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

```c
#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:

(a) The pooled sample of all the observations is ranked. Average ranks are assigned to tied scores.

(b) The ranks of the observations in each sample are summed, to give the rank sums $R_i$, for $i = 1, 2, \ldots, k$.

(c) The Kruskal–Wallis’ test statistic $H$ is computed as:

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

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 $\chi^2_{k-1}$ 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


5 Arguments

1:  k – Integer
    Input
    On entry: the number of samples, \( k \).
    Constraint: \( k \geq 2 \).

2:  \( l[k] \) – const Integer
    Input
    On entry: \( l[i-1] \) must contain the number of observations \( l_i \) in sample \( i \), for \( i = 1, 2, \ldots, k \).
    Constraint: \( l[i-1] > 0 \), for \( i = 1, 2, \ldots, k \).

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: \( lx = \sum_{i=1}^{k} l[i-1] \).

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 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_ALLOC_FAIL
    Dynamic memory allocation failed.

NE_ARRAY_CONS
    The contents of array \( l \) are not valid.
    Constraint: \( l[i-1] > 0 \), for \( i = 1, 2, \ldots, k \).

NE_INT
    On entry, \( lx = \langle \text{value} \rangle \).
    Constraint: \( lx = \sum_{i=1}^{k} l[i-1] \), for \( i = 1, 2, \ldots, k \).

NE_INT_ARG_LT
    On entry, \( k = \langle \text{value} \rangle \).
    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.
7Accuracy

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

8Parallelism and Performance

Not applicable.

9Further 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.

10Example

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.

10.1Program Text

/* nag_kruskal_wallis_test (g08afc) Example Program. *
 * Copyright 2014 Numerical Algorithms Group. *
 * Mark 6, 2000. */
#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
");
    /* 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++)
```c
#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?"":"
);        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).
%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

<p>| | | | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>23</td>
<td>27</td>
<td>26</td>
<td>19</td>
<td>30</td>
<td>29</td>
<td>25</td>
<td>33</td>
<td>36</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>28</td>
<td>30</td>
<td>31</td>
<td>38</td>
<td>31</td>
<td>28</td>
<td>35</td>
<td>33</td>
<td>36</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>27</td>
<td>28</td>
<td>22</td>
<td>33</td>
<td>34</td>
<td>32</td>
<td>31</td>
<td>33</td>
<td>31</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

10.3 Program Results

nag_kruskal_wallis_test (g08afc) Example Program Results

Kruskal-Wallis test

Data values

<table>
<thead>
<tr>
<th>Group</th>
<th>Observations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>23 27 26 19 30</td>
</tr>
<tr>
<td>2</td>
<td>29 25 33 36 32 28 30 31</td>
</tr>
<tr>
<td>3</td>
<td>38 31 28 35 33 36</td>
</tr>
<tr>
<td>4</td>
<td>30 27 28 22 33 34 34 32</td>
</tr>
<tr>
<td>5</td>
<td>31 33 31 28 30 24 29 30</td>
</tr>
</tbody>
</table>

Test statistic 10.537
Degrees of freedom 4
Significance 0.032