblem %4ld\n", j);
Vprintf("Number of classes %4ld\n", kmin1);

g01adc(k, x, ifreq, &xmean, &xsd, &xskew, &xkurt, &n, &fail);

if (fail.code == NE_NOERROR)
{
  Vprintf("Successful call of g01adc\n\n");
  Vprintf("      Class      Frequency\n\n");
  for (i = 1; i <= kmin1; ++i)
    Vprintf("%10.2f%10.2f%12ld\n", x[i - 1], x[i], ifreq[i - 1]);

  Vprintf("\n Mean %16.4f\n", xmean);
  Vprintf(" Std devn%13.4f\n", xsd);
  Vprintf(" Skewness%13.4f\n", xskew);
  Vprintf(" Kurtosis%13.4f\n", xkurt);
  Vprintf(" Number of cases%8ld\n", n);
}
else
{
  Vprintf("Error from g01adc.\n%s\n", fail.message);
  exit_status = 1;
}
if (x) NAG_FREE(x);
if (ifreq) NAG_FREE(ifreq);
}
END:
return exit_status;
}

```

## 9.2 Program Data

g01adc Example Program Data

```

1
14
9.3      3      12      19      14      52      16      96
18      121     20      115     22      86      24      70
26      49      28      31      30      16      32      6
34      8       36      7       39.7

```

## 9.3 Program Results

g01adc Example Program Results

```

Problem      1
Number of classes  14
Successful call of g01adc

```

	Class	Frequency
	9.30	12.00
	12.00	14.00
	14.00	16.00
	16.00	18.00
	18.00	20.00
		3
		19
		52
		96
		121

20.00	22.00	115
22.00	24.00	86
24.00	26.00	70
26.00	28.00	49
28.00	30.00	31
30.00	32.00	16
32.00	34.00	6
34.00	36.00	8
36.00	39.70	7
Mean	21.4932	
Std devn	4.9325	
Skewness	0.7072	
Kurtosis	0.5738	
Number of cases	679	

---