

NAG Library Function Document

nag_summary_stats_1var (g01aac)

1 Purpose

nag_summary_stats_1var (g01aac) calculates the mean, standard deviation, coefficients of skewness and kurtosis, and the maximum and minimum values for a set of ungrouped data. Weighting may be used.

2 Specification

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

void nag_summary_stats_1var (Integer n, const double x[], const double wt[],
    Integer *nvalid, double *xmean, double *xsd, double *xskew, double *xkurt,
    double *xmin, double *xmax, double *wsum, NagError *fail)
```

3 Description

The data consist of a single sample of n observations, denoted by x_i , with corresponding weights, w_i , for $i = 1, 2, \dots, n$.

If no specific weighting is required, then each w_i is set to 1.

The quantities computed are:

- (a) The sum of the weights

$$W = \sum_{i=1}^n w_i.$$

- (b) Mean

$$\bar{x} = \frac{\sum_{i=1}^n w_i x_i}{W}.$$

- (c) Standard deviation

$$s_2 = \sqrt{\frac{\sum_{i=1}^n w_i (x_i - \bar{x})^2}{d}}, \quad \text{where} \quad d = W - \frac{\sum_{i=1}^n w_i^2}{W}.$$

- (d) Coefficient of skewness

$$s_3 = \frac{\sum_{i=1}^n w_i (x_i - \bar{x})^3}{d \times s_2^3}.$$

- (e) Coefficient of kurtosis

$$s_4 = \frac{\sum_{i=1}^n w_i (x_i - \bar{x})^4}{d \times s_2^4} - 3.$$

- (f) Maximum and minimum elements of the sample.

- (g) The number of observations for which $w_i > 0$, i.e., the number of **valid** observations. Suppose m observations are valid, then the quantities in (c), (d) and (e) will be computed if $m \geq 2$, and will be based on $m - 1$ degrees of freedom. The other quantities are evaluated provided $m \geq 1$.

4 References

None.

5 Arguments

- 1: **n** – Integer *Input*
On entry: n , the number of observations.
Constraint: $n \geq 1$.
- 2: **x[n]** – const double *Input*
On entry: the sample observations, x_i , for $i = 1, 2, \dots, n$.
- 3: **wt[n]** – const double *Input*
On entry: if weights are being supplied then the elements of **wt** must contain the weights associated with the observations, w_i , for $i = 1, 2, \dots, n$.
 If weights are not supplied then **wt** must be set to the null pointer, i.e., (double *)0.
- 4: **nvalid** – Integer * *Output*
On exit: is used to indicate the number of valid observations, m ; see Section 3 (g).
- 5: **xmean** – double * *Output*
On exit: the mean, \bar{x} .
- 6: **xsd** – double * *Output*
On exit: the standard deviation, s_2 .
- 7: **xskew** – double * *Output*
On exit: the coefficient of skewness, s_3 .
- 8: **xkurt** – double * *Output*
On exit: the coefficient of kurtosis, s_4 .
- 9: **xmin** – double * *Output*
On exit: the smallest value in the sample.
- 10: **xmax** – double * *Output*
On exit: the largest value in the sample.
- 11: **wsum** – double * *Output*
On exit: the sum of the weights in the array **wt**, that is $\sum_{i=1}^n w_i$. This will be n if weighted estimates are not used.
- 12: **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.

NE_BAD_PARAM

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

NE_CASES_ONE

The number of valid cases is one. In this case, standard deviation and coefficients of skewness and of kurtosis cannot be calculated.

NE_CASES_ZERO

The number of valid cases is zero.

NE_INT_ARG_LE

On entry, $n = \langle value \rangle$.
Constraint: $n \geq 1$.

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.

NE_REAL_ARG_LT

On entry, $wt[\langle value \rangle] = \langle value \rangle$.
Constraint: $wt[\langle value \rangle] \geq 0.0$.

7 Accuracy

The method used is believed to be stable.

8 Further Comments

The time taken by `nag_summary_stats_1var` (g01aac) is approximately proportional to n .

9 Example

This example summarises a number of datasets. For each dataset the observations and, optionally, weights are read and printed. `nag_summary_stats_1var` (g01aac) is then called and the calculated quantities are printed.

9.1 Program Text

```
/* nag_summary_stats_1var (g01aac) Example Program.
 *
 * Copyright 1990 Numerical Algorithms Group.
 *
 * Mark 1, 1990.
 *
 * Mark 5 revised, 1998.
 * Mark 8 revised, 2004.
 *
 */

#include <nag.h>
#include <nagx04.h>
#include <stdio.h>
```

```

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

int main(int argc, char *argv[])
{
    FILE      *fpin, *fpout;
    Integer   exit_status = 0, i, j, n, nprob, nvalid, weight;
    NagError  fail;
    double    wsum, *wt = 0, *x = 0, xkurt, xmax, xmean, xmin, xsd, xskew;

    INIT_FAIL(fail);

    /* Check for command-line IO options */
    fpin = nag_example_file_io(argc, argv, "-data", NULL);
    fpout = nag_example_file_io(argc, argv, "-results", NULL);
    /* Skip heading in data file */
    fscanf(fpin, "%*[\n]");
    fprintf(fpout, "nag_summary_stats_lvar (g01aac) Example Program Results\n");
    fscanf(fpin, "%ld", &nprob);
    for (j = 1; j <= nprob; j++)
    {
        fscanf(fpin, "%ld %ld", &n, &weight);
        fprintf(fpout, "Problem %5ld\n", j);
        fprintf(fpout, "Number of cases %ld\n", n);
        if (n >= 1)
        {
            if (!(wt = NAG_ALLOC(n, double)) ||
                !(x = NAG_ALLOC(n, double)))
            {
                fprintf(fpout, "Allocation failure\n");
                exit_status = -1;
                goto END;
            }
        }
        else
        {
            fprintf(fpout, "Invalid n.\n");
            exit_status = 1;
            return exit_status;
        }
        for (i = 0; i < n; i++)
            fscanf(fpin, "%lf", &x[i]);
        fprintf(fpout, "Data as input -\n");
        for (i = 0; i < n; i++)
            fprintf(fpout, "%12.1f%c", x[i], (i%5 == 4 || i == n-1)?'\n':' ');
        if (weight)
        {
            fprintf(fpout, "Weights as input -\n");
            for (i = 0; i < n; i++)
                fscanf(fpin, "%lf", &wt[i]);
            for (i = 0; i < n; i++)
                fprintf(fpout, "%12.1f%c", wt[i], (i%5 == 4 || i == n-1)?'\n':' ');
            /* nag_summary_stats_lvar (g01aac).
             * Mean, variance, skewness, kurtosis, etc., one variable,
             * from raw data
             */
            nag_summary_stats_lvar(n, x, wt, &nvalid, &xmean, &xsd, &xskew,
                                  &xkurt, &xmin, &xmax, &wsum, &fail);
        }
        else
            /* nag_summary_stats_lvar (g01aac), see above. */
            nag_summary_stats_lvar(n, x, (double *) 0, &nvalid, &xmean, &xsd,
                                  &xskew, &xkurt, &xmin, &xmax, &wsum, &fail);

        if (fail.code == NE_NOERROR)
        {
            fprintf(fpout, "\n");
            fprintf(fpout, "Successful call of "
                    "nag_summary_stats_lvar (g01aac)\n");
            fprintf(fpout, "No. of valid cases %5ld\n", nvalid);
            fprintf(fpout, "Mean %13.1f\n", xmean);
        }
    }
}

```

```

        fprintf(fpout, "Std devn      %13.1f\n", xsd);
        fprintf(fpout, "Skewness    %13.1f\n", xskew);
        fprintf(fpout, "Kurtosis    %13.1f\n", xkurt);
        fprintf(fpout, "Minimum     %13.1f\n", xmin);
        fprintf(fpout, "Maximum     %13.1f\n", xmax);
        fprintf(fpout, "Sum of weights %13.1f\n", wsum);
    }
else
    {
        fprintf(fpout, "Unsuccessful call of "
                "nag_summary_stats_lvar (g01aac)\n");
        fprintf(fpout, "%s \n", fail.message);
        if (fail.code == NE_CASES_ONE)
            {
                fprintf(fpout, "No. of valid cases %5ld\n", nvalid);
                fprintf(fpout, "Mean          %13.1f\n", xmean);
                fprintf(fpout, "Minimum       %13.1f\n", xmin);
                fprintf(fpout, "Maximum       %13.1f\n", xmax);
                fprintf(fpout, "Sum of weights %13.1f\n", wsum);
                fprintf(fpout, "Std devn and coeffts of skewness\n");
                fprintf(fpout, "and kurtosis not defined\n");
                exit_status = 2;
            }
        else
            {
                exit_status = 1;
                goto END;
            }
    }

if (wt)
    {
        NAG_FREE(wt);
        wt = 0;
    }
if (x)
    {
        NAG_FREE(x);
        x = 0;
    }
}
END:
if (fpin != stdin) fclose(fpin);
if (fpout != stdout) fclose(fpout);
if (wt) NAG_FREE(wt);
if (x) NAG_FREE(x);
return exit_status;
}

```

9.2 Program Data

```

nag_summary_stats_lvar (g01aac) Example Program Data
1
24 0
193.0 215.0 112.0 161.0 92.0 140.0 38.0 33.0 279.0 249.0
473.0 339.0 60.0 130.0 20.0 50.0 257.0 284.0 447.0 52.0
67.0 61.0 150.0 2200.0

```

9.3 Program Results

```

nag_summary_stats_lvar (g01aac) Example Program Results
Problem 1
Number of cases 24
Data as input -
    193.0      215.0      112.0      161.0      92.0
    140.0       38.0       33.0      279.0     249.0
    473.0     339.0       60.0     130.0      20.0
    50.0      257.0     284.0     447.0     52.0
    67.0       61.0      150.0     2200.0

```

```
Successful call of nag_summary_stats_lvar (g01aac)
No. of valid cases      24
Mean                    254.2
Std devn                433.5
Skewness                 3.9
Kurtosis                14.7
Minimum                 20.0
Maximum                 2200.0
Sum of weights          24.0
```
