

## NAG Library Function Document

### nag\_robust\_trimmed\_1var (g07ddc)

#### 1 Purpose

nag\_robust\_trimmed\_1var (g07ddc) calculates the trimmed and Winsorized means of a sample and estimates of the variances of the two means.

#### 2 Specification

```
#include <nag.h>
#include <nagg07.h>

void nag_robust_trimmed_1var (Integer n, const double x[], double alpha,
    double *tmean, double *wmean, double *tvar, double *wvar, Integer *k,
    double sx[], NagError *fail)
```

#### 3 Description

nag\_robust\_trimmed\_1var (g07ddc) calculates the  $\alpha$ -trimmed mean and  $\alpha$ -Winsorized mean for a given  $\alpha$ , as described below.

Let  $x_i$ , for  $i = 1, 2, \dots, n$ , represent the  $n$  sample observations sorted into ascending order. Let  $k = [\alpha n]$  where  $[y]$  represents the integer nearest to  $y$ ; if  $2k = n$  then  $k$  is reduced by 1.

Then the trimmed mean is defined as:

$$\bar{x}_t = \frac{1}{n - 2k} \sum_{i=k+1}^{n-k} x_i,$$

and the Winsorized mean is defined as:

$$\bar{x}_w = \frac{1}{n} \sum_{i=k+1}^{n-k} x_i + (kx_{k+1}) + (kx_{n-k}).$$

nag\_robust\_trimmed\_1var (g07ddc) then calculates the Winsorized variance about the trimmed and Winsorized means respectively and divides by  $n$  to obtain estimates of the variances of the above two means.

Thus we have

$$\text{Estimate of } \text{var}(\bar{x}_t) = \frac{1}{n^2} \sum_{i=k+1}^{n-k} (x_i - \bar{x}_t)^2 + k(x_{k+1} - \bar{x}_t)^2 + k(x_{n-k} - \bar{x}_t)^2$$

and

$$\text{Estimate of } \text{var}(\bar{x}_w) = \frac{1}{n^2} \sum_{i=k+1}^{n-k} (x_i - \bar{x}_w)^2 + k(x_{k+1} - \bar{x}_w)^2 + k(x_{n-k} - \bar{x}_w)^2.$$

#### 4 References

Hampel F R, Ronchetti E M, Rousseeuw P J and Stahel W A (1986) *Robust Statistics. The Approach Based on Influence Functions* Wiley

Huber P J (1981) *Robust Statistics* Wiley

## 5 Arguments

- |     |  |                     |
|-----|--|---------------------|
| 1:  | <b>n</b> – Integer   | <i>Input</i>        |
|     | <i>On entry:</i> the number of observations, $n$ .   |                     |
|     | <i>Constraint:</i> $n \geq 2$ .  |                     |
| 2:  | <b>x[n]</b> – const double   | <i>Input</i>        |
|     | <i>On entry:</i> the sample observations, $x_i$ , for $i = 1, 2, \dots, n$ .                               |                     |
| 3:  | <b>alpha</b> – double  | <i>Input</i>        |
|     | <i>On entry:</i> the proportion of observations to be trimmed at each end of the sorted sample, $\alpha$ . |                     |
|     | <i>Constraint:</i> $0.0 \leq \mathbf{alpha} < 0.5$ .   |                     |
| 4:  | <b>tmean</b> – double *  | <i>Output</i>       |
|     | <i>On exit:</i> the $\alpha$ -trimmed mean, $\bar{x}_t$ .  |                     |
| 5:  | <b>wmean</b> – double *  | <i>Output</i>       |
|     | <i>On exit:</i> the $\alpha$ -Winsorized mean, $\bar{x}_w$ .   |                     |
| 6:  | <b>tvar</b> – double *   | <i>Output</i>       |
|     | <i>On exit:</i> contains an estimate of the variance of the trimmed mean.                                  |                     |
| 7:  | <b>wvar</b> – double *   | <i>Output</i>       |
|     | <i>On exit:</i> contains an estimate of the variance of the Winsorized mean.                               |                     |
| 8:  | <b>k</b> – Integer *   | <i>Output</i>       |
|     | <i>On exit:</i> contains the number of observations trimmed at each end, $k$ .                             |                     |
| 9:  | <b>sx[n]</b> – double  | <i>Output</i>       |
|     | <i>On exit:</i> contains the sample observations sorted into ascending order.                              |                     |
| 10: | <b>fail</b> – NagError *   | <i>Input/Output</i> |
|     | The NAG error argument (see Section 3.6 in the Essential Introduction).                                    |                     |

## 6 Error Indicators and Warnings

### NE\_INT\_ARG\_LT

On entry, **n** =  $\langle value \rangle$ .  
 Constraint:  $n \geq 2$ .

### 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\_GE

On entry, **alpha** must not be greater than or equal to 0.5: **alpha** =  $\langle value \rangle$ .

### NE\_REAL\_ARG\_LT

On entry, **alpha** must not be less than 0.0: **alpha** =  $\langle value \rangle$ .

## 7 Accuracy

The results should be accurate to within a small multiple of *machine precision*.

## 8 Parallelism and Performance

Not applicable.

## 9 Further Comments

The time taken by `nag_robust_trimmed_1var` (g07ddc) is proportional to  $n$ .

## 10 Example

The following program finds the  $\alpha$ -trimmed mean and  $\alpha$ -Winsorized mean for a sample of 16 observations where  $\alpha = 0.15$ . The estimates of the variances of the above two means are also calculated.

### 10.1 Program Text

```

/* nag_robust_trimmed_1var (g07ddc) 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 <nagg07.h>

#define NMAX 1000
int main(void)
{
    /* Local variables */
    Integer exit_status = 0, i, k, n;
    NagError fail;
    double alpha, propn, *sx = 0, tmean, tvar, wmean, wvar, *x = 0;

    INIT_FAIL(fail);

    printf(
        "nag_robust_trimmed_1var (g07ddc) Example Program Results\n\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 >= 2)
    {
        if (!(x = NAG_ALLOC(NMAX, double)) ||
            !(sx = NAG_ALLOC(NMAX, 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
#ifdef _WIN32
    scanf_s("%lf ", &alpha);
#else
    scanf("%lf ", &alpha);
#endif

  /* nag_robust_trimmed_lvar (g07ddc).
   * Trimmed and winsorized mean of a sample with estimates of
   * the variances of the two means
   */
  nag_robust_trimmed_lvar(n, x, alpha, &tmean, &wmean, &tvar, &wvar, &k, sx,
                        &fail);
  if (fail.code != NE_NOERROR)
  {
    printf("Error from nag_robust_trimmed_lvar (g07ddc).\n%s\n",
          fail.message);
    exit_status = 1;
    goto END;
  }

  propn = (double) k / n;
  propn = 100.0 - propn * 200.0;
  printf("Statistics from middle %6.2f%% of data\n\n", propn);
  printf("          Trimmed-mean = %11.4f\n", tmean);
  printf("    Variance of Trimmed-mean = %11.4f\n\n", tvar);
  printf("          Winsorized-mean = %11.4f\n", wmean);
  printf("    Variance of Winsorized-mean = %11.4f\n", wvar);
  END:
  NAG_FREE(x);
  NAG_FREE(sx);
  return exit_status;
}

```

## 10.2 Program Data

nag\_robust\_trimmed\_lvar (g07ddc) Example Program Data

```

16
26.0 12.0 9.0 2.0 5.0 6.0 8.0 14.0 7.0 3.0 1.0 11.0 10.0 4.0 17.0 21.0
0.15

```

## 10.3 Program Results

nag\_robust\_trimmed\_lvar (g07ddc) Example Program Results

Statistics from middle 75.00% of data

Trimmed-mean =	8.8333
Variance of Trimmed-mean =	1.5434
Winsorized-mean =	9.1250
Variance of Winsorized-mean =	1.5381

---