NAG Library Function Document

nag_summary_stats_onevar (g01atc)

1 Purpose

nag_summary_stats_onevar (g01atc) calculates the mean, standard deviation, coefficients of skewness and kurtosis, and the maximum and minimum values for a set of (optionally weighted) data. The input data can be split into arbitrary sized blocks, allowing large datasets to be summarised.

2 Specification

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

```
void nag_summary_stats_onevar (Integer nb, const double x[],
     const double wt[], Integer *pn, double *xmean, double *xsd,
     double *xskew, double *xkurt, double *xmin, double *xmax,
     double rcomm[], NagError *fail)
```

3 Description

Given a sample of n observations, denoted by $x = \{x_i : i = 1, 2, \ldots, n\}$ and a set of non-negative weights, $w = \{w_i : i = 1, 2, \ldots, n\}$, nag_summary_stats_onevar (g01atc) calculates a number of quantities:

(a) Mean

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

(b) Standard deviation

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

(c) Coefficient of skewness

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

(d) Coefficient of kurtosis

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

(e) Maximum and minimum elements, with $w_i \neq 0$.

These quantities are calculated using the one pass algorithm of West (1979).

For large datasets, or where all the data is not available at the same time, x and w can be split into arbitrary sized blocks and nag_summary_stats_onevar (g01atc) called multiple times.

4 References

West D H D (1979) Updating mean and variance estimates: An improved method *Comm. ACM* **22** 532–555

5 Arguments

1: **nb** – Integer *Input*

On entry: b, the number of observations in the current block of data. The size of the block of data supplied in **x** and **wt** can vary; therefore **nb** can change between calls to nag_summary_stats_onevar (g01atc).

Constraint: **nb** ≥ 0.

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

On entry: the current block of observations, corresponding to x_i, for $i = k+1, \ldots, k+b$, where k is the number of observations processed so far and b is the size of the current block of data.

3: **wt**[**nb**] – const double *Input*

On entry: if **wt** is not **NULL**, **wt** must contain the user-supplied weights corresponding to the block of data supplied in **x**, that is w_i, for $i = k+1, \ldots, k+b$.

If **wt** is **NULL**, $w_i = 1$ for all i.

Constraint: **wt**$[i-1] \geq 0$, for $i = 1, 2, \ldots,$ **nb**.

4: **pn** – Integer * *Input/Output*

On entry: the number of valid observations processed so far, that is the number of observations with $w_i > 0$, for $i = 1, 2, \ldots, k$. On the first call to nag_summary_stats_onevar (g01atc), or when starting to summarise a new dataset, **pn** must be set to 0.

If **pn** $\neq 0$, it must be the same value as returned by the last call to nag_summary_stats_onevar (g01atc).

On exit: the updated number of valid observations processed, that is the number of observations with $w_i > 0$, for $i = 1, 2, \ldots, k+b$.

Constraints:

pn ≥ 0;
if **rcomm** is **NULL**, **pn** $= 0$.

5: **xmean** – double * *Output*

On exit: $\bar{x}$, the mean of the first $k+b$ observations.

6: **xsd** – double * *Output*

On exit: s_2, the standard deviation of the first $k+b$ observations.

7: **xskew** – double * *Output*

On exit: s_3, the coefficient of skewness for the first $k+b$ observations.

8: **xkurt** – double * *Output*

On exit: s_4, the coefficient of kurtosis for the first $k+b$ observations.

9: **xmin** – double * *Output*

On exit: the smallest value in the first $k+b$ observations.

10: **xmax** – double * *Output*

On exit: the largest value in the first $k + b$ observations.

11: **rcomm**[**20**] – double *Communication Array*

On entry: communication array, used to store information between calls to nag_summary_stats_onevar (g01atc). If **pn** $= 0$, **rcomm** need not be initialized, otherwise it must be unchanged since the last call to this function.

If **rcomm** is **NULL**, **rcomm** is not referenced and all the data must be supplied in one go.

On exit: the updated communication array. The first five elements of **rcomm** hold information that may be of interest with

$$\begin{aligned}
\mathbf{rcomm}[0] &= \sum_{i=1}^{k+b} w_i \\
\mathbf{rcomm}[1] &= \left(\sum_{i=1}^{k+b} w_i\right)^2 - \sum_{i=1}^{k+b} w_i^2 \\
\mathbf{rcomm}[2] &= \sum_{i=1}^{k+b} w_i (x_i - \bar{x})^2 \\
\mathbf{rcomm}[3] &= \sum_{i=1}^{k+b} w_i (x_i - \bar{x})^3 \\
\mathbf{rcomm}[4] &= \sum_{i=1}^{k+b} w_i (x_i - \bar{x})^4
\end{aligned}$$

the remaining elements of **rcomm** are used for workspace and so are undefined.

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

The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_BAD_PARAM

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

NE_CASES_ONE

On exit we were unable to calculate **xsd**, **xskew** or **xkurt**. A value of 0 has been returned.

NE_CASES_ZERO

On entry, the number of valid observations is zero.

NE_ILLEGAL_COMM

rcomm has been corrupted between calls.

NE_INT

On entry, **nb** $=$ ⟨*value*⟩.
Constraint: **nb** ≥ 0.

On entry, **pn** $=$ ⟨*value*⟩.
Constraint: if **rcomm** is **NULL** then **pn** $= 0$.

On entry, **pn** $=$ ⟨*value*⟩.
Constraint: **pn** ≥ 0.

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_NEG_WEIGHT

On entry, **wt**$[\langle value \rangle] = \langle value \rangle$.
Constraint: if **wt** is not **NULL** then **wt**$[i-1] \geq 0$, for $i = 1, 2, \ldots,$ **nb**.

NE_PREV_CALL

On entry, **pn** $= \langle value \rangle$.
On exit from previous call, **pn** $= \langle value \rangle$.
Constraint: if **pn** > 0, **pn** must be unchanged since previous call.

NE_ZERO_VARIANCE

On exit we were unable to calculate **xskew** or **xkurt**. A value of 0 has been returned.

7 Accuracy

Not applicable.

8 Parallelism and Performance

nag_summary_stats_onevar (g01atc) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

Please consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

Both nag_summary_stats_onevar (g01atc) and nag_summary_stats_onevar_combine (g01auc) consolidate results from multiple summaries. Whereas the former can only be used to combine summaries calculated sequentially, the latter combines summaries calculated in an arbitrary order allowing, for example, summaries calculated on different processing units to be combined.

10 Example

This example summarises some simulated data. The data is supplied in three blocks, the first consisting of 21 observations, the second 51 observations and the last 28 observations.

10.1 Program Text

```
/* nag_summary_stats_onevar (g01atc) Example Program.
 *
 * Copyright 2013 Numerical Algorithms Group.
 *
 * Mark 24, 2013.
 */

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

int main(void)
{
  /* Integer scalar and array declarations */
  Integer  b, i, ierr, iwt, nb, pn;
  Integer  exit_status = 0;
```

```
/* NAG structures and types */
NagError fail;

/* Double scalar and array declarations */
double   xkurt, xmax, xmean, xmin, xsd, xskew;
double   rcomm[20];
double   *wt = 0, *x = 0;

/* Initialise the error structure */
INIT_FAIL(fail);

printf("nag_summary_stats_onevar (g01atc) Example Program Results\n\n");

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

/* Initialise the number of valid observations processed so far */
pn = 0;

/* Loop over each block of data */
for (b = 0;; )
  {
    /* Read in the number of observations in this block and a flag indicating
       whether weights have been supplied (iwt = 1) or not (iwt = 0) */
    ierr = scanf("%ld%ld", &nb, &iwt);
    if (ierr == EOF || ierr < 2) break;
    scanf("%*[^\n] ");

    /* Keep a running total of the number of blocks of data */
    b++;

    /* Reallocate X to the required size */
    NAG_FREE(x);
    if (!(x = NAG_ALLOC(nb, double)))
      {
        printf("Allocation failure\n");
        exit_status = -1;
        goto END;
      }

    /* Read in the data for this block */
    if (iwt)
      {
        /* Weights supplied, so reallocate WT to the required size */
        NAG_FREE(wt);
        if (!(wt = NAG_ALLOC(nb, double)))
          {
            printf("Allocation failure\n");
            exit_status = -2;
            goto END;
          }
        for (i = 0; i < nb; i++)
          scanf("%lf%lf", &x[i], &wt[i]);
      }
    else
      {
        /* No weights */
        NAG_FREE(wt);
        wt = 0;

        for (i = 0; i < nb; i++)
          scanf("%lf", &x[i]);
      }
    scanf("%*[^\n] ");

    /* Call nag_summary_stats_onevar (g01atc) to update the summaries for
       this block of data */
    nag_summary_stats_onevar(nb, x, wt, &pn, &xmean, &xsd, &xskew, &xkurt,
                             &xmin, &xmax, rcomm, &fail);
    if (fail.code != NE_NOERROR && fail.code != NE_CASES_ONE &&
          fail.code != NE_ZERO_VARIANCE && fail.code != NE_CASES_ZERO)
        {
          printf("Error from nag_summary_stats_onevar (g01atc).\n%s\n",
                 fail.message);
          exit_status = 1;
          goto END;
        }
    }

  /* Display the results */
  printf(" Data supplied in %ld blocks\n", b);
  if (fail.code == NE_CASES_ZERO)
    printf(" No valid observations supplied. All weights are zero.\n");
  else
    {
      printf(" %ld valid observations\n", pn);
      printf(" Mean          %13.2f\n", xmean);
      if (fail.code == NE_CASES_ONE)
        {
          printf("   Unable to calculate the standard deviation,");
          printf(" skewness or kurtosis\n");
        }
      else
        {
          printf(" Std devn      %13.2f\n", xsd);
          if (fail.code == NE_ZERO_VARIANCE)
            printf("   Unable to calculate the skewness and kurtosis\n");
          else
            {
              printf(" Skewness      %13.2f\n", xskew);
              printf(" Kurtosis      %13.2f\n", xkurt);
            }
        }
      printf(" Minimum       %13.2f\n", xmin);
      printf(" Maximum       %13.2f\n", xmax);
    }

 END:
  NAG_FREE(x);
  NAG_FREE(wt);

  return(exit_status);
}
```

10.2 Program Data

```
nag_summary_stats_onevar (g01atc) Example Program Data
21 1                                 :: nb,iwt (1st block)
 -0.62 4.91     -1.92 0.25
 -1.72 3.90     -6.35 3.75
  2.00 1.17      7.65 3.19
  6.15 2.66      3.81 0.02
  4.87 3.59     -0.51 3.63
  6.88 4.83     -5.85 3.72
 -0.72 1.72      0.66 0.78
  2.23 4.74     -1.61 1.72
 -0.15 3.94     -1.15 1.33
 -8.74 0.51     -3.94 2.40
  3.61 3.90                          :: End of x,wt for 1st block
51 0                                 :: n,iwt (2nd block)
 -0.66 -2.39 -6.25  1.23  2.27 -2.27
 10.12  8.29 -2.99  8.71 -0.74  0.02
  1.22  1.70  4.30  2.99 -0.83 -1.00
  6.57  2.32 -3.47 -1.41 -5.26  0.53
  1.80  4.79 -3.04  1.20 -3.21 -3.75
  0.86  1.27 -5.95 -5.27  1.63  3.59
 -0.01 -1.38 -4.71 -4.82  3.55  0.46
  2.57  1.76 -4.05  1.23 -1.99  3.20
 -0.65  8.42 -6.01                    :: End of x for 2nd block
28 0                                  :: n,iwt (3rd block)
```

```
  1.13 -8.86  5.92 -1.71 -3.99  6.57
 -2.01 -2.29 -1.11  7.14  4.84 -4.44
 -3.32 10.25 -2.11  8.02 -7.31  2.80
 -1.20  1.01  1.37 -2.28  1.28 -3.95
  3.43 -0.61  4.85 -0.11              :: End of x for 3rd block
```

10.3 Program Results

```
nag_summary_stats_onevar (g01atc) Example Program Results

 Data supplied in 3 blocks
 100 valid observations
 Mean                   0.51
 Std devn               4.24
 Skewness               0.18
 Kurtosis              -0.59
 Minimum               -8.86
 Maximum               10.25
```