

## NAG Library Function Document

### nag\_kruskal\_wallis\_test (g08afc)

#### 1 Purpose

nag\_kruskal\_wallis\_test (g08afc) performs the Kruskal–Wallis one-way analysis of variance by ranks on  $k$  independent samples of possibly unequal sizes.

#### 2 Specification

```
#include <nag.h>
#include <nagg08.h>

void nag_kruskal_wallis_test (Integer k, const Integer l[], const double x[],
    Integer lx, double *h, double *p, NagError *fail)
```

#### 3 Description

The Kruskal–Wallis test investigates the differences between scores from  $k$  independent samples of unequal sizes, the  $i$ th sample containing  $l_i$  observations. The hypothesis under test,  $H_0$ , often called the null hypothesis, is that the samples come from the same population, and this is to be tested against the alternative hypothesis  $H_1$  that they come from different populations.

The test proceeds as follows:

- The pooled sample of all the observations is ranked. Average ranks are assigned to tied scores.
- The ranks of the observations in each sample are summed, to give the rank sums  $R_i$ , for  $i = 1, 2, \dots, k$ .
- The Kruskal–Wallis' test statistic  $H$  is computed as:

$$H = \frac{12}{N(N+1)} \sum_{i=1}^k \frac{R_i^2}{l_i} - 3(N+1), \quad \text{where } N = \sum_{i=1}^k l_i,$$

i.e.,  $N$  is the total number of observations. If there are tied scores,  $H$  is corrected by dividing by:

$$1 - \frac{\sum (t^3 - t)}{N^3 - N}$$

where  $t$  is the number of tied scores in a group and the summation is over all tied groups.

nag\_kruskal\_wallis\_test (g08afc) returns the value of  $H$ , and also an approximation,  $p$ , to the probability of a value of at least  $H$  being observed,  $H_0$  is true. ( $H$  approximately follows a  $\chi_{k-1}^2$  distribution).  $H_0$  is rejected by a test of chosen size  $\alpha$  if  $p < \alpha$ . The approximation  $p$  is acceptable unless  $k = 3$  and  $l_1, l_2$  or  $l_3 \leq 5$  in which case tables should be consulted (e.g., O of Siegel (1956)) or  $k = 2$  (in which case the Median test (see nag\_median\_test (g08acc)) or the Mann–Whitney  $U$  test (see nag\_mann\_whitney (g08amc)) is more appropriate).

#### 4 References

- Moore P G, Shirley E A and Edwards D E (1972) *Standard Statistical Calculations* Pitman
- Siegel S (1956) *Non-parametric Statistics for the Behavioral Sciences* McGraw–Hill

## 5 Arguments

- 1: **k** – Integer *Input*  
*On entry:* the number of samples,  $k$ .  
*Constraint:*  $k \geq 2$ .
- 2: **I[k]** – const Integer *Input*  
*On entry:* **I**[ $i - 1$ ] must contain the number of observations  $l_i$  in sample  $i$ , for  $i = 1, 2, \dots, k$ .  
*Constraint:* **I**[ $i - 1$ ]  $> 0$ , for  $i = 1, 2, \dots, k$ .
- 3: **x[lx]** – const double *Input*  
*On entry:* the elements of **x** must contain the observations in the **k** groups. The first  $l_1$  elements must contain the scores in the first group, the next  $l_2$  those in the second group, and so on.
- 4: **lx** – Integer *Input*  
*On entry:* the total number of observations,  $N$ .  
*Constraint:*  $lx = \sum_{i=1}^k I[i - 1]$ .
- 5: **h** – double \* *Output*  
*On exit:* the value of the Kruskal–Wallis test statistic,  $H$ .
- 6: **p** – double \* *Output*  
*On exit:* the approximate significance,  $p$ , of the Kruskal–Wallis test statistic.
- 7: **fail** – NagError \* *Input/Output*  
The NAG error argument (see Section 2.7 in How to Use the NAG Library and its Documentation).

## 6 Error Indicators and Warnings

### NE\_ALLOC\_FAIL

Dynamic memory allocation failed.

### NE\_ARRAY\_CONS

The contents of array **I** are not valid.  
*Constraint:* **I**[ $i - 1$ ]  $> 0$ , for  $i = 1, 2, \dots, k$ .

### NE\_INT

*On entry,* **lx** =  $\langle value \rangle$ .  
*Constraint:*  $lx = \sum_{i=1}^k I[i - 1]$ , for  $i = 1, 2, \dots, k$ .

### NE\_INT\_ARG\_LT

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

### 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\_X\_IDEN**

On entry, all elements of  $\mathbf{x}$  are equal.

**7 Accuracy**

For estimates of the accuracy of the significance  $p$ , see `nag_prob_chi_sq` (g01ecc). The  $\chi^2$  approximation is acceptable unless  $k = 3$  and  $l_1, l_2$  or  $l_3 \leq 5$ .

**8 Parallelism and Performance**

`nag_kruskal_wallis_test` (g08afc) is not threaded in any implementation.

**9 Further Comments**

The time taken by `nag_kruskal_wallis_test` (g08afc) is small, and increases with  $N$  and  $k$ .

If  $k = 2$ , the Median test (see `nag_median_test` (g08acc)) or the Mann–Whitney  $U$  test (see `nag_mann_whitney` (g08amc)) is more appropriate.

**10 Example**

This example is taken from Moore *et al.* Moore *et al.* (1972). There are 5 groups of sizes 5, 8, 6, 8 and 8. The data represent the weight gain, in pounds, of pigs from five different litters under the same conditions.

**10.1 Program Text**

```

/* nag_kruskal_wallis_test (g08afc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagg08.h>

int main(void)
{
    Integer count, exit_status = 0, i, ii, k, *l = 0, lx, nhi, ni, nlo;
    NagError fail;
    double h, p, *x = 0;

    INIT_FAIL(fail);

    printf("nag_kruskal_wallis_test (g08afc) Example Program Results\n");

    /* Skip heading in data file */
#ifdef _WIN32
    scanf_s("%*[\n]");
#else
    scanf("%*[\n]");
#endif

    k = 5;
    if (!(l = NAG_ALLOC(k, Integer)))
    {
        printf("Allocation failure\n");
        exit_status = -1;
        goto END;
    }

```

```

    }
    for (i = 1; i <= k; i++)
#ifdef _WIN32
        scanf_s("%" NAG_IFMT " ", &l[i - 1]);
#else
        scanf("%" NAG_IFMT " ", &l[i - 1]);
#endif
    printf("\n");
    printf("%s\n", "Kruskal-Wallis test");
    printf("\n");
    printf("%s\n", "Data values");
    printf("\n");
    printf("%s\n", " Group      Observations");

    lx = 0;
    for (i = 1; i <= 5; ++i)
        lx += l[i - 1];

    if (!(x = NAG_ALLOC(lx, double)))
    {
        printf("Allocation failure\n");
        exit_status = -1;
        goto END;
    }
    for (i = 1; i <= lx; ++i)
#ifdef _WIN32
        scanf_s("%lf", &x[i - 1]);
#else
        scanf("%lf", &x[i - 1]);
#endif
    }

    nlo = 1;
    for (i = 1; i <= k; ++i) {
        ni = l[i - 1];
        nhi = nlo + ni - 1;
        printf(" %5" NAG_IFMT "      ", i);
        count = 1;
        for (ii = nlo; ii <= nhi; ++ii) {
            printf("%4.0f%s", x[ii - 1], count % 10 ? " " : "\n");
            count++;
        }
        nlo += ni;
        printf("\n");
    }
}
/* nag_kruskal_wallis_test (g08afc).
 * Kruskal-Wallis one-way analysis of variance on k samples
 * of unequal size
 */
nag_kruskal_wallis_test(k, l, x, lx, &h, &p, &fail);
if (fail.code != NE_NOERROR) {
    printf("Error from nag_kruskal_wallis_test (g08afc).\n%s\n",
        fail.message);
    exit_status = 1;
    goto END;
}
printf("\n");
printf("%s%9.3f\n", "Test statistic      ", h);
printf("%s%9" NAG_IFMT "\n", "Degrees of freedom      ", k - 1);
printf("%s%9.3f\n", "Significance          ", p);
END:
NAG_FREE(l);
NAG_FREE(x);
return exit_status;
}

```

**10.2 Program Data**

```
nag_kruskal_wallis_test (g08afc) Example Program Data
 5 8 6 8 8
 23 27 26 19 30 29 25 33 36 32
 28 30 31 38 31 28 35 33 36 30
 27 28 22 33 34 34 32 31 33 31
 28 30 24 29 30
```

**10.3 Program Results**

```
nag_kruskal_wallis_test (g08afc) Example Program Results
```

```
Kruskal-Wallis test
```

```
Data values
```

Group	Observations
1	23 27 26 19 30
2	29 25 33 36 32 28 30 31
3	38 31 28 35 33 36
4	30 27 28 22 33 34 34 32
5	31 33 31 28 30 24 29 30

```
Test statistic          10.537
Degrees of freedom      4
Significance             0.032
```

---