NAG 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

```c
#include <nag.h>
#include <nagg01.h>

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 \( i = 1, 2, \ldots, k - 1 \), and the boundary values of the classes \( x_i \), for \( i = 1, 2, \ldots, k \). 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

\[
y_i = (x_{i+1} + x_i)/2, \quad i = 1, 2, \ldots, k - 1.
\]

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,

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

(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 \( y_1, \ldots, y_{k-1} \), then the boundary values for the classes may be defined as follows:

(i) for \( k > 2 \),

\[
x_1 = \frac{3y_1 - y_2}{2}
\]
\[
x_j = \frac{y_{j-1} + y_j}{2}, \quad j = 2, \ldots, k - 1
\]
\[
x_k = \frac{3y_{k-1} - y_k}{2}
\]

(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 Arguments

1: \textbf{k} – Integer \hspace{1cm} \textit{Input}

\textit{On entry:} \( k \), the number of class boundaries, which is one more than the number of classes of the frequency distribution.

\textit{Constraint:} \( k > 1. \)

2: \textbf{x[k]} – const double \hspace{1cm} \textit{Input}

\textit{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 \( i = 1, 2, \ldots, k - 1. \)

\textit{Constraint:} \( x[i] < x[i+1] \), for \( i = 0, 1, \ldots, k - 2. \)

3: \textbf{ifreq[k]} – const Integer \hspace{1cm} \textit{Input}

\textit{On entry:} the \( i \)th element of \textbf{ifreq} must contain the frequency associated with the \( i \)th class, for \( i = 1, 2, \ldots, k - 1. \) \textbf{ifreq[k-1]} is not used by the function.

\textit{Constraints:}

\[
\sum_{i=1}^{k-1} \text{ifreq}[i-1] \geq 0,
\]
\[
\text{ifreq}[i-1] > 0, \text{ for } i = 1, 2, \ldots, k - 1;
\]

4: \textbf{xmean} – double * \hspace{1cm} \textit{Output}

\textit{On exit:} the mean value, \( \bar{y} \).

5: \textbf{xsd} – double * \hspace{1cm} \textit{Output}

\textit{On exit:} the standard deviation, \( s_2 \).

6: \textbf{xskew} – double * \hspace{1cm} \textit{Output}

\textit{On exit:} the coefficient of skewness, \( s_3 \).

7: \textbf{xkurt} – double * \hspace{1cm} \textit{Output}

\textit{On exit:} the coefficient of kurtosis, \( s_4 \).

8: \textbf{n} – Integer * \hspace{1cm} \textit{Output}

\textit{On exit:} the total frequency, \( n \).
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_FREQ_CONS**
Either \(\text{ifreq}[i] > 0\) for some \(i\), or the sum of frequencies is zero.

**NE_FREQ_SUM**
The total frequency is less than 2.

**NE_INT**
On entry, \(k = \langle\text{value}\rangle\).
Constraint: \(k > 1\).

**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.

**NE_NOT_INCREASING**
On entry, \(I = \langle\text{value}\rangle\), \(x[I-2] = \langle\text{value}\rangle\) and \(x[I-1] = \langle\text{value}\rangle\).
Constraint: \(x[I-2] \leq x[I-1]\).

7 Accuracy
The method used is believed to be stable.

8 Parallelism and Performance
Not applicable.

9 Further Comments
The time taken by nag_summary_stats_freq (g01adc) increases linearly with \(k\).
10 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.

10.1 Program Text

```c
/* nag_summary_stats_freq (g01adc) Example Program. */
/* Copyright 2014 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 = 0, i, j, k, kmin1, n, nprob;

    NagError fail;

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

    INIT_FAIL(fail);

    printf("nag_summary_stats_freq (g01adc) 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] ", &nprob);
    #else
        scanf("%"NAG_IFMT"%*[\n] ", &nprob);
    #endif
    for (j = 1; j <= nprob; ++j)
    {
        #ifdef _WIN32
            scanf_s("%"NAG_IFMT"%*[\n] ", &kmin1);
        #else
            scanf("%"NAG_IFMT"%*[\n] ", &kmin1);
        #endif
        k = kmin1 + 1;

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

        for (i = 1; i <= kmin1; ++i)
```

---

**Note:** The snippet above is incomplete and requires continuation to properly end the program. The source code snippet was intended to be part of a larger program, and was cut off at the end. The full code would include the necessary code to process the input data, call `nag_summary_stats_freq`, and print the results.
ifdef _WIN32
    scanf_s("%lf"NAG_IFMT", &x[i - 1], &ifreq[i - 1]);
else
    scanf("%lf"NAG_IFMT", &x[i - 1], &ifreq[i - 1]);
endif
ifdef _WIN32
    scanf_s("%lf%*\[\n"] , &x[k - 1]);
else
    scanf("%lf%\n", &x[k - 1]);
endif
printf("Problem %4"NAG_IFMT"
", j);
printf("Number of classes %4"NAG_IFMT"
", k);
/* nag_summary_stats_freq (g01adc).
 * Mean, variance, skewness, kurtosis, etc., one variable,
 * from frequency table
 */
nag_summary_stats_freq(k, x, ifreq, &xmean, &xsd, &xskew, &xkurt, &n, &fail);
if (fail.code == NE_NOERROR)
{
    printf("Successful call of ", n);  
    printf("nag_summary_stats_freq (g01adc)\n\n");
    printf(" Class Frequency\n\n");
    for (i = 1; i <= kmin1; ++i)  
        printf("%10.2f%10.2f%12"NAG_IFMT"
", x[i-1], x[i], ifreq[i-1]);
    printf("\n Mean %16.4f
", xmean);
    printf(" Std devn%13.4f
", xsd);
    printf(" Skewness%13.4f
", xskew);
    printf(" Kurtosis%13.4f
", xkurt);
    printf(" Number of cases%8"NAG_IFMT"
", n);
}
else  
{
    printf("Error from nag_summary_stats_freq (g01adc).\n\n", fail.message);
    exit_status = 1;
}  
NAG_FREE(x);
NAG_FREE(ifreq);
END:
NAG_FREE(x);
NAG_FREE(ifreq);
return exit_status;

10.2 Program Data

nag_summary_stats_freq (g01adc) Example Program Data

<table>
<thead>
<tr>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.3 3 12 19 14 52 16 96</td>
</tr>
<tr>
<td>18 121 20 115 22 86 24 70</td>
</tr>
<tr>
<td>26 49 28 31 30 16 32 6</td>
</tr>
<tr>
<td>34 8 36 7 39.7</td>
</tr>
</tbody>
</table>

10.3 Program Results

nag_summary_stats_freq (g01adc) Example Program Results

Problem 1
Number of classes 14
Successful call of nag_summary_stats_freq (g01adc)

<table>
<thead>
<tr>
<th>Class</th>
<th>Frequency</th>
</tr>
</thead>
</table>

Mark 25
<table>
<thead>
<tr>
<th>Time (hrs)</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.30</td>
<td>12.00</td>
</tr>
<tr>
<td>12.00</td>
<td>14.00</td>
</tr>
<tr>
<td>14.00</td>
<td>16.00</td>
</tr>
<tr>
<td>16.00</td>
<td>18.00</td>
</tr>
<tr>
<td>18.00</td>
<td>20.00</td>
</tr>
<tr>
<td>20.00</td>
<td>22.00</td>
</tr>
<tr>
<td>22.00</td>
<td>24.00</td>
</tr>
<tr>
<td>24.00</td>
<td>26.00</td>
</tr>
<tr>
<td>26.00</td>
<td>28.00</td>
</tr>
<tr>
<td>28.00</td>
<td>30.00</td>
</tr>
<tr>
<td>30.00</td>
<td>32.00</td>
</tr>
<tr>
<td>32.00</td>
<td>34.00</td>
</tr>
<tr>
<td>34.00</td>
<td>36.00</td>
</tr>
<tr>
<td>36.00</td>
<td>39.70</td>
</tr>
</tbody>
</table>

Mean: 21.4932  
Std devn: 4.9325  
Skewness: 0.7072  
Kurtosis: 0.5738  
Number of cases: 679