NAG Toolbox

nag_nonpar_randtest_gaps (g08ed)

1 Purpose

nag_nonpar_randtest_gaps (g08ed) performs a gaps test on a sequence of observations.

2 Syntax

```
[ngaps, ncount, ex, chi, df, prob, ifail] = nag_nonpar_randtest_gaps(cl, x, m, rlo, rup, totlen, ngaps, ncount, 'n', n, 'maxg', maxg)
[ngaps, ncount, ex, chi, df, prob, ifail] = g08ed(cl, x, m, rlo, rup, totlen, ngaps, ncount, 'n', n, 'maxg', maxg)
```

3 Description

Gaps tests are used to test for cyclical trend in a sequence of observations. nag_nonpar_randtest_gaps (g08ed) computes certain statistics for the gaps test.

nag_nonpar_randtest_gaps (g08ed) may be used in two different modes:

(i) a single call to nag_nonpar_randtest_gaps (g08ed) which computes all test statistics after counting the gaps;

(ii) multiple calls to nag_nonpar_randtest_gaps (g08ed) with the final test statistics only being computed in the last call.

The second mode is necessary if all the data does not fit into the memory. See argument **cl** in Section 5 for details on how to invoke each mode.

The term gap is used to describe the distance between two numbers in the sequence that lie in the interval (r_l, r_u). That is, a gap ends at x_j if $r_l \leq x_j \leq r_u$. The next gap then begins at x_{j+1}. The interval (r_l, r_u) should lie within the region of all possible numbers. For example if the test is carried out on a sequence of $(0, 1)$ random numbers then the interval (r_l, r_u) must be contained in the whole interval $(0, 1)$. Let t_{len} be the length of the interval which specifies all possible numbers.

nag_nonpar_randtest_gaps (g08ed) counts the number of gaps of different lengths. Let c_i denote the number of gaps of length i, for $i = 1, 2, \ldots, k-1$. The number of gaps of length k or greater is then denoted by c_k. An unfinished gap at the end of a sequence is not counted unless the sequence is part of an initial or intermediate call to nag_nonpar_randtest_gaps (g08ed) (i.e., unless there is another call to nag_nonpar_randtest_gaps (g08ed) to follow) in which case the unfinished gap is used. The following is a trivial example.

Suppose we called nag_nonpar_randtest_gaps (g08ed) twice (i.e., with **cl** = 'F' and then with **cl** = 'L') with the following two sequences and with **rlo** = 0.30 and **rup** = 0.60:

(0.20 0.40 0.45 0.40 0.15 0.75 0.95 0.23) and

(0.27 0.40 0.25 0.10 0.34 0.39 0.61 0.12).

Then after the second call nag_nonpar_randtest_gaps (g08ed) would have counted the gaps of the following lengths:

2, 1, 1, 6, 3 and 1.

When the counting of gaps is complete nag_nonpar_randtest_gaps (g08ed) computes the expected values of the counts. An approximate χ^2 statistic with k degrees of freedom is computed where

$$X^2 = \frac{\sum_{i=1}^{k}(c_i - e_i)^2}{e_i},$$

where

$e_i = ngaps \times p \times (1-p)^{i-1}$, if $i < k$;

$e_i = ngaps \times (1-p)^{i-1}$, if $i = k$;

$ngaps =$ the number of gaps found and

$p = (r_u - r_l)/t_{\text{len}}$.

The use of the χ^2-distribution as an approximation to the exact distribution of the test statistic improves as the expected values increase.

You may specify the total number of gaps to be found. If the specified number of gaps is found before the end of a sequence nag_nonpar_randtest_gaps (g08ed) will exit before counting any further gaps.

4 References

Dagpunar J (1988) *Principles of Random Variate Generation* Oxford University Press

Knuth D E (1981) *The Art of Computer Programming (Volume 2)* (2nd Edition) Addison–Wesley

Morgan B J T (1984) *Elements of Simulation* Chapman and Hall

Ripley B D (1987) *Stochastic Simulation* Wiley

5 Parameters

5.1 Compulsory Input Parameters

1: **cl** – CHARACTER(1)

Indicates the type of call to nag_nonpar_randtest_gaps (g08ed).

cl = 'S'
This is the one and only call to nag_nonpar_randtest_gaps (g08ed) (single call mode). All data are to be input at once. All test statistics are computed after the counting of gaps is complete.

cl = 'F'
This is the first call to the function. All initializations are carried out before the counting of gaps begins. The final test statistics are not computed since further calls will be made to nag_nonpar_randtest_gaps (g08ed).

cl = 'I'
This is an intermediate call during which the counts of gaps are updated. The final test statistics are not computed since further calls will be made to nag_nonpar_randtest_gaps (g08ed).

cl = 'L'
This is the last call to nag_nonpar_randtest_gaps (g08ed). The test statistics are computed after the final counting of gaps is complete.

Constraint: **cl** = 'S', 'F', 'I' or 'L'.

2: **x**(**n**) – REAL (KIND=nag_wp) array

The sequence of observations.

3: **m** – INTEGER

The maximum number of gaps to be sought. If $\mathbf{m} \leq 0$ then there is no limit placed on the number of gaps that are found.

m should not be changed between calls to nag_nonpar_randtest_gaps (g08ed).

Constraint: if $\mathbf{cl} = \text{'S'}$, $\mathbf{m} \leq \mathbf{n}$.

4: **rlo** – REAL (KIND=nag_wp)

The lower limit of the interval to be used to define the gaps, r_l.

rlo must not be changed between calls to nag_nonpar_randtest_gaps (g08ed).

5: **rup** – REAL (KIND=nag_wp)

The upper limit of the interval to be used to define the gaps, r_u.

rup must not be changed between calls to nag_nonpar_randtest_gaps (g08ed).

Constraint: $\mathbf{rup} > \mathbf{rlo}$.

6: **totlen** – REAL (KIND=nag_wp)

The total length of the interval which contains all possible numbers that may arise in the sequence.

Constraint: $\mathbf{totlen} > 0.0$ and $\mathbf{rup} - \mathbf{rlo} < \mathbf{totlen}$.

7: **ngaps** – INTEGER

If $\mathbf{cl} = \text{'S'}$ or 'F', **ngaps** need not be set.

If $\mathbf{cl} = \text{'I'}$ or 'L', **ngaps** must contain the value returned by the previous call to nag_nonpar_randtest_gaps (g08ed).

8: **ncount**(**maxg**) – INTEGER array

If $\mathbf{cl} = \text{'S'}$ or 'F', **ncount** need not be set.

If $\mathbf{cl} = \text{'I'}$ or 'L', **ncount** must contain the values returned by the previous call to nag_nonpar_randtest_gaps (g08ed).

5.2 Optional Input Parameters

1: **n** – INTEGER

Default: the dimension of the array **x**.

n, the length of the current sequence of observations.

Constraint: $\mathbf{n} \geq 1$.

2: **maxg** – INTEGER

Default: the dimension of the array **ncount**.

k, the length of the longest gap for which tabulation is desired.

maxg must not be changed between calls to nag_nonpar_randtest_gaps (g08ed).

Constraints:

$\mathbf{maxg} > 1$;
if $\mathbf{cl} = \text{'S'}$, $\mathbf{maxg} \leq \mathbf{n}$.

5.3 Output Parameters

1: **ngaps** – INTEGER

The number of gaps actually found, $ngaps$.

2: **ncount**(**maxg**) – INTEGER array

The counts of gaps of the different lengths, c_i, for $i = 1, 2, \ldots, k$.

3: **ex**(**maxg**) – REAL (KIND=nag_wp) array

If **cl** = 'S' or 'L' (i.e., if it is a final exit) then **ex** contains the expected values of the counts.

Otherwise the elements of **ex** are not set.

4: **chi** – REAL (KIND=nag_wp)

If **cl** = 'S' or 'L' (i.e., if it is a final exit) then **chi** contains the χ^2 test statistic, X^2, for testing the null hypothesis of randomness.

Otherwise **chi** is not set.

5: **df** – REAL (KIND=nag_wp)

If **cl** = 'S' or 'L' (i.e., if it is a final exit) then **df** contains the degrees of freedom for the χ^2 statistic.

Otherwise **df** is not set.

6: **prob** – REAL (KIND=nag_wp)

If **cl** = 'S' or 'L' (i.e., if it is a final exit) then **prob** contains the upper tail probability associated with the χ^2 test statistic, i.e., the significance level.

Otherwise **prob** is not set.

7: **ifail** – INTEGER

ifail = 0 unless the function detects an error (see Section 5).

6 Error Indicators and Warnings

Note: nag_nonpar_randtest_gaps (g08ed) may return useful information for one or more of the following detected errors or warnings.

Errors or warnings detected by the function:

ifail = 1

On entry, **cl** = ⟨*value*⟩.
Constraint: **cl** = 'S', 'F', 'I' or 'L'.

ifail = 2

Constraint: **n** ≥ 1.

ifail = 3

Constraint: if **cl** = 'S', **m** ≤ **n**.

ifail = 4

Constraint: if **cl** = 'S', **maxg** ≤ **n**.

Constraint: **maxg** > 1.

ifail $= 5$

Constraint: **rup** $>$ **rlo**.

Constraint: **rup** $-$ **rlo** $<$ **totlen**.

Constraint: **totlen** > 0.0.

ifail $= 6$

No gaps were found. Try using a longer sequence, or increase the size of the interval **rup** $-$ **rlo**.

ifail $= 7$

The expected frequency in class is zero. The value of $(\mathbf{rup} - \mathbf{rlo})/\mathbf{totlen}$ may be too close to 0.0 or 1.0. or **maxg** is too large relative to the number of gaps found.

ifail $= 8$ (*warning*)

The number of gaps requested were not found, only $\langle value \rangle$ out of the requested $\langle value \rangle$ where found.
All statistics are returned and may still be of use.

ifail $= 9$ (*warning*)

The expected frequency of at least one class is less than one.
This implies that the χ^2 may not be a very good approximation to the distribution of the test statistics.
All statistics are returned and may still be of use.

ifail $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

ifail $= -399$

Your licence key may have expired or may not have been installed correctly.

ifail $= -999$

Dynamic memory allocation failed.

7 Accuracy

The computations are believed to be stable. The computation of **prob** given the values of **chi** and **df** will obtain a relative accuracy of five significant places for most cases.

8 Further Comments

The time taken by nag_nonpar_randtest_gaps (g08ed) increases with the number of observations n, and depends to some extent whether the call is an only, first, intermediate or last call.

9 Example

The following program performs the gaps test on 500 pseudorandom numbers. nag_nonpar_randtest_gaps (g08ed) is called 5 times with 100 observations on each call. All gaps of length 10 or more are counted together.

9.1 Program Text

```
     function g08ed_example

fprintf('g08ed example results\n\n');

% Initialize the base generator to a repeatable sequence
seed  = [nag_int(324213)];
genid = nag_int(1);
subid = nag_int(1);
[state, ifail] = g05kf( ...
                        genid, subid, seed);

m      = nag_int(0);
rlo    = 0.4;
rup    = 0.6;
totlen = 1;
ncount = zeros(10, 1, nag_int_name);
ngaps  = nag_int(0);
nsampl = 5;
n      = nag_int(100);
cl    = 'F';

for i=1:nsampl
  % Generate a sample from U(0,1)
  [state, x, ifail] = g05sq( ...
                             n, 0, 1, state);
  % Process the sample
  [ngaps, ncount, ex, chi, df, prob, ifail] = ...
  g08ed( ...
         cl, x, m, rlo, rup, totlen, ngaps, ncount);
  % Adjust CL
  cl = 'I';
  if i==nsampl-1
     cl = 'L';
  end
end

fprintf('Total number of gaps found = %d\n', ngaps);
fprintf('\n%33s\n', 'Count');
head = sprintf('%7d', [0:8]);
head = sprintf('%s     >8', head);
fprintf('%s\n', head);
fprintf('%7d', ncount);

fprintf('\n\n%34s\n', 'Expect');
fprintf('%s\n', head);
fprintf('%7.1f', ex);
fprintf('\n\n');

fprintf('Chisq = %10.4f\n', chi);
fprintf('DF    = %7.1f\n', df);
fprintf('Prob  = %10.4f\n', prob);
```

9.2 Program Results

```
     g08ed example results

Total number of gaps found = 99

                           Count
      0      1      2      3      4      5      6      7      8     >8
     22     11     10     13      6     12      4      6      2     13

                           Expect
      0      1      2      3      4      5      6      7      8     >8
```

```
   19.8   15.8   12.7   10.1    8.1    6.5    5.2    4.2    3.3   13.3

Chisq =     9.9540
DF    =     9.0
Prob  =     0.3542
```